



AVEVA™ InTouchHMI

© 2015-2024 by AVEVA Group Limited or its subsidiaries. All rights reserved.

本ガイドのいかなる部分も、AVEVA Group Limited の書面による事前の許可なく、いかなる形式、手段的、機械的、複写、記録、またはその他の手段を含むによっても、複製すること、検索システムに保存すること、および送信することはできません。ここに記載されている情報の使用に関しては、いかなる責任も負いません。

このマニュアルの準備には予防措置が取られていますが、AVEVA はエラーや省略に対して責任を負いません。このマニュアルに記載されている情報は予告なしに変更される場合があります、AVEVA 側の誓約を示すものではありません。このマニュアルに記載されているソフトウェアは、ライセンス契約に基づき提供されます。本ソフトウェアは、そのようなライセンス契約の条件に従ってのみ使用またはコピーできます。AVEVA、AVEVA ロゴおよびロゴタイプ、OSIsoft、OSIsoft ロゴおよびロゴタイプ、ArcheStrA、Avantis、Citect、DYNsIM、eDNA、EYESIM、InBatch、InduSoft、InStep、IntelaTrac、InTouch、Managed PI、OASyS、OSIsoft Advanced Services、OSIsoft Cloud Services、OSIsoft Connected Services、OSIsoft EDS、PIPEPHASE、PI ACE、PI Advanced Computing Engine、PI AF SDK、PI API、PI Asset Framework、PI Audit Viewer、PI Builder、PI Cloud Connect、PI Connectors、PI Data Archive、PI DataLink、PI DataLink Server、PI Developers Club、PI Integrator for Business Analytics、PI Interfaces、PI JDBC Driver、PI Manual Logger、PI Notifications、PI ODBC Driver、PI OLEDB Enterprise、PI OLEDB Provider、PI OPC DA Server、PI OPC HDA Server、PI ProcessBook、PI SDK、PI Server、PI Square、PI System、PI System Access、PI Vision、PI Visualization Suite、PI Web API、PI WebParts、PI Web Services、PRiSM、PRO/II、PROVISION、ROMeo、RLINK、RtReports、SIM4ME、SimCentral、SimSci、Skelta、SmartGlance、Spiral Software、WindowMaker、WindowViewer、Wonderware は AVEVA および/または子会社の商標です。その他のすべてのブランドは、それぞれの所有者の商標である可能性があります。

米国政府の権利

米国政府による使用、複製、開示は、AVEVA Group Limited または子会社とのライセンス契約および DFARS 227.7202、DFARS 252.227-7013、FAR 12-212、FAR 52.227-19、またはこれらを継承するものに記載されている制限に準じるものとします。

発行日 : Tuesday, September 10, 2024

パブリケーション ID : 1425708

Contact information (お問い合わせ先)

AVEVA Group Limited
High Cross
Maddingley Road
Cambridge
CB3 0HB. UK

<https://sw.aveva.com/>

セールスおよびカスタマー トレーニングへの連絡方法については、<https://sw.aveva.com/contact> を参照してください。

テクニカル サポートへの連絡方法については、<https://sw.aveva.com/support> を参照してください。

AVEVA のナレッジ ベースとサポート センターには、<https://softwaresupport.aveva.com> からアクセスできます。

目次

Standard	22
統合開発環境	23
InTouch アプリケーション マネージャについて	23
アプリケーション エクスプローラの使用	24
アプリケーション エクスプローラでのナビゲート	27
アプリケーション エクスプローラへのアプリケーションの追加	27
ツールバーの管理	28
WindowMaker 環境設定	28
マウス ショートカット	32
全画面モードの使用	33
[ファイル] メニューからのウィンドウの管理	33
フォントのデフォルト設定	34
矢印キーを使用したオブジェクトの移動	34
カラーパレットの使用	35
カラーパレットを開く	35
カスタム カラーの作成	36
カスタム カラーのインポートとエクスポート	37
パンとズーム	37
サムネイル ウィンドウを使用したパンとズーム	38
マウス ホイールを使用したズームとパン	39
パンとズームの制限	39
画面グリッドとルーラーの使用	39
グリッドへのオブジェクトの配置	40
ルーラーの使用	40
WindowViewer について	40
ランタイム環境のカスタマイズ	40
リモートセッションで実行するアプリケーションへのユーザー アクセスの設定	46
WindowViewer のメモリ管理について	47
WindowViewer ウィンドウの操作	56
産業用グラフィック	57
産業用グラフィックの作成	58
産業用グラフィック エディタ	58
アプリケーション	60
スタンドアロン InTouch アプリケーション	60
マネージド InTouch アプリケーション	61
パブリッシュ済み InTouch アプリケーション	62
InTouchView アプリケーション	63
Application Server のアーキテクチャ	64

Galaxy との通信.....	65
スタンドアロン、マネージド、およびパブリッシュ済みの InTouch アプリケーションの比較....	66
アプリケーションの構築.....	67
アプリケーションの実行.....	68
タグ.....	69
InTouch タグ変数の操作.....	69
InTouch タグ変数のタイプ.....	70
メモリ型タグ.....	71
I/O タグ変数.....	72
間接型タグ変数.....	73
その他のタグ変数.....	74
システム タグ変数.....	75
システム タグ変数リファレンス.....	75
タグ プロパティ.....	80
メモリ型タグ変数のプロパティ.....	80
I/O 型タグ変数のプロパティ.....	82
リモート タグ リファレンス.....	84
間接型タグ変数の定義.....	84
間接型タグ変数とスクリプトの使用.....	85
間接型タグ変数とローカル タグ変数の使用.....	86
間接型タグとリモート参照の使用.....	86
再使用可能なタグ構造の定義.....	88
スーパータグ テンプレートとメンバー タグの管理.....	90
スーパータグ インスタンス.....	91
スーパータグ 属性.....	96
スーパータグ メンバーの参照.....	97
Bulk Import Utility を使用したスーパータグのインポート.....	98
MapApp ウィジェットとスーパータグの使用.....	98
他のアプリケーションから履歴タグ変数値へのアクセス.....	105
DDE アイテムを使用した履歴データの表示.....	106
DDE によるログ データへのアクセス.....	109
他のアプリケーションからの履歴データへのアクセス.....	115
HistData エラーのトラブルシューティング.....	116
IEEE 小数ユニット.....	118
Historian HMI における浮動小数点数の表示.....	118
InTouch OPC UA サーバーの設定および使用.....	119
OPC UA の設定とチェックリスト.....	119
InTouch OPC UA サーバーの設定.....	120
OPC UA サービス用のファイアウォールの設定.....	122
サードパーティ製 OPC UA クライアント アプリケーション用のサーバーおよびクライアント 証明書の設定.....	126
OI Gateway を使用したクライアント セキュリティ証明書の設定.....	130
アラーム.....	134
InTouch アラームの概要.....	134
アラーム優先度.....	135
アラームのサブステート.....	135
アラームの確認.....	136
アラーム グループ.....	136

InTouch イベントの概要.....	136
InTouch アラームのタイプ.....	137
論理値アラーム.....	138
アナログアラーム.....	138
InTouch 分散アラート システム.....	139
アラーム プロバイダとコンシューマ.....	141
分散アラーム グループ リスト.....	142
サマリ アラームと履歴アラーム.....	143
アラームの無効化、抑止、非表示.....	143
ターミナル サービス、アラームのサポート.....	144
分散アラーム システム データの保存.....	144
Build.	145
InTouch HMI アプリケーションの管理.....	146
アプリケーション マネージャの起動.....	146
アプリケーション マネージャの使用.....	147
InTouch アプリケーションの作成.....	148
新規 InTouchView アプリケーションの作成.....	150
InTouch アプリケーションから InTouchView アプリケーションへの変更.....	150
WindowMaker と WindowViewer でアプリケーションを開く.....	151
アプリケーション マネージャ ウィンドウのカスタマイズ.....	152
アプリケーション タイルの使用.....	153
InTouch アプリケーションのロックおよびロック解除.....	154
InTouch アプリケーションの変更.....	155
アプリケーション マネージャからの InTouch アプリケーションの削除.....	157
InTouch アプリケーションの検索.....	158
アプリケーション テンプレートでの作業.....	159
InTouch アプリケーション フォルダの表示.....	161
テンプレートとして使用するための InTouch アプリケーションのエクスポート.....	161
アプリケーション マネージャを使用した Web Client 設定の更新.....	162
InTouch OMI ViewApp の起動.....	163
AVEVA Identity Manager での登録.....	163
資格情報マネージャの操作.....	164
名前付き資格情報の管理.....	165
名前付き資格情報のエクスポートとインポート.....	166
IDE を使用した InTouch アプリケーションの管理.....	168
マネージド InTouch アプリケーションの作成.....	169
アプリケーション テンプレートからのマネージド アプリケーションの作成.....	172
IDE からの WindowMaker の起動.....	174
マネージド InTouch アプリケーションの言語の切り替え.....	175
InTouch アプリケーションの変更の送信.....	176
InTouch アプリケーションのインポート.....	176
InTouchViewApp オブジェクトのインポートとエクスポート.....	178
タグ データのエクスポートとインポート.....	178
タグの値およびパラメータの保持.....	179
マネージド InTouch アプリケーションの削除.....	180
InTouchViewApp 内のすべての Galaxy グラフィックの関連付け.....	180

ウィンドウ.....	182
アプリケーション ウィンドウの作成.....	182
テンプレート ウィンドウとしてのアプリケーション ウィンドウの設定.....	185
テンプレート ウィンドウからのアプリケーション ウィンドウの作成.....	186
枠ウィンドウの操作.....	186
プロパティ パネルの使用.....	187
枠に埋め込まれたグラフィックの操作.....	189
枠ウィンドウのリボンの使用.....	191
アプリケーション ウィンドウの変更.....	191
ウィンドウを開く、保存する、閉じる.....	192
ウィンドウ サムネイルプレビューの表示.....	193
ウィンドウの複製.....	194
ウィンドウの削除.....	194
クイック アクセス ツールバーの使用.....	194
InTouch ウィンドウに関する情報の印刷.....	196
コマンドプロンプトからのウィンドウ情報の印刷.....	198
ウィンドウを印刷するための .CSV 形式.....	198
コマンドプロンプトから印刷するための構文.....	199
コンテキスト メニューの深さの設定.....	199
グラフィック要素.....	201
WindowMaker オブジェクト.....	201
WindowMaker オブジェクト.....	201
シンプル オブジェクト.....	202
複合オブジェクト.....	204
共通の操作.....	206
すべてのオブジェクトの特殊操作.....	213
特殊オブジェクトの特殊操作.....	215
オブジェクトのアニメーション化.....	219
産業用グラフィック エディタの操作.....	257
産業用グラフィックの管理.....	257
HMI/SCADA アプリケーションと共に使用する産業用グラフィック エディタ ワークフロー.....	258
シンボル変更の反映.....	263
シンボルのダイナミック サイズの反映.....	263
WindowMaker と産業用グラフィック エディタの比較.....	263
InTouch タグへのアニメーションの接続.....	273
WindowMaker での産業用グラフィックの使用.....	275
WindowMaker での産業用グラフィックの使用について.....	276
InTouch ウィンドウへの産業用グラフィックの埋め込み.....	277
埋め込まれた産業用グラフィックのサイズ変更.....	282
WindowMaker での産業用グラフィックの設定.....	283
産業用グラフィック エディタでの産業用グラフィックの編集.....	293
InTouch タグを使用したグラフィック要素と産業用グラフィックの作成.....	295
産業用グラフィック エディタの [ツールボックス] タブからのグラフィックの埋め込み.....	297
WindowViewer での産業用グラフィックのテスト.....	298
グラフィック パフォーマンスの予測.....	300
新しいオートメーション インスタンスの作成.....	300
新しい Application Server オブジェクト インスタンスの自動作成.....	300
InTouch ウィンドウから産業用グラフィックへの変換.....	301

ウィンドウの変換の準備.....	301
ウィンドウの変換.....	301
アニメーションスクリプトの変換.....	302
ウィンドウ変換の既知の制限事項.....	302
ウィンドウを変換した後.....	303
ウィンドウ変換手順の完了.....	304
ウィンドウ変換エラーの診断.....	305
シンボル ウィザード エディタでのシンボル ウィザードの作成.....	306
シンボル ウィザードの作成.....	307
シンボル ウィザードのデザイナー ワークフロー.....	307
シンボル ウィザードのコンシューマ ワークフロー.....	308
トレンド ペンの履歴データ検索について.....	308
ランタイム時のトレンド ペン プロパティの変更.....	310
MinValue プロパティ.....	310
MaxValue プロパティ.....	311
StartTime プロパティ.....	311
EndTime プロパティ.....	312
PlotType プロパティ.....	312
TimeMode プロパティ.....	312
FillTrend プロパティ.....	312
マルチ ペン トレンドの設定.....	313
ペン名と参照の設定.....	314
ペンの詳細の設定.....	315
ペン オプションの設定.....	316
ソースの設定.....	317
マルチ ペン トレンドのカスタマイズ.....	317
ランタイム時のマルチ ペン トレンドの使用.....	318
SmartSymbol について.....	326
SmartSymbol マネージャおよびライブラリ.....	326
InTouch SmartSymbol および産業用グラフィック SmartSymbol.....	326
SmartSymbol の制限.....	329
SmartSymbol テンプレートとインスタンスの作成.....	329
SmartSymbol の管理.....	337
SmartSymbol の編集.....	342
InTouch SmartSymbol の移行.....	348
タグ.....	355
タグ名ディクショナリを使用したタグの管理.....	355
タグ変数の使用法の計画.....	356
新しいタグの作成.....	357
タグ プロパティの設定.....	358
タグ変数の変更.....	365
OPC UA サーバーからの InTouch タグの作成.....	366
タグ変数の削除.....	370
タグ変数リストと使用情報の印刷.....	370
タグ ドットフィールドを使用したタグ プロパティの表示または変更.....	370
利用可能なタグ タイプのドットフィールド.....	371
タグ変数のしきい値の変更.....	379
I/O を使用したデータ アクセス.....	388

Gateway Communications Driver の操作.....	389
サポートされている InTouch 通信プロトコル.....	391
アクセス名の設定.....	400
I/O タグ変数による I/O データへのアクセス.....	402
タグのリモート参照への変換.....	411
リモート参照による I/O データアクセス.....	414
InTouch から Application Server データへのアクセス.....	418
I/O タグ変数の時間スタンプと品質情報の表示.....	428
ランタイム時の I/O 接続の初期化と再設定.....	439
アクセス名によるフェイルオーバー機能の使用.....	443
I/O 接続ステータスの監視.....	449
他のアプリケーションからの InTouch タグ変数データへのアクセス.....	452
タグ値の記録.....	453
履歴ログの設定.....	454
ランタイムでの履歴ログの開始と停止.....	461
タグデータのトレンド.....	462
InTouch トレンドのタイプ.....	462
保存済みタグの値の履歴トレンド表示.....	463
履歴トレンドオブジェクトの使用.....	464
履歴トレンドウィザードの使用.....	481
スクリプトの使用による履歴トレンドウィザードのコントロール.....	486
リアルタイムトレンドオブジェクトの使用.....	497
ランタイム中のトレンド印刷.....	500
他の InTouch ノードや Historian Server の履歴タグ値の表示.....	501
ユーザー定義型.....	506
UDT の概要.....	506
UDT の仕様.....	507
手動でのユーザー定義型の作成.....	508
新しいデータ型の作成.....	508
新しいメンバーの作成.....	509
新しい派生データ型の作成.....	510
新しいインスタンスの作成.....	511
別のデータ型の下でのメンバーの入れ子.....	512
インポートによるユーザー定義型のセットの構築.....	514
データ型のエクスポート.....	517
ユーザー定義型の管理.....	518
ユーザー定義型の一括編集.....	519
ユーザー定義型の表示.....	521
[ユーザー定義型] ビュー.....	521
[モデル-タグ名] ビュー.....	522
プロパティグリッドでのユーザー定義型の編集.....	523
変更の反映.....	526
既存の機能の UDT サポート.....	527
Intellisense.....	528
アニメーション.....	529
スクリプト.....	529
参照の変更.....	530
UDT メンバーの参照.....	531

インスタンスの UDT メンバーの産業用グラフィック エディタ キャンバスへのドラッグアンドドロップ.....	532
アニメーションとスクリプト.....	533
アラームおよびアラーム クライアント コントロール.....	533
履歴およびトレンド クライアント.....	535
I/O タグとしての UDT メンバーの設定.....	536
所有オブジェクトとしての UDT インスタンスの使用.....	538
Web Client.....	539
MapApp.....	540
クロス リファレンス.....	541
未使用タグの削除.....	542
スーパータグとの共存.....	543
ライセンス.....	544
ランタイム Tag Viewer.....	545
UDT の制限事項.....	546
アラーム.....	547
アラームの設定.....	547
アラーム階層の定義.....	548
アラーム状況を使用したタグ変数の設定.....	551
個々のタグのイベントプロパティの設定.....	557
アラームとイベントのグローバル設定.....	557
アラーム グループ リスト ファイルの作成.....	560
アラーム クエリー.....	561
アラーム クエリーの例.....	563
詳細な InTouch クエリー情報の取得.....	563
アラーム冗長性による工場セキュリティの強化.....	564
Hot Backup の理解.....	565
Hot Backup ペアの設定.....	566
Hot Backup ペアの例.....	576
Hot Backup ペアについての注記.....	580
アラーム監査記録の作成.....	581
認証を要求するアラームの確認.....	581
分散アラーム表示オブジェクトの操作.....	584
分散アラーム表示オブジェクトの操作.....	584
分散アラーム表示オブジェクトについて.....	584
デザイン時の分散アラーム表示オブジェクトの設定.....	585
ランタイムでの分散アラーム表示オブジェクトの使用.....	595
関数とドットフィールドを使用した分散アラーム表示オブジェクトの制御.....	600
アラーム階層の表示.....	641
Alarm Viewer コントロールの設定.....	641
ランタイムでの Alarm Tree Viewer コントロールの使用.....	649
Alarm Tree Viewer コントロールの ActiveX プロパティの使用.....	650
Alarm Tree Viewer コントロールの ActiveX メソッドの使用.....	652
メソッドとプロパティを使用するときのエラー処理.....	657
Alarm Tree Viewer コントロール ActiveX イベントを使用したスクリプトのトリガ.....	657
アラームの印刷.....	658
アラーム印刷とログの設定.....	659
Alarm Printer でのアラームの印刷の概要.....	670

アラームのファイルへのログ.....	671
特定の設定での Alarm Printer の起動.....	671
スクリプトを使用した Alarm Printer の制御.....	672
アラーム データベースへのアラームの記録.....	688
Alarm DB Logger Manager 用の SQL Server アカウント.....	689
Alarm DB Logger Manager の使用.....	689
アラーム データベースのビュー.....	696
アラーム データベースのストアードプロシージャ.....	702
レコードされたアラームの表示.....	704
Alarm DB View コントロールの設定.....	704
ランタイムでの Alarm DB View コントロールの使用.....	726
Alarm DB View ActiveX プロパティの使用.....	727
Alarm DB View ActiveX メソッドの使用.....	751
メソッドとプロパティを使用するときのエラー処理.....	756
Alarm DB View ActiveX イベントを使用したスクリプトのトリガ.....	756
複数のタグにわたるアラーム分散の分析.....	757
Alarm Pareto ActiveX コントロールの設定.....	757
ランタイムでの Alarm Pareto コントロールの使用.....	768
Alarm Pareto ActiveX プロパティの使用.....	769
Alarm Pareto ActiveX メソッドの使用.....	771
メソッドとプロパティを使用するときのエラー処理.....	775
Alarm Pareto ActiveX イベントを使用したスクリプトのトリガ.....	775
スクリプト.....	776
基本的なスクリプトの概念.....	776
スクリプトのタイプ.....	777
高度なスクリプトの概念.....	777
OLE オブジェクト.....	777
ActiveX コントロールを使用したスクリプト.....	778
スクリプトの作成と編集.....	778
InTouch スクリプト エディタの操作.....	778
編集用にスクリプトを開く.....	783
スクリプトへの変更の保存または破棄.....	784
テキストのコピー、切り取り、貼り付け.....	784
スクリプト内での検索.....	784
コード要素の挿入.....	791
スクリプト関数のヘルプへのアクセス.....	791
正確な構文を確認するためのスクリプトの検証.....	791
スクリプトの印刷.....	792
スクリプトの削除.....	793
スクリプトを表示するズーム オプションの調整.....	793
スクリプト トリガ.....	793
スクリプト トリガのタイプ.....	794
複数のトリガの使用.....	795
定期的なスクリプトの実行.....	795
アプリケーション スクリプトの設定.....	795
ウィンドウ スクリプトの設定.....	796
キー スクリプトの設定.....	798
条件スクリプトの設定.....	800

データ変化スクリプトの設定.....	802
アクション スクリプトの設定.....	803
ActiveX イベント スクリプトの設定.....	807
ランタイムでのスクリプト実行の一時停止.....	810
スクリプト言語.....	810
基本構文規則.....	811
標準関数の呼び出し.....	813
カスタム関数（クイック関数）の呼び出し.....	814
値の割り当てと演算子.....	815
条件付きプログラム分岐構造の使用.....	823
プログラム ループの使用.....	824
ローカル変数の使用.....	827
カスタム スクリプト関数.....	828
クイック 関数について.....	828
クイック 関数の設定.....	828
クイック 関数の呼び出し.....	830
非同期クイック関数の作成.....	830
組み込み関数.....	831
アニメーション表示リンクでの強制更新.....	832
数学計算.....	832
文字列操作.....	839
データ タイプの変換.....	848
ランタイムでの InTouch ウィンドウの操作.....	852
日付と時刻情報の操作.....	864
他のアプリケーションとの対話.....	873
ファイルの操作.....	880
システム関連情報の取得.....	886
InTouch 関連情報の取得.....	889
セキュリティ関連スクリプト.....	891
その他のスクリプト.....	892
OLE オブジェクトを使用したスクリプト.....	897
OLE オブジェクトの作成、確認および解放.....	897
OLE オブジェクトのプロパティとメソッドの使用.....	899
同じ OLE オブジェクトへの複数のポインタの割り当て.....	900
OLE エラーのトラブルシューティング.....	901
OLE を使用して行える操作.....	903
ActiveX コントロールのスクリプト.....	906
ActiveX コントロール メソッドの呼び出し.....	906
InTouch HMI からの ActiveX コントロール プロパティへのアクセス.....	907
ActiveX イベント スクリプトの作成と再使用.....	909
ActiveX イベント スクリプトのインポート.....	912
ActiveX コントロール.....	913
ActiveX コントロールの使用.....	913
ActiveX コントロールの設定.....	915
ActiveX コントロールの命名.....	915
ActiveX コントロールの標準操作.....	916
ActiveX コントロールのインストールとアンインストール.....	916
ウィザード.....	918

ウィザードの操作.....	918
ウィザードのタイプ.....	918
ツールバーへのウィザードの追加.....	919
ウィザード インスタンスの貼り付け.....	920
ウィザードの設定.....	920
ウィザードの標準操作の実行.....	920
ウィザードのインストールとアンインストール.....	920
トレンドオブジェクト.....	922
ウィンドウ コントロール ウィザード.....	922
ウィンドウ コントロールの作成と設定.....	923
テキスト ボックスの作成.....	923
リスト ボックスの作成.....	924
コンボ ボックスの作成.....	925
チェック ボックスの作成.....	926
ラジオ ボタン グループの作成.....	927
ウィンドウ コントロールのスクリプト記述.....	928
コントロールの値の取得または設定.....	928
ユーザー入力用のコントロールの有効化または無効化.....	930
ウィンドウ コントロールの動的非表示.....	931
コンボ ボックスのアイテムの追加と削除.....	931
ファイルに対するリスト アイテムのロードと保存.....	935
コンボ ボックスまたはリストのアイテムの検索.....	937
コンボ ボックスまたはリストのアイテム インデックスの操作.....	938
リスト ボックスまたはコンボ ボックス アイテムのカウント.....	941
リスト アイテムの値の取得または設定.....	942
リスト アイテムの名前の取得.....	943
テキスト ボックスの内容のロード.....	944
テキスト ボックスが読み取り専用であるかどうかの確認.....	946
チェック ボックスのラベルの取得または設定.....	947
ウィンドウ コントロール エラー メッセージの理解.....	947
XML ファイルを使用したウィンドウのインポート.....	949
XML ファイルの準備.....	950
コマンド ファイルの準備.....	950
最小コマンド ファイルの作成.....	951
ファイルへの印刷情報の送信.....	951
ファイルへのクロス リファレンスの送信.....	951
ログ ファイルの作成.....	952
コマンドの構文.....	952
アプリケーションの作成.....	953
新しく作成されたアプリケーションへのタグの追加.....	954
ウィンドウの削除.....	954
ウィンドウの名前の変更.....	955
ウィンドウのインポート.....	955
エラーの処理.....	956
SmartSymbol の欠落.....	956
産業用グラフィックの欠落.....	956
式、タグ名、およびスクリプト.....	957
コマンド プロンプトからの WindowMaker の実行.....	957

System Platform IDE 拡張機能.....	957
マネージド InTouch アプリケーションからの WindowMaker の実行.....	958
コマンドプロンプトからの DBDump の実行.....	958
マネージド InTouch アプリケーションに対する DBDump の実行.....	959
コマンドプロンプトからの DBLoad の実行.....	959
マネージド InTouch アプリケーションに対する DBLoad の実行.....	960
XML の形式.....	960
一般的な XML ファイル形式.....	961
共通の要素定義.....	961
ウィンドウ要素.....	968
サポートされていない InTouch 機能.....	1032
AVEVA Connect での産業用グラフィックの操作.....	1033
AVEVA Connect へのログイン.....	1033
共有ドライブ間のナビゲーション.....	1034
グラフィックのアップロード/ダウンロード.....	1034
AVEVA Connect へのグラフィックのアップロード.....	1034
Download Graphics to the Local Visualization Folder.....	1035
さまざまなクラウドコンテンツのアップロード/ダウンロードのサポート.....	1036
AVEVA Connect でのグラフィックの管理.....	1036
Managing Graphics with Multiple Users.....	1036
クラウドグラフィックのバージョン管理.....	1037
クラウドでのグラフィック コンテンツの上書き.....	1037
Cloud Graphic Versioning - Scenarios and Examples.....	1038
Deploy.....	1041
ターミナル サービスとリモートデスクトップ サービスの配置と操作.....	1042
ターミナル サーバー アプリケーションの計画上の考慮事項.....	1042
ターミナル サービス環境へのマネージド InTouch アプリケーションの配置.....	1043
ターミナル サービス環境のアラーム.....	1043
ターミナル サービス環境のセキュリティ.....	1043
ターミナル サービス環境の I/O.....	1044
ターミナル サービス環境でのスクリプトの実行.....	1044
InTouch を実行するためのターミナルセッションへの適切なログオン.....	1044
ターミナル サービス環境のアラーム クエリーの構文.....	1045
ターミナル サービス環境のその他の制限.....	1045
スクリプトを使用した InTouch クライアントセッションの情報の取得.....	1046
TseGetClientId() 関数.....	1046
TseGetClientNodeName() 関数.....	1046
TseQueryRunningOnConsole() 関数.....	1046
TseQueryRunningOnClient() 関数.....	1047
リモートデスクトップ サービスの概要.....	1047
リモートデスクトップ サービスの役割サービス.....	1047
リモートデスクトップ サービス (RDS) 接続の保護.....	1048
Windows Server 2016 のリモートデスクトップ サービスのベストプラクティス.....	1049
アプリケーションの分散.....	1051
サポートされている InTouch アーキテクチャ.....	1051
単一のコンピュータのアーキテクチャ.....	1051

クライアント ベースのアーキテクチャ.....	1051
サーバー ベースのアーキテクチャ.....	1052
ネットワーク アプリケーション開発 (NAD)	1052
ネットワーク アプリケーションの計画上の考慮事項.....	1053
ネットワーク アプリケーションの I/O データ アクセス.....	1053
共有ファイルへのアクセス.....	1055
分散環境でのデータのログ.....	1057
特殊なネットワークの考慮事項.....	1061
NAD 用 InTouch アプリケーションの設定.....	1061
自動 NAD 更新の実行.....	1063
手動による NAD 更新の実行.....	1064
アプリケーション編集のロック.....	1066
NAD 更新中のアプリケーションの変更.....	1066
ランタイム時のアプリケーションの解像度のスケーリング.....	1067
アプリケーションの解像度のロック.....	1069
リモート ノードへのアプリケーションのパブリッシュ.....	1071
発行したファイルのコンテンツ.....	1072
スタンドアロン InTouch アプリケーションのパブリッシュ.....	1074
マネージド InTouch アプリケーションのパブリッシュ.....	1077
マネージド InTouch アプリケーションのパブリッシュ.....	1078
Insight へのアプリケーションのパブリッシュ.....	1079
ランタイム時のマネージド InTouch アプリケーションの使用.....	1080
マネージド InTouch アプリケーションの配置.....	1080
初めての InTouchViewApp オブジェクトの配置.....	1081
マネージド InTouch アプリケーションへの変更の配置.....	1081
マネージド InTouch アプリケーションの起動.....	1081
マネージド アプリケーションを配置するときの WindowViewer の再起動待機時間の制御.....	1082
オペレータ ノードでの新しいアプリケーションバージョンの受け入れ.....	1083
埋め込まれた産業用グラフィックでの ArchestrA スクリプトの実行.....	1085
ターミナル サービス環境での InTouchViewApp オブジェクトの配置.....	1086
Operate.....	1088
ランタイムのアプリケーションの表示.....	1089
ランタイム時のアプリケーションの表示の概要.....	1089
別のターゲット解像度サイズでのラインタイムのアプリケーションの表示.....	1089
InTouch Web Client について.....	1090
元のアプリケーション解像度.....	1090
ランタイム時のキーボード、マウス、およびタッチ ジェスチャの操作によるパンとズーム.....	1091
ランタイム時のズーム.....	1091
ランタイム時のパン.....	1093
タッチ ジェスチャのアニメーションサポート.....	1094
枠ウィンドウと ShowGraphic() 関数の使用.....	1095
インターネット上での InTouch ウィンドウの実行.....	1095
ランタイム時にウィンドウを表示する設定.....	1096
Web Client でのアプリケーション グラフィックの表示.....	1097
Web Client の使用.....	1097
InTouch Web Client アプリケーションの設計.....	1098

Web Client アクセスの保護.....	1099
Web Client でのユーザー アクセスの設定.....	1103
InTouch Web Client の仮想アカウントの使用.....	1103
Web Client 機能の有効化.....	1104
Web Client のカスタマイズ.....	1105
Web ブラウザで表示するためのグラフィックの設定.....	1106
Web Client ホームのシンボルの設定.....	1107
Web Client ホーム アイコンの動作の理解.....	1107
Web ブラウザでのアプリケーションの表示.....	1108
Web Client の高速切り替え.....	1108
Web Client ページの理解.....	1109
アプリケーション グラフィックおよびブラウザのサイズ調整.....	1112
パンとズームのサポート.....	1112
外部 Web サイトでの InTouch アプリケーション グラフィックのホスティング.....	1114
iFrame コード ブロックの生成.....	1115
サポートされているグラフィック要素と既知の制限事項.....	1116
既知の制限事項.....	1116
すべてのサポート対象グラフィック要素でサポートされるプロパティ.....	1117
すべてのグラフィック要素でサポートされるプロパティ (いくつかの制限あり)	1117
サポートされているグラフィック要素と追加のプロパティ.....	1118
サポートされるアニメーション.....	1124
Web Client モバイル アプリ.....	1126
オペレーティング システムの要件.....	1126
モバイル アプリへのログイン.....	1126
Web Client モバイル アプリの使用.....	1127
ランタイムでの言語の切り替え.....	1128
ランタイム時の言語の切り替えの概要.....	1128
ランタイム時の言語の切り替えのための言語の設定.....	1128
設定した言語のフォント設定の変更.....	1129
ランタイム時の言語の切り替え機能の追加.....	1130
SwitchDisplayLanguage() 関数.....	1132
\$Language システム タグ.....	1133
オフライン翻訳用のアプリケーション テキストのエクスポート.....	1133
既存のディクショナリ ファイルへのテキストのエクスポート.....	1135
エクスポートされたディクショナリ ファイルの翻訳.....	1135
翻訳されたディクショナリ ファイルのインポート.....	1136
アラームのランタイム時の言語切り替え.....	1137
翻訳のためのアラーム コメントのエクスポート.....	1137
既存のアラーム コメント ファイルへのエクスポート.....	1140
翻訳されたアラーム コメントのインポート.....	1142
翻訳のためのアラーム グループ名のエクスポート.....	1143
翻訳されたアラーム グループ名のインポート.....	1144
ランタイム時の言語の切り替え機能のテスト.....	1144
ネットワーク アプリケーション開発 (NAD) クライアントへのローカライズされたファイルの配布.....	1145
タグ.....	1146
Tag Viewer の概要.....	1146
Tag Viewer の有効化.....	1146

Tag Viewer の起動.....	1148
Tag Viewer でのナビゲート.....	1149
Tag Viewer を閉じる.....	1150
Tag Viewer の使用.....	1150
タグの検索.....	1150
クイック検索を行う.....	1152
タグの管理.....	1152
監視ウィンドウの管理.....	1155
タグの使用数の削減.....	1157
InTouch と Historian.....	1158
タグ変数の使用数の確認.....	1158
未使用タグの削除.....	1168
アラーム.....	1170
現在のアラームの表示.....	1170
Alarm Viewer コントロールの設定.....	1171
ランタイムでの Alarm Viewer コントロールの使用.....	1184
Alarm Viewer コントロールの ActiveX プロパティの使用.....	1187
Alarm Viewer コントロールの ActiveX メソッドの使用.....	1194
メソッドとプロパティを使用するときのエラー処理.....	1208
ActiveX イベントを使用したスクリプトのトリガ.....	1208
リアルタイムでのアラームの確認.....	1209
アラーム確認モデルの理解.....	1210
アラームを確認するためのドットフィールドの使用.....	1212
アラームを確認するためのスクリプト関数の使用.....	1217
タグ値が平常に復帰するときの自動確認の使用.....	1218
アラームを確認するためのアラーム クライアントの使用.....	1219
アラーム コメントと確認コメントの使用.....	1220
ランタイム時のアラームの解除.....	1220
ドットフィールドを使用したアラームの解除.....	1221
ランタイム時のタグとグループのアラーム プロパティの制御.....	1222
タグ変数またはアラーム グループがアラーム状況にあるかどうかの確認.....	1227
InTouch 7.1 の動作にアラーム状況を戻す.....	1237
タグ変数のアラームしきい値設定の確認.....	1237
タグ変数またはアラームグループのアラームの有効化と無効化.....	1242
タグ変数のアラームしきい値の変更.....	1256
タグ変数のアラーム デッドバンドの変更.....	1262
タグ変数に関連付けられたアラーム コメントの変更.....	1263
ユーザー定義情報とアラーム インスタンスとの関連付け.....	1264
タグ変数またはアラーム グループの抑止タグ変数の確認.....	1266
アクティブまたは未確認のアラーム数の確認.....	1272
Maintain.....	1280
アプリケーションの移行およびアップグレード.....	1281
従来のアプリケーションから新しいスタンドアロン アプリケーションへの移動.....	1281
古いアプリケーションの移行およびアップグレード.....	1281
以前の InTouch アプリケーションの現在のバージョンへの移行.....	1282
古いアラーム表示の変換.....	1283
アプリケーション設定の管理.....	1283

InTouch アプリケーションのインポート.....	1283
System Platform IDE を使用した InTouch アプリケーションの管理.....	1285
InTouchViewApp オブジェクト.....	1286
InTouchViewApp テンプレートと InTouch アプリケーションの関連付け.....	1287
マネージド InTouch アプリケーションの編集.....	1287
マネージド InTouch アプリケーションのテスト.....	1287
InTouchViewApp オブジェクトの配置.....	1287
InTouchViewApp オブジェクトのエクスポートとインポート.....	1288
InTouchViewApp オブジェクトの属性.....	1288
InTouchViewApp オブジェクトと他のオートメーション オブジェクトとの違い.....	1289
ViewEngine オブジェクト.....	1290
InTouch コンポーネントのエクスポートとインポート.....	1291
マネージド InTouch アプリケーションに関連付けられているタグ変数データのエクスポートとインポート.....	1291
タグ定義のエクスポート.....	1291
エクスポートされたタグ定義の表示.....	1292
タグ定義のインポート.....	1293
タグ名ディクショナリ インポート ファイル形式.....	1293
インポート ファイルテンプレートの作成.....	1294
ディクショナリ インポート ファイルの操作モードの設定.....	1295
アクセス名とアラーム グループの設定.....	1297
タグ変数タイプのキーワードと属性の定義.....	1302
インポート ファイルでの空白文字列の使用.....	1324
フィールドに対するデフォルト値の使用.....	1324
スーパータグ インスタンスの作成.....	1325
DBLoad を使用したタグ定義のインポート.....	1325
InTouch アプリケーション間での InTouch ウィンドウのエクスポートとインポート.....	1327
ウィンドウのインポート.....	1327
インポートされたウィンドウのプレースホルダ タグの変換.....	1329
ウィンドウのエクスポート.....	1330
スクリプトのインポート.....	1331
インポートされたスクリプトでのプレースホルダ タグの変換.....	1333
インポートされたウィンドウとスクリプト用のタグ変数プレースホルダ.....	1334
アプリケーションからの産業用グラフィックのエクスポート.....	1335
アプリケーションへの産業用グラフィックのインポート.....	1336
産業用グラフィック ツールボックスで選択したシンボルのエクスポート.....	1337
カスタム クライアント コントロールのインポートと埋め込み.....	1338
重複するクライアント コントロールをインポートする際の競合の解決.....	1338
産業用グラフィックへのクライアント コントロールの埋め込み.....	1340
HTML5 ウィジェットのインポート.....	1340
カラーセル ウィジェット.....	1341
Web ブラウザ ウィジェット.....	1342
QR コード スキャナ.....	1343
Map_App ウィジェット.....	1344
InTouch アプリケーションへのスクリプト関数ライブラリのインポート.....	1346
.NET スクリプト ライブラリ内の競合するメソッドのインポートの解決.....	1347
アプリケーションのアプリケーション スタイル ライブラリの設定.....	1348
アプリケーション スタイル ライブラリのエクスポートとインポート.....	1349

アプリケーションのアラーム優先度マッピングの設定.....	1350
アプリケーションから産業用グラフィック テキスト文字列のエクスポート.....	1351
アプリケーションへの産業用グラフィックのテキスト文字列のインポート.....	1352
シンボルからのローカリゼーション文字列のエクスポート.....	1354
産業用グラフィック ライブラリのインポート.....	1355
マルチ モニタ システムの設定.....	1356
マルチ モニタの設定.....	1356
単一のビデオ カードの設定.....	1356
複数のビデオ カードの設定.....	1357
マルチ モニタ アプリケーションの計画.....	1358
マルチ モニタ ビデオ カードの選択.....	1359
アプリケーションの画面解像度の決定.....	1359
アプリケーションを表示するモニタの数の決定.....	1359
アプリケーション ウィンドウの場所の決定.....	1360
マルチ モニタ InTouch アプリケーションの開発.....	1361
マルチ モニタ パラメータの設定.....	1361
画面解像度の変換の設定.....	1362
アプリケーションの配置およびマルチ モニタの設定の確認.....	1362
ランタイム中のマルチ モニタ サポートの確認.....	1363
タブレット PC での InTouch の使用.....	1364
ビジュアル化画面に注釈を付け、電子メール メッセージで送信する.....	1364
ウィンドウの注釈の作成.....	1365
ウィンドウの注釈の選択、コピー、および削除.....	1365
注釈の付いたウィンドウの保存、印刷、電子メールによる送信.....	1365
AnnotateLayout() 関数.....	1366
画面の向きの変更.....	1367
InTouch サービスの管理.....	1368
InTouch サービスの管理の概要.....	1368
サービスとしての WindowViewer の実行.....	1369
サービスとして起動するための WindowViewer の設定.....	1369
WindowViewer でアプリケーションをサービスとして実行するための WIN.INI の編集.....	1371
手動によるサービスの起動.....	1371
サービスの停止.....	1371
InTouch サービスのユーザー アカウントの設定.....	1372
InTouch サービスのトラブルシューティング.....	1372
サービスのエラー メッセージの表示.....	1373
サービス ユーザー アカウントの問題のトラブルシューティング.....	1373
有効 I/O アイテムの使用解除.....	1373
InTouch サービスのレジストリ キー.....	1374
アラーム.....	1375
従来のアラーム システムからの移行.....	1375
従来のアラーム システムからの移行の概要.....	1375
標準アラーム システムから分散アラーム システムへの移行.....	1375
アラーム データベースの維持.....	1376
アラーム データベースの維持の概要.....	1376
ページ設定またはアーカイブ設定.....	1377
アラーム データベースのリストア.....	1388

INTOUCH.ini ファイルからのアプリケーション設定のカスタマイズ.....	1393
カスタム INTOUCH.ini パラメータ.....	1393
カスタム ログ プロパティの設定.....	1394
WindowMaker ショートカット メニューの無効化.....	1395
カスタム WindowViewer プロパティの設定.....	1395
Diagnose.	1399
QuickScript のトラブルシューティング.....	1400
Log Viewer へのメッセージのログ記録.....	1400
LogMessage() 関数.....	1400
Log Viewer メッセージの表示.....	1401
エラー メッセージの理解.....	1402
メイン ウィンドウ.....	1402
タグ リファレンスを追加.....	1402
タグ値の変更.....	1403
監視ウィンドウの名前の変更.....	1403
Managing the InTouch Web Client.	1404
Web Client でのエラー処理.....	1404
ブラウザとモバイルのサポート.....	1404
Web ブラウザでのグラフィックの表示に関するトラブルシューティング.....	1405
無効な証明書エラー.....	1405
Manage.	1407
AVEVA Operations Control 接続エクスペリエンス.....	1408
Operations Control 接続エクスペリエンスについて.....	1408
Operations Control 接続エクスペリエンスの設定.....	1408
認証とエンタイトルメント.....	1414
AVEVA Identity Manager を使用した認証.....	1415
接続損失時の動作.....	1418
ライセンスとエンタイトルメント.....	1419
サブスクリプション情報の表示.....	1420
WindowMaker の起動と実行.....	1421
WindowViewer の起動と実行.....	1423
Operations Control 接続エクスペリエンスでのアプリケーションセキュリティの設定.....	1425
マネージドアプリケーションでのセキュリティ設定.....	1426
スタンドアロンアプリケーションでのセキュリティ設定.....	1428
混合モードでの操作.....	1429
Application Manager の起動と実行.....	1431
アプリケーション マネージャからのアプリケーションの選択と実行.....	1432
InTouch Access Anywhere の起動と実行.....	1432
ネットワーク アプリケーション開発 (NAD) アプリケーションの起動.....	1433
セキュリティ書き込みおよび確認書き込みの実行.....	1434
Web Client (オンプレミス) の起動と実行.....	1435
トレンドコントロールの起動と実行.....	1437
AVEVA アプリケーションからのサインアウト.....	1441
サポートされている InTouch HMI 機能.....	1442

サポートされていない InTouch HMI 機能.....	1444
Operation Control モード以外での InTouch の使用.....	1445
InTouch HMI でのライセンス.....	1446
ライセンス タグの数について.....	1446
InTouch HMI で使用できるライセンス.....	1448
RDS と非 RDS 環境の InTouch ライセンス.....	1449
InTouchView アプリケーション ライセンスについて.....	1449
サーバー上の InTouch ライセンスの管理に関するベスト プラクティス.....	1451
ライセンス情報の表示.....	1452
起動後における別のライセンスの使用の管理.....	1456
フリーモードとデモモードでの操作.....	1457
猶予期間での操作.....	1458
リモート タグを数える関数.....	1459
InTouch Web Client のライセンス付与.....	1464
ライセンスの取得.....	1465
ライセンス機能.....	1466
シングルセッション モード.....	1467
猶予期間.....	1468
定期的な更新.....	1468
ライセンスの解放.....	1468
補足コンポーネントについて.....	1469
レシピ マネージャの使用.....	1469
InTouch からの SQL データベースの操作.....	1491
16 ペン トレンド ウィザードの使用.....	1540
Symbol Factory.....	1554
InTouch のセキュリティ保護.....	1566
InTouch セキュリティ機能.....	1566
認証ベースおよび承認ベースのセキュリティ.....	1574
ユーザーの管理と承認レベルの設定.....	1584
ログオンとログオフ.....	1592
オペレータまたはアクセス レベルに基づく機能の有効化と無効化.....	1599
現在ログオンしているオペレータに関する情報の取得.....	1601
セキュリティ システム タグと関数の概要.....	1605
管理者以外のユーザーに許可されたアプリケーション マネージャの操作.....	1606
アプリケーション フォルダへの役割ベースのセキュリティの適用.....	1608
マネージド InTouch アプリケーションおよび NAD InTouch アプリケーションのローカル作業ディレクトリの完全性チェック.....	1610
InTouch HMI のセキュリティの管理.....	1612
セキュリティに関する全体的な考察.....	1612
はじめに.....	1612
ホストのセキュリティ.....	1613
ホストのセキュリティに関する一般的なガイドライン.....	1613
Windows の更新.....	1614
ICS ソフトウェアの更新.....	1615
Scanning the Host.....	1615
ホスト上のアプリケーションおよびコンテンツの保護.....	1616
SQL Server での暗号化の設定.....	1617
ネットワークの保護.....	1620

ICS ネットワークのセグメント分割.....	1620
ネットワーク サービスおよびポートの管理.....	1621
クライアントとホストの間の通信のセキュリティ.....	1622
クラウド ベースのシステム.....	1623
認証および承認によるシステムのセキュリティ.....	1623
Windows でのユーザーとグループの管理.....	1624
ICS ソフトウェアでのユーザーおよびグループの管理.....	1625
緊急時対応計画.....	1625
監査とログ.....	1626
事業継続計画.....	1626
障害復旧計画.....	1626
まとめ.....	1627
InTouch HMI のセキュリティ設定.....	1627

Standard

章 1 統合開発環境

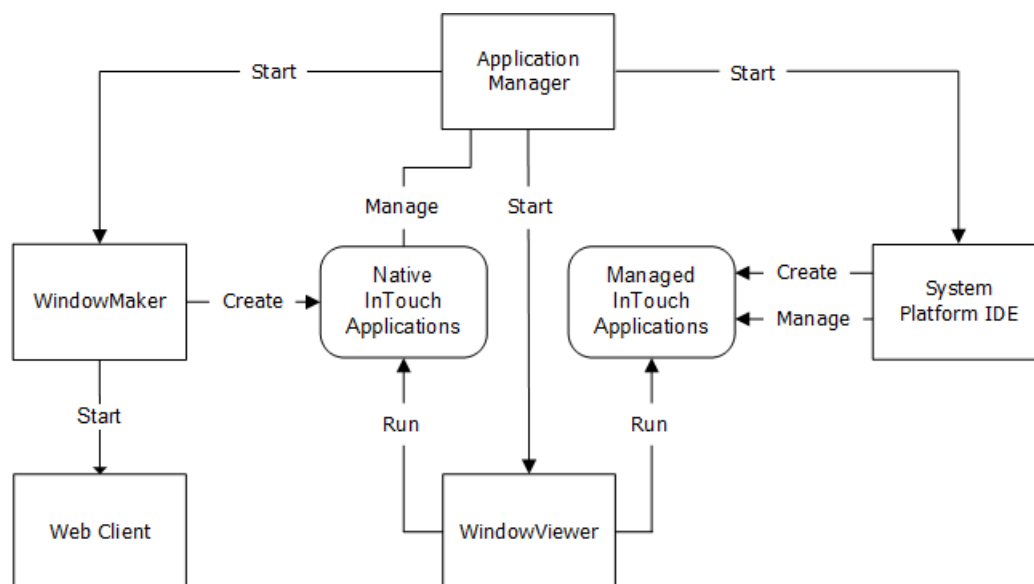
WindowMaker は、InTouch HMI アプリケーションを作成するために使用する開発環境です。製造プロセスを表示および管理するためにオペレータが使用するビジュアルインターフェイスを作成するには、WindowMaker を使用します。InTouch HMI のインターフェイスでは、運用環境のデータへの表示、および運用環境へのデータ書き込みが行われます。

WindowMaker を使用すると、InTouch アプリケーションの以下のビジュアルインターフェイス要素を設定できます。

- ウィンドウには、工場のオペレータが製造プロセスを管理するための対話型の視覚要素が含まれています。
- 基本的なオブジェクトとは、長方形、円、線、およびテキストなどのシンプルなグラフィック要素です。
- ユーザー定義の複合オブジェクトは、バルブおよびタンクなどの、運用環境の要素を表す 1 つまたは複数の基本的なオブジェクトで構成されます。
- 事前定義済み複合オブジェクトを使用すると、アラーム、履歴トレンド、シンボル ウィザードなどの特定の機能を実行できます。
- アニメーションリンクは、外観をアニメーション化しユーザーの入力をタスクおよび製造データの変更に送信するための、シンプルオブジェクトおよび複合オブジェクトのプロパティです。
- ウィザードは、特定の関数を実行する、またはスライダーやメーターなどの特定の外観を持つ定義済み複合オブジェクトです。
- ActiveX コントロールとは、現在のアラームを表示するなどの特定の関数を実行するために InTouch ウィンドウに配置できるコントロールです。

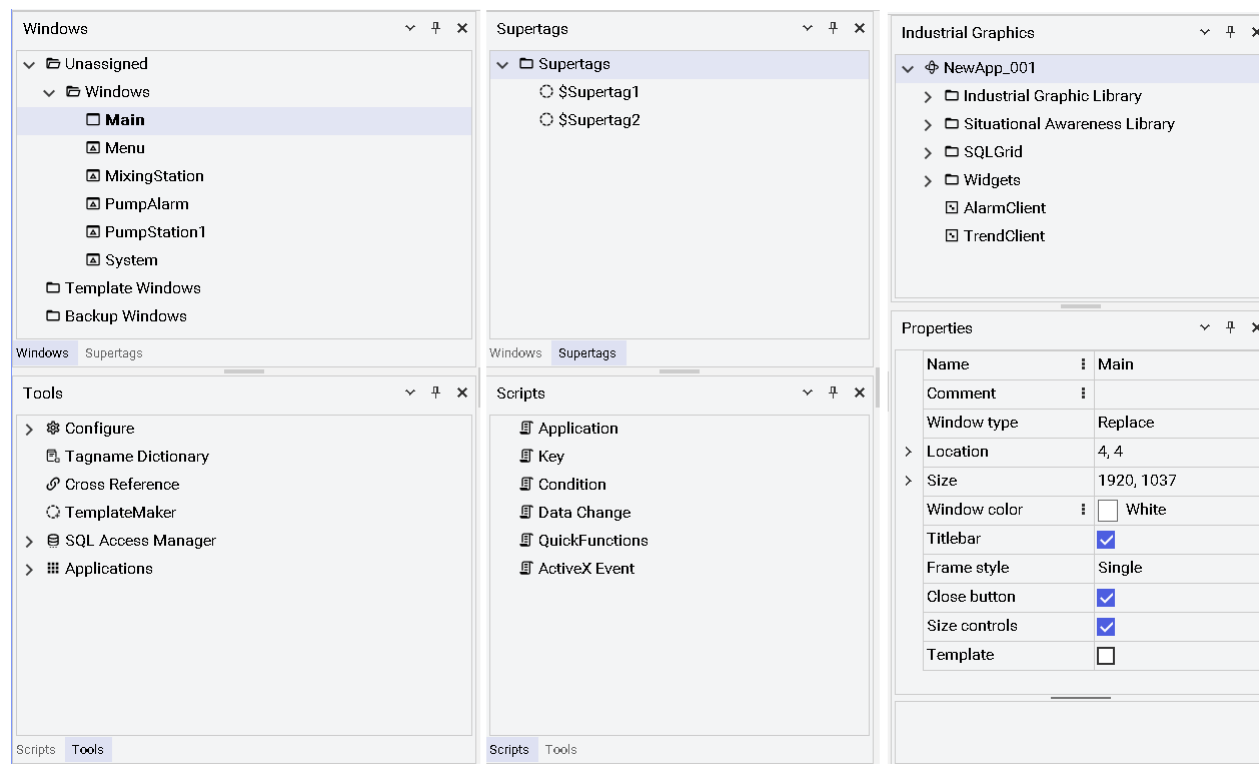
InTouch アプリケーション マネージャについて

InTouch アプリケーションの作成、削除、および変更などの大部分のグローバルタスクを管理するには、InTouch アプリケーション マネージャを使用します。アプリケーション マネージャには、現在の InTouch アプリケーションのリストが表示されます。WindowMaker または WindowViewer を開くには、リストからアプリケーションを選択します。



アプリケーション エクスプローラの使用

アプリケーション エクスプローラには、ウィンドウ/スーパータグ、スクリプト/ツール、および産業用グラフィック ツールボックスのタブ形式セクションがあります。他のツールバーと同様に、開くこと、閉じること、ピン留めすること、およびドッキングすることができます。デフォルトでは、各ペインは以下の図に示す場所に表示されます。



これらのビューからは、すべてのアプリケーション ウィンドウ、スクリプト、設定メニュー、タグ名ディクショナリ、およびウィザードにアクセスできます。

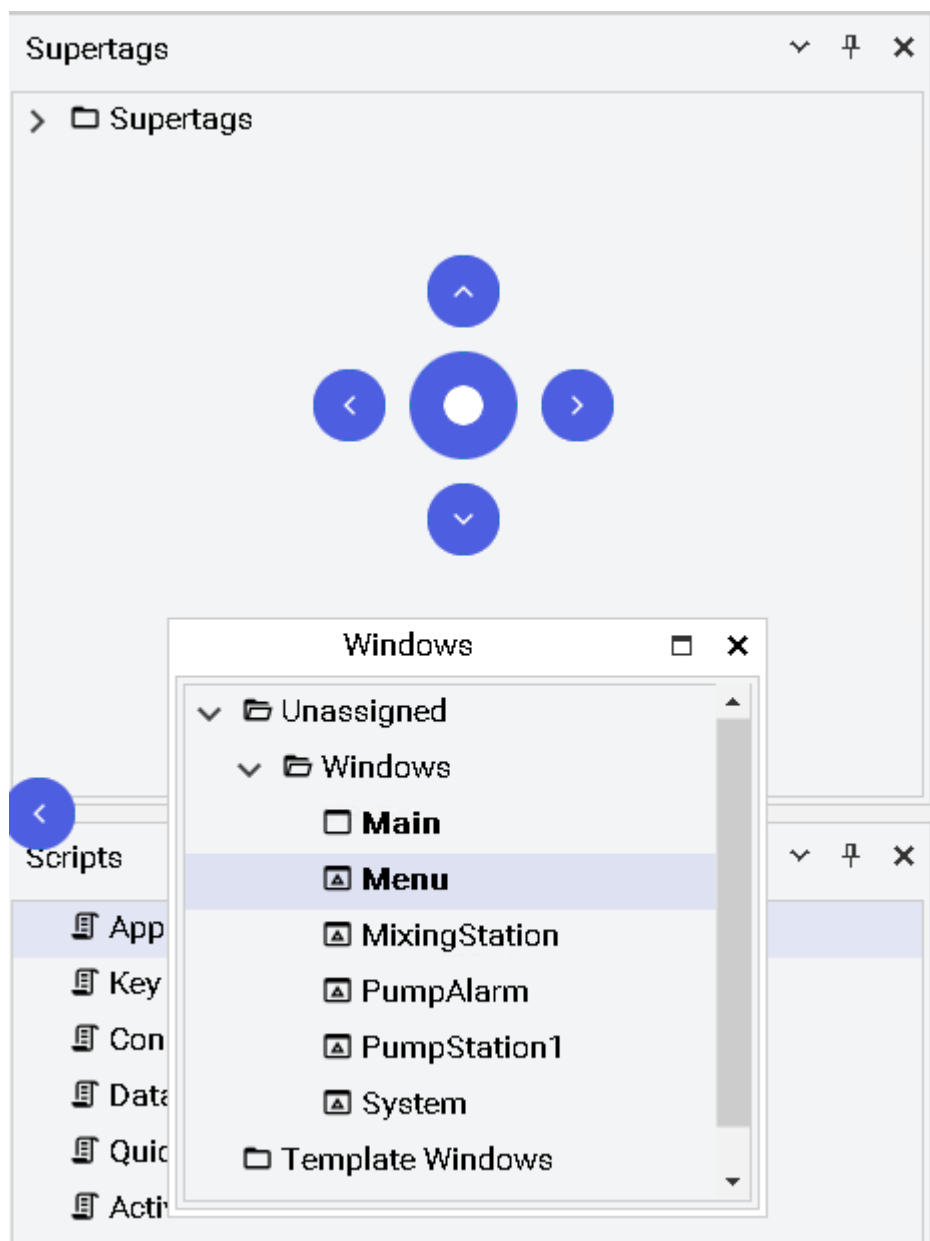
ペインのドッキング

これらの各ペインは移動して、画面の上下左右の位置にドッキングすることができます。ペインをドッキングするには、ペインの見出しをクリックします。カーソルが二股の矢印に変わります。ペインをドラッグして画面上の目的の場所にドッキングします。

ナビゲーション エイドを使用してフローティング ペインをドッキングすることもできます。画面の特定の場所で以下の操作を実行します。

- 上向きの矢印をクリックすると、画面の上部にペインがドッキングします。
- 下向きの矢印をクリックすると、画面の下部にペインがドッキングします。
- 右向きの矢印をクリックすると、画面の右側にペインがドッキングします。
- 左向きの矢印をクリックすると、画面の左側にペインがドッキングします。

ペインは、垂直および水平方向にドッキングできます。



ペインの位置を変更した場合、WindowMaker を閉じてから再度開いた後でもペインのカスタム設定は維持されます。

ペインの位置変更とドッキングに関する情報は DockSettings.xml に保存されます。ドッキング設定を変更した後にナビゲーションの問題が発生した場合は、以下の手順を実行します。

1. WindowMaker を閉じます。
2. C:\Users\<ユーザー名>\AppData\Local\Wonderware に移動します。
3. DockSettings.xml ファイルを見つけて削除します。
4. WindowMaker を再度開きます。

アプリケーション エクスプローラでのナビゲート

いずれかのアプリケーション エクスプローラ ツールバーにあるフォルダは、表示または非表示にすることができます。

[アプリケーション] ビューには、インストールされているその他のアプリケーションが表示されます。

アプリケーション エクスプローラ フォルダを表示または非表示にするには

1. フォルダまたはアイコンをダブルクリックして、グループ メンバーを表示します。
2. メンバーをダブルクリックして、そのメンバーを開きます。

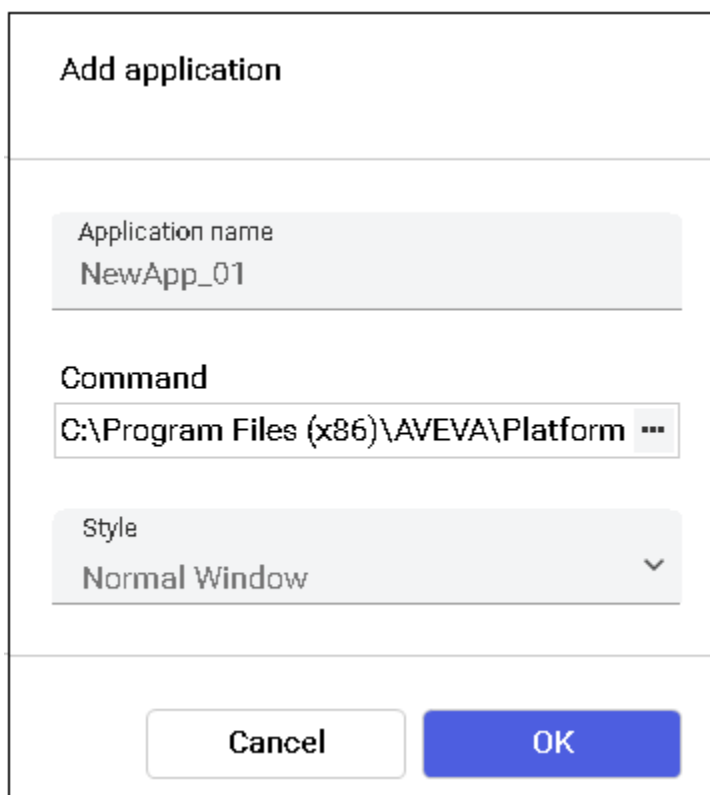
アプリケーション エクスプローラへのアプリケーションの追加

WindowMaker アプリケーション エクスプローラは、WindowMaker 内から他のアプリケーションを起動できます。たとえば、アプリケーションの開発と同時に I/O サーバーを実行して、設定することができます。Windows のメモ帳、ワードパッド、Microsoft Excel、Microsoft Word、Microsoft ペイントなどのサードパーティ プログラムを起動できます。

アプリケーション エクスプローラは、文書やスプレッドシートなどの特定のファイルを開くように設定することもできます。

新しいアプリケーションをアプリケーション エクスプローラに追加するには

1. [ツール] ペインで、[アプリケーション] を右クリックし、[新規] をクリックします。
[アプリケーションの追加] ウィンドウが開きます。



The screenshot shows a dialog box titled "Add application". It contains three input fields: "Application name" with the text "NewApp_01", "Command" with the text "C:\Program Files (x86)\AVEVA\Platform ...", and "Style" with a dropdown menu showing "Normal Window". At the bottom are "Cancel" and "OK" buttons.

2. [アプリケーション名] ボックスに、アプリケーションの名前を入力します。

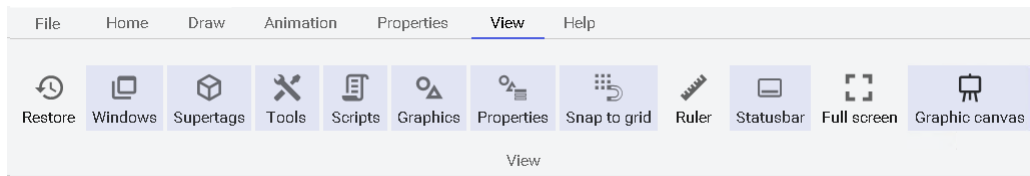
3. [コマンド] ボックスにアプリケーションのフルパスを入力します。アプリケーションを参照するには [...] ボタンをクリックします。
[コマンドライン] ボックスには、アプリケーションに対するコマンドラインパラメータを追加できます。
4. [スタイル] リストから、アプリケーションの起動時の表示方法を選択します。使用可能なオプションは、[狭いウィンドウ]、[最小化]、および [最大化] です。
5. [OK] をクリックします。

[アプリケーション] の下の [アプリケーションエクスプローラ] にアプリケーションが追加されます。これで、アプリケーションを WindowMaker から直接起動できるようになりました。

注記: WindowViewer (view.exe) はアプリケーションエクスプローラに追加しないでください。WindowViewer を正しく起動させるには、[ファイル] メニューの [WindowViewer] をクリックするか、ツールバーの [実行] 高速切り替えをクリックします。

ツールバーの管理

WindowMaker のツールバーを使用すると、頻繁に使用するコマンドにすばやくアクセスできます。WindowMaker を起動すると、デフォルトですべてのツールバーが表示されます。ツールバーの管理、およびツールバーの表示と非表示の切り替えは、[表示] メニューを使用して行うことができます。



ツールバーをドラッグして、フローティングさせたりドッキングさせたりすることができます。非表示されていたドッキングツールバーを表示すると、ツールバーはウィンドウで最後にドッキングされていた位置で再び表示されます。ツールバーはデフォルトの位置から開発ウィンドウ内の任意の位置に移動できます。フローティングツールバーのタイトルバーを使用すると、サイズを変更できます。

ツールバーを表示または非表示にするには

- [表示] メニューで、ツールバー アイコンをクリックします。

フローティング ツールバーのサイズを変更するには

1. カーソルをツールバーの端に移動します。カーソルが両向き矢印に変わります。
2. ツールバーの端をドラッグして、ツールバーを移動したり、サイズ変更したりします。

カーソルを移動すると、マウス ボタンを離した際のツールバーのサイズを示すボックスが表示されます。

WindowMaker 環境設定

WindowMaker の設定画面で、WindowMaker の動作に影響を与える環境設定およびオプションを設定できます。以下の操作を実行できます。

- タイトルバー テキストを変更する
- グリッドを表示または非表示にする

- グリッドのピクセル間のスペースを変更する
- タグの数を表示する
- テキストおよびボタンのデフォルト フォントを変更する
- マウス カーソルの反応範囲を設定する
- WindowViewer に切り替えるときに WindowMaker を閉じるオプションを設定する
- 枠線だけのシンボルやセルの中のオブジェクトを選択する
- WindowMaker から WindowViewer への高速切り替えを有効にする
- 元に戻るレベルの数を設定する

WindowMaker のプロパティを設定するには

1. [ファイル] メニューの [設定] をクリックし、[WindowMaker] をクリックします。
WindowMaker の設定画面が表示されます。

WindowMaker

Configuration of editing environment

Title bar text
InTouch - WindowMaker

Supertag template path: C:\ProgramData\InTouchDem...

☒ Store supertags data in application directory

☒ Show tag count ☒ Lock window size

☒ Pick through hollow objects ☒ Enable fast switch

☒ Close on transfer to WindowViewer ☒ Show grid

☒ Disable script autocomplete

Line selection precision: 4 Pixels

Level of undo: 10

Grid spacing: 10 Pixels

Text font: Arial Size: 12

Button font: Tahoma Size: 10

Script editor font: Courier New Size: 9

2. 環境編集領域の WindowMaker の設定で、タイトルバーの外観を設定します。以下のいずれかを実行します。
- [タイトルバーのテキスト] ボックスに、デザイン時にタイトルバーに表示するテキストを入力します。
 - [スーパータグ テンプレート パス] で、スーパータグ テンプレートを保存するパスを指定します。

3. スーパータグデータをアプリケーションディレクトリに保存するには、[アプリケーションディレクトリにスーパータグデータを保存] チェックボックスをオンにします。このオプションを選択すると、[スーパータグテンプレートパス] フィールドで指定したパスが上書きされます。
4. その他のウィンドウプロパティを設定します。以下のいずれかを実行します。

- タグ名ディクショナリのタグ名の数をメニューバーに表示するには、[タグの数の表示] チェックボックスをオンにします。大量のタグがある場合にタグの数を表示すると、タグ名ディクショナリのパフォーマンスに影響が生じることがあります。

これは、制限されたタグ名ディクショナリサイズでアプリケーションを作成する場合に役立ちます。タグ名の数には、リモートタグ名参照またはシステムタグは含まれていません。リモートタグ名参照の使用を確認するには、[ホーム] タブの [使用カウントを更新] をクリックします。

- InTouch アプリケーションウィンドウのサイズをロックするには、[ウィンドウサイズをロック] チェックボックスをオンにします。

このオプションを選択すると、ウィンドウとグラフィックのスケールなしでアプリケーションを別の解像度に変換できます。

- [枠線だけのシンボルやセルの中のオブジェクトを選択] チェックボックスをオンにすると、枠線だけのシンボルやセルの背面にあるオブジェクトを選択できます。

これにより、フレームを背景に移動する必要なく、フレーム内のオブジェクトを選択することができます。

- WindowMaker と WindowViewer を切り替える「高速切り替え」を使用するには、[高速切り替えを有効にする] チェックボックスをオンにします。

高速切り替えは、WindowMaker の右上隅に表示される [実行] という単語です。WindowViewer では、[開発] が表示されます。

高速切り替えを有効にすると、WindowViewer に切り替える前に WindowMaker で開いているすべてのウィンドウに対する変更が自動的に保存されます。

- WindowViewer の起動時に WindowMaker を自動的に閉じるには、[WindowViewer への移行時に自動的に終了] チェックボックスをオンにします。

このオプションの目的は、制限されたメモリを節約することです。メモリが問題ではなく、WindowViewer と WindowMaker を頻繁に切り替える場合は、このオプションを選択しないでください。

[WindowViewer への移行時に自動的に終了] をオンにすると、[WindowViewer のプロパティ] 画面の [全般] タブの対応コマンドである [WindowViewer を自動的に終了する] もオンになります。

- グリッドを表示するには、[グリッドの表示] チェックボックスをオンにします。
- オートコンプリートの推奨リストボックスの表示を無効にするには、[スクリプトのオートコンプリートを無効化] を選択します。
- [マウスカーソルの反応範囲] ボックスに、カーソルがラインから離れていても選択できる範囲をピクセル単位で入力します。

ほとんどの場合、デフォルト設定の 4 が適当です。

- **【元に戻すレベル】** ボックスに、**【元に戻す】** および **【やり直す】** コマンドで維持するレベルの数を入力します。

最大レベルは 25 です。0 と入力すると、「元に戻す/やり直す」機能がオフになります。

1 つのレベルは 1 回のアクションを表しています。新しいウィンドウを作成するか、既存のウィンドウを開くと、**【元に戻す】** および **【やり直す】** のスタックは空になります。ウィンドウを閉じると、両方のスタックが空になります。

- **【グリッド間隔】** ボックスに、グリッド間隔をピクセル数で入力します。
5. テキスト、ボタン、およびスクリプトエディタのフォントプロパティも **【テキストフォント】**、**【ボタンフォント】**、および **【スクリプトエディタ フォント】** フィールドを使用して設定できます。
 6. 任意のウィンドウでのこれらのデフォルト値は、該当する画面の **【フォント】** 設定を使用して上書きできます。
 7. **【OK】** をクリックします。
 8. 変更内容を適用するには、**WindowMaker** を再起動します。

マウス ショートカット

以下のショートカットを使用して、ダイアログ ボックスを開き、その他の一般タスクを実行します。

WindowMaker のアイテム用のメニュー コマンドにアクセスするには

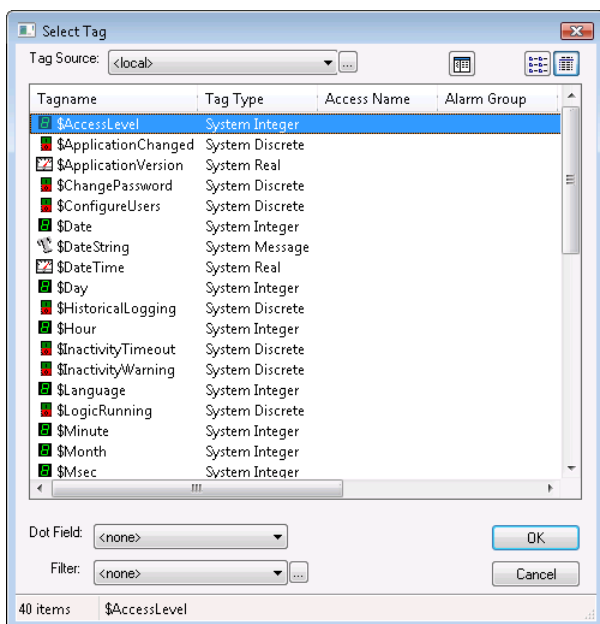
- アイテムを右クリックします。アイテムには、グラフィック オブジェクトやビュー内のフォルダ名などが含まれます。

【アニメーション リンク】 ダイアログ ボックスを開くには

- オブジェクトまたはシンボルをダブルクリックします。

【タグ変数を選択してください】 ダイアログ ボックス（タグ ブラウザ）を開くには

- リンク定義ダイアログ ボックス内の空白の式入力フィールドをダブルクリックします。**【タグ変数を選択してください】** ダイアログ ボックスが表示されます。



タグ変数ドットフィールドにアクセスするには

- [タグ変数] または [タグ変数/式] 入力ボックスに、タグ変数の後にピリオドを入力してから、ピリオドの右側をダブルクリックします。ピリオドのみを入力してその右側をダブルクリックすることもできます。[フィールド名の選択] ダイアログボックスが開き、すべてのタグ変数ドットフィールドが表示されます。

タグ変数ディクショナリでタグ変数定義を開くには

- タグ変数をダブルクリックします。

全画面モードの使用

全画面モードでは、開いているウィンドウとフローティング ツールバーを除くすべてのプログラム要素が非表示となります。

全画面モードのオン/オフを切り替えるには

- [表示] ツールバーで、[全画面] をクリックして、標準モードから全画面モードに切り替えます。
[表示] ツールバーが自動的に [元のサイズに戻す] ツールバーに変化して最上部にフロートされます。

[ファイル] メニューからのウィンドウの管理

[ファイル] メニューを使用して、ウィンドウを開く、保存、閉じる、または削除などの操作を行う場合、選択したコマンドを使用できるすべてのウィンドウの名前が一覧表示されます。

ウィンドウ リストの表示

ウィンドウは、アイコン ビュー、一覧ビュー、または詳細ビューで表示できます。

- アイコン ビュー: [アイコン] をクリックすると、ウィンドウのリストがグリッドに配置されます。
- リスト ビュー: [リスト] をクリックすると、ウィンドウが複数列のリストに配置されます。

- 詳細ビュー: [詳細] をクリックすると、ウィンドウの詳細を含むウィンドウのリストが表示されます。説明は、ウィンドウのプロパティ パネルの [コメント] フィールドから取得されます。

ウィンドウの選択

- ウィンドウを選択するには、ウィンドウ アイコンまたはエントリをクリックします。選択したウィンドウが強調表示されます。
- 複数のウィンドウを選択するには、目的のウィンドウ アイコンまたはエントリをクリックします。選択したウィンドウが強調表示されます。
- すべてのウィンドウを選択するには、[すべて選択] チェック ボックスをオンにします。

ウィンドウの選択解除

- 選択したウィンドウの選択を解除するには、ウィンドウ アイコンまたはエントリを再度クリックします。
- すべてのウィンドウの選択を解除するには、[すべて選択] チェック ボックスをクリックしてクリアします。

ウィンドウの検索

- リストからウィンドウを検索するには、検索バーを使用します。

フォントのデフォルト設定

テキスト オブジェクトおよびテキストを持つボタン オブジェクトのデフォルト フォントを設定できます。これらのデフォルト設定は、フォント セクションを使用してウィンドウまたはボタン テキストをカスタマイズすることによって上書きできます。

フォントのデフォルトを設定するには

1. [ファイル] メニューの [設定] をクリックし、[WindowMaker] をクリックします。
WindowMaker の設定画面が表示されます。
2. テキスト、ボタン、およびスクリプト エディタのフォント プロパティは、[テキスト フォント]、[ボタン フォント]、および [スクリプト エディタ フォント] フィールドを使用して設定できます。
これらのデフォルト設定は [フォント] セクションを使用して上書きできます。
3. [OK] をクリックします。

矢印キーを使用したオブジェクトの移動

WindowMaker で、矢印キーを使用して、選択したオブジェクトまたはオブジェクトのグループを移動できます。

矢印キーを使用してオブジェクトを移動する場合、オブジェクトの移動距離はグリッドが表示されているかどうかによって異なります。

グリッドが表示されている場合、オブジェクトの移動ピクセル数は WindowMaker の設定画面で設定したグリッド間隔に依存します。デフォルト設定のグリッド間隔は 10 ピクセルです。

グリッドが表示されている場合：

- 矢印キーを押すと、オブジェクトが 1 グリッド ポイント移動します。
- Shift キーと矢印キーを押すと、オブジェクトが 2 グリッド ポイント移動します。

- Ctrl キーと矢印キーを押すと、オブジェクトが 4 グリッドポイント移動します。

グリッドが表示されていない場合：

- 矢印キーを押すと、オブジェクトが 1 ピクセル移動します。
- Shift キーと矢印キーを押すと、オブジェクトが 10 ピクセル移動します。
- Ctrl キーと矢印キーを押すと、オブジェクトが 50 ピクセル移動します。

カラーパレットの使用

カラーパレットを使用すると、線、長方形、角丸長方形、楕円形、折れ線、多角形、およびテキストの静的プロパティおよび動的プロパティに対して、色を適用できます。ウィンドウの背景色およびビットマップの透明色を選択できるため、ビットマップの背後にあるオブジェクトを表示できます。

パレットには、最大 1670 万色という幅広い選択範囲の色が提供されています。使用できる色の数は、ご使用のビデオカードの性能によって制限される可能性があります。

カスタムカラーを定義して追加することもできます。

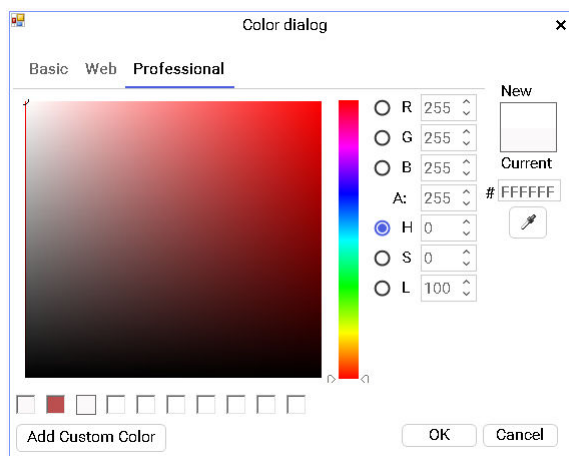
カラーパレットを開く

カラーパレットは、[プロパティ] パネルにある色付きの四角形をクリックするか、選択したオブジェクトに線、塗りつぶし色、またはテキスト色を適用する色ツールをクリックすると表示されます。

カラーパレットを開くには

1. 設定画面の [色] フィールドの横にある色付きの四角形をクリックします。

[色] ダイアログが表示されます。



1. 他のクラシック カラーパレットにアクセスするには、[基本] および [Web] タブをクリックします。
2. 以下のいずれかを実行します。
 - カラー表示の任意の場所をクリックして、スライダを使用してカラーを調整します。白と黒を含まない 100% の色を指定するには、ALT + O を押します。

- [赤]、[緑]、および [青] のボックスに値を入力して色を定義します。これらの値はカラー マトリックスを確認しながら調整できます。輝度、彩度、および明度の値も変わることにご注意してください。
- [輝度]、[彩度]、および [明度] のボックスに値を入力して、色を定義します。これらの値を変更すると、赤、緑、青のスケールがそれに応じて変化します。

輝度は論理色の値です。0 は赤、60 は黄、120 は緑、180 はシアン、200 はマゼンタ、240 は青です。

彩度は特定の輝度での色の度合で、最大 240 です。

明度は色の明るさです。

3. [OK] をクリックします。

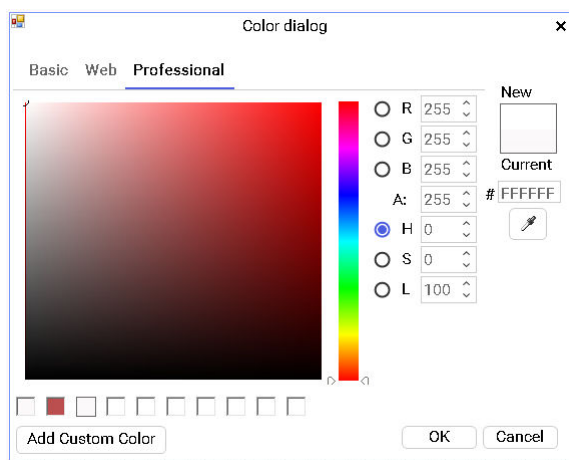
[色] ダイアログが閉じて、選択した色が適用されます。

カスタム カラーの作成

カスタム カラーのパレットを作成することができます。

カスタム カラーを作成するには

1. カラー パレットを開きます。
2. [カスタム パレット] 領域で色を選択し、[カスタム カラーを追加] をクリックします。



3. [色 | 純色] ボックスで表示される色を確認します。

使用しているモニタが 256 色表示に設定されている場合は、[色 | 純色] ボックスには 2 つの色が表示されます。右側には選択した色が純色としてどのように表示されるかを示します。左側にはディザ色または 256 色の 2 色を使用した特定の色の類似色が表示されます。

4. [OK] をクリックします。

スポイト ツールを使用してカスタム カラーを作成することもできます。

注記: この機能は、透過ビットマップを作成するときに使用します。

スポイト ツールを使用してカスタム カラーを選択するには

1. カラー パレットを開きます。

2. スポイト ツールをクリックして、追加する色をクリックします。

カスタム カラーのインポートとエクスポート

カスタム カラー パレットを定義した場合、InTouch アプリケーションからエクスポートして別の InTouch アプリケーションにインポートできます。

カスタム パレットをインポートするには

1. カラー パレットを開きます。
2. カスタム パレットの下向き矢印をクリックします。
3. [パレットの読み込み] をクリックします。Windows 標準の [ファイルを開く] ダイアログ ボックスが表示されます。
4. 希望の色定義を持つ .pal パレット ファイルを探し、選択します。
5. [開く] をクリックします。パレット ファイルに含まれている色が **カスタム パレット** にロードされます。

カスタム パレットをエクスポートするには

1. カラー パレットを開きます。
2. カスタム パレットの下向き矢印をクリックします。
3. [パレットのエクスポート] をクリックします。Windows 標準の [名前を付けて保存] ダイアログ ボックスが表示されます。
4. パレット ファイルの名前を指定して、[保存] をクリックします。

パンとズーム

編集している要素をズーム インまたはズーム アウトして表示を調節し、オブジェクトが正確に整列しているか、または正しく配置されているかを確認することができます。

[パンおよびズーム] ツールバーがデフォルトで画面の右下に表示されます。他のツールバーと同じ様に、他の場所にフローティングさせたりドッキングさせたりすることができます。

画面解像度と異なるターゲット解像度を指定した場合、境界キャンバスは、パンとズームに応じて調整されます。

以下の操作を実行できます。

- 100% ～ 500% のズーム インおよびズーム アウト
- ラバー バンド ツールを使用した特定の領域へのズーム
- 特定のパーセンテージでウィンドウをズームする
- クリックしてドラッグし、ウィンドウをパンする
- 標準デフォルト ビューに戻す

[パンおよびズーム] ツールバーを表示または非表示にするには

- [表示] メニューで、[ステータス バー] をクリックします。

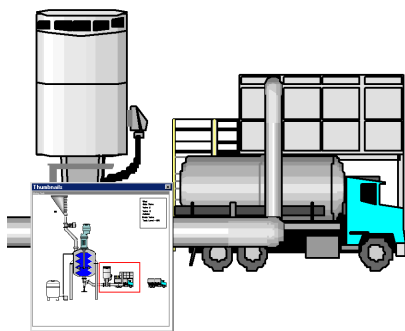
サムネイル ウィンドウを使用したパンとズーム

アプリケーション ウィンドウの詳細にズーム インすると、サムネイル ウィンドウにアプリケーション 全体に対する詳細の関係が表示されます。

[サムネイル] ウィンドウには開発領域の概要と詳細の両方が表示されます。

[サムネイル] ウィンドウを表示または非表示にするには、[サムネイル] ボタンをクリックします。

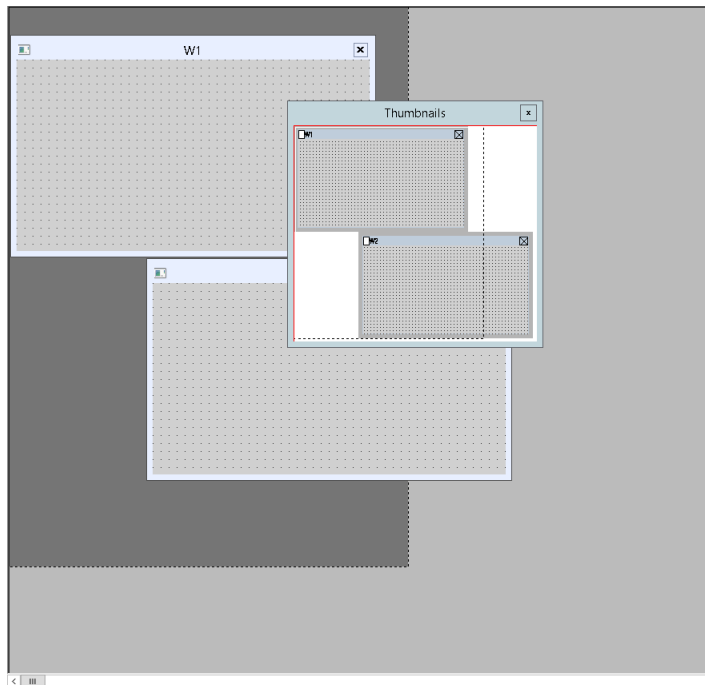
赤い長方形がウィンドウ内のズームされた領域の境界です。



- 赤い長方形をドラッグすると、ウィンドウの別の部分が表示されます。
- ウィンドウの別の領域をクリックすると、長方形がその領域に移動します。
- 長方形のサイズを変更すると、表示領域のズーム レベルが変わります。

サムネイル ビューでは、**ActiveX** コントロールなど、ズームできないオブジェクトには白い長方形が表示されます。

画面解像度と異なるアプリケーション ターゲット解像度を指定した場合、指定したターゲット解像度の寸法を示す境界キャンバスもサムネイル ビューに表示されます。



注: ターゲット解像度の境界の寸法を超えるアプリケーション ウィンドウは、上の例に示すようにサムネイル ビューに引き続き表示されます。

マウス ホイールを使用したズームとパン

マウスにホイールが付いている場合は、Ctrl キーを押しながらホイールを回転させると画像のズーム レベルを変更できます。

- マウス ホイールを回転させると、クリックの度にズーム レベルが **20%** ずつ変化します。
- カーソルを InTouch ウィンドウに置いてマウス ホイールを押すことによって、ウィンドウ内をナビゲートすることもできます。マウス ホイールを押すと、**4** つの矢印が付いたアイコンが表示されます。ウィンドウをナビゲートするには、マウスを移動します。

パンとズームの制限

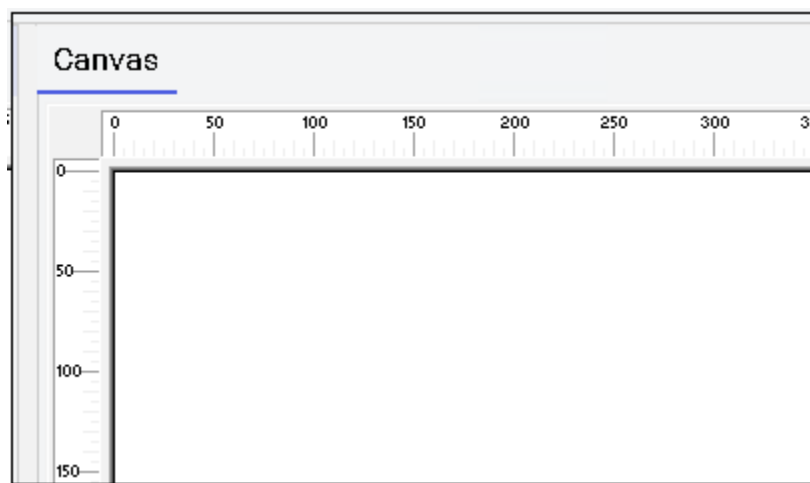
パンとズームは以下のコントロールには適用されません。

- ActiveX コントロール
- 分散アラーム オブジェクト
- 16 ペン トレンド
- テキスト ボックス
- チェック ボックス
- リスト ボックス
- コンボ ボックス
- ラジオ グループ オブジェクト

ビューが **100%** を超えるズームになっているときに、これらのコントロールの **1** つが表示領域にある場合は、コントロールの名前が付いた長方形がコントロールが使われている領域に表示されます。

画面グリッドとルーラーの使用

開発領域のグリッドおよびルーラーを表示して、オブジェクトの配置および整列に使用できます。



グリッドへのオブジェクトの配置

オブジェクトを定義済みグリッドポイントに配置するには、**[グリッドに配置]** を選択します。

WindowMaker を起動したときデフォルトでは、グリッドは 10 ピクセルで表示されるように設定されています。グリッドのピクセル間隔は、WindowMaker の設定画面で設定します。

グリッドを表示するには、WindowMaker の設定画面の **[グリッドの表示]** を選択して、**[表示]** メニューの **[グリッドに配置]** を選択します。

グリッドを設定するには

1. **[ファイル]** メニューの **[設定]** をクリックし、**[WindowMaker]** をクリックします。WindowMaker の設定画面が表示されます。
2. **[グリッド幅]** ボックスで、グリッド座標間の間隔をピクセル数で入力します。
3. **[グリッドに配置]** が選択されているときにグリッドを表示する場合は、**[グリッドの表示]** チェック ボックスをオンにします。

[グリッドの表示] をオンにしないと、**[グリッドに配置]** を選択してもウィンドウにグリッドは表示されません。

ルーラーの使用

ルーラーを使用すると、ウィンドウでオブジェクトを正確に整列させることができます。ルーラーは開発環境ウィンドウの最上部から一方のサイドに沿って表示されます。

ルーラーを表示または非表示にするには

1. **[表示]** メニューの **[表示]** グループで **[ルーラー]** をクリックします。
2. ルーラーを非表示にするには、手順を繰り返します。

WindowViewer について

WindowViewer は、InTouch アプリケーションのランタイム環境を提供します。アプリケーションの動作要件に基づいて、WindowViewer がアプリケーションをサポートする方法を設定できます。たとえば、アプリケーションのセキュリティ要件に応じて、WindowViewer でオペレータが使用できるメニューおよびコマンドを設定できます。

ランタイム環境のカスタマイズ

プロパティを設定して、ランタイムの WindowViewer 環境をカスタマイズすることができます。

- 一般プロパティにより、InTouch アプリケーションを実行するための環境条件が決まります。
- ウィンドウ設定プロパティにより、WindowViewer で実行されている InTouch アプリケーションとユーザーがデータをやり取りする際にアクセスできるメニュー、コマンド、およびウィンドウ コンポーネントが決まります。

一般的な WindowViewer プロパティの設定

InTouch アプリケーションを実行するときの WindowViewer の特性を決定する一連の一般プロパティを設定できます。これらの一般プロパティを変更した後に変更を有効にするには、WindowViewer を再起動する必要があります。

注記: オペレーティングシステムの地域設定を変更する前に WindowViewer を閉じてください。

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[WindowViewer] をクリックします。
WindowViewer の設定画面が表示されます。

WindowViewer

Preferences

Application type

Window

Memory

Startup

Advanced format

WindowViewer startup

☐ Startup as icon

☒ Enable tag viewer

Minimum access level: 9999

Inactivity in secs

Warning: 0

Timeout: 0

Blink frequency in msec

Slow: 1000

Medium: 500

Fast: 250

I/O

Retry initiates: 0 secs

☒ Start local servers

☐ Reinitialize default

Transfer to WindowMaker

☒ Close WindowViewer

☒ Close all open windows

Time/Timer control in msec

Tick interval: 100

Update for time variables: 1000

Miscellaneous

☐ Beep when object touched

☐ Debug scripts

☐ Update all trends "fast"

☐ Use old sendkeys

Hotlinks

☐ Show halo around hotlink

☐ Show halo around ActiveX control

☐ Halo follows object shape

Keyboard

☐ Allow decimal notation

☒ InTouch keyboard

☐ Windows keyboard

☐ Resizeable keyboard

Font: Microsoft Sans Serif

Size: 8

Alpha numeric keyboard

X Location: 0

Width: 5568

Y Location: 6016

Height: 0

Numeric keyboard

X Location: 5568

Width: 0

Y Location: 5568

Height: 5568

Cancel

Save

3. [環境設定] タブの [WindowViewer 起動時の設定] セクションで、以下の手順を実行します。

- WindowViewer を最小化した状態で起動するには、**[起動時アイコンにする]** チェック ボックスをオンにします。
 - Tag Viewer がランタイム時に起動するようにするには、**[Tag Viewer の有効化]** チェック ボックスをオンにします。Tag Viewer では、ランタイム時にタグの監視やタグ値の変更を行うことができます。詳細については、『AVEVA™ InTouch HMI アプリケーション ランタイム ガイド』の「[Tag Viewer の有効化](#)」を参照してください。
 - **[最小アクセス レベル]** ボックスに、Tag Viewer アプリケーションを実行するために必要な最小のセキュリティ アクセス レベルを入力します。値は 0 から 9999 の間の数値である必要があります。詳細については、『AVEVA™ InTouch HMI アプリケーション ランタイム ガイド』の「[Tag Viewer の有効化](#)」を参照してください。
4. **[無操作時間の設定 (秒)]** セクションで、オペレータの無操作時間に対する警告およびタイムアウト期間を設定します。
- 警告の設定およびタイムアウト時間の詳細については、『AVEVA™ InTouch HMI 管理ガイド』の「[無操作タイムアウトの設定](#)」を参照してください。
5. **[点滅速度の設定 (ミリ秒)]** セクションに、**[低速]**、**[中速]**、および **[高速]** の点滅アニメーションリンクの間隔の長さをミリ秒で入力します。
6. **[I/O]** 領域で、以下の手順を実行します。
- **[I/O 通信リトライ時間]** ボックスに、I/O サーバーへの接続の試行に失敗した後に InTouch アプリケーションが接続を再試行するまで待機する秒数を入力します。最初の接続試行で InTouch が I/O サーバーに正常に接続された場合、**[I/O 通信リトライ時間]** ボックスの設定は何の影響も及ぼしません。
 - WindowViewer の起動時に、通信しようとしている I/O サーバーが実行中でない場合にダイアログ ボックスを表示するには、**[ローカルのサーバーを起動]** チェック ボックスをオンにします。
 - デフォルト設定でアクセス名を再初期化するには、**[デフォルトの再初期化]** チェック ボックスをオンにします。アクセス名に割り当てられた現在の値は無視され、元の設定で再初期化されます。
7. **[WindowMaker 移行時の設定]** 領域で、以下の手順を実行します。
- WindowMaker を起動するときに WindowViewer を自動的に終了するには、**[WindowViewer を自動的に終了する]** チェック ボックスをオンにします。
- このオプションを選択すると、WindowMaker の設定画面の **[全般]** タブの **[WindowViewer への移行時に自動的に終了]** オプションも自動的に選択されます。メモリの容量に問題があり、WindowViewer と WindowMaker 間的高速切り替えを使用する場合は、このオプションは選択しないでください。
- WindowViewer から WindowMaker に切り替えるする際に、開いているすべてのウィンドウを自動的に閉じるには、**[全ウィンドウを自動的に閉じる]** チェック ボックスをオンにします。
8. **[データ更新時間の設定 (ミリ秒)]** 領域で、以下の手順を実行します。
- **[内部イベント チェック周期]** ボックスに、InTouch HMI が内蔵タイマーをチェックする間隔を入力します。
- この間隔に基づいて、アプリケーションの **[実行中]**、ウィンドウの **[表示中]**、条件の **[True の時]** または **[False の時]**、キーとタッチ押しボタンアクションの **[押している間]** などの QuickScript が実行されるときが決定されます。

このオプションにより、INTOUCH.ini ファイル内の **TimerTickInterval** パラメータの値が設定されます。100 ミリ秒ごとに実行するようスケジュールされたスクリプトでは、[内部イベント チェック周期] は 50 ミリ秒を超えないように設定する必要があります。Windows XP または Windows 2003 を実行するコンピュータでは、最も低い内部イベント チェック周期は 10 ミリ秒です。

- **[時間システム変数の更新周期]** ボックスに、システム タグ (\$Msec、\$Second、\$Minute など) の時間が更新される間隔をミリ秒単位で入力します。

デフォルト設定 (1000 ミリ秒) を使用することが推奨されます。このオプションを 0 に設定すると、時間変数は更新されません。

9. [その他の設定] 領域で、以下の手順を実行します。

- ウィンドウ上のタッチ対応オブジェクトをオペレータが選択したときに InTouch アプリケーションがビーブ音を発するようになるには、**[オブジェクト クリック時ビーブする]** を選択します。
- QuickScript が実行されるたびにロガーにメッセージを書き込む場合、**[スクリプトのデバッグ]** を選択します。

[メニュー/操作設定] プロパティ ページで **[デバッグ]** を選択した場合、WindowViewer の **[システム]** メニューから QuickScript のログ記録の有効化と無効化を切り替えることができます。

- トレンドオブジェクトの更新スピードを速くする場合、**[トレンドの更新を速くする]** を選択します。

このオプションは、アプリケーション ウィンドウ上のランタイム トレンドにオブジェクトが重なっていない場合にのみ選択してください。オブジェクトがトレンドに重なっている場合にこのオプションを選択すると、オブジェクトを正しく描画できない場合があります。

- InTouch バージョン 3.26 以前で開発された国際アプリケーションを使用している場合、**[旧バージョンの使用]** チェック ボックスをオンにします。

10. [Hotlink] 領域で、以下の手順を実行します。

- ランタイム画面上のオブジェクトにユーザーがカーソルを移動したときにオブジェクトを強調表示するには、**[HotLink の周りにハローを表示する]** チェック ボックスをオンにします。
- ActiveX コントロールの周りにハローを表示するには、**[ActiveX コントロールの周りにハローを表示する]** チェック ボックスをオンにします。
- オブジェクトの輪郭の周りのハローにユーザーがカーソルを移動したときに強調表示するには、**[オブジェクトの形に従いハローを表示する]** チェック ボックスをオンにします。

11. [キーボード] 領域で、使用するキーボードのタイプを必要に応じて選択します。

WindowViewer でのキーボード オプションの設定の詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[オブジェクトのアニメーション化](#)」を参照してください。

12. ランタイム時に 10 進値だけを許可して数値以外の文字を制限するには、**[10 進法]** チェック ボックスをオンにします。(**[10 進法]** オプションは英語以外のオペレーティングシステムではテストされていません。)
13. **[英数字キーボード]** および **[数値キーボード]** 領域で、X、Y の位置、および **[幅]** と **[高さ]** の値を設定します。
14. **[保存]** をクリックします。

WindowViewer の表示特性の設定

InTouch アプリケーションを実行するときの WindowViewer の表示特性を決めるプロパティを設定できます。これらのプロパティにより、WindowViewer ウィンドウに表示されるメニュー、コマンド、および標準的なコントロールが決まります。

WindowViewer の表示特性を設定するには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[WindowViewer] をクリックします。
WindowViewer の設定画面が表示されます。
3. [ウィンドウ] タブをクリックします。
表示特性のチェック ボックスをオンにします。

WindowViewer

Preferences

Application type

Window

Memory

Startup

Advanced format

Target resolution

Target resolution

Screen resolution

☒ Show target resolution boundary canvas

Width:

1920

Pixels

Height:

1080

Pixels

Aspect ratio:

16 : 9

Window

☐ Hide title bar

Title bar text

InTouch - WindowViewer

☒ Show application directory in titlebar

☒ Control menu
 ☒ Minimize box
 ☒ Maximize box
 ☒ Size controls

Menus

☒ Menu bar

☒ File
 ☒ Logic
 ☐ Debug
 ☒ Special
 ☒ Security

☒ WindowMaker

☒ Start uninit conversations
 ☒ Log on

☒ Reinitialize I/O
 ☒ Change password

☒ Reinitialize all
 ☒ Configure users

☒ Select...
 ☒ Log off

☒ Restart historical log
 ☒ Language

☒ Stop historical logging

☐ Tag viewer

Miscellaneous

☐ Impossible to close
 ☐ Disable ALT key
 ☐ Disable ESC key
 ☐ Disable WIN key

☒ Allow CTRL-Break to stop scripts
 ☐ Hide cursor
 ☐ Always maximize
 ☒ Enable fast switch

Cancel

Save

4. WindowViewer を再起動します。

リモート セッションで実行するアプリケーションへのユーザー アクセスの設定

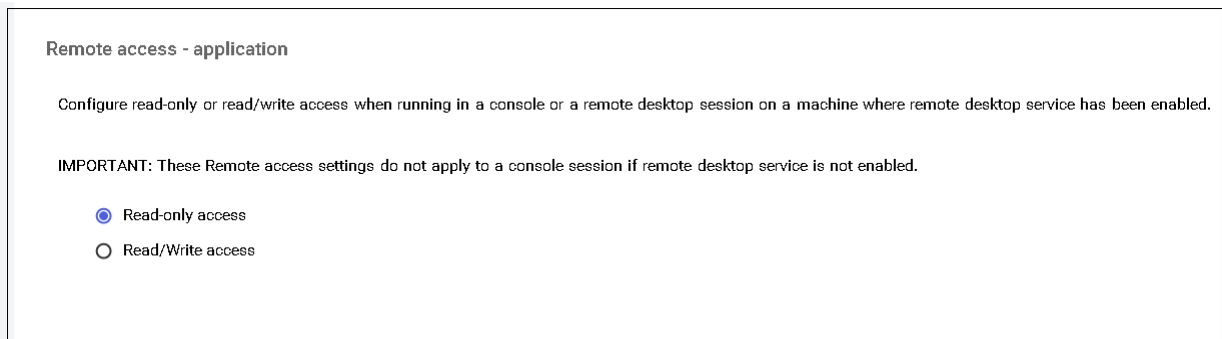
リモート デスクトップ 接続またはターミナル サービス セッションで実行する分散 InTouch アプリケーションへの読み取り専用アクセスを割り当てることができます。読み取り専用アクセスは、アプリケーションを表示する必要があるがあっても書き込み許可を与えるべきでない非オペレータに適切です。たとえば、マネージャは、保護されていない公共ネットワークで InTouch Access Anywhere を使用してモバイル

デバイス上のアプリケーションを表示する必要があります。読み取り専用アクセスを使用すると、公共ネットワークからアクセスできる分散 InTouch アプリケーションの保護を向上させることができます。

リモート ノードで実行するアプリケーションにユーザー アクセスを設定するには

1. WindowMaker でアプリケーションを開きます。
2. [ファイル] メニューの [設定] をクリックし、[WindowViewer] をクリックします。
3. WindowViewer 設定画面で [アプリケーション タイプ] タブを選択します。

[リモート アクセス] セクションが表示され、WindowMaker で開いているアプリケーションへの [読み取り/書き込み] または [読み取り専用] アクセス権限を付与するオプションが表示されます。[読み取り/書き込み] アクセスがデフォルト設定です。



4. 選択を行い、[OK] をクリックしてダイアログを閉じます。

初期化中、WindowViewer は、アプリケーションがリモート セッションで実行しているかどうか、および読み取り専用として指定されているかどうかを検証します。また、リモート ノードに読み取り専用ライセンスがあることのチェックも行われます。これらの条件がすべて満たされた場合、アプリケーションの InTouch リンクおよびユーザー入力アニメーションは読み取り専用モードでのみ表示できます。

WindowViewer のメモリ管理について

ランタイム時のパフォーマンスを改善するために、WindowViewer がウィンドウとポップアップ シンボルにメモリを使用する方法を設定することができます。特定の条件下では、ウィンドウ、および ShowSymbol アニメーションの使用によって表示されるポップアップ シンボルと ShowGraphic スクリプト関数は、データ取得の高速化を可能にするため、ランタイム時にメモリ上に保存できます。

産業用グラフィックスのメモリ内キャッシングは、マネージド InTouch アプリケーションまたはパブリッシュ済み InTouch アプリケーションでのみ使用できます。この機能は、スタンドアロン InTouch アプリケーションでは無効化されています。

また、定期的なメモリのヘルス チェックの周期、およびヒープ メモリ セグメントの設定を指定することもできます。新しいポップアップ シンボルが開くたびに、事前に設定されている間隔に関わらず、メモリのヘルス チェックがトリガされます。

WindowViewer への高速切り替え、または WindowViewer から WindowMaker への高速切り替えは、グラフィック キャッシュとウィンドウ メモリ キャッシュの両方を消去します。

ShowGraphic 関数によって表示されるシンボルのメモリ管理の詳細については、『産業用グラフィック エディタ ユーザー ガイド』を参照してください。

WindowViewer ウィンドウのメモリ使用量の設定

ランタイム時のパフォーマンスを改善するために、WindowViewer アプリケーション ウィンドウのメモリ使用量を設定できます。

特定の条件下では、キャッシュされた閉じたウィンドウを再び開くと、ディスクからではなく、メモリから読み取ります。

特定のウィンドウに高い優先度のメモリ使用量を指定し、それらのウィンドウだけに個別のメモリ設定を構成することができます。

WindowViewer のメモリ オプションのいずれかを変更した後、WindowViewer が再起動するまで変更は反映されません。

メモリのプロパティを設定するには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[WindowViewer] をクリックします。
WindowViewer の設定画面が表示されます。
3. [メモリ] タブをクリックします。

Preferences Application type Window **Memory** Startup Advanced format Managed application

In-memory caching

☒ Use In-Memory cache

☐ Cache Industrial Graphics not embedded in InTouch windows

☐ Reset Industrial Graphic cached values and windows properties

Memory limit for in-memory graphics: 70 % In-memory graphics expiration time: 0 hours

High priority window caching

☒ Enable high priority window caching

Memory limit for high priority windows: 90 %

High priority windows Search Select all

4. [メモリ内キャッシュ] 領域で、以下の手順を実行します。
 - a. ランタイム時にすべての閉じたウィンドウをメモリにキャッシュする場合、[メモリ内キャッシュを使用する] チェック ボックスをオンにします。
 - b. 産業用グラフィック シンボル キャッシュを有効にする場合、[InTouch ウィンドウに埋め込まれていない産業用グラフィックをキャッシュ] チェック ボックスをオンにします。
 - c. ウィンドウを閉じて再度開いたときにカスタム プロパティとウィンドウ プロパティをリセットするには、[産業用グラフィックのキャッシュ値とウィンドウ プロパティをリセット] チェック ボックスをオンにします。
 - d. [メモリ内ウィンドウのメモリ制限] ボックスに、ランタイム時にキャッシュ メモリに保持される閉じたメモリ内ウィンドウの制限値を入力します。デフォルトのメモリ制限は、プロセス メモリの 70% です。

このメモリ制限を超えると、高優先度ウィンドウとしてマークされていない限り、ランタイム時にキャッシュから最も古い閉じたメモリ内ウィンドウが自動的に削除されます。

メモリ内ウィンドウのメモリ制限は、高優先度ウィンドウのメモリ制限よりも常に少なくなります。

- e. **[メモリ内ウィンドウのキャッシュ期限]** ボックスに、閉じたメモリ内ウィンドウをランタイム時にキャッシュ メモリに保持しておく期限を入力します。0 ～ 8760 時間の値を入力することができます。デフォルト値は 0 時間（時間制限なし）です。

注記: InTouch ウィンドウと産業用グラフィックは、メモリ内キャッシュを共有します。設定されたメモリ制限を超えると、キャッシュから最も古いメモリ内グラフィックが自動的に削除されます。

メモリ制限またはキャッシュ期限は、どちらの制限に最初に達したかに応じて適用されます。

5. **[高優先度ウィンドウのキャッシュ]** 領域で、以下の手順を実行します。

- a. **[高優先度ウィンドウのキャッシュを有効にする]** チェック ボックスをオンにすると、一部のウィンドウを高優先度としてマークできます。これらのウィンドウはランタイム時に閉じられると、常にキャッシュ メモリに保持されます。
- b. **[高優先度ウィンドウのメモリ制限]** ボックスに、ランタイム時にキャッシュ メモリに保持される閉じた高優先度ウィンドウの制限値を入力します。デフォルトのメモリ制限は 90% です。ランタイム時にメモリの使用パーセンテージがこの制限を超えると、最も古いメモリ内ウィンドウが最初に削除され、次に最も古い高優先度ウィンドウが削除されます。
- c. **[高優先度ウィンドウ]** ボックスで、高優先度ウィンドウとしてマークするウィンドウを選択します。

注記: **[メモリ内キャッシュを使用する]** または **[高優先度ウィンドウのキャッシュを有効にする]** チェック ボックスがオンになっている場合、**[産業用グラフィックのキャッシュ値とウィンドウプロパティをリセット]** チェック ボックスもオンになります。**[産業用グラフィックのキャッシュ値とウィンドウプロパティをリセット]** チェック ボックスがオンになっている場合、リセットアクションは、**[メモリ内キャッシュを使用する]** または **[高優先度ウィンドウのキャッシュを有効にする]** オプションにも影響します。

6. **[保存]** をクリックします。

メモリのヘルス チェックの周期の設定

システムは、メモリとグラフィカルデバイス インターフェイス (GDI) の数を一定の周期でチェックします。メモリまたは GDI の数の制限が超えている場合、システムはウィンドウを削除し始めます。デフォルトでは、5 分周期に設定されています。

周期を変更するには、InTouch.ini ファイルに新しいエントリを追加して、新しい周期の値を指定します。

チェックを行わない場合には、新しいエントリを追加して、その値を 0 に設定します。

周期を変更した後で、変更を適用するには、WindowViewer を再起動する必要があります。

ウィンドウが削除される方法の詳細については、「[WindowViewer ウィンドウのメモリ使用量の設定](#)」を参照してください。

メモリのヘルス チェックの周期を設定するには

1. アプリケーションのフォルダの InTouch.ini ファイルを開きます。

2. [Intouch] セクションに、次のスクリプトを追加します。ここで *<interval>* は、希望する周期を分単位で表したものです。

```
MemoryHealthCheckInterval = <interval>
```

新しいポップアップシンボルまたは新しいウィンドウを開くと、事前に設定されている間隔に関わらず、それはメモリのヘルスチェックをトリガします。

wwHeap メモリの設定

wwHeap はヒープメモリ セグメントを管理するメモリ マネージャです。このメモリ マネージャは、1 つまたは複数のプログラムが仮想メモリを共有するためのメカニズムを提供します。

注意: wwHeap のメモリ設定は、ロガーでメモリの競合が報告された場合にのみ、変更してください。

wwHeap のサイズと場所を指定することで、wwHeap メモリを設定することができます。デフォルトのサイズ、デフォルトの開始場所、および可能な場所の範囲は、オペレーティングシステムごとに異なります。

デフォルトのサイズについては、次の表を参照してください。

オペレーティング システム	デフォルトのサイズ
32 ビット	1519 MB
32 ビット (/3GB スイッチ有効)	2048 MB
64 ビット	2048 MB

デフォルトの場所と可能な場所の範囲については、次の表を参照してください。

オペレーティング システム	デフォルトの開始場所	可能な範囲
32 ビット	0x21000000	0x00010000 ~ 0x7FFEFFFF
32 ビット (/3GB スイッチ有効)	0x40000000	0x00010000 ~ 0xBFFEFFFF
64 ビット	0x80000000	0x00010000 ~ 0xFFFFEFFFF

wwHeap メモリを設定するには

- アプリケーション マネージャを起動します。
- [ツール] メニューで、[ノードのプロパティ] をクリックします。
[ノードのプロパティ] 画面が表示されます。
- [メモリ設定] タブをクリックします。

Node properties

App development Resolution **Memory Settings** Performance

wwHeap is a memory manager that manages the heap memory segment and provides a mechanism for one or more programs to share virtual memory.

Enter the heap memory size and start location below

Size: 2048 MB

Start location: 0x80000000

Defaults

Restoring defaults will reset the wwHeap to the default size and location for this node.

Reset to defaults

Cancel Ok

4. 以下の手順を実行します。

- **【サイズ】** ボックスに、wwHeap メモリのサイズを入力します。1 ～ 2048 MB の値を入力することができます。
- **【開始場所】** ボックスに、開始場所のアドレスを入力します。

5. デフォルト値に戻す場合は、**【デフォルトの復元】** をクリックします。

6. **【OK】** をクリックします。

WindowViewer オプションの **【地域の設定】** の選択

InTouch WindowMaker には、WindowViewer の詳細な書式設定 **【地域の設定】** 設定オプションが含まれます。

地域ロケールによる数値形式を有効にするには、デザイン時に **【地域の設定】** オプションを選択して、産業用グラフィックの数値を **【地域】** 設定で選択されている国に設定する必要があります。デフォルトでは、**【地域の設定】** オプションは無効になっています。

OS の地域の設定がチェックされるのは、WindowViewer の起動時だけです。したがって、次のいずれかの操作を行った場合、WindowViewer を再起動する必要があることがあります。

- WindowViewer が実行している間に **【地域の設定】** オプションを選択する。

- **〔地域の設定〕** オプションが選択されている状態で **WindowViewer** が実行しているときに OS の地域の設定を変更する。

詳細な書式設定のプロパティの設定

WindowViewer では、値の大きさに基づいて、ランタイム時に表示する小数点以下の桁数を設定することができます。この機能は、**〔値の表示〕** または **〔ユーザー入力〕** アニメーションで **〔実数〕** を選択した場合にのみ使用可能になります。

フィールドデバイスから収集したデータの品質が低い場合、または大きすぎて表示できない場合に、ランタイム時に表示される特殊文字を指定できます。

産業用グラフィックに表示される数値は、**Windows** のコントロールパネルの **〔地域〕** 設定の主な使用場所に設定された国の形式で表示されます。ランタイム時、産業用グラフィックの数値は、OS の地域の設定で指定された国の数値形式に一致する **1000** 桁区切り文字を使用して表示できます。

詳細な書式設定のプロパティを設定するには

1. **WindowMaker** を開きます。
2. **〔ファイル〕** メニューの **〔設定〕** をクリックし、**〔WindowViewer〕** をクリックします。
WindowViewer の設定画面が表示されます。
3. **〔詳細な書式設定〕** タブをクリックします。

Value range	Precision
Values less than 1 (from 0 to 0.999)	4
Values in the ones (from 1 to 9)	2
Values in the tens (from 10 to 99)	2
Values in the hundreds (from 100 to 999)	2
Values in the thousands (from 1,000 to 9,999)	2
Values in the ten thousands (from 10,000 to 99,999)	0
Values in the hundred thousands (from 100,000 to 999,999)	0
Values in the millions (from 1,000,000 to 9,999,999)	0
Values in the ten millions (from 10,000,000 to 99,999,999)	0
Values in the hundred million or greater (from 100,000,000 on)	0

Special characters to show at runtime

Bad quality with no value
!

Value too large for fixed field
*

☐ Follow regional settings for industrial graphics

Restore Default

Cancel Save

4. [小数点精度の書式設定] セクションに、実数タイプの値の範囲ごとに、ランタイム時に表示される数値の小数点以下の桁数を入力します。
5. [ランタイム時に表示する特殊文字] セクションで、以下の手順を実行します。
 - [値を持たない低クオリティ] ボックスに、データ ポイントの品質が低すぎるか、または良い品質の最後の既知の値がない場合に、ランタイム時に表示される文字を入力します。デフォルトの文字は ! ですが、スペース以外の ASCII 文字を入力することができます。
 - [固定幅よりも大きい値] ボックスに、値が大きすぎて表示できない場合に、ランタイム時に表示される文字を入力します。デフォルトの文字は * ですが、スペース以外の ASCII 文字を入力することができます。
6. コンピュータの [地域] 設定で指定されている国の形式で産業用グラフィック内に数値を表示するには、[産業用グラフィックの地域の設定に従う] チェック ボックスをオンにします。

デフォルトでは、[地域の設定] オプションは無効になっています。

注記: OS の地域の設定がチェックされるのは、WindowViewer の起動時だけです。したがって、

- 1) WindowViewer が実行しているときに **【地域の設定】** オプションを選択した場合、または
- 2) **【地域の設定】** オプションが選択されている状態で WindowViewer が実行しているときに中に OS の地域の設定を変更した場合、WindowViewer を再起動する必要があることがあります。

7. デフォルトの値に戻すには、**【デフォルトの復元】** をクリックします。

HMI/SCADA アプリケーションをホストするコンピュータの地域ロケールの設定

地域ロケールによる数値形式を有効にするには、HMI/SCADA アプリケーションを実行するコンピュータの地域を産業用グラフィックの数値の書式として使用する国に設定します。

地域の設定は、Windows の **【コントロールパネル】** からアクセスできます。産業用グラフィックの数値を米国以外の形式で表示するには、**【形式】** タブを選択して、**【形式】** フィールドで国を選択します。

ターミナル サーバー環境での WindowViewer のコア親和性の設定

コンピュータに複数のプロセッサが搭載されている場合、ターミナル サービス クライアントの実行時に、WindowViewer を実行するために CPU 0 以外の CPU（コア）を使用できます。これは、ターミナル サーバー環境で動作する InTouch アプリケーションが、ターミナル サーバーのマルチコア機能を活用できるようにするためです。ただし、WindowViewer がターミナル サーバー コンソールで動作している場合、このオプションは使用できません。InTouch アプリケーションは、利用できるプロセッサ数には関わりなく、常に 1 つのプロセッサ上で動作します。

WindowViewer が起動すると、システムは、コンピュータに複数のプロセッサが搭載されているかということと、どのプロセッサで WindowViewer のインスタンスを実行することが許されているかをチェックします。WindowViewer は次にプロセッサを順にチェックして、実行されている View インスタンスの最も少ないプロセッサで実行を開始します。

オペレータが管理特権を持っていて **【パフォーマンス】** タブにアクセスできる場合には、プロセッサの「プール」を設定して、そこから WindowViewer が動作するプロセッサを選択しようにすることができます。

WindowViewer のコア親和性は、アプリケーション マネージャで設定します。タスク マネージャで WindowViewer のコア親和性を手動で調整しなくても済むように、WindowViewer のコア選択プロセスは、タスク マネージャのコア親和性設定を考慮しないようになっています。

プロセッサ「プール」を設定するには

1. アプリケーション マネージャを起動します。
2. **【ツール】** メニューで、**【ノードのプロパティ】** をクリックします。**【ノードのプロパティ】** ダイアログ ボックスが表示されます。
3. **【パフォーマンス】** タブをクリックします。

Node properties

App development

Resolution

Memory Settings

Performance

Each WindowViewer View Application runs on a single processor. Upon startup, WindowViewer selects sequentially the next processor on the system.

☒ Allow WindowViewer to select from all available processors.

☐ WindowViewer is limited to use only the processors selected below.

☒ CPU 0

☒ CPU 1

Limit to 0

Allow all

Cancel

Ok

- WindowViewer が利用可能なプロセッサをすべて使用できるようにするには、**[WindowViewer がすべての利用可能なプロセッサから選択することを可能にします]** をクリックします。
- WindowViewer が使用するプロセッサを制限するには、**[WindowViewer は、以下の選択されたプロセッサのみの使用に制限されます]** をクリックして、次のいずれかを実行します。
 - WindowViewer が動作できる各 CPU に対応する **[CPU]** チェック ボックスをオンにします。
 - [0 に制限]** をクリックすると、WindowViewer はプロセッサ 0 でのみ実行できます。このボタンをクリックすると、**[CPU 0]** チェック ボックスが自動的にオンになります。
 - すべてのチェック ボックスをオンにするには、**[すべてを許可]** をクリックします。

選択したプロセッサは、いつでもオフにすることができます。また、リスト内の新しいプロセッサをオンにすることもできます。また、一度に複数のプロセッサをオンにすることもできます。ある

プロセッサのチェック ボックスをオフにしても、WindowViewer のインスタンスはそのプロセッサでの動作を続けます。

6. **[OK]** をクリックします。WindowViewer は、他の View セッションに基づき、次の CPU で動作を開始します。

WindowViewer ウィンドウの操作

通常、InTouch アプリケーションには、生産プロセスを管理するためにオペレータが操作するウィンドウがいくつかあります。WindowViewer の設定画面の **[ウィンドウ]** タブで設定したプロパティに基づいて、オペレータは WindowViewer の **[ファイル]** メニューから標準的なコマンドを実行してウィンドウを開くことや閉じることができます。

さまざまなモードでの WindowViewer の表示

[詳細] をクリックして、一覧表示を詳細表示に切り替えます。ウィンドウのタイプ、ウィンドウが最後に変更された日時などの詳細が表示されます。

詳細表示では、チェック ボックスだけではなく、行の部分をクリックしても、現在開いていないウィンドウを選択したり選択解除できます。選択すると行全体がハイライトされます。

- 選択したウィンドウを開くには、**[OK]** をクリックします。
- 選択をキャンセルしてダイアログ ボックスを閉じるには、**[キャンセル]** をクリックします。
- ダイアログ ボックスを元の一覧表示に戻すには、**[リスト]** をクリックします。
- 表示されたウィンドウをすべて選択するには、**[すべて選択]** をクリックします。
- すべての選択ウィンドウを解除するには、**[全てを消去]** をクリックします。
- 昇順または降順でリストをソートするには、カラム ヘッダーをクリックします。

WindowViewer からウィンドウを開く

[ファイル] メニューを表示するように WindowViewer を設定すると、オペレータは InTouch アプリケーション ウィンドウを開いたり閉じたりできます。**[ファイル]** メニューでオペレータが **[ウィンドウを開く]** コマンドまたは **[ウィンドウを閉じる]** コマンドのどちらかをクリックすると、選択したコマンドに対応するダイアログ ボックスが表示されます。

選択したコマンドに適用可能なすべてのウィンドウの名前が、リストに表示されます。たとえば、**[ウィンドウを開く]** コマンドをクリックした後に、**[表示するウィンドウ]** 画面が表示されます。

[ファイル] メニューを表示するように WindowViewer が設定されている場合、オペレータは InTouch アプリケーション ウィンドウを開くことができます。

WindowViewer からウィンドウを開くには

1. **[ファイル]** メニューの **[ウィンドウを開く]** をクリックします。
[ウィンドウを開く] ダイアログ ボックスが表示されます。
2. 開く各ウィンドウの名前の横にあるチェック ボックスをオンにします。
1. **[OK]** をクリックしてダイアログ ボックスを閉じ、選択したウィンドウを開きます。

注記: 「リプレース」タイプのウィンドウを選択すると、このウィンドウと交差しているウィンドウはすべて閉じます。

WindowViewer からウィンドウを閉じる

[ファイル] メニューを表示するように WindowViewer が設定されている場合、オペレータは InTouch アプリケーション ウィンドウを閉じることができます。

開いているウィンドウを閉じるには

1. [ファイル] メニューの [ウィンドウを閉じる] をクリックします。
[閉じるウィンドウを選択] 画面が表示されます。
2. 閉じる 1 つまたは複数のウィンドウの名前の横にあるチェック ボックスをオンにします。
3. [OK] をクリックして、選択したウィンドウを閉じます。

WindowViewer から WindowMaker への切り替え

InTouch アプリケーションを開発するとき、[ファイル] メニューの [WindowMaker] コマンドまたはツールバーの [開発] コマンドをクリックすることにより、WindowMaker と WindowViewer を簡単に切り替えることができます。これを高速切り替えと言います。

高速切り替えは、短期間での開発テスト時専用です。運環境では使用しないでください。WindowMaker に切り替えるためのコマンドを非表示にすることができます。

WindowViewer から WindowMaker に切り替えるには

1. WindowViewer を起動します。
2. [ファイル] メニューの [WindowMaker] をクリックします。
[ウィンドウの編集] ダイアログ ボックスが表示されます。
3. WindowMaker に切り替えるときに開く各ウィンドウの名前の横にあるチェック ボックスをオンにします。
4. [OK] をクリックしてダイアログ ボックスを閉じ、WindowMaker に切り替えます。

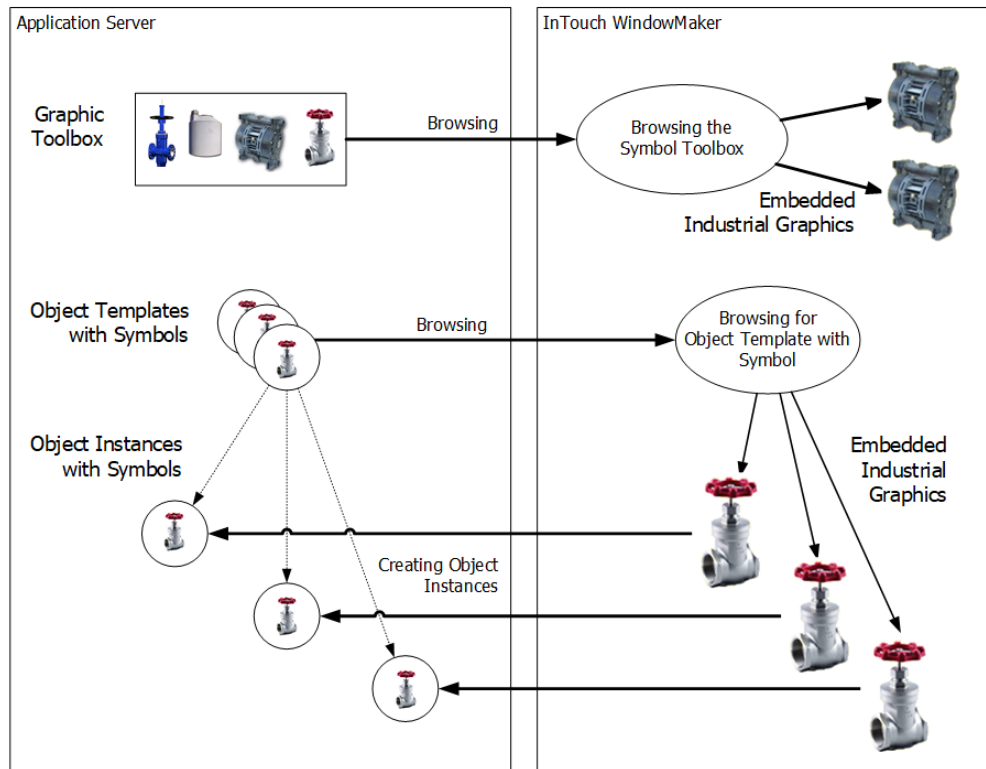
注記: 開発中に WindowViewer のプロパティを設定したときに、アプリケーション開発者が [WindowViewer を自動的に終了する] オプションを選択した場合、WindowMaker に切り替えるときに WindowViewer が自動的に終了します。

産業用グラフィック

System Platform IDE には、製造プロセスの表現やオートメーション オブジェクトへの HMI インターフェイスの提供を行うシンボルを作成する産業用グラフィック エディタが含まれています。

以下の図は、産業用グラフィック エディタで作成したシンボルが InTouch アプリケーションでどのように使用されるのかを示しています。

Using Industrial Graphics



産業用グラフィックの作成

産業用グラフィックは IDE の産業用グラフィック エディタで作成します。

以下を作成できます。

- 産業用グラフィック（産業用グラフィック ツールボックスを使用します）。これらは、特定のオブジェクト テンプレートやオブジェクト インスタンスには関連付けられていません。
- 特定のオブジェクト テンプレートまたはインスタンスに含まれる産業用グラフィック。

産業用グラフィック エディタ

System Platform IDE 内で InTouch アプリケーションを管理する利点に加え、産業用グラフィック エディタでグラフィックを作成することにより製造環境をモデル化できます。産業用グラフィック エディタは System Platform IDE に完全に統合されており、強力な編集機能をサポートします。

産業用グラフィック エディタでは、シンボル要素に要素スタイルを適用できます。要素スタイルは、グラフィック要素に適用されるテキスト、要素の塗りつぶし、要素の輪郭、および線の表示プロパティの定義済みセットです。要素スタイルを使用すると、標準の表示スタイルをグラフィック要素に追加して、一貫したシンボル標準を容易に確立できます。

産業用グラフィック エディタにはシンボル ウィザード エディタも含まれているので、さまざまな表示設定および機能設定を含むシンボル ウィザードを作成できます。シンボル ウィザード エディタを使用すると、左右の弁を個々に設定できるポンプシンボルを作成することなどが可能です。**Situational Awareness Library** シンボルは、構築済みの設定を含むシンボル ウィザードのサンプルのコレクションで、シンボルのウィザード オプションから選択できます。単一の **Situational Awareness Library** シンボル

は、容易に設定して温度計、風量系、または圧力計を表すことができます。シンボル ウィザードがマネージド InTouch アプリケーションに埋め込まれている場合、必要な設定を選択します。

WindowMaker には、産業用グラフィック エディタを開くことなく InTouch アプリケーションの産業用シンボルと Situational Awareness Library シンボルを選択できる産業用グラフィック ツールボックスが組み込まれています。シンボルは、WindowMaker から InTouch ウィンドウに直接ドラッグアンドドロップすることによって追加できます。産業用シンボルと Situational Awareness Library シンボルの操作に関する詳細については、『産業用グラフィック エディタ ユーザー ガイド』または WindowMaker ヘルプを参照してください。

注記: Microsoft のレンダリング技術の最近のバージョンにおいては、特定のグラデーション機能が廃止されました。将来的な対応が考慮されたグラフィックを使用できるよう、影響を受ける機能が設定環境から削除されました。廃止された機能を使用して以前に設定されたグラフィックについては、引き続き期待通りにレンダリングされます。『産業用グラフィック エディタ ユーザー ガイド』の「廃止された機能を含むグラフィックのロード」を参照してください。

章 2 アプリケーション

InTouch アプリケーション マネージャまたは IDE を使用して、InTouch アプリケーションを管理できます。

InTouch アプリケーションは、管理方法、サポートされるシンボルの種類、およびパブリッシュ元によって分類されます。

- **スタンドアロン InTouch アプリケーション:** スタンドアロン アプリケーションはアプリケーション マネージャで作成されるデフォルトのアプリケーションで、InTouch シンボルと産業用グラフィックの柔軟性を提供します。
- **マネージド InTouch アプリケーション:** マネージド アプリケーションは、IDE を使用して作成および管理されます。
- **パブリッシュ済み InTouch アプリケーション:** 派生した InTouchViewApp テンプレートからマネージド アプリケーションをエクスポートすることにより、パブリッシュ済みアプリケーションを作成できます。
- **InTouchView アプリケーション:** InTouchView アプリケーションは InTouch アプリケーション マネージャまたは IDE から作成できます。

InTouch HMI は、古いモダン アプリケーションの移行およびアップグレードをサポートします。

スタンドアロン InTouch アプリケーション

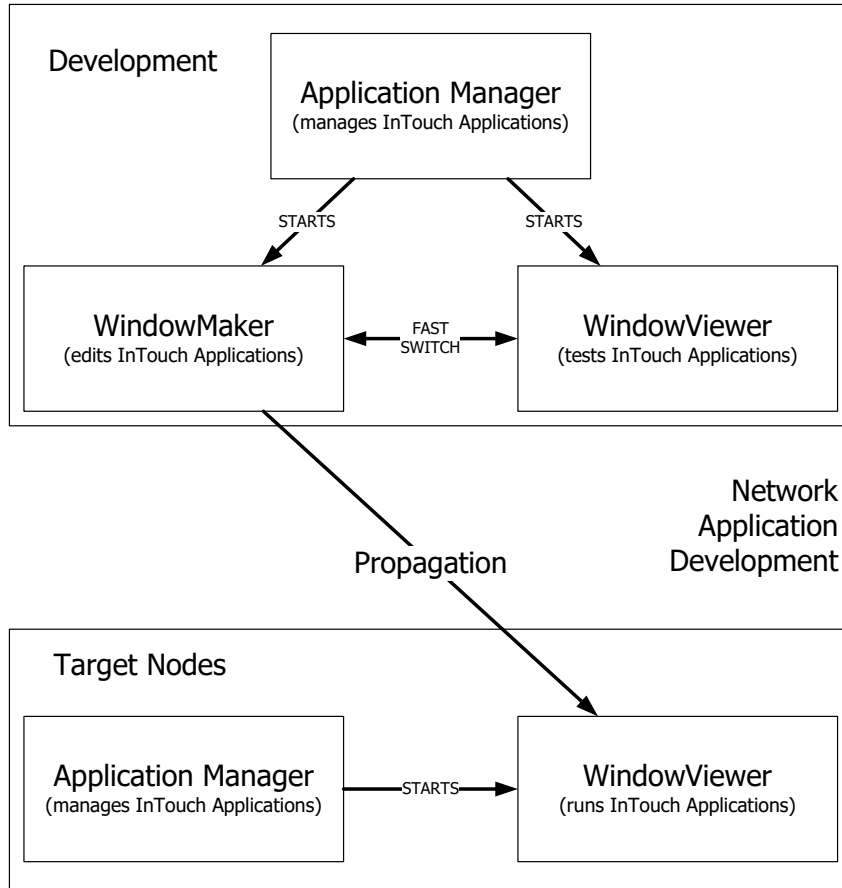
スタンドアロン InTouch アプリケーションは、InTouch アプリケーション マネージャによって管理されます。これらは、InTouch アプリケーション マネージャに **[スタンドアロン]** として表示されます。

アプリケーション マネージャを使用すると、以下を行うことができます。

- スタンドアロン InTouch アプリケーションの作成と管理。
- WindowMaker の起動と InTouch アプリケーションの編集
- WindowViewer の起動と InTouch アプリケーションの実行

また、アプリケーションをテストまたは実行したり、アプリケーションに変更を加えるために切り替える目的で、WindowMaker と WindowViewer 間で直接切り替えることもできます。スタンドアロン アプリケーションは、ディレクトリ ファイルシステムにある InTouch HMI によって保持されている一連のファイルで構成されます。これには産業用グラフィックを含めることもできます。スタンドアロン アプリケーションは複数のネットワーク ノード上に配置でき、単一のノードに制限されていません。IDE にインポートすることやマネージド アプリケーションに変換することができます。

開発ノード上の InTouch アプリケーションから対象ノードで実行中の InTouch アプリケーションへの変更の反映は、ネットワーク アプリケーション開発によって管理されます。



マネージド InTouch アプリケーション

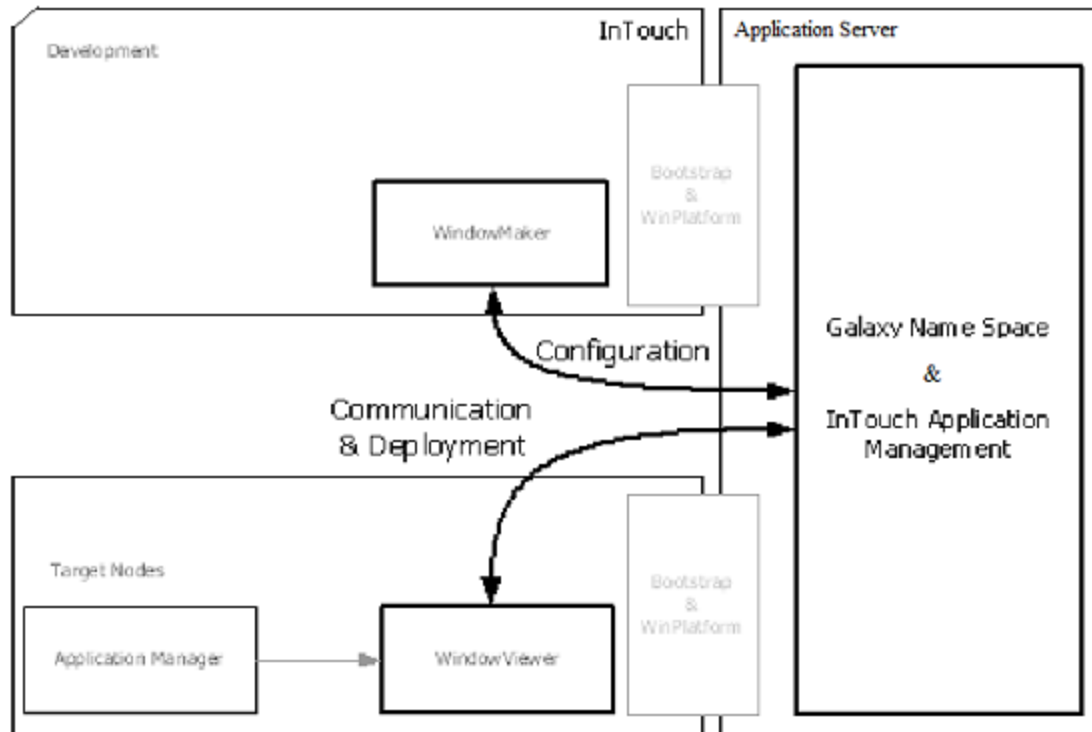
System Platform IDE（統合開発環境）を使用して、InTouch アプリケーションを管理できます。これらのアプリケーションは、マネージド InTouch アプリケーションと呼ばれます。

これらのアプリケーションは、InTouch アプリケーションマネージャに **【マネージド】** として表示されます。マネージド InTouch アプリケーションは ArchestrA 環境と密接に統合され、高度なグラフィックをサポートします。

InTouch アプリケーションは、Galaxy の 1 つのノード上で WindowMaker を使用して開発します。その後、WindowViewer が実行されている 1 つまたは複数の対象ノードに配置します。

IDE のシステムプラットフォーム機能を使用して InTouch アプリケーションを管理すると、以下を行うことができます。

- どのノードでどの InTouch アプリケーションが実行されているのかの参照。
- InTouch アプリケーションの中央リポジトリの使用。
- リモート ノードで実行している WindowViewer への変更の配置。

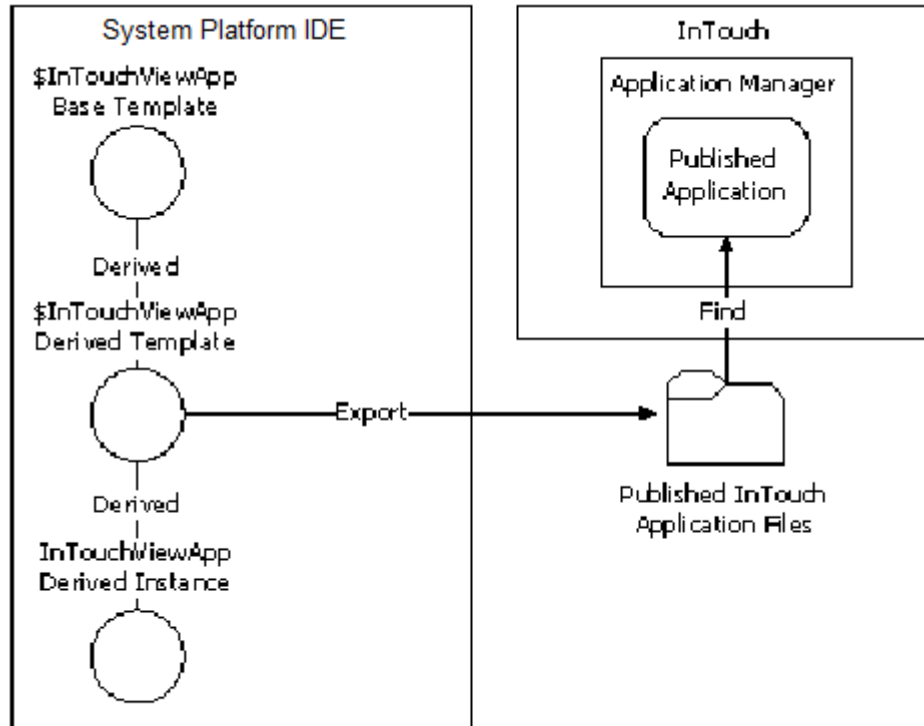


WindowViewer でマネージドアプリケーションを起動できるのは、アプリケーション マネージャからだけです。**WindowViewer** を起動すると、ランタイム中にマネージドアプリケーションのファイルがフォルダにコピーされます。

各マネージドアプリケーションは、ベース テンプレートから派生した **InTouchViewApp** オブジェクトに関連付けられています。**InTouchViewApp** オブジェクトには、マネージド **InTouch** アプリケーション フォルダおよびマネージド **InTouch** アプリケーションのその他の動作固有の情報への参照のみが含まれています。アプリケーション ファイルは、**Galaxy** ファイル リポジトリ内の個別のフォルダに保存されます。1 つのフォルダには最新のチェックインバージョンの **InTouch** アプリケーションが含まれ、その他のフォルダには最新のチェックアウト バージョンが含まれます。IDE には産業用グラフィック エディタが含まれており、**InTouch HMI** アプリケーション内で製造プロセスを表すシンボルの作成に使用できます。IDE から **WindowMaker** を開いた場合のみ、**WindowMaker** と **WindowViewer** をすばやく切り替えてマネージドアプリケーションをテストできます。

パブリッシュ済み InTouch アプリケーション

マネージド **InTouch** アプリケーションを編集した後、パブリッシュすることができます。派生した **InTouchViewApp** テンプレートからマネージドアプリケーションをパブリッシュできます。マネージドアプリケーションをパブリッシュすると、**InTouch** アプリケーション ファイルおよびアプリケーションに埋め込まれた産業用グラフィックが含まれるユーザー定義フォルダが作成されます。フォルダを検索するには、アプリケーション マネージャの **[検索]** ユーティリティを使用します。その後、変換されたアプリケーションが、パブリッシュ済みアプリケーションとしてアプリケーション マネージャに表示されます。



パブリッシュ済み InTouch アプリケーションの利点は、スタンドアロン アプリケーションのように分散しつつ、産業用グラフィックの機能を引き続きサポートする点です。

ただし、以下を行うことはできなくなります。

- System Platform IDE の使用による InTouch アプリケーションの配置。
- InTouch アプリケーションの産業用グラフィックの編集または追加。
- パブリッシュ済み InTouch アプリケーションの編集。

WindowMaker を使用して、パブリッシュ済みアプリケーションをバージョン 11.1 (2014 R2 Patch 01) 以前のバージョンからバージョン 11.1 Patch 01 の InTouch に移行できます。その後、アプリケーションソース ファイルを更新して再度パブリッシュすることにより、パブリッシュ済みアプリケーションを変更できます。マネージドアプリケーションをパブリッシュした後でも、埋め込まれた産業用グラフィックを引き続き使用して、Galaxy にデータを書き込むことやデータを視覚化することができます。パブリッシュ済みアプリケーションを Galaxy に再度インポートすることはできません。パブリッシュ済みマネージドアプリケーションを移行した後、アプリケーションを再度パブリッシュする必要があります。再パブリッシュの際、埋め込まれたアラーム コントロールが新しいバージョンにアップグレードされます。

InTouchView アプリケーション

InTouchView アプリケーションには、Application Server 環境で使用するための視覚的なインターフェイスが表示されます。InTouchView アプリケーションは WindowViewer で動作し、HMI のほとんどの機能は Application Server から提供されます。InTouch タグ サーバーとして機能するようアプリケーションを設定して、その他のノードにセキュアなタグ情報を提供できます。InTouchView アプリケーションは、ク

ライアント ノードで必要な機能がデータへのアクセスだけで、開発ノードの完全な機能は必要ない場合に便利です。

InTouchView アプリケーションでは、フル機能の InTouch アプリケーションで利用できる標準的な関数の一部のみが提供されています。InTouchView アプリケーションの特徴を以下に示します。

- Application Server Galaxy や InTouch タグ サーバー以外の I/O ソースには接続できません。
- アラームを生成できません。しかし、Application Server オブジェクトや InTouch アラームなどのリモート アラーム プロバイダからアラームを表示および確認できます。
- アプリケーション データやイベントをログに記録しません。InTouchView アプリケーションでは、SYS およびユーザー関連イベントのみが生成されます。
- アプリケーションで消費される Galaxy のデータを保護できるのは、ArchestrA セキュリティだけです。
- 埋め込まれた産業用グラフィック内のタグを参照できません。

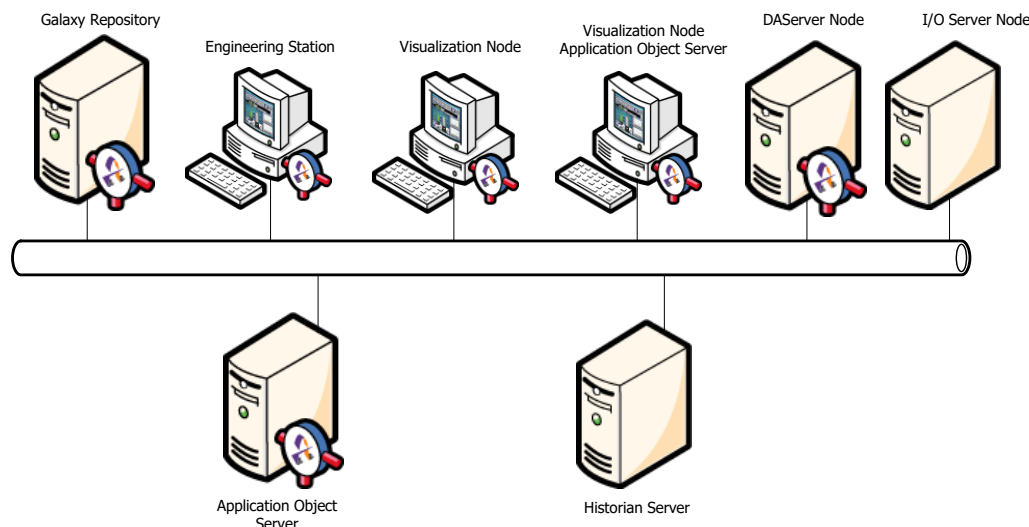
InTouch アプリケーションは WindowMaker で開発し、WindowViewer で実行します。その後、InTouch アプリケーションを InTouchView アプリケーションに変更します。InTouchView では、Application Server を介して InTouch アプリケーションを管理できます。同様に、InTouchView アプリケーションを InTouch アプリケーションに変更できます。

InTouchView アプリケーションを作成するときに使用できない WindowMaker コマンドおよび [タグ名ディクショナリ] オプションを次のリストに示します。

- 使用できないコマンド:
 - アクセス名
 - アラーム グループ
 - 設定...アラーム
 - 設定...履歴ログ
 - 設定...分散名前マネージャ
- 使用できないタグ名ディクショナリ オプション:
 - アラーム
 - 両方
 - データ ログ
 - イベント ログ
 - 優先度

Application Server のアーキテクチャ

Application Server は、分散 InTouch HMI アプリケーションにサービスを提供するために ArchestrA 技術を使用します。Application Server のサービスは、アプリケーション インターフェイスを提供するために、ビジュアル化ノードに InTouch HMI がインストールされている一連のノード上に分散されています。



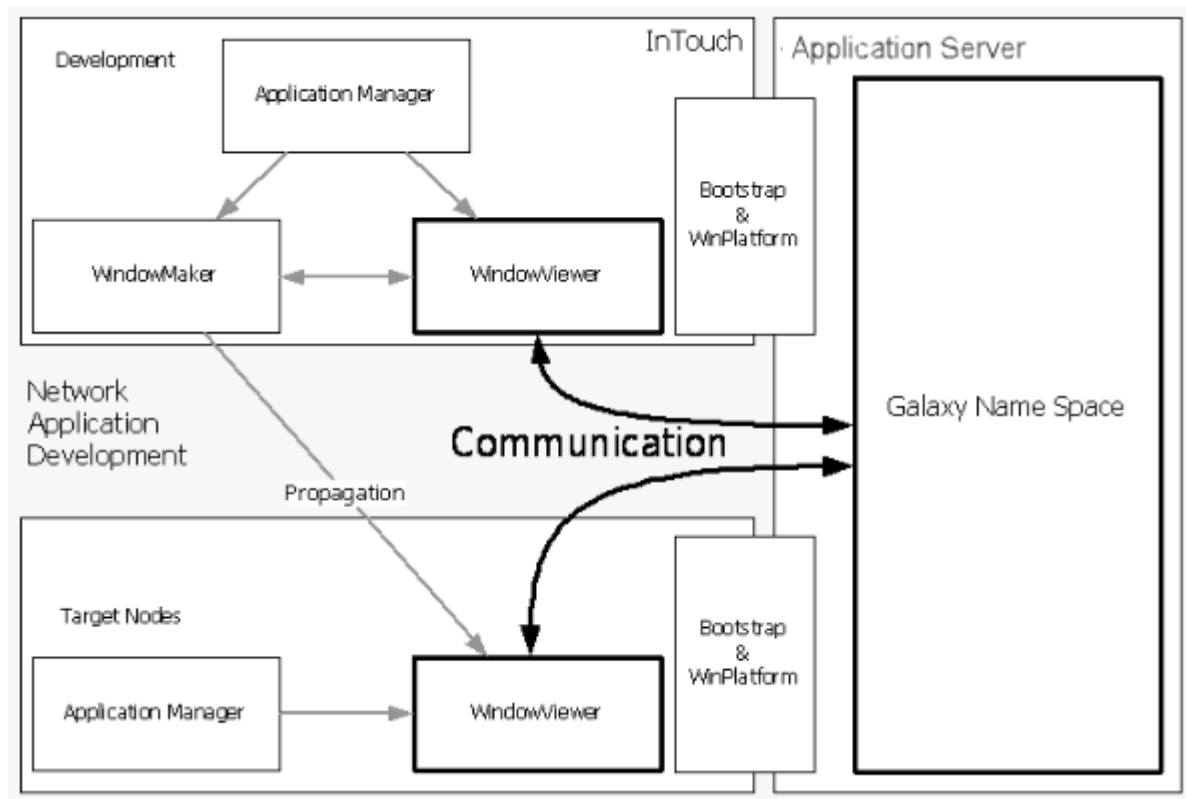
Application Server Galaxy は、複数のノードに分散されたさまざまなコンポーネントを組み込んだ、単一の論理ネームスペースに基づいたシステムの設定を表します。

Application Server には InTouch タグ変数に類似した製造データを保存できますが、より多くのデータタイプおよび配列がサポートされています。Application Server では、InTouch アラームシステムと互換性のあるアラーム機能が提供されています。Application Server では、.NET 関数の有効な機能強化がサポートされた、InTouch HMI と互換性のあるスクリプト言語が提供されています。Galaxy データの読み取りおよび書き込みを可能にするには、InTouch ノードに Application Server Bootstrap コンポーネントをインストールする必要があります。Galaxy の参照を可能にするには、Application Server IDE コンポーネントをインストールする必要があります。

詳細については、Application Server のマニュアルを参照してください。

Galaxy との通信

ArchestrA を使用すると、Galaxy 全体で名前空間を使用する際に、生産関連のデータを含めたり処理することが可能になります。また、ハイ レベルなビジュアル化、および製造環境で InTouch が実行されているさまざまなノードからのデータ アクセスの管理も可能になります。



スタンドアロン、マネージド、およびパブリッシュ済みの InTouch アプリケーションの比較

以下の表で説明するように、スタンドアロン、マネージド、およびパブリッシュ済みの InTouch アプリケーションにはいくつかの相違点と類似点があります。

	スタンドアロン InTouch アプリケーション	マネージド InTouch アプリケーション	パブリッシュ済み InTouch アプリケーション
アプリケーションの作成	アプリケーションマネージャ	System Platform IDE <ul style="list-style-type: none"> 新しいアプリケーション スタンドアロンアプリケーションのインポート SmartSymbol のインポート 	非サポート
アプリケーションの編集	アプリケーションマネージャから起動した WindowMaker	IDE 内から起動した WindowMaker	非サポート
アプリケーションの削除	アプリケーションマネージャからの削除	InTouchViewApp テンプレートで削除	アプリケーションマネージャからの削除

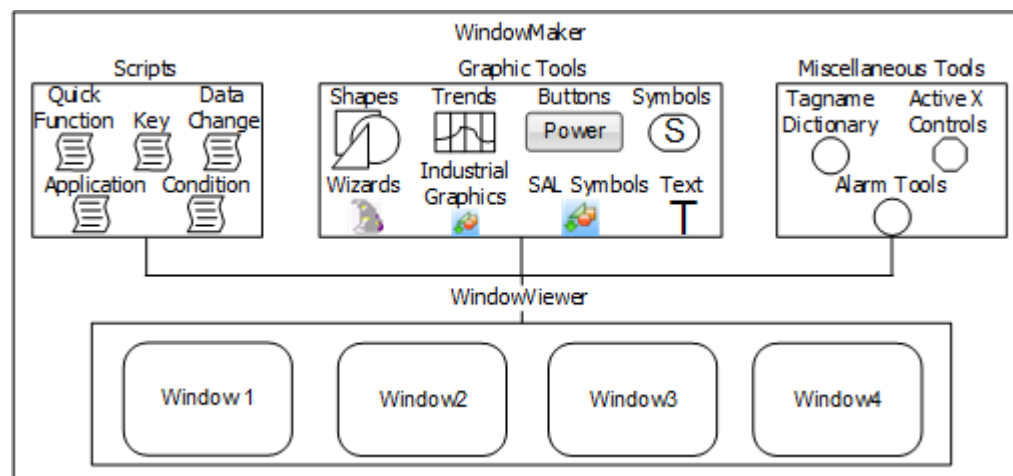
	スタンドアロン InTouch アプリケー ション	マネージド InTouch アプリケー ション	パブリッシュ済み InTouch アプリケーシ ョン
産業用グラフィックのサポート（シンボルウィザードを含む）	サポート	すべての操作でサポート	表示のみサポート、作成および編集は非サポート
DB ダンプと DB ロードのサポート	サポート（アプリケーションマネージャ内の関数）	IDE 内の関数でサポート	サポート（アプリケーションマネージャ内の関数）
元の解像度でのアプリケーションの編集における変換	サポート	非サポート	サポート
分散アプリケーションの管理	ネットワークアプリケーション開発（NAD）	System Platform IDE	ネットワークアプリケーション開発（NAD）
新しい InTouch アプリケーションバージョンが受け入れられる方法の設定	アプリケーションマネージャで設定（ネットワークアプリケーション開発）	WindowMaker で設定	アプリケーションマネージャで設定（ネットワークアプリケーション開発）
アプリケーションのテストで高速切り替えの使用	サポート	サポート	サポート
タグ変数の値およびタグ変数パラメータの保持の使用	サポート	サポート（ローカルディレクトリの設定も必要）	サポート

アプリケーションの構築

WindowMaker では、アプリケーションのウィンドウに表示されるオブジェクトの動作を定義するグラフィック ツール、スクリプト言語、およびタグ変数管理ユーティリティが提供されます。WindowMaker を使用すると、ウィンドウ オブジェクトに関連付けられているデータ ポイントを表すタグ変数を作成できます。製造工程のデータは、最終的にはタグ変数の値として関連付けられます。このタグ変数デー

タは、アラームの監視、トレンドの作成、およびランタイム中のアプリケーションの動作を決定するためにアプリケーションで使用できます。

以下の図は、InTouch HMI アプリケーションを作成するためのいくつかの WindowMaker のツールを示しています。



より複雑なオブジェクトを作成するために組み合わせることができるシンプルな図形から、事前定義されたプロパティを持つ標準の産業用グラフィックまで、豊富な種類のグラフィック ツールを使用できます。トリガのメカニズムに基づいて、異なるタイプのスクリプトを作成できます。また、事前定義された InTouch 関数をスクリプトに挿入することもできます。タグ変数がいつ通常の状態またはアラーム状況になるのかを決定するタグ変数ディクショナリを使用すると、さまざまなタグ変数の値のしきい値を定義できます。

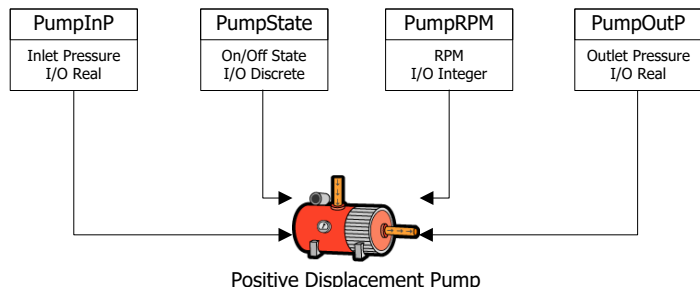
アプリケーションの実行

WindowViewer を使用して、すべてのタイプの InTouch アプリケーションを実行します。System Platform IDE からマネージドアプリケーションを配置した後、アプリケーションマネージャを使用して WindowViewer で開きます。豊富な種類のランタイムトリガを使用して、アプリケーションの実行中にスクリプトを開始できます。アプリケーションデータおよびアラームをファイルまたは SQL Server データベースに保存するように、WindowViewer を設定できます。オペレータに WindowViewer へのログオンを要求し、WindowViewer を実行するコンピュータへの変更を行うことを禁止することでセキュリティを強化できます。WindowViewer のメニュー コマンドを選択することにより、オペレータはアプリケーションの履歴のロギングを開始および停止できます。タグ変数データの保存方法および分散方法に基づいて、WindowViewer が実行されているコンピュータを、クライアントまたはサーバーとして機能するよう設定できます。

章 3 タグ

InTouch ヒューマン マシン インターフェイス (HMI) は、製造環境におけるコンポーネントをグラフィック表示するアプリケーションです。このグラフィカル インターフェイスを使用して、工場のオペレータは製造工程を監視および管理することができます。

以下の図は、製造工程のコンポーネントであるポンプの例を示しています。ポンプには、値に関連付けられているプロパティがあります。圧力、RPM、およびステータスはポンプのプロパティであり、その値は HMI から監視されます。

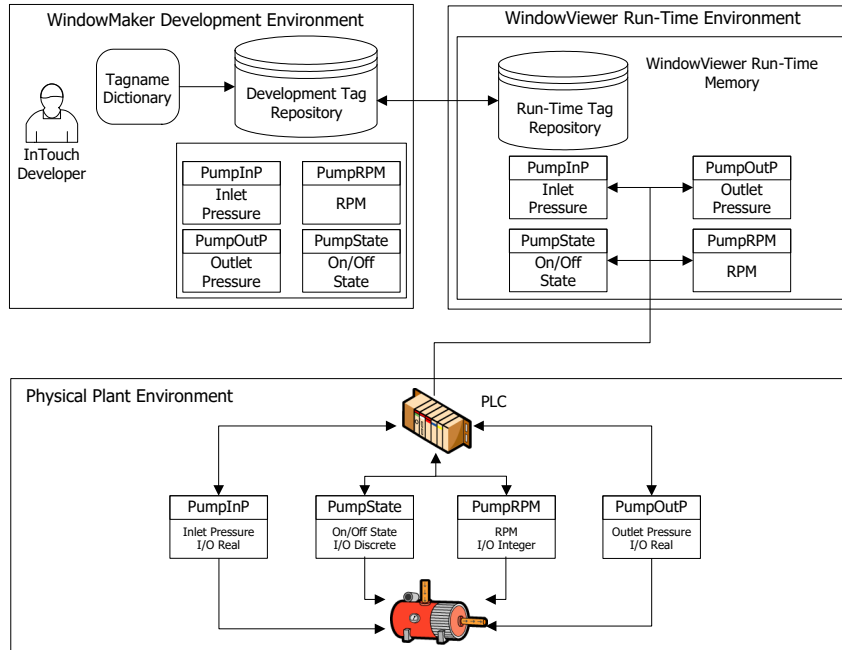


タグは InTouch HMI アプリケーションのデータ アイテムを表します。タグを使用すると、製造環境からの特定のデータ アイテムとして、特定のコンポーネント プロパティをアクセス可能にできます。上の図では、**PumpState** タグはポンプがオンであるかオフであることを示しています。製造環境において、InTouch アプリケーションでプロパティを監視または制御するコンポーネントに対してタグを作成します。

製造コンポーネントから収集された異なるタイプのデータに対しては異なるタイプのタグを使用できます。たとえば、**PumpState** タグは、ポンプが実行中であるか停止中であることを示すブール型のオン/オフ値を返します。アプリケーションの一部とするデータ型に対しては、適切なタイプの InTouch タグを割り当てます。

InTouch タグ変数の操作

InTouch アプリケーションを作成することによって開始します。WindowMaker のツールであるタグ変数ディクショナリを使用して、アプリケーションのタグ変数を定義します。以下の図は、InTouch の開発環境およびランタイム環境を示しています。



タグ変数ディクショナリを使用して、タグ変数の名前とタイプを割り当てます。タグ変数のタイプによっては、タグ変数ディクショナリのその他のオプションを使用して、タグ変数の追加のプロパティを指定できます。たとえば、I/O タイプのタグ変数には、リモートデータソースへの接続を指定するための追加オプションが含まれています。

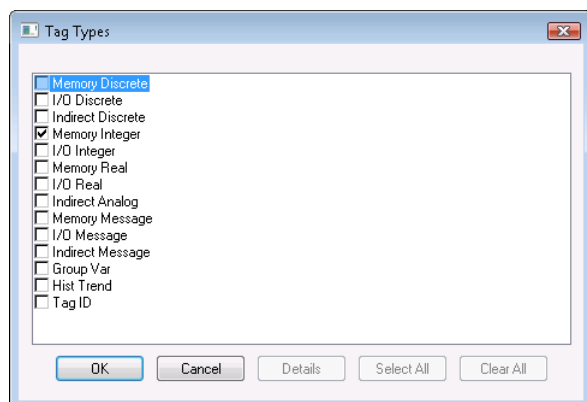
InTouch アプリケーションは、WindowViewer 環境で動作します。WindowViewer がアプリケーションを起動すると、開発リポジトリからタグ変数を読み取り、ランタイムメモリに配置します。

InTouch アプリケーションは、アニメーションリンクやスクリプトを使用して、ランタイムメモリに配置されたタグ変数と通信します。InTouch アプリケーションは、タグ変数に割り当てられたコンポーネントプロパティから現在値とその他のステータス情報を追跡します。

InTouch タグ変数のタイプ

タグ変数を定義する場合、そのタグ変数に関連付けられた処理データに応じて、特定のタイプを割り当てます。たとえば、タグ変数がポンプの RPM を表示する場合、このタグ変数を整数型として割り当てます。

タグ変数ディクショナリで、[タグ変数タイプ] ダイアログボックスを使用して、作成したタグ変数にタグ変数タイプを割り当てます。



タグ変数タイプを割り当てると、タグ変数ディクショナリでは、選択したタグ変数のタイプに対して、特定のオプションが一覧表示されます。

メモリ型タグ

メモリ型タグは、InTouch アプリケーション内の内部システム定数や変数を定義します。たとえば、実数 3.414 を内部定数として定義できます。プロセス シミュレーションでは、カウンタとして動作することによって、メモリ型タグは QuickScript のバックグラウンドのアクションを制御できます。タグ変数に関連付けられているカウントに基づき、QuickScript はさまざまなアニメーション効果を起動できます。また、メモリ型タグは、他のプログラムによってアクセスされる計算済みの変数としても動作できます。

メモリ型タグのタイプは、タグに関連付けられた処理データに基づいて、次の 4 種類から選択します。

- **メモリ論理型**

メモリ論理型タグ変数は、プロセス コンポーネントの状態プロパティに関連付けられています。メモリ論理型タグ変数に割り当てられた値は、考えられる次のような 2 つのブール型の状態で示されます。

- 0 または 1
- False または True
- On または Off
- High または Low

- **メモリ整数型（アナログ）**

メモリ整数型タグ変数には、-2,147,483,648 ～ 2,147,483,647 の範囲の 32 ビット符号付き整数値を割り当てることができます。

- **メモリ実数型（アナログ）**

メモリ整数型タグ変数には、 $-3.4 \times 10^{38} \sim 3.4 \times 10^{38}$ の範囲の浮動小数点数を指定できます。2 つの実数型タグが比較されると、2 つの実数型タグの差分は FLT_EPSILON（値 1.19209290E-07F、10 進数では 0.0000001192092896）よりも大きくなる必要があります。浮動小数点の計算はすべて 64 ビットの解像度で行われますが、その結果は 32 ビットの 10 進数で保存されます。実数の最大の精度については、「[IEEE 小数ユニット](#)」を参照してください。

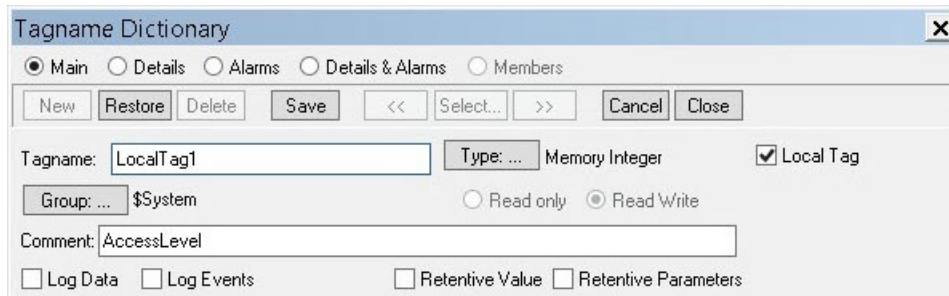
- **メモリ メッセージ型**

メモリ メッセージ型タグには、半角で最大 131 文字のテキスト文字列を割り当てることができます。

ローカル タグ

ローカル タグを使用すると、Web Client で使用するセッションごとのメモリ タグを作成できます。たとえば、ナビゲーションに使用するローカル メモリ タグを設定できます。ランタイム時、ユーザーがそのタグにリンクされているナビゲーション シンボルをクリックすると、各セッションは独立して動作します。

注記: ドット フィールドはローカル タグでサポートされていません。



ローカル タグを作成するには:

- メモリ タグを作成し、[ローカル タグ] チェック ボックスをオンにします。その他のタグ プロパティを設定します。新しいタグの作成の詳細については、「[新しいタグの作成](#)」を参照してください。
- [保存] をクリックします。

I/O タグ変数

I/O 型タグ変数は、外部ソースに対して、InTouch アプリケーションのデータの読み取りまたは書き込みを行います。外部データには、プログラマブル コントローラ、プロセス コンピュータ、および各ネットワーク ノードからの入力および出力が含まれています。I/O 型タグ変数のデータ値は、以下のプロトコルを介してリモートでアクセスされます。

- Microsoft Dynamic Data Exchange (DDE)
- SuiteLink™

I/O 型タグ変数の値がランタイム メモリで変更されると、InTouch HMI によってリモート アプリケーションが更新されます。反対に、対応するデータ アイテムの値がリモート アプリケーションで変更されるたびに、InTouch の I/O 型タグ変数の値は更新されます。

InTouch HMI には、タグ変数に関連付けられているプロセス データに基づいて、次の 4 種類の I/O 型タグ変数が用意されています。これら 4 種類の I/O 型タグは、メモリ型タグに類似しています。

• I/O 論理型

I/O 論理型タグ変数は、コンポーネント プロセスのプロパティに関連付けられており、値は考えられる次のような 2 つの状態で表されます。

- 0 または 1
- False または True
- On または Off

- High または Low
- **I/O 整数型** (アナログ)

I/O 整数型タグ変数には、-2,147,483,648 ～ 2,147,483,647 の範囲の 32 ビット符号付き整数値を割り当てることができます。
- **I/O 実数型** (アナログ)

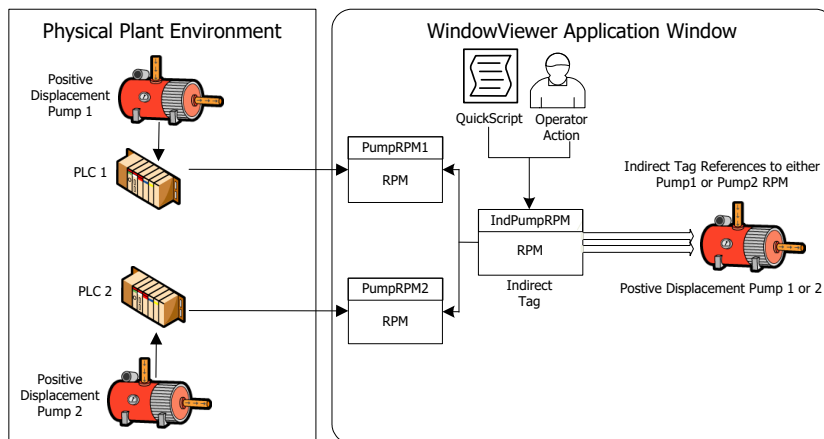
I/O 実数型タグ変数には、 $-3.4 \times 1038 \sim 3.4 \times 1038$ の範囲の浮動小数点数を割り当てることができます。2 つの実数型タグが比較されると、2 つの実数型タグの差分は **FLT_EPSILON** (値 1.19209290E-07F、10 進数では 0.0000001192092896) よりも大きくなる必要があります。I/O 実数型タグの浮動小数点の計算はすべて 64 ビットの解像度で行われますが、その結果は 32 ビットの数字で保存されます。I/O 実数の最大の精度については、「[IEEE 小数ユニット](#)」を参照してください。
- **I/O メッセージ型**

I/O メッセージ型タグには、半角で最大 131 文字のテキスト文字列を割り当てることができます。

間接型タグ変数

間接型タグ変数は、他のタグ変数に対して「ポインタ」として動作します。たとえば、単一の InTouch ウィンドウを作成し、複数の異なる一連のタグ変数の間接型タグ変数を使用して、データを表示することができます。

以下の図は、複数のポンプを表示できるアプリケーション ウィンドウの例を示しています。ポンプごとに個別のウィンドウを作成する代わりに、1 つのウィンドウで間接型タグ変数を使用して、個々のポンプに関連付けられた異なるソース タグ変数の値を表示できます。



QuickScript またはオペレータのアクションによって、間接型タグ変数がソース タグ変数に指定されます。たとえば、次のスクリプト ステートメントは、**PumpNo** タグ変数の値に基づいて、2 つの **PumpRPM** タグ変数を **IndPumpRPM** と呼ばれる間接アナログ型タグ変数に割り当てます。

```
IF PumpNo == 1 THEN
    IndPumpRPM.Name = "PumpRPM1";
ELSE
    IndPumpRPM.Name = "PumpRPM2";
ENDIF;
```

間接型タグ変数を別のソース タグ変数と等式化すると、間接型タグ変数はソース タグ変数のように動作します。元のソースに関連付けられた値と間接型の重複タグ変数は、共に同期しています。ソース タ

グ変数の値が変化すると、間接型タグ変数の値はその変化を反映します。また、間接型タグ変数の値が変化すると、それに対応してソースタグ変数の値も変化します。

論理型、アナログ型、メッセージ型の間接型タグ変数を使用できます。この 3 種類の間接型タグ変数は、メモリ型タグ変数や I/O 型タグ変数に類似しています。

間接型タグ変数の詳細については、「[間接型タグ変数の定義](#)」を参照してください。

その他のタグ変数

特定の限られた用途に設計された、その他のタイプの InTouch タグ変数を使用できます。これらのタグを使用すると、動的なアラーム表示の作成、履歴トレンドの作成、および履歴トレンドペンに割り当てられたタグの変更を行うことができます。

履歴トレンドタグ変数

履歴トレンドタグ変数は、履歴トレンドグラフを参照するために使用できます。履歴トレンドに関連付けられたすべてのドットフィールドは、**履歴トレンド型タグ変数**に適用できます。

履歴トレンドタグ変数の定義および使用方法については、「[タグデータのトレンド](#)」を参照してください。

タグ変数 ID タグ変数

タグ変数 ID タグ変数は、InTouch 履歴トレンドグラフに値がプロットされているタグ変数からの情報を取得します。一般的に、タグ変数 ID タグ変数を使用して、特定のトレンドペンに割り当てられたタグ変数の名前を表示したり、トレンドペンに割り当てられたタグ変数を変更したりします。

QuickScript にステートメントを挿入し、タグ変数 ID 型タグ変数を使用して、新しいタグ変数を任意の履歴トレンドのペンに割り当てることができます。たとえば、次の QuickScript ステートメントは、履歴トレンドペンに関連付けられたタグ変数を変更します。

```
HistTrend.Pen1=MyLoggedTag.TagID;
```

この QuickScript を実行すると、履歴トレンドの Pen1 は MyLoggedTag に対してログ記録されたデータの履歴トレンドの作成を開始します。

履歴トレンドタグ変数の定義および使用方法については、「[履歴トレンドウィザードの使用](#)」を参照してください。

スーパータグ

スーパータグは、関連するタグのセットを含むテンプレートです。たとえば、ポンプのすべてのプロパティに割り当てられたタグセットを含むスーパータグテンプレートを作成できます。

生産工程で同一の装置がある場合は、スーパータグを使用します。設備ごとにタグセットを作成するのではなく、スーパータグテンプレートのインスタンスを同じ工程の各アイテムに割り当てます。

スーパータグの詳細については、「[再使用可能なタグ構造の定義](#)」を参照してください。

旧版のタグ変数

グループ変数タグ変数を使用すると、InTouch の標準アラームシステムを使用して、動的なアラーム表示、ディスクへの動的ログ記録、および動的印刷を作成できます。グループ変数タグ変数は、InTouch バージョン 7.11 以前のソフトウェアで開発されたアプリケーションとの下位互換性用のためだけに含まれています。InTouch バージョン 7.11 以降で開発されたアプリケーションでは、グループ変数タグ変数を使用しないでください。

システム タグ変数

システム タグ変数の使用によって、システム関連情報や InTouch スクリプト用の日付や時間などの標準関数が提供されます。システム タグ変数はすべてのアプリケーションの一部です。また、実行中のアプリケーションの以下のような動作を管理するために、スクリプトでシステム タグ変数を使用することもできます。

- アプリケーションセキュリティの監視と管理。
- リモート ノードへの I/O 通信の確立。
- アラームが発生した場合の検出。
- 履歴のロギングの開始と終了。
- NAD を使用した、新しいバージョンへのアプリケーションの更新。

タグ名ディクショナリでは、システム タグはドル記号 (\$) がタグ名の最初の文字として識別されます。システム タグ変数は削除できません。システム タグ変数に関連付けられたコメントのみが変更できます。

システム タグ変数リファレンス

以下の表は、InTouch システム タグ変数を説明しています。

システム タグ変数	説明	関連情報
\$AccessLevel	現在ログオンしているオペレータに関連付けられているアクセス レベルを指定する読み取り専用の整数型タグ変数。 この情報は、オペレータによる特定の InTouch 関数へのアクセスを制御するアニメーション リンクまたはスクリプトで使用できます。	「 InTouch のセキュリティ保護 」を参照してください。
\$ApplicationChanged	マスター アプリケーションが NAD 環境で変更されたかどうかを示す読み取り専用の論理型タグ変数。	「アプリケーションの分散」を参照してください。
\$ApplicationVersion	WindowViewer で実行中のアプリケーションの現在のバージョンを示す読み取り専用の実数型タグ変数。	「アプリケーションの分散」を参照してください。

システム タグ変数	説明	関連情報
\$ChangePassword	1に設定されると「パスワードの変更」ダイアログボックスを示す書き込み専用の論理型タグ変数。	「 InTouch のセキュリティ保護 」を参照してください。
\$ConfigureUsers	セキュリティ ユーザー名リストを編集するための「ユーザーの設定」ダイアログボックスを表示する書き込み専用の論理型タグ変数。	「 InTouch のセキュリティ保護 」を参照してください。
\$Date	1970 年 1 月 1 日から経過した日数の整数を表示する読み取り専用の整数型タグ変数。	「 組み込み関数 」を参照してください。
\$DateString	Windows の「地域と言語のオプション」ダイアログボックスで指定されているものと同じ形式で日付を表す読み取り専用のメッセージ型タグ。	「 組み込み関数 」を参照してください。
\$DateTime	1970 年 1 月 1 日から経過した日数の少数を表示する読み取り専用の実数型タグ変数。	「 組み込み関数 」を参照してください。
\$Day	現在の日付（1 ～ 31）を表示する読み取り専用の整数型タグ変数。	「 組み込み関数 」を参照してください。
\$False	式内で FALSE 値を返す論理型読み取り専用タグ変数。 \$False システム タグ変数は、アプリケーションを以前のバージョンの InTouch を現在のバージョンに更新する際に旧版のタグ変数のインスタンスを置き換えるために使用します。	追加情報はありません。
\$HistoricalLogging	InTouch アプリケーションの実行中の履歴ログ記録を開始または停止するために使用される読み取りまたは書き込みの論理型タグ変数。	「 タグ値の記録 」を参照してください。

システム タグ変数	説明	関連情報
\$Hour	現在の時間を 0 ～ 23 の値で表示する読み取り専用の整数型タグ変数。	「 組み込み関数 」を参照してください。
\$InactivityTimeout	ユーザーの無操作時間が経過したことを示す読み取り専用の論理型タグ変数。1 に設定すると、非アクティブ期間が経過するとユーザーが WindowViewer から自動的にログオフされます。	「 InTouch のセキュリティ保護 」を参照してください。
\$InactivityWarning	無操作警告時間が経過したことを示す読み取り専用の論理型タグ変数。 \$InactivityWarning の値は、オペレータへの無操作警告を発行するために使用できます。	「 InTouch のセキュリティ保護 」を参照してください。
\$LogicRunning	アプリケーション スクリプトの開始および停止に使用できる読み取りまたは書き込みの論理型タグ変数。	「 InTouch のセキュリティ保護 」を参照してください。
\$Minute	現在の分 (0 ～ 59) を表示する読み取り専用の整数型タグ変数。	「 組み込み関数 」を参照してください。
\$Month	現在の月 (1 ～ 12) を表示する読み取り専用の整数型タグ変数。	「 組み込み関数 」を参照してください。
\$Msec	現在のミリ秒 (0 ～ 999) を表示する読み取り専用の整数型タグ変数。	「 組み込み関数 」を参照してください。
\$NewAlarm	新しいローカルアラームが発生したことを示す読み取りまたは書き込みの論理型タグ変数。	「 ランタイム時のタグとグループのアラームプロパティの制御 」を参照してください。

システム タグ変数	説明	関連情報
\$ObjHor	画面上で選択したオブジェクトの中心の水平ピクセル位置を表示する読み取り専用の整数型タグ変数。	『AVEVA™ InTouch HMI アプリケーション開発者ガイド』の「 オブジェクトのアニメーション化 」を参照してください。
\$ObjVer	画面上で選択したオブジェクトの中心の垂直ピクセル位置を表示する読み取り専用の整数型タグ変数。	『AVEVA™ InTouch HMI アプリケーション開発者ガイド』の「 オブジェクトのアニメーション化 」を参照してください。
\$Operator	InTouch アプリケーションにログオンしたオペレータの名前を表示する読み取り専用のメッセージ型タグ。	「 InTouch のセキュリティ保護 」を参照してください。
\$OperatorDomain	アプリケーションがオペレーティング システム ベースのセキュリティで保護されている場合に、ログオン時に指定されたドメイン名またはマシン名を含む読み取り専用のメッセージ型タグ。	「 InTouch のセキュリティ保護 」を参照してください。
\$OperatorDomainEntered	<p>InTouch アプリケーションへのログイン試行を行うオペレータのドメインが割り当てられた書き込み専用のメッセージ型タグ。</p> <p>ログイン試行は、\$PasswordEntered システムタグ変数に値を割り当てるまで開始されません。</p>	「 InTouch のセキュリティ保護 」を参照してください。
\$OperatorEntered	<p>InTouch アプリケーションへのログイン試行を行うオペレータのユーザー アカウント名が割り当てられた読み取り/書き込みメッセージ型タグ。</p> <p>ログイン試行は、\$PasswordEntered システムタグ変数に値を割り当てるまで開始されません。</p>	「 InTouch のセキュリティ保護 」を参照してください。
\$OperatorName	オペレーティング システム ベースまたは Archestra® 認証が使用される場合に、オペレータのフルネームを表示する読み取り専用のメッセージ型タグ。	「 InTouch のセキュリティ保護 」を参照してください。

システム タグ変数	説明	関連情報
\$PasswordEntered	InTouch アプリケーションへのログイン試 行を行うオペレータのパスワードが割り 当てられた書き込みのメッセージ型タグ。 このタグ変数に値を書き込むと、 \$OperatorDomainEntered、 \$OperatorEntered、および \$PasswordEntered システム タグ変数の値を使用してログイン 試行が開始されます。	「InTouch のセキュリテ ィ保護」 を参照してく ださい。
\$Second	現在の秒 (0 ～ 59) を表示する読み取り専 用の整数型タグ変数。	「組み込み関数」 を参照 してください。
\$StartDdeConversations	初期化されていない通信をランタイムで 開始するために使用される読み取りまた は書き込みの論理型タグ変数	「I/O を使用したデー タアクセス」 を参照して ください。
\$System	ルート アラーム グループを識別する読み 取り専用のタグ変数。	「アラームとイベント の概要」 を参照してく ださい。
\$Time	現在の日付の午前 0 時からの経過時間を ミリ秒単位で表示する読み取り専用の整 数型タグ変数。	「組み込み関数」 を参照 してください。
\$TimeString	Windows の [地域と言語のオプション] タ イアログ ボックスで指定されているもの と同じ形式で現在の時刻を表示する読み 取り専用のメッセージ型タグ。	「組み込み関数」 を参照 してください。
\$VerifiedUserName	確認されたユーザーのフル ネームまたは Null が含まれる読み取り専用のメッセー ジ型タグ。	「InTouch のセキュリテ ィ保護」 を参照してく ださい。
\$Year	現在の年を 4 桁の数字で表示する読み取 り専用の整数型タグ変数。	「組み込み関数」 を参照 してください。

タグ プロパティ

InTouch タグの各タイプには、そのタグに関連付けられたデータの特性を説明するプロパティ セットがあります。InTouch タグに関連付けられた 4 種類の主要データ型は、以下のとおりです。

- 論理値
- 整数
- 実数
- テキスト メッセージ

タグ名ディクショナリでタグを作成すると、タグのすべてのプロパティには初期値が設定されます。以下の図は、I/O 整数型タグのプロパティを示しています。

The screenshot shows the 'Tagname Dictionary' dialog box with the 'Main' tab selected. The tag name is 'PumpRPM', type is 'I/O Integer', and it is not a 'Local Tag'. The group is '\$System' and access is 'Read/Write'. The comment is 'AccessLevel'. There are checkboxes for 'Log Data', 'Log Events', 'Retentive Value', and 'Retentive Parameters'. The initial value is 0, and the deadband is 0. The minimum and maximum values are -32768 and 32767 respectively. The unit is 'RPM'. The conversion is set to 'Linear'. The alarm model is 'Condition'.

Alarm Value	Priority	Alarm Inhibitor	Value Deadband
LoLo	0	1	0
Low	0	1	0
High	0	1	0
HiHi	0	1	0

% Deviation	Target	Priority	Alarm Inhibitor	Deviation Deadband %
Minor Deviation	0	1	0	0
Major Deviation	0	1	0	0

Rate of Change: 0 % per: Sec Min Hr Priority: 1 Alarm Inhibitor: 0

タグ名ディクショナリからタグ プロパティの初期値を設定すると、ほとんどのタグのプロパティはドットフィールドを使用して動的に変更できます。ドットフィールドは、InTouch アプリケーションの稼動中に、スクリプトで監視または変更できるタグ プロパティを識別します。ドットフィールドは、スクリプトのタグの名前に追加します。

ドットフィールドを使用したタグ プロパティの動的な変更の詳細については、「[タグ ドットフィールドを使用したタグ プロパティの表示または変更](#)」を参照してください。

メモリ型タグ変数のプロパティ

次の表は、メモリ型タグ変数の 4 種類のタイプのプロパティを一覧表示しています。各プロパティは、タグ変数ディクショナリのオプションとして選択または変更できます。詳細については、「[新しいタグの作成](#)」を参照してください。

タグ変数のプロパティ	論理型	整数	メッセージ	実数
% 偏差		•		•
% /		•		•
確認モデル	•	•		•
アラーム コメント	•	•	•	•
アラーム グループ	•	•	•	•
アラーム抑止タグ変数	•	•		•
アラーム状況	•			
アラーム値		•		•
コメント	•	•	•	•
デッドバンド		•		•
工学単位		•		•
偏差デッドバンド %		•		•
Hi		•		•
HiHi		•		•
初期値	•	•	•	•
データ ログ	•	•		•
ログ デッドバンド		•		•
イベント ログ	•	•	•	•
Lo		•		•
LoLo		•		•
最大文字数			•	
大偏差		•		•
最大値		•		•
最小値		•		•
小偏差		•		•
オフ メッセージ	•			
オン メッセージ	•			

タグ変数のプロパティ	論理型	整数	メッセージ	実数
優先度	•	•	•	•
変化率		•		•
読み取り専用	•	•	•	•
読み書き	•	•	•	•
パラメータ値保持		•		•
変数値保持	•	•	•	•
ターゲット		•		•
デッドバンド値		•		•

I/O 型タグ変数のプロパティ

メモリ型タグ変数と同じように、I/O 型タグ変数のプロパティもタグ変数ディクショナリのオプションとして選択または変更できます。詳細については、「[新しいタグの作成](#)」を参照してください。

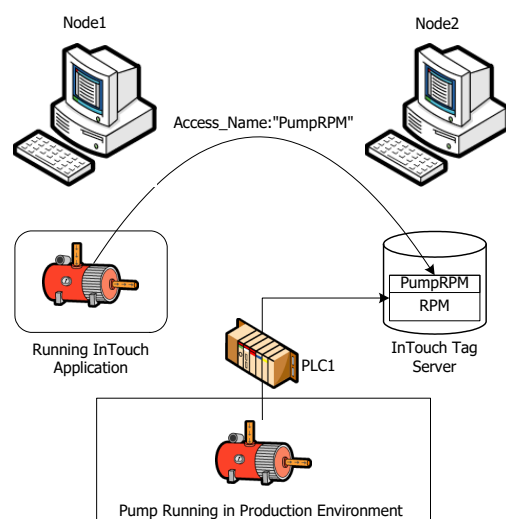
タグ変数のプロパティ	論理型	整数	メッセージ	実数
% 偏差		•		•
% /		•		•
アクセス名	•	•	•	•
確認モデル	•	•		•
アラーム コメント	•	•	•	•
アラーム グループ	•	•	•	•
アラーム抑止タグ変数	•	•		•
アラーム状況	•			
アラーム値		•		•
コメント	•	•	•	•
変換方法		•		•
デッドバンド		•		•
偏差デッドバンド %		•		•
工学単位		•		•

タグ変数のプロパティ	論理型	整数	メッセージ	実数
Hi		•		•
HiHi		•		•
初期値	•	•	•	•
入力データの変換	•			
項目	•	•	•	•
データ ログ	•	•		•
ログ デッドバンド		•		•
イベント ログ	•	•	•	•
Lo		•		•
LoLo		•		•
最大文字数			•	
大偏差		•		•
工学値最大値		•		•
生データ最大値		•		•
最大値		•		•
工学値最小値		•		•
生データ最小値		•		•
最小値		•		•
小偏差		•		•
オフ メッセージ	•			
オン メッセージ	•			
優先度	•	•	•	•
変化率		•		•
書き込み禁止	•	•	•	•
読み書き	•	•	•	•
パラメータ値保持		•		•
変数値保持	•	•	•	•

タグ変数のプロパティ	論理型	整数	メッセージ	実数
二乗根変換		•		•
ターゲット		•		•
タグ変数名をアイテム名として使用	•	•	•	•
ディッドバンド		•		•

リモート タグ リファレンス

InTouch アプリケーションが稼動しているノードとは別のノード上で稼動しているタグ変数サーバーで、分散 InTouch アプリケーションを作成できます。以下の図は、別のノードで稼動しているタグ変数サーバーから PumpRPM タグ変数へのリモート リファレンスを作成する InTouch アプリケーションを示しています。



リモート ノードにあるタグ変数を参照する InTouch アプリケーションを作成するには、次の 2 つの方法があります。

- I/O 型タグ変数に、リモート サーバーをタグ変数ソースとして識別するアクセス名を関連付ける。I/O 型タグ変数のアクセス名の定義については、「[アクセス名の設定](#)」を参照してください。
- タグ変数を直接参照するリモート リファレンスを使用する。たとえば、PLC1:PumpRPM です。

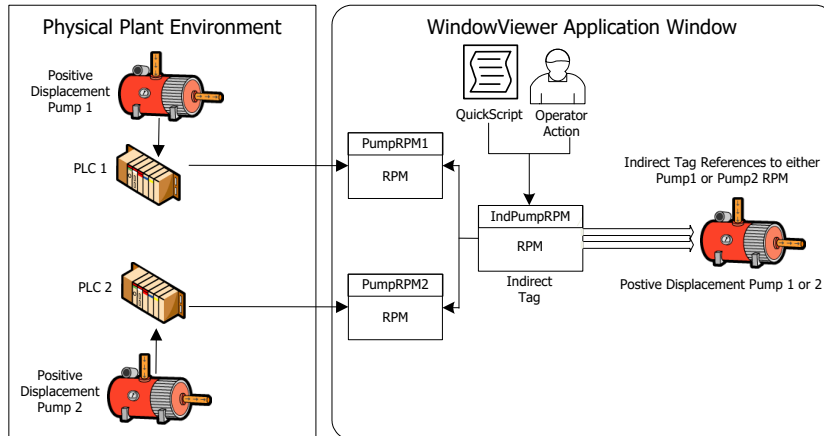
詳細については、「[リモート参照による I/O データアクセス](#)」を参照してください。

間接型タグ変数の定義

間接型タグ変数を使用して、複数のタグ変数からの値を表示するウィンドウ オブジェクトを持つアプリケーションを作成できます。

以下の図は、アプリケーション ウィンドウ内のポンプ オブジェクトを示しています。ポンプ オブジェクトは、間接型タグ変数から設定された値に基づいて可能な 2 つのプロセス ポンプを表示します。

QuickScript またはオペレータの操作によって、間接型タグ変数に関連付けられたソース タグ変数が選択されます。



間接型タグ変数によって、開発時間が最小化されます。運用環境で実行中の複数のプロセスを単一のウィンドウオブジェクトで表すことができるため、作成するアプリケーションウィンドウの数を少なくできます。

間接型タグ変数とスクリプトの使用

スクリプトを使用すると、入力ソース タグ変数を間接型タグ変数に割り当てることができます。入力ソース タグ変数を間接型タグ変数に割り当てするには、ソース タグ変数の名前を間接型タグ数の **.Name** ドットフィールドに割り当てます。

たとえば、**IndPumpRPM** という名前の間接アナログ型タグ変数を作成する場合、以下の例のようなスクリプトステートメントを使用して、2つのソース **PumpRPM** タグ変数が **IndPumpRPM** に割り当てられます。

```
IF PumpNo == 1 THEN
    IndPumpRPM.Name = "PumpRPM1";
ELSE
    IndPumpRPM.Name = "PumpRPM2";
ENDIF;
```

間接型タグ変数の割り当てスクリプトは、アプリケーションイベントまたはウィンドウ ボタンをクリックするなどのオペレータによる操作によってトリガできます。

間接型タグ変数を別のソース タグ変数と等式化すると、間接型タグ変数はソース タグ変数のように動作します。ソース タグ変数の値が変化すると、間接型タグ変数の値はその変化を反映します。また、間接型タグ変数の値が変化すると、それに対応してソース タグ変数の値も変化します。

間接型タグ変数の **.Name** ドットフィールドは簡単な文字列であるため、ランタイムで間接型タグ変数を動的に割り当てることができます。たとえば、**Number** タグ変数の値が変わるたびに実行されるデータ変化 QuickScript を作成すると、その変化に応じて間接型タグ変数 **IndPumpRPM** に割り当てられるソース タグ変数が変化します。

```
IndPumpRPM.Name = "PumpRPM" + Text(Number, "#");
```

このスクリプトが実行されると、アナログ型タグ変数 **Number** の値がテキストに変換されて、**PumpRPM** 文字列に追加されます。**Number** が **1** の場合には、間接型タグ変数 **IndPumpRPM** の名前は **PumpRPM1** に設定されます。

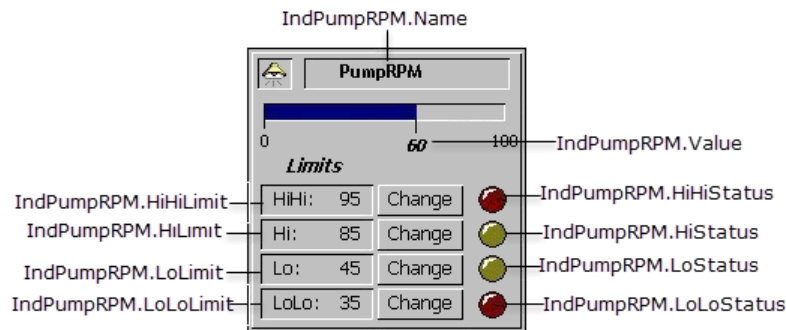
間接アナログ型タグ変数は、整数型と実数型のタグ変数の両方に使用できます。タグ変数のタイプが同じであれば、他のどのタグ変数にも間接型タグ変数をマップできます。

また、間接型タグ変数に保持属性を割り当てることもできます。保持属性を使用すると、アプリケーションが再起動されたときに、間接型タグ変数では最後のタグ変数割り当てが保持された状態となります。

間接型タグ変数とローカル タグ変数の使用

一般的に、間接型タグ変数はローカルのタグ変数ディクショナリで定義されたタグ変数と共に使用されます。間接型ローカルタグ変数を使用すると、ローカルタグ変数の複数の属性を表示する視覚的なオブジェクトを作成できます。たとえば、アプリケーション ウィンドウにフェイスプレートを作成できます。フェイスプレートには、異なるドットフィールドに割り当てられた間接型ローカルタグ変数にリンクする、選択可能なアイテムが含まれています。以下の例では、オペレータは間接型ローカルタグ変数 **PumpRPM** にリンクしているドットフィールドのアラームしきい値を変更できます。

以下の図は、ポンプ RPM のアラームしきい値へのアニメーションリンクを持つフェイスプレートを示しています。間接型タグ変数は、アラームしきい値フェイスプレートに対して異なるローカルタグ変数の属性を割り当てます。



フェイスプレートを適切なタグ変数にリダイレクトするには、QuickScript 内にステートメントを含めます。

```
Indirect_tag_name.Name = "tag_name";
```

このスクリプトの例では、**tag_name** はローカルのタグ変数ディクショナリで定義された実際のタグ変数の名前です。このスクリプトが実行されると、このローカルタグ変数に関連付けられているすべてのドットフィールド値は、間接型タグ変数を介してアプリケーション オブジェクトにアクセスできるようになります。

間接型タグとリモート参照の使用

リモート間接型タグ参照は、ローカルタグ参照とは異なります。リモート参照の構文は以下のとおりです。

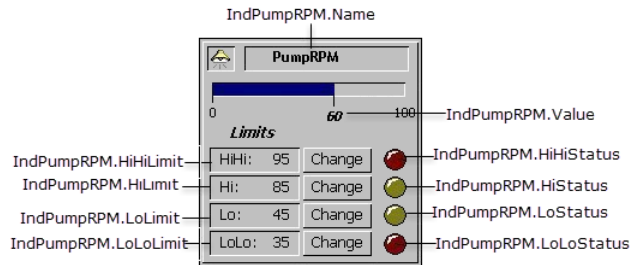
AccessName:Item

各項目の説明

- **AccessName** は、任意の有効な InTouch アクセス名です。
- **Item** は、アクセス名の定義で指定した、I/O サーバーによってサポートされる任意の有効なアイテム名です。

リモート参照を使用すると、サーバーはタグ構造ではなく値をクライアントに返します。この値には、時刻スタンプと品質スタンプが含まれています。そのため、リモート参照に割り当てられている間接型タグは、値、時刻、品質に関するもの以外、タグ ドットフィールドにアクセスすることはできません。たとえば、間接型タグは、アラームしきい値を指定するために、リモート参照を通じてタグ属性にアクセスすることはできません。

考えられる解決策の 1 つは、間接型タグのセットを使用してフェイスプレートを作成することです。以下の図は、ポンプのアラームしきい値を変更するフェイスプレートを示しています。



この例では、黙示的な **.Value** ドットフィールドに関連付けられた 10 個の間接型タグがフェイスプレートで使用されています。アラーム フェイスプレートは、**TagServer1** というリモート InTouch ノード上のリモート参照タグ **IndPumpRPM** にリダイレクトされています。InTouch アクセス名は以下のように設定されます。

アクセス名 : TagSource1

ノード名 : TagServer1

アプリケーション名: View

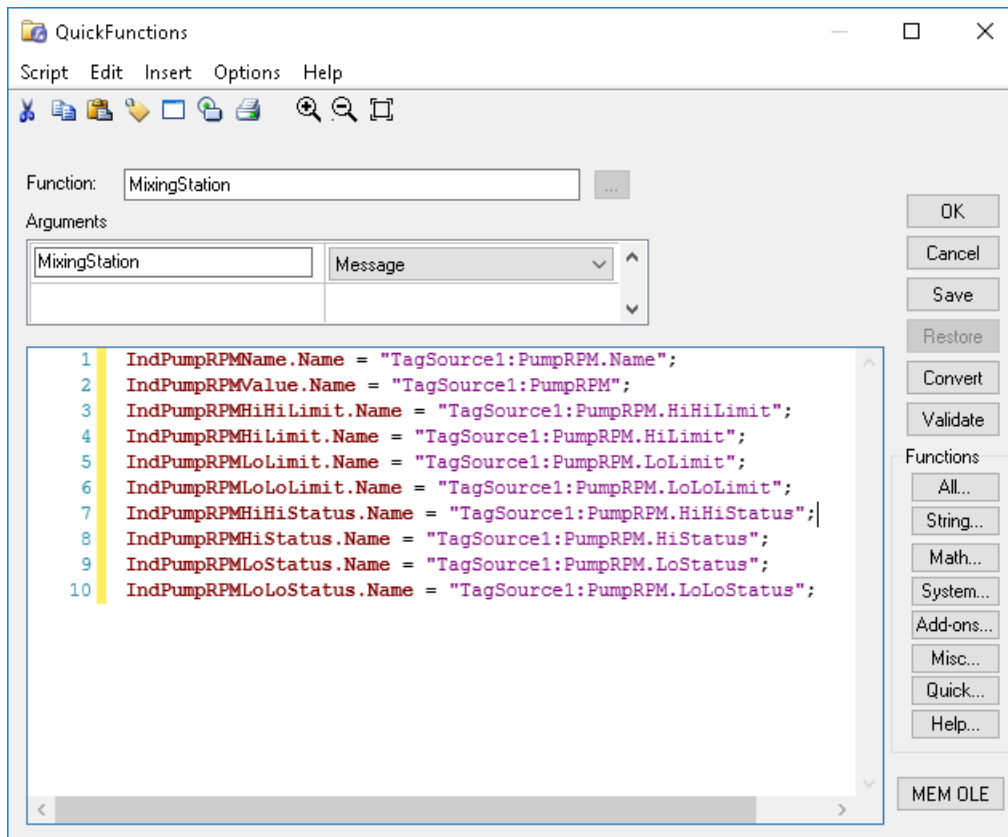
トピック名 : Tagname

フェイスプレートをリモート参照タグ **PumpRPM** にリダイレクトするには、以下の **QuickScript** を実行します。

```
IndPumpRPMName.Name = "TagSource1:PumpRPM.Name";
IndPumpRPMValue.Name = "TagSource1:PumpRPM";
IndPumpRPMHiHiLimit.Name = "TagSource1:PumpRPM.HiHiLimit";
IndPumpRPMHiLimit.Name = "TagSource1:PumpRPM.HiLimit";
IndPumpRPMLoLimit.Name = "TagSource1:PumpRPM.LoLimit";
IndPumpRPMLoLoLimit.Name = "TagSource1:PumpRPM.LoLoLimit";
IndPumpRPMHiHiStatus.Name = "TagSource1:PumpRPM.HiHiStatus";
IndPumpRPMHiStatus.Name = "TagSource1:PumpRPM.HiStatus";
IndPumpRPMLoStatus.Name = "TagSource1:PumpRPM.LoStatus";
IndPumpRPMLoLoStatus.Name = "TagSource1:PumpRPM.LoLoStatus";
```

このスクリプトは、フェイスプレートをリダイレクトするたびに実行する必要があります。別の解決策は、単独のスクリプトを記述して、そのスクリプトにリモート参照の名前を渡す InTouch クイック関数を作成することです。同じクイック関数を呼び出す複数のフェイスプレートを使用することによって、スクリプトのコーディング作業を軽減できます。

たとえば、類似したスクリプト コマンドのセットを使用して、**RedirectAlarmFacePlate** というクイック関数を定義できます。



RedirectAlarmFacePlate 関数を呼び出すと、リダイレクト処理全体を行えます。この処理を実行するには、この関数が別の InTouch QuickScript によって呼び出される必要があります。次に例を示します。

CALL RedirectAlarmFacePlate ("TagSource1:PumpRPM");

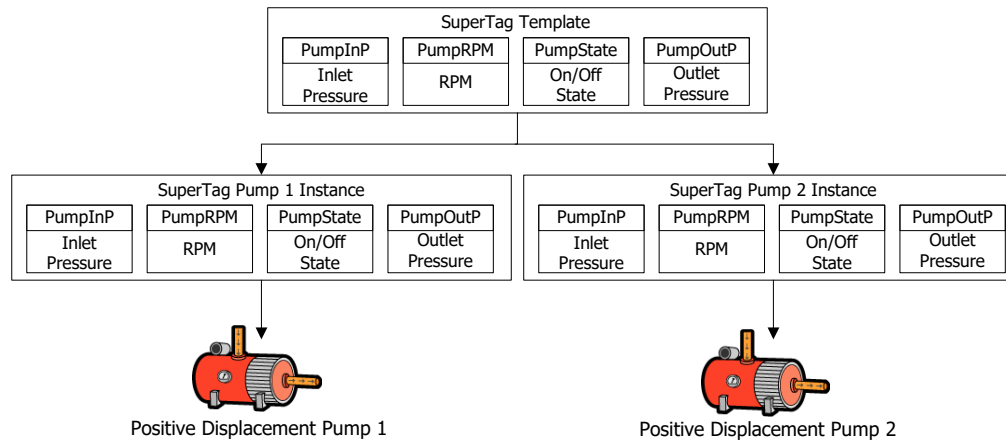
再使用可能なタグ構造の定義

スーパータグは、関連するタグのセットのテンプレートです。スーパータグ テンプレートに属しているタグは、生産工程で 1 つのコンポーネントの共通のプロパティに関連付けられます。

スーパータグを利用することで、開発期間が短縮されます。生産工程でコンポーネントごとにタグのセットを作成する代わりに、1 つのスーパータグ テンプレートを複製し、同じプロパティを持つ工程内のすべてのコンポーネントに個別のインスタンスを作成することができます。

スーパータグ テンプレート

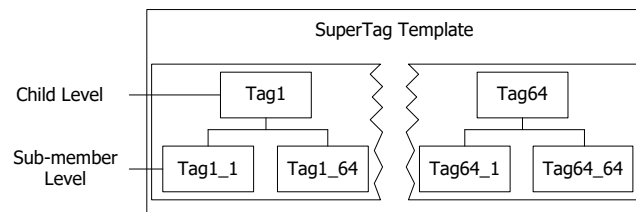
下図に、ポンプのデータに関連付けられた関連タグのセットで構成されるスーパータグ テンプレートを示します。このテンプレートを複製して、プロセスの中ですべての同じポンプに対してスーパータグのインスタンスを作成することができます。



スーパータグのインスタンスに含まれるタグはすべて、普通のタグとまったく同じように機能します。メンバータグのデータ型には、InTouch の論理型、整数型、実数型、メッセージ型を指定することが可能です。タグは、トレンド、アラーム、およびすべてのタグドットフィールドをサポートしています。

スーパータグテンプレートは、メンバーのタグを 2 階層の入れ子構造にまとめることができます。スーパータグテンプレートには、最大 64 個の子タグを組み込むことができます。各子タグも最大 64 個のサブメンバータグを持つことができます。つまり、スーパータグテンプレートには合計 4095 個のタグが入ります。

以下の図は、スーパータグテンプレート内のタグの構造を示しています。



1 つのスーパータグテンプレートの親が別のスーパータグテンプレートに埋め込まれると、子メンバーになります。

スーパータグの親テンプレートを作成すると、タグ名ディクショナリでは **[タグタイプ]** ダイアログボックスにタグタイプとして一覧表示されます。新しいタグを作成した場合、タグタイプとして直ちにそのテンプレートを選択できるようになります。新しく作成したスーパータグタイプを使用するタグを定義するために **WindowMaker** を再起動する必要はありません。

スーパータグテンプレートの保存

デフォルトでは、すべてのスーパータグテンプレートは、**C:\ProgramData\Wonderware\InTouch** フォルダの **Supertag.dat** ファイルに保存されます。スーパータグテンプレートは別の場所に保存することもできます。

スーパータグテンプレートを保存する場所を指定するには

1. **WindowMaker** を開きます。
2. **[ファイル]** メニューの **[設定]** をクリックし、**[WindowMaker]** をクリックします。

WindowMaker の設定画面が表示されます。

3. [スーパータグのパス] テキスト ボックスで、スーパータグ テンプレートを保存するパスを指定します。

スーパータグのインポート

InTouch アプリケーション移行の際にスーパータグをインポートするには、「InTouch アプリケーション移行中のアセットのインポート」を参照してください。

スーパータグ テンプレートとメンバー タグの管理

InTouch HMI 2023 のリリースでは、テンプレート メーカー ユーティリティが廃止されました。スーパータグは、キャンバスのスーパータグ ウィンドウを使用して管理できます。スーパータグ テンプレートおよびそのメンバー タグの作成、編集、および削除を行うことができます。スーパータグ テンプレートを作成した後、タグ名ディクショナリの [タグタイプ] ダイアログ ボックスには、スーパータグ テンプレートとタグタイプとしてのテンプレートの間接型が一覧表示されます。

スーパータグ テンプレートまたはスーパータグ メンバーのタグは、いつでも修正できます。あるテンプレートから派生した既存のスーパータグ インスタンスには、このテンプレートへの変更内容は継承されません。ただし、新しいインスタンスはすべて、修正されたテンプレートの変更内容を継承します。

スーパータグ テンプレートは、テンプレート全体を一括して削除することも、テンプレートに含まれるメンバーのうち選択したタグだけを削除することもできます。削除したテンプレートは、タグ名ディクショナリの [タグタイプ] ダイアログ ボックスに一覧表示されなくなります。

既存のスーパータグのテンプレートまたはメンバーのタグを編集するには

注記: スーパータグ ウィンドウ ペインを使用してスーパータグを管理することが推奨されます。

1. [スーパータグ] ペインを開きます。
2. 編集するスーパータグ テンプレート名またはメンバー タグに移動します。
3. スーパータグ テンプレートを右クリックし、[スーパータグを編集] をクリックします。
または、スーパータグ テンプレートをダブルクリックします。
キャンバスにスーパータグ ウィンドウが表示されます。
4. スーパータグ テンプレートの名前や説明を変更することができます。
5. 必要な変更を行います。
6. [OK] をクリックします。

スーパータグ テンプレートまたはメンバー タグを削除するには

1. [スーパータグ] ペインを開きます。
2. 削除するスーパータグ テンプレート名またはメンバー タグに移動します。
3. スーパータグ テンプレートを右クリックし、[削除] をクリックします。
選択した項目を削除するか確認を要求するメッセージが表示されます。
4. [はい] をクリックして、選択したテンプレートを削除します。

重要: スーパータグ テンプレートのインスタンスのメンバー タグは削除できません。たとえば、PumpRPM が TankPump スーパータグ テンプレートのメンバー タグである場合、PumpRPM は TankPump のいずれのインスタンスからも削除することはできません。スーパータグ テンプレートから削除できるのはタグだけです。

スーパータグ インスタンス

スーパータグ テンプレートとテンプレート インスタンスは異なります。インスタンスは、InTouch HMI アプリケーションにスーパータグ テンプレートを実際に実装したものです。

テンプレートとインスタンスの最も大きな違いは、親のテンプレート名が各インスタンスのタグ名で置き換えられる点です。子テンプレートの名前とサブメンバー タグは変更されません。

スーパータグ インスタンスの作成

タグ名ディクショナリを使用して、スーパータグ インスタンスを作成することができます。新しいスーパータグ インスタンスを定義すると、タグ名ディクショナリによってすべてのメンバー タグとサブメンバー タグが自動的に作成されます。

テンプレートからスーパータグ インスタンスを作成するには

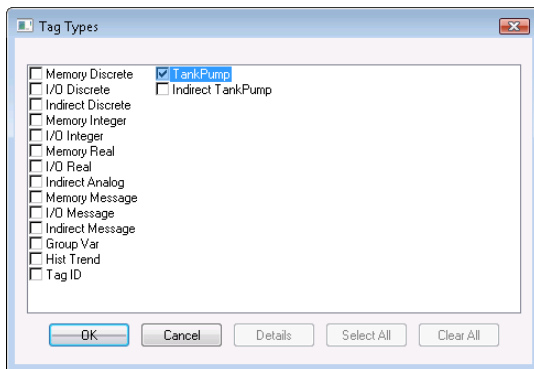
1. **[スーパータグ]** ペインで、インスタンスを作成するスーパータグ テンプレートを選択します。
2. スーパータグ テンプレートを右クリックし、**[新しいインスタンス]** を選択します。

新しいスーパータグ インスタンスが作成されます。

3. 新しいインスタンスをダブルクリックしてタグ名ディクショナリを開きます。
4. **[タグ名]** ボックスに新しいスーパータグ インスタンスの名前を入力します。

スーパータグ インスタンスの名前には、最大 32 文字まで指定できます。インスタンス名の命名規則は、通常の InTouch のタグと同じです。

5. **[タイプ]** をクリックして、**[タグ名タイプ]** ダイアログ ボックスを開きます。
6. リストからスーパータグ テンプレートの名前を選択します。



7. **[OK]** をクリックします。**[タグ名ディクショナリ]** ダイアログ ボックスが展開され、詳細なオプションが表示されます。

[タグ名] ボックスに入力した新しいタグは、選択したスーパータグ テンプレートに含まれるすべてのメンバー タグの親になります。

8. タグのプロパティを設定します。以下の手順を実行します。
 - a. **[メンバー リスト]** ボックスで、スーパータグ テンプレート リストからタグを選択します。
 - b. **[データ アクセス]** から、**[メモリ]** または **[I/O]** を選択して、それぞれ **[メモリ]** ダイアログ ボックスまたは **[I/O 型詳細]** ダイアログ ボックスを開きます。
 - c. 通常の InTouch タグと同じように、詳細設定を入力します。

- d. リストから残りのメンバー タグも選択し、それらについても設定します。
9. スーパータグ インスタンスに属しているすべてのメンバー タグの詳細設定を指定したら、[閉じる] をクリックします。

スーパータグ インスタンスの複製

タグ名ディクショナリを使用して、既存のインスタンスを複製することができます。インスタンスを複製すると、タグは直ちにアニメーション リンクや InTouch QuickScript で使用できるようになります。

タグ名ディクショナリからスーパータグ インスタンスを複製するには

1. タグ名ディクショナリを開きます。
2. [選択] をクリックして、アプリケーションに定義されているタグが一覧表示された [タグの選択] ダイアログ ボックスを開きます。
3. 新しいインスタンスのテンプレートとして使用するスーパータグ インスタンスをリストから選択します。
4. [OK] をクリックします。
[タグ名] ボックスに選択したテンプレートの名前が表示されます。
5. [新規] をクリックします。スーパータグ インスタンスを複製するか確認を要求するメッセージが表示されます。
6. [はい] をクリックします。[名前を入力してください] ダイアログ ボックスが開き、新しいスーパータグの名前を入力するように求められます。
通常のタグの命名規則に従い、最大 32 文字の名前を入力します。
7. [OK] をクリックします。タグ名ディクショナリに新しいスーパータグ インスタンスが表示されます。
8. 必要に応じて、通常の InTouch タグと同じようにメンバー タグを編集します。
9. [閉じる] をクリックします。

スーパータグ インスタンスへのタグの追加

タグ名ディクショナリを使用して、既存のスーパータグ インスタンスのメンバーとしてタグを追加することができます。

タグを追加するには、スーパータグ インスタンスの正確な名前に続いて円記号 (\) の区切り文字と新しいメンバー タグの名前を入力します。

次に例を示します。

Pump_8\PumpSTS

注記: Bulk Import Utility を使用して InTouch HMI アプリケーションから Application Server にタグを移行することを計画している場合、標準の円記号 (\) 区切り文字の置換の詳細については、「[Bulk Import Utility を使用したスーパータグのインポート](#)」を参照してください。

タグをスーパータグ インスタンスに追加するには

1. タグ名ディクショナリを開きます。
2. タグをスーパータグ インスタンスに追加するには、以下の手順を実行します。
 - a. [新規] をクリックします。

- b. **[タグ名]** ボックスに、正確なスーパータグのインスタンス名に続いて円記号 (\) の区切り文字と新しいメンバー タグの名前を入力します。
 - c. **[タグ タイプ]** ボタンをクリックします。
 - d. インスタンスに追加しようとしている新しいメンバー タグのタグ タイプを選択します。
 - e. **[OK]** をクリックします。メンバー タグのタイプに応じて、詳細ダイアログ ボックスが表示されます。
 - f. 通常の InTouch タグと同じように、必要な詳細設定を入力します。
3. **[保存]** をクリックします。
 4. **[選択]** をクリックします。
 5. メンバー タグを追加したスーパータグ インスタンスを選択します。
 6. **[OK]** をクリックします。

[メンバー リスト] ボックスに、このスーパータグ テンプレートに含まれるすべてのメンバー タグが一覧表示されます。

リストには、新しいメンバー タグも表示されます。

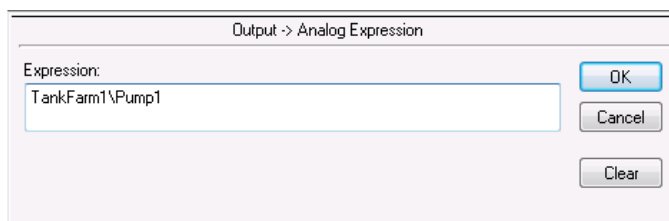
スーパータグのその他の作成方法

スーパータグは次の方法を使用して作成することもできます。

- アニメーション リンクの式の入力ボックス
- InTouch QuickScript 内
- InTouch DBLoad ユーティリティを使用してアプリケーションにロードした外部ファイル

注記: 代替方法を使用してメンバーを作成すると、スーパータグ設定画面では、スーパータグ テンプレート定義にメンバーが表示されません。

スーパータグをアニメーション リンクの式または InTouch QuickScript で作成する場合には、有効なスーパータグの形式を使用する必要があります。次に例を示します。



注記: アニメーション リンクの式または QuickScript で指定したスーパータグ インスタンスおよびメンバー タグが現在定義されていない場合には、タグを定義するように求められます。**[OK]** をクリックします。タグ名ディクショナリが開き、スーパータグ インスタンスと作成したメンバー タグが表示されます。

次の例は有効な構文を示しています。

ParentInstance\ChildMember ParentInstance\ChildMember\Submember

次の例は無効な構文を示しています。

ParentInstance\
ParentInstance\ChildMember\

無効な書式を使用した場合、スーパータグの構文にエラーがあることを示すエラーメッセージボックスが表示されます。

所有オブジェクトとしてのスーパータグインスタンスの設定

所有オブジェクトは、相対参照に使用できます。OwningObject プロパティは、1つのグラフィックを別のグラフィックに埋め込んでいる場合に使用できます。所有オブジェクトプロパティは、産業用グラフィックエディタの設定 [プロパティ] タブの [ランタイム動作] セクションで設定できます。マネージド InTouch アプリケーションとスタンドアロン InTouch アプリケーションの両方の InTouch: 接頭辞付きのインスタンス名を入力します。例えば、「InTouch:Tank01」と入力します。所有オブジェクトは、スクリプトを使用して設定することもできます。OwningObject プロパティは、ShowGraphic() スクリプト関数によって表示されるグラフィックの所有オブジェクトを設定します。これは定数文字列および参照文字列の連結である場合があります。例: `graphicInfo.OwningObject = "UserDefined_001";`

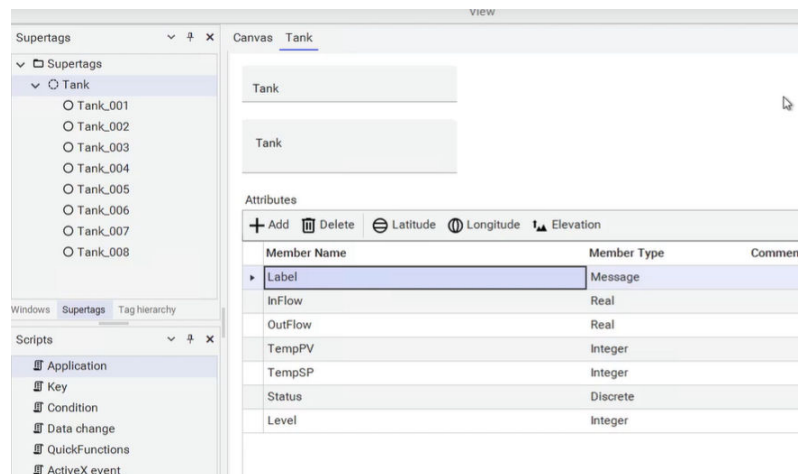
注記: OwningObject プロパティはグラフィックの参照を設定しますが、シンボルがオブジェクトウィザードの一部である場合、GraphicName プロパティに関連付けられません。したがって、所有オブジェクトを使用してシンボルのスクリプトを作成する場合、所有オブジェクトの名前を GraphicName プロパティの一部として指定します (UserDefined_001.Pump_001 など)。

スーパータグ内の所有オブジェクト

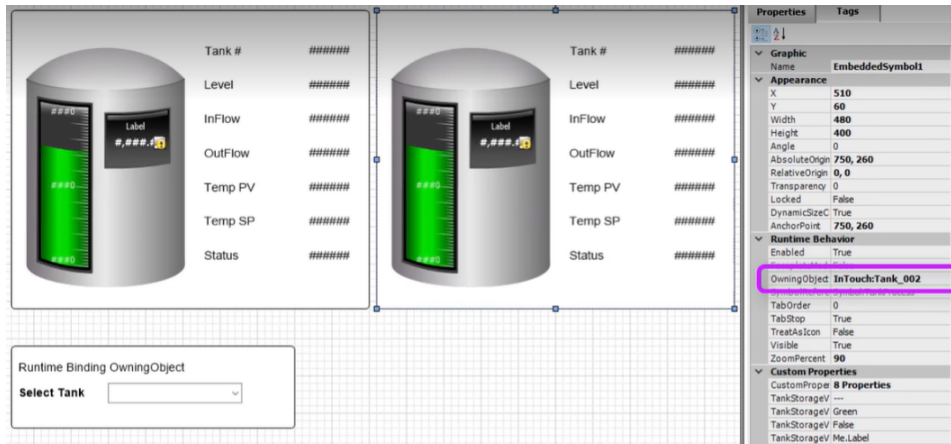
スーパータグで相対参照を使用できます。スーパータグに基づいて、シンボルを作成し、所有オブジェクトプロパティを使用できます。この機能を使用すると、スーパータグインスタンスをグラフィックの所有オブジェクトとして設定できます。個々のシンボルを設定する必要がなくなるので、アプリケーションを迅速に開発できます。

以下に例を示します。

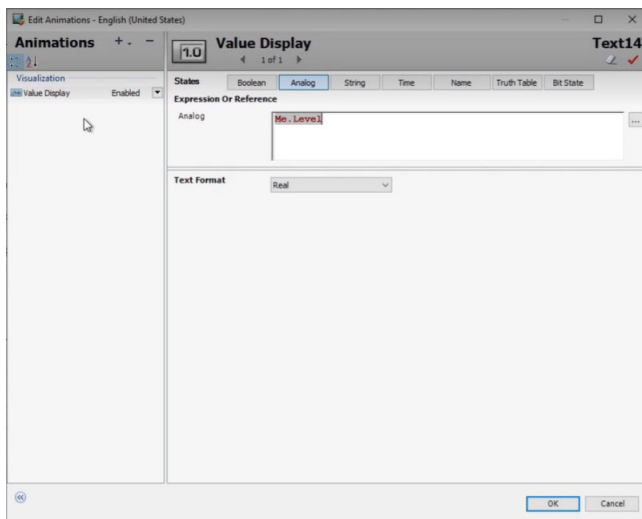
7つの属性が追加され、次に示すような8つのインスタンス (Tank_001、Tank_002 ... Tank_008) がある Tank という名前のスーパータグを考えてみます。



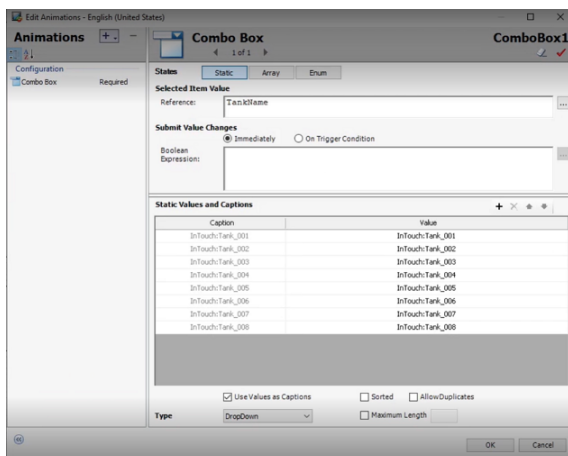
個々の Tank スーパータグインスタンスは、所有オブジェクトとして設定されています。これらの Tank インスタンスを表すために、これらのタンクシンボルが産業用グラフィックエディタに追加されます。



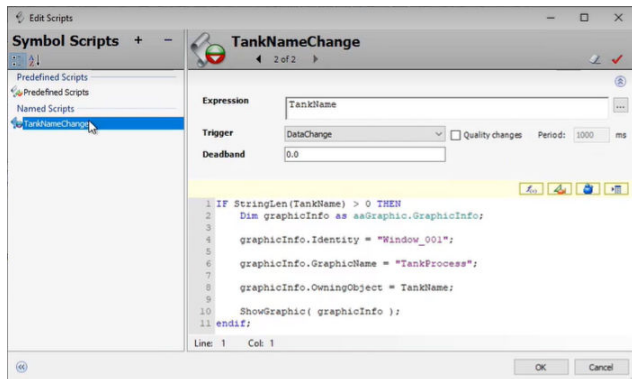
各タンク シンボルに対して、属性の値表示が相対参照として設定されます（埋め込まれたグラフィックには、相対参照が機能するための "me.Attribute" が含まれます。参照される "me.Level" などの属性は、そのスーパータグの属性にランタイム時にバインドされます）。



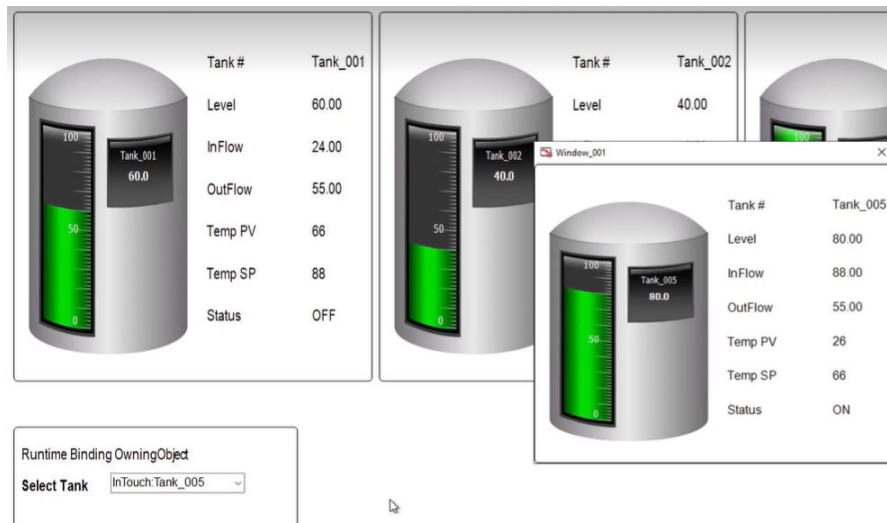
TankName カスタム プロパティを参照するコンボ ボックスがグラフィックに追加され、コンボ ボックスの値がスーパータグ インスタンスの名前として設定されます。



このグラフィックのスクリプトで、**TankName** カスタムプロパティが変更されたときに実行するようデータ変更スクリプトが設定されます。このスクリプトは、グラフィック表示オプションを使用してタンクを表示し、所有オブジェクトを **TankName** カスタムプロパティの値に設定します。



ランタイムに切り替えた後、コンボ ボックスでタンクを選択し、対応するタンクとその属性値を表示することができます。



スーパータグ属性

キャンバスのスーパータグ ウィンドウを使用して、スーパータグ テンプレートに属性を追加できます。各属性に対して、メンバー名やメンバー タイプなどのプロパティも設定できます。

メンバー属性を追加または変更するには

1. キャンバスでスーパータグ ウィンドウを開きます。
2. [属性] セクションで [追加] をクリックします。
新しい属性がグリッドに追加されます。
3. [メンバー名] セルをダブルクリックして名前を変更します。
4. [メンバー タイプ] セルをダブルクリックして使用可能なオプション（整数、実数、メッセージ、および論理）から選択します。デフォルトでは、[メンバー タイプ] フィールドは [整数] に設定されています。

5. [コメント] セルをダブルクリックして説明を変更します。

場所属性を追加または変更するには

1. キャンバスでスーパータグ ウィンドウを開きます。
2. [属性] セクションで [緯度]、[経度]、または [標高] をクリックします。
緯度、経度、および標高の新しい属性がグリッドに追加されます。
3. メンバー名には、選択した場所属性の **緯度**、**経度**、または **標高** がそれぞれ表示されます。

注記: 場所属性のメンバー名は変更できません。

4. [メンバー タイプ] セルをダブルクリックして使用可能なオプション（整数、実数、メッセージ、および論理）から選択します。デフォルトでは、場所の [メンバー タイプ] 属性は [実数] に設定されています。
5. [コメント] セルをダブルクリックして説明を変更します。

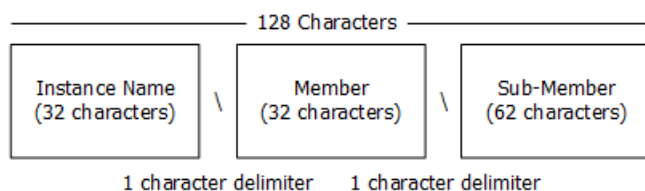
属性を削除するには

1. キャンバスでスーパータグ ウィンドウを開きます。
2. [属性] セクションで、削除する属性を選択します。
3. [削除] をクリックします。
選択した属性が削除されます。

スーパータグ メンバーの参照

InTouch タグ名には、最大 128 文字まで指定できます。各スーパータグ ParentInstance\ChildMember\Sub-member の長さは最大 128 文字です。タグ名に最大長が設定されていることにより、スーパータグの参照に制限が適用されます。

スーパータグの参照には、次の図に示すように最高 2 つのテンプレート（ParentInstance\ChildMember）および 1 つの階層のメンバーだけを含むことができます。



注記: 属性タイプが既存のスーパータグに設定されていない場合、スーパータグのサブメンバーは最大 62 文字の英数字です。スーパータグ タイプに基づくサブメンバーは、最大 32 の英数字の長さです。

スーパータグ テンプレートの各メンバーにアクセスする際は、通常の InTouch タグ タイプのドットフィールドにアクセスするために現在使用している標準の形式を使用できます。スーパータグ参照の構文は、通常のタグを使用できる InTouch アプリケーション全体で使用できます。たとえば、有効なスーパータグ ドットフィールド参照は以下のとおりです。

```
TankFarm\Tank1\Pump1RPM.RawValue
```

リモート タグ参照にスーパータグを使用することもできます。たとえば、有効なスーパータグ リモート参照は以下のとおりです。

```
PLC1:"TankFarm\Tank1\Pump1RPM.RawValue"
```

Bulk Import Utility を使用したスーパータグのインポート

Bulk Import Utility を使用して、タグの定義を InTouch から Application Server のオブジェクト構造に変換することができます。Bulk Import Utility を使用すると、InTouch タグを Application Server に効率的に移行できます。さらに、Bulk Import Utility では、通常の InTouch タグだけでなくスーパータグも移行することができます。

スーパータグを InTouch アプリケーションから Application Server に移行する場合は、スーパータグ リファレンスで使用されている円記号 (\) をサポートされているアンダースコア (_) などの文字に置き換えてください。

例:

```
TankFarm_Tank1_Pump1RPM.RawValue
```

注記: このリリースの時点では、DBDump および DBLoad プロセスに既知の制限があります。.CSV ファイルからのインポートによって作成されたスーパータグ インスタンスはタグ ディクショナリにのみ表示され、[スーパータグ] ペインには表示されません。

MapApp ウィジェットとスーパータグの使用

MapApp ウィジェットを使用して、実行中のアプリケーションからグラフィックを含むマップを表示できます。ランタイム時にマップを使用すると、ユーザーはマップのさまざまな領域をパンすることや、ズームインまたはズームアウトして詳細表示のレベルを調整できます。一般的に、マップ内に配置されたグラフィックは、マップが示す領域内に配置されたビジネスアセットを表します。

詳細については、ウィジェットのヘルプの Map_App ウィジェットのプロパティを参照してください。

スーパータグと MapApp ウィジェットを統合して、WindowViewer または Web Client で表示できます。

InTouch HMI で MapApp ウィジェットを設定するワークフローに従ってください。

ステップ 1: スーパータグおよびスーパータグ インスタンスを作成します。

ステップ 2: スーパータグのグラフィックを作成します。

ステップ 3: スーパータグのマップ設定を設定します。

ステップ 4: MapApp ウィジェットを設定して使用します。

ステップ 5: WindowViewer または Web Client で MapApp を表示します。

スーパータグおよびスーパータグ インスタンスの作成

MapApp ウィジェットは、スーパータグの詳細、および WindowMaker のスーパータグ ペインで設定されたインスタンスの場所属性を使用します。

場所属性を追加するには

1. スーパータグおよびインスタンスを作成します。
たとえば、アセット \$Tank および Tank1 と Tank2 というインスタンスを作成できます。
2. 各インスタンスに属性を追加します。たとえば、Flow（流量）や Temp（温度）などを追加します。
3. "Latitude"（緯度）や "Longitude"（経度）などの場所属性を追加します。
4. タグ ディクショナリを使用して属性値を設定します。

例:

Tank1 をダブルクリックしてタグディクショナリを開きます。以下のメンバー（属性）値を設定します。

Tank1\Flow = 11

Tank1\Latitude = 33

Tank1\Longitude = -114

Tank2 をダブルクリックしてタグディクショナリを開きます。以下のメンバー（属性）値を設定します。

Tank2\Flow = 21

Tank2\Latitude = 36.746

Tank2\Longitude = -119.773

アセットのグラフィックの作成

グラフィック ツールボックスを使用して、アセットを表現するグラフィックを作成します。

例: 作成されたグラフィックは Tank1Sym と Tank2Sym です。

グラフィックをアセットに追加するには

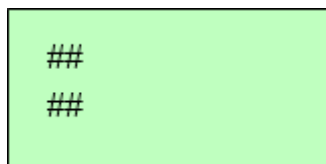
- グラフィックの名前を入力します。

または、グラフィック ツールボックスからグラフィックをドラッグアンドドロップします。

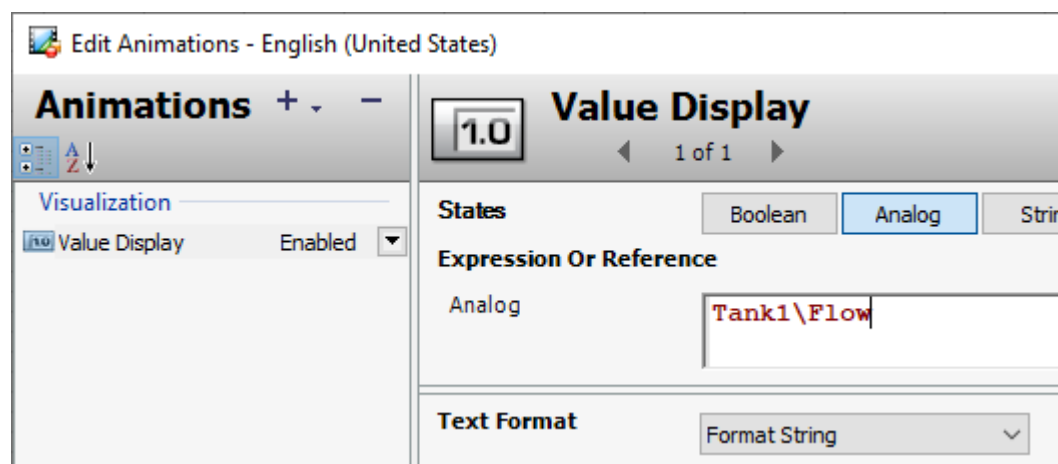
例:

Tank1 アセットのグラフィックに Tank1Sym を使用します。

Tank2 アセットのグラフィックに Tank2Sym を使用します。



グラフィックには、アセット インスタンスのその他の属性（流量や温度など）への参照を設定できます。

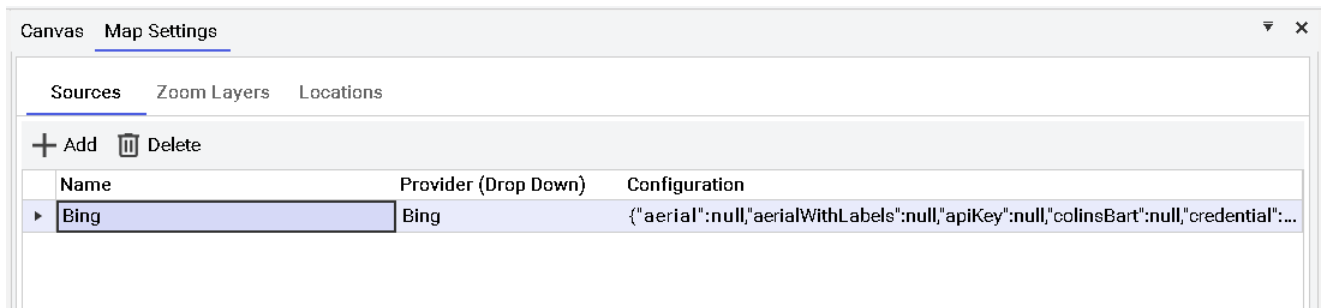


アセットのマップ設定の構成

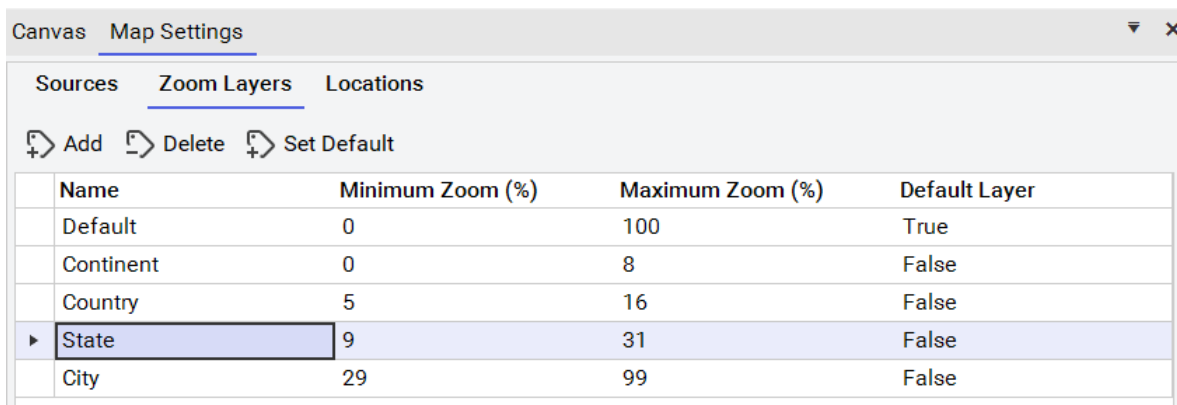
MapApp ウィジェットを使用して、各アセットインスタンスのマップ設定を構成します。

マップ設定を構成するには

1. WindowMaker を開きます。
2. 産業用グラフィック ツールボックス ペインで [ウィジェット] に移動し、[Map_App] をダブルクリックします。
[マップ設定] 画面がキャンバス上のタブとして表示されます。
3. マップのソースの設定



- [追加] をクリックします。
[ソース] グリッドに新しいエントリが作成されます。
 - ソースの名前を入力します。
 - プロバイダのリストからマッププロバイダ (Bing など) を選択します。
これはオプションです。マッププロバイダを選択しない場合、MapApp はデフォルトの Map App プロバイダを使用します。
 - 構成値を入力します。
APIKey: An6Grh5_YKz-_CYnREoNn3nOaBU_rlnVEoc7o8HpSj3GWCvjseEguPvN3rJkZ95T
4. ズーム レイヤーを設定します。
ズーム レイヤーを使用すると、マップのズーム パーセンテージを追加できます。



ズーム レイヤーを追加するには

- a. [追加] をクリックします。

- b. ズーム レイヤーの名前を入力します。ここで指定した名前は、[場所] タブの [レイヤー] カラムに反映されます。

たとえば、Tank1 のレイヤーを Default と設定し、Tank2 を State として設定できます。

- c. 最小および最大のズーム パーセンテージを入力します。

たとえば、State レイヤーの最小および最大ズーム パーセンテージにそれぞれ 9% と 31% を設定した場合、グラフィック Tank2Sym は、ズーム パーセンテージが 9% から 31% のときにのみマップに表示されます。

- d. デフォルト レイヤーを設定します。デフォルトとして設定されたズーム レイヤーは True を表示し、その他のエントリは False を表示します。

ズーム レイヤーを削除するには、[削除] をクリックします。

ズーム レイヤーをデフォルトとして設定するには、[デフォルトに設定] をクリックします。

5. 場所の設定

各アセットの場所の詳細は、アセット設定画面で設定されたアセット属性値から取得されます。

Canvas Map Settings						
Sources		Zoom Layers	Locations			
	Asset	Graphic	Layer	Latitude	Longitude	Position
▶	Tank1	Tank1Sym	Default	33	-114	bottom - center
	Tank2	Tank2Sym	State	36.746	-119.773	bottom - center

レイヤー情報は、[ズーム レイヤー] タブから取得されます。

たとえば、Tank1 のレイヤーを Default と設定し、Tank2 を State として設定できます。

この場合、グラフィック Tank2Sym は、ズーム パーセントが 9% から 31% の場合にのみ表示されます。

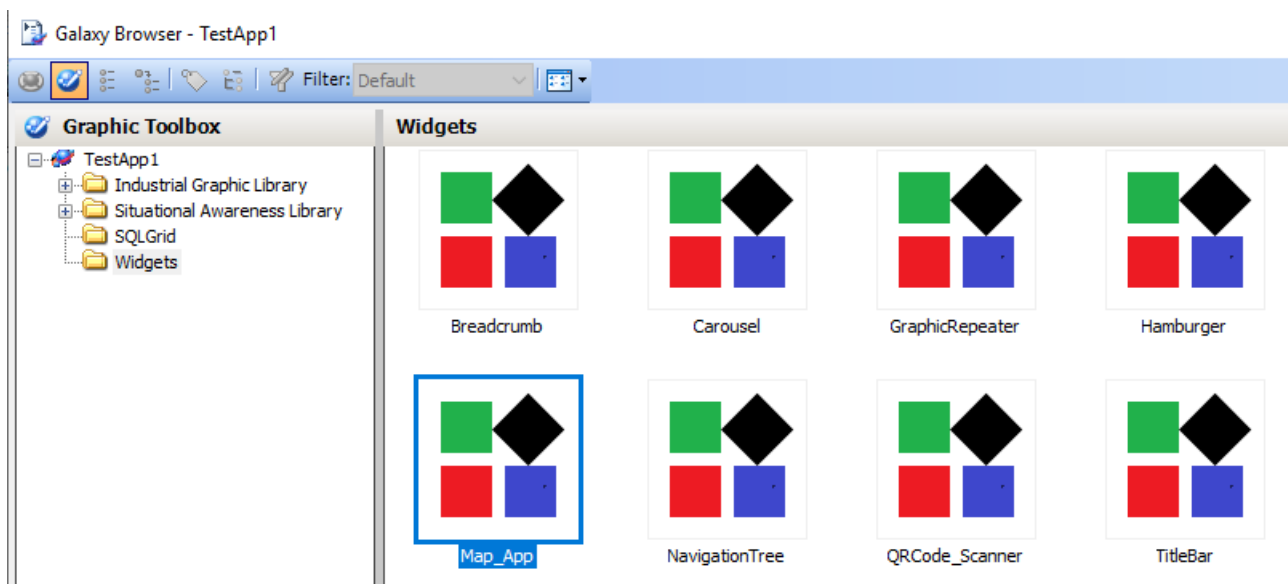
6. [OK] をクリックして保存し、ページを閉じます。

MapApp ウィジェットの埋め込み

マップ上に MapApp ウィジェットを表示するには、グラフィックに MapApp ウィジェットを埋め込む必要があります。

MapApp ウィジェットを埋め込むには

1. 新規グラフィックを作成します (MyApp など)。
2. このグラフィックをグラフィック エディタで編集します。
3. グラフィック エディタ内の任意の場所を右クリックして、[産業用グラフィックの埋め込み] を選択します。
4. Galaxy ブラウザでグラフィック ツールボックスの [ウィジェット] を選択します。
5. ウィジェットのリストから [Map_App] を選択します。



6. Map_App ウィジェットを選択してキャンバスに埋め込み、サイズを調整します。

7. Map_App ウィジェットのウィジェット プロパティを編集します。

[Config name] (設定名) は必須のフィールドです。少なくとも、ConfigName、CurrentZoom、InitialLatitude、InitialLongitude、および InitialZoom の属性値を更新することが推奨されます。

たとえば、次のように属性値を設定できます。

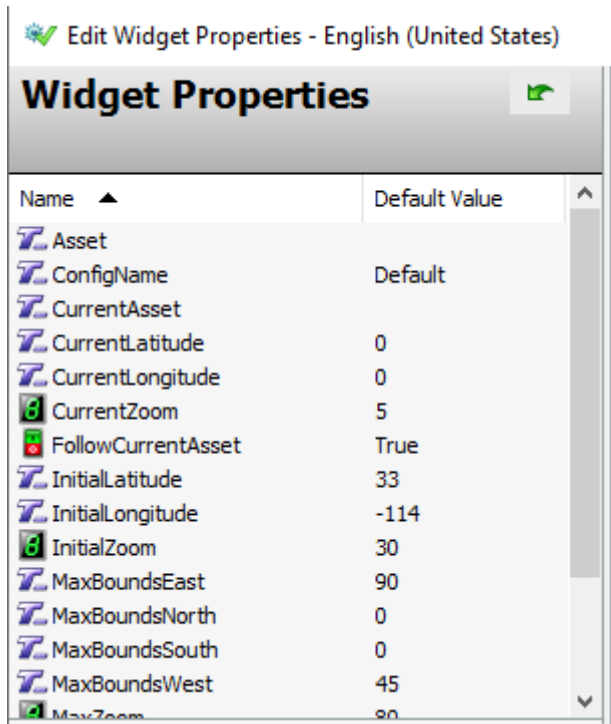
ConfigName = Default

CurrentZoom = 5

InitialLatitude = 33

InitialLongitude = -114

InitialZoom = 30



8. [OK] をクリックします。

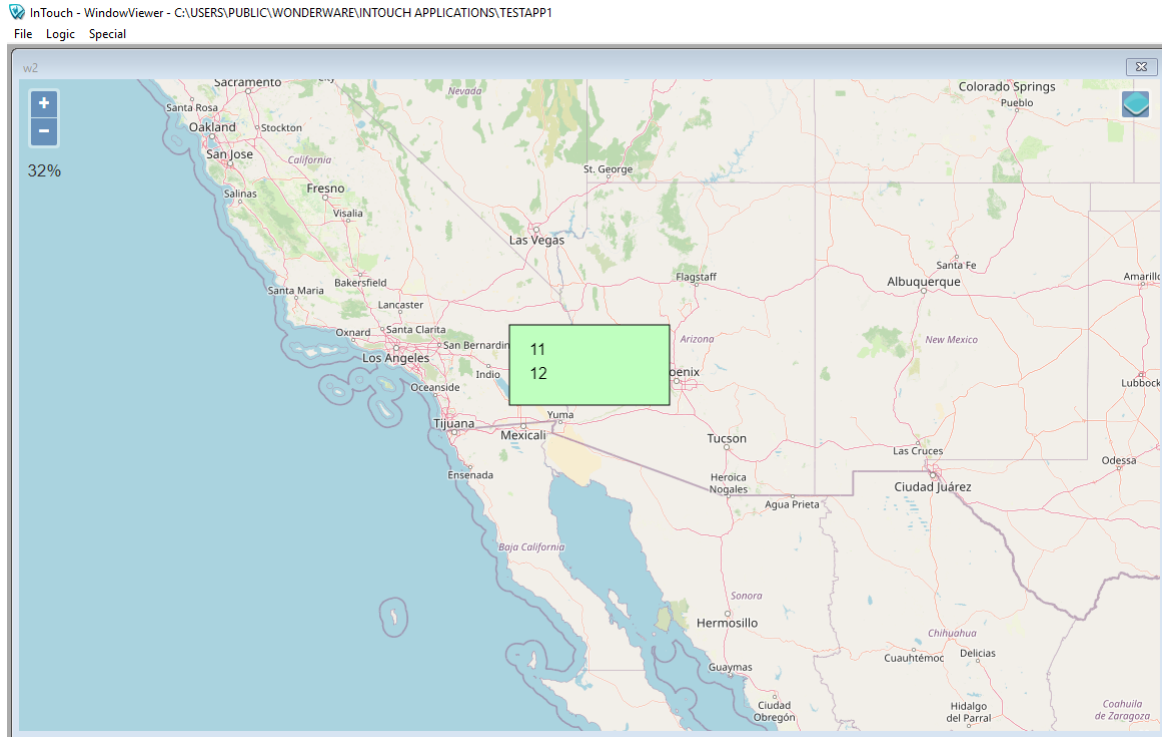
9. グラフィックを枠ウィンドウに埋め込み、ウィンドウを閉じて保存します。

WindowViewer または Web Client での MapApp の表示

設定したアセットと MapApp は WindowViewer または Web Client のいずれかで表示できます。

MapApp を WindowViewer で表示するには

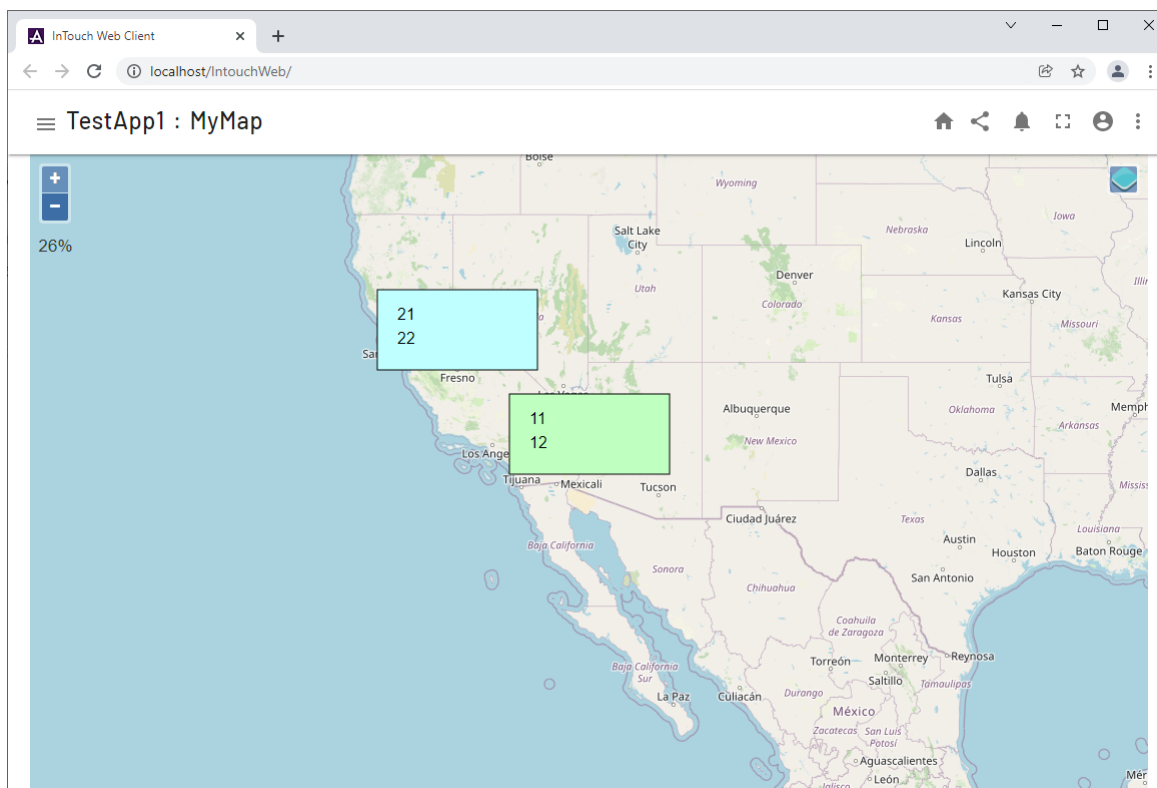
1. WindowViewer への高速切り替えを行い、MyMap グラフィックを含むウィンドウを開きます。
2. Map_App ウィジェットとスーパータグがマップの設定された場所に表示されます。



3. ズーム パーセンテージを調整してマップをズーム インまたはズーム アウトします。

Web Client で MapApp を表示するには

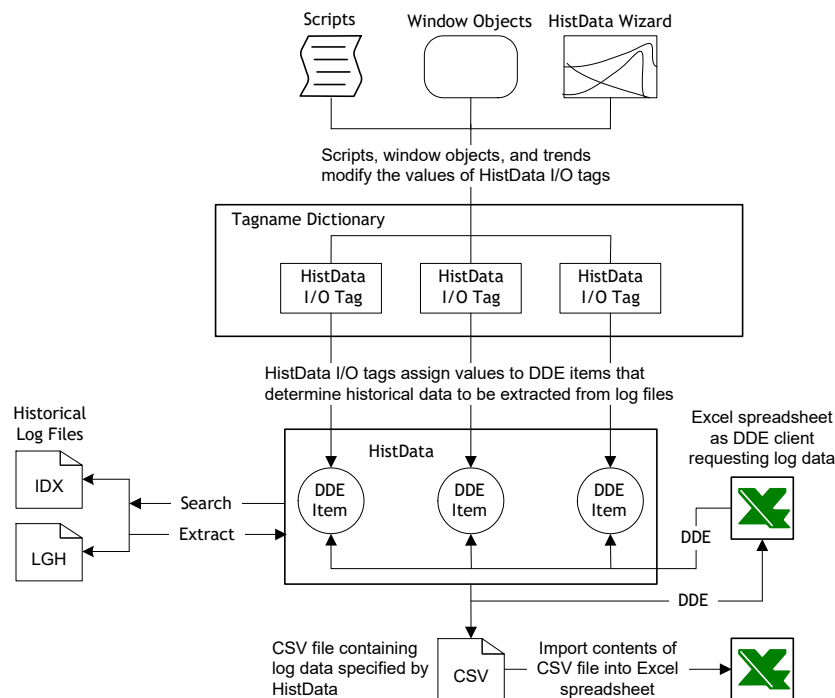
1. Chrome ブラウザをインストールしてデフォルトのブラウザに設定します。
2. WindowViewer が実行している状態で Web Client を起動します。
3. メニューから "MyMap" グラフィックをクリックします。
Map_App ウィジェットとスーパータグがマップの設定された場所に表示されます。
4. ズーム パーセンテージを調整してマップをズーム インまたはズーム アウトします。



他のアプリケーションから履歴タグ変数値へのアクセス

InTouch HistData ユーティリティを使用すると、履歴ログ ファイルからカンマ区切り形式 (.csv) ファイルにデータを抽出することができます。Excel などのアプリケーションは、DDE クライアントとして HistData から直接 InTouch ログ データを抽出したり、または HistData ユーティリティで作成された出力ファイルからログ データをインポートすることが可能です。

下図は、選択した履歴ログ データをファイルまたは DDE クライアント アプリケーションに保存するプロセスを示しています。



DDE アイテムを使用した履歴データの表示

HistData プログラムには、ログ ファイルから履歴データを抽出する方法を指定した DDE アイテムのセットが含まれています。これらのアイテムは、HistData 内部データベースの一部です。各アイテムに値を割り当てます。

次の表は、HistData プログラムで定義されている HistData アイテムをまとめたものです。

項目	データタイプ	説明
DATADIR	メッセージ型	履歴ログ ファイルの含まれるフォルダのパス
DBDIR	メッセージ型	InTouch タグ変数ディクショナリの内容が含まれるフォルダのパス
STARTDATE	メッセージ型	ログ ファイルからデータを抽出する開始日。開始日の形式は MM/DD/YY です。
STARTTIME	メッセージ型	ログ ファイルからデータを抽出する開始時間。開始時間の形式は、24 時間表示で HH:MM:SS です。

項目	データ タイプ	説明
DURATION	メッセージ型	<p>ログ ファイルからのデータ収集期間の長さ。DURATION は、次のように表記することができます。</p> <ul style="list-style-type: none">• 週 (w)• 日 (d)• 時間 (h)• 分 (m)• 秒 (s) <p>DURATION には、秒単位の間隔も指定可能です。たとえば、DURATION=0.5m は 30 秒を意味します。サンプリングを 1 回だけ要求する場合は、DURATION を 0 に設定します。</p>
INTERVAL	メッセージ型	<p>データ収集期間の長さ。INTERVAL は週、日、時間、分、秒で指定します。INTERVAL 期間を表わす時間の単位は、DURATION 期間と同じです。</p> <p>秒単位の間隔も指定可能です。たとえば、INTERVAL=0.25d は 6 時間を意味します。</p> <p>DURATION または INTERVAL の最大期間は 6 週間です。最大期間の 6 週間は、DURATION または INTERVAL のすべての時間単位に適用されます。たとえば、DURATION または INTERVAL で指定できる最大日数は 42 日です。</p>
FILENAME	メッセージ型	<p>履歴ログ ファイルから抽出されたデータを含むファイルの名前およびフォルダの場所。</p>
WRITEFILE	整数型	<p>HistData の出力ファイルへの書き込み操作のステータスを示すフラグ。1 に設定すると、HistData は要求データを FILENAME アイテム名で指定したファイルに書き込みます。ファイルの更新が完了すると、WRITEFILE は自動的に 0 にリセットされます。</p>
ERROR	メッセージ型	<p>ログ ファイルからのデータの抽出中に発生した最後のエラーを説明する文字列。STATUS が 1 にセットされると、ERROR 文字列は None にセットされます。STATUS が 0 にセットされると、ERROR 文字列にはエラーメッセージが入ります。</p>

項目	データ タイプ	説明
TAGS1、TAGS2 など	メッセージ型	<p>ログ ファイルから抽出されたデータを含む 1 つまたは複数のタグ変数の名前を含む文字列。</p> <p>TAGS 文字列には、WindowViewer では 131 文字、Excel で 255 文字を指定することができます。</p> <p>これ以上に長い文字列が必要な場合は、Tagsn という名前のタグ変数アイテムを追加することにより、文字列を追加することができます。ここで、<i>n</i> は追加する文字数を表す整数値です。</p> <p>タグ変数にタグ テキストを追加する必要がある場合は、文字列の最後にプラス記号 (+) を付けます。</p> <p>以下に例を示します。</p> <pre>TAGS="\$Date,ProdLevel,ProdTemp,+" TAGS1="ReactLevel,Temp,GasLevel,+" TAGS2="MotorStatus"</pre> <p>タグ変数名を重複して付けることはできません。また、各タグ文字列の最大長は 512 バイトです。</p>
PRINTTAGNAMES	論理型	<p>タグ変数の名前を関連する値の列の上に印刷するかどうかを示すフラグ。1 にセットされると、タグ変数名が印刷されます。0 にセットされると、タグ変数名は印刷されません。</p>
DATA	メッセージ型	<p>要求データを .CSV 形式で HistData プログラムに保存します。他のアプリケーションから DDE 経由でデータを ADVISE または REQUEST する際に使用します。</p>
STATUS	論理型	<p>HistData の操作の最新のステータス。</p> <p>1 は、HistData がログ ファイルから履歴データを正常に抽出したことを示します。0 は、エラーが発生したことを示します。</p>
SENDDATA	整数型	<p>HistData の更新操作のステータスを示すフラグ。1 に設定すると、HistData は DATA アイテムを要求データで更新します。更新が完了すると、SENDDATA は自動的に 0 にリセットされます。</p> <p>SENDDATA で要求されたデータ量が多すぎるというエラー メッセージを受信した場合は、DURATION の期間を短くするか、または要求タグ変数の数を減らします。タグ変数名を重複して付けることはできません。また、各タグ文字列の最大長は 512 バイトです。</p>

DDE によるログ データへのアクセス

ログ データを出力ファイルに抽出するには、2 つの方法があります。

- 8 個以上のタグ変数を含む履歴ログを出力ファイルに保存する場合は、手動の方法で行います。
- 現在履歴トレンドに割り当てられたペンにマップされているログ データを保存するだけなら、[履歴データ] ウィザードを使用します。

HistData を使用したログ データの手動抽出

ログ データを出力ファイルに手動で抽出することができます。以下の操作を順番に実行します。

1. HistData アクセス名の作成
2. HistData 用 I/O タグの作成
3. HistData ウィンドウの作成
4. HistData の実行

HistData アクセス名の作成

InTouch で HistData プログラムからデータを要求する場合は、以下の手順でアクセス名を定義する必要があります。

アクセス名を定義するには

1. [ホーム] メニューの [タグ] グループで [アクセス名] をクリックします。
[アクセス名] ダイアログ ボックスが表示されます。
2. [追加] をクリックします。
[アクセス名の追加] ダイアログ ボックスが表示されます。

Add access name

Primary source

Access name Node name Application name Topic name

Which protocol to use? When to advise server?

☐ DDE ☒ Suitelink ☐ All items ☒ Only active items

Secondary source

☒ Enable secondary source

Node name Application name Topic name

Which protocol to use? When to advise server?

☐ DDE ☒ SuiteLink ☐ All items ☒ Only active items

Failover

☒ Enable failover

Expression (optional) Deadband: 0 sec

☒ Switchback to primary when conditions clear

Deadband: 0 sec

Cancel Add

3. [アクセス名] ボックスに、最大 32 文字の半角英数字で名前を入力します。[アクセス名] と [トピック名] には、同じ値を入れてください。
4. [ノード名] ボックスに、ログ ファイルが現在置かれているノード名を入力します。
5. [アプリケーション名] ボックスに、ファイル拡張子 .exe を付けずに HistData と入力します。
6. [トピック名] ボックスに、[アクセス名] ボックスで指定した名前を入力します。[アクセス名] と [トピック名] には、同じ値を入力してください。
7. 適切な通信プロトコルを使用可能なオプション ([DDE] および [Suitelink]) から選択します。
8. [サーバーへの通知] 領域で、HistData を使用する場合は常に [すべてのアイテムを要求] を選択します。
9. [追加] をクリックします。

HistData タグ変数の作成

アクセス名を定義したら、以下の I/O タイプのタグ変数を作成して、ログ ファイルからのデータを含む出力ファイルを 1 つ生成します。前の手順で作成したアクセス名をタグ変数に割り当てます。

タグ	I/O 型タグ	タイプ アイテム
HDWDATADIR	メッセージ型	DataDir
HDWDBDIR	メッセージ型	DbDir
HDWDURATION	メッセージ型	Duration
HDWERROR	メッセージ型	Error
HDWFILENAME	メッセージ型	FileName
HDWINTERVAL	メッセージ型	Interval
HDWSTARTDATE	メッセージ型	StartDate
HDWSTARTTIME	メッセージ型	StartTime
HDWSTATUS	メッセージ型	Status
HDWTAGS、HDWTAGS1、 HDWTAGS2	メッセージ型	Tags
PRINTTAGNAMES	論理型	PrintTagNames
HDWWRITEFILE	整数型	WriteFile

注: [HistData ウィザード] では、これらのタグを自動的に作成することができます (ただし、PRINTTAGNAMES タグは除きます)。

ログデータをデータ アイテムに送信して、他のアプリケーションからもアクセスできるようにするには、さらに 2 つのタグ変数を作成します。また、[HistData ウィザード] では、HDWSendDate と HDWData タグは自動的に作成されません。

タグ	I/O 型タグ	タイプ アイテム
HDWSendDate	論理型	SendData
HDWData	メッセージ型	Data

HistData ウィンドウの作成

I/O タイプのタグを作成してから、[HistData] と呼ばれる新しいウィンドウを作成します。これは、以下の例に似ています。

HistData

Status Write File Initialize Data Write File

Path to the historical data file: # (Drive:\Path)

Path to the InTouch Database: # (Drive:\Path)

Start date for requested historical data: # (MM/DD/YY)

Start time for requested historical data using the 24 hour clock: # (HH:MM:SS)

Duration for the requested data to be returned: # (w=week, d=day, h=hr, m=min, s=sec, 0 = one sample)

Length of time between samples: # (1d = 1 day, 1w = 1 week, 25d = 6 hour)
(Note: maximum length of time allowed for Duration & Interval is 6 weeks)

Complete Pathname of the file to write data to: # (Drive:\Path)

List of tag names to return historical data for: #
(Note: Date & Time for a sample can be requested via the System tags, \$Date and \$Time)

Error Message = #

シンボルは、ユーザー入力リンクにリンクします。たとえば、# シンボルには、HDWDataDir タグへのユーザー入力/文字列リンクがあります。ユーザー入力リンクでは、ランタイム中にタグの値を変更できます。

[ステータス] ボタンは、HDWStatus タグに基づいた塗りつぶし色-論理型式にリンクしています。

Object type: Rectangle Prev Link Next Link OK Cancel

Fill Color -> Discrete Expression

Expression: HDWStatus==1 OK Cancel

Colors: 1,TRUE,On: (green) 0,FALSE,Off: (red) Clear

[ファイル書き込み] ボタンは、HDWWriteFile タグに基づいた塗りつぶし色-論理型式にリンクしています。

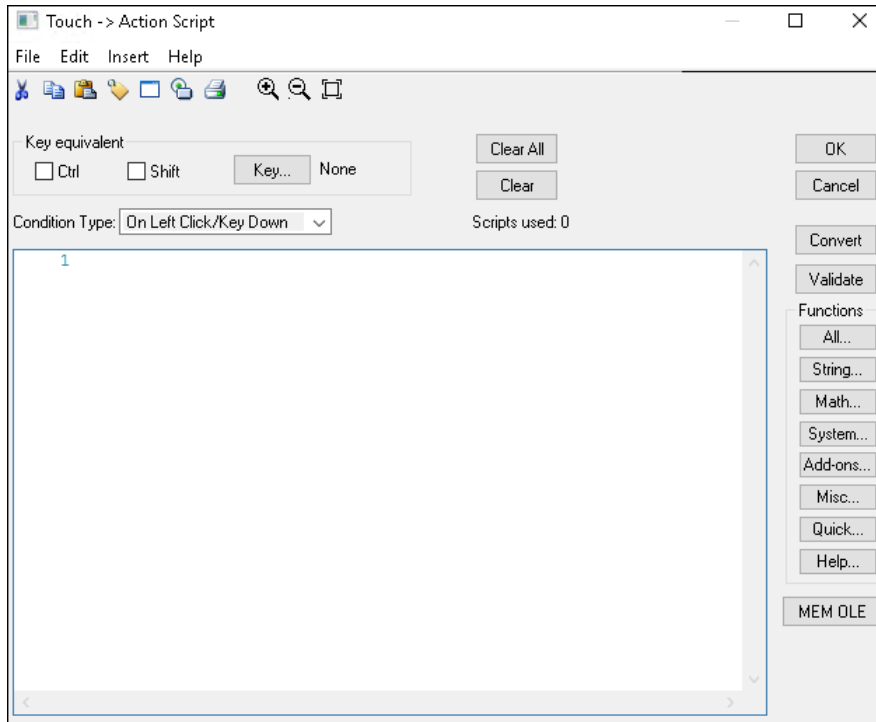
Object type: Symbol Prev Link Next Link OK Cancel

Fill Color -> Discrete Expression

Expression: HDWWriteFile==1 OK Cancel

Colors: 1,TRUE,On: (red) 0,FALSE,Off: (green) Clear

[データ初期化] ボタンは、[タッチ押しボタン] アクションスクリプトにリンクしています。



[**データ初期化**] ボタンをクリックすると、**HistData** アイテムは目的の値に初期化されます。ユーザー入力リンクを使用すると、必要に応じてランタイム中にアイテムの値を変更できます。

[**ファイル書き込み**] ボタンは、[タッチ押しボタン] アクションスクリプトにリンクしています。

[**ファイル書き込み**] ボタンをクリックすると、出力ファイルが生成されます。

HistData の実行

HistData ウィンドウを作成してから、以下の手順を完了させて、**WindowViewer** で実行します。

HistData ウィンドウを実行するには

1. [HistData] を開始して最小化します。
2. WindowViewer を起動して、[HistData] ウィンドウを開きます。
3. [初期化] ボタンをクリックし、HistData アイテムを変更します。
4. [ファイル書き込み] ボタンをクリックします。

操作が成功すると、ステータスの値はオンになり、オンステータスに関連付けられている色が表示されます。操作が失敗すると、ステータスの値はオフになり、失敗の原因を示すエラーメッセージが表示されます。

HistData ウィザードを使用したログデータの抽出

履歴トレンドに表示されるログデータの含まれる出力ファイルを作成することができます。InTouch には [HistData ウィザード] が用意されており、ログファイルから自動的にデータを抽出します。

HistData の出力ファイルには履歴トレンドに表示されるログデータが書き込まれるため、このファイルには履歴トレンドのペンに現在割り当てられているタグからのデータだけが入ります。

「HistData ウィザード」を使用してログデータを抽出するには

1. WindowMaker を起動します。
2. 履歴トレンドが表示されたウィンドウを開きます。
3. 「描画」メニューの「挿入」グループで「ウィザード」をクリックします。
「ウィザードの選択」ダイアログボックスが表示されます。
4. 左ペインから、「トレンド」グループを選択します。
5. 右ペインから、「HistData ウィザード」アイコンを選択し、「OK」をクリックします。
6. テレンドウィンドウ上の HistData オブジェクトを配置する場所にマウスポインタを移動させます。
7. テレンドウィンドウで、HistData オブジェクトを配置する場所をクリックします。出力ファイルが作成されるパスが表示された「ファイル」ボックスとボタンで構成されたウィンドウオブジェクトが開きます。
8. テレンドウィンドウで「HistData ウィザード」オブジェクトをダブルクリックします。「履歴データパネルウィザード」ダイアログボックスが表示されます。

HistData Panel Wizard

This Wizard is designed to work in conjunction with a particular Hist Trend. Enter the tag for this trend below:

Hist Trend: (Hist Trend)

OK
Cancel
Suggest

If the tag that you enter above does not exist, the Wizard will create it. Click Suggest for suggestions on a name. If you have previously used one of the other HistTrend Wizards, Suggest will provide the tag connected with it. If not, Suggest will provide a unique unused tag.

The Wizard will also create 11 other tags for use in communicating with the Historical Data Manager (HistData). The Wizard will use existing tags, if possible. If not, the tags it will create will all begin with 'HDW'.

Number of Records to Write per CSV File:

At runtime, pressing the button on this Panel will save to a CSV file all the data data between the Scooters on the Trend. To do this, the Panel will write a fixed number of records to the file (specified above), no matter what the actual time duration of the data is. These records will be evenly spaced in time.

9. 「履歴トレンド」ボックスに HistTrend タグの名前を入力します。
10. 「CSV ファイルごとに書き込むレコード数」ボックスに、出力ファイルに書き込むレコード数を入力します。
11. 「OK」をクリックします。HDW 接頭辞で識別されるタグのセットが作成されます。

「[HistData タグ変数の作成](#)」に記載されているタグが作成されます。HisDataViewSt のアクセス名にタグが割り当てられます。

12. WindowViewer で履歴トレンドウィンドウを実行します。

13. HistData ウィンドウ オブジェクトの中にある **【ファイルへ保存】** をクリックします。ウィンドウ オブジェクトに表示されているフォルダの場所に、出力ファイルが作成されます。

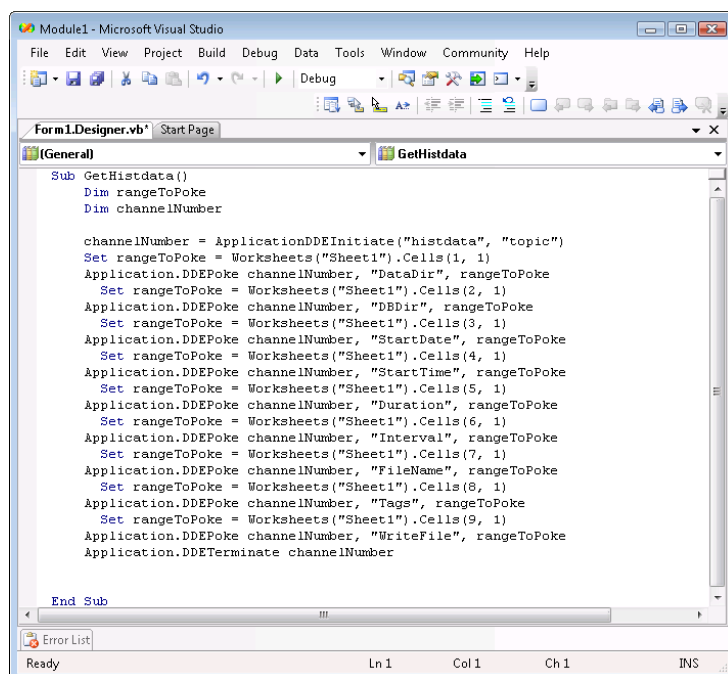
注記: 新しく作成された *.idx ファイルおよび *.lgh ファイルで InTouch を起動後 1 時間以内にタグの値が更新されない場合、HistData は .csv ファイルに正しく入力されません。

HistData データには、履歴トレンドスクータと同じ解像度がありません。HistData の解像度は、時間範囲内で要求された値の数に基づきます。これは *.idx ファイルと *.lgh ファイルの値が正確に反映されたものではありません。

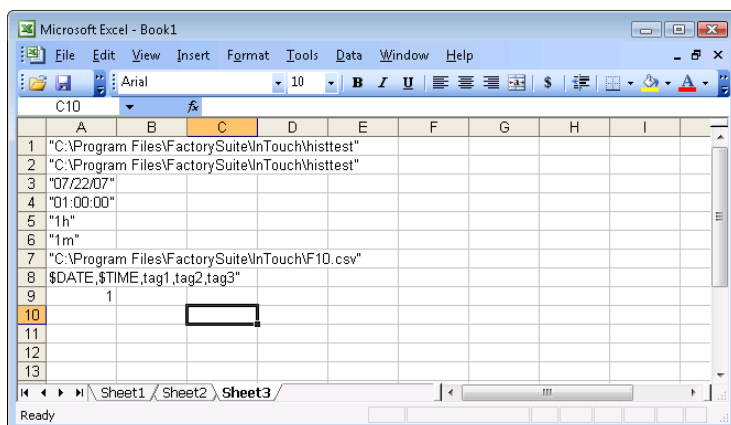
他のアプリケーションからの履歴データへのアクセス

Excel マクロを記述して、HistData ファイルからデータを取得できます。HistData プログラムは、マクロの INITIATE、POKE、および TERMINATE 関数に対応しています。キーワード（内部データベース アイテム）を持つ POKE 関数は、クエリーを定義するパラメータの設定に使用します。クエリーを正しく設定すると、マクロが実行され、HistData ファイルから指定された履歴データを要求します。

以下に、VBA で書いたマクロの例を示します。



上の例では、POKE したデータは Sheet1 に入っています。次に、POKE したデータの例を示します。



HistData エラーのトラブルシューティング

HistData でログデータを抽出しているときに発生する可能性のあるエラーを確認することができます。以下の表には、左の列に標準的な HistData の問題またはエラーメッセージが一覧表示されています。右の列は、その問題の考えられる原因と解決方法です。

エラーメッセージまたは状況	原因/解決方法
エラーメッセージ: 要求するデータが多すぎます。期間を短くするか、または要求するタグ変数の数を減らしてください。	このエラーは、SendData アイテムに要求されるデータ量が多すぎる場合に発生します。 SendData アイテムは、ログファイルからのデータが入った出力ファイルを作成するためだけに使用しないでください。
エラーメッセージ: ファイル C: \FILES1\HISTDATA.CSV を開くことができません	フォルダパスが存在しないか、またはフォルダパスのスペルが間違っています。
エラーメッセージ: ファイル C:\FILES\を開くことができません	出力ファイルが定義されていません。
エラーメッセージ: DATADIR アイテムが無効です	DataDir アイテムで指定された宛先フォルダパスが存在しません。フォルダパスのスペルを確認してください。
エラーメッセージ: STARTDATE アイテムが無効です	StartDate アイテムに無効な形式の開始日が含まれています。Windows の場合は、コンピュータの日付形式を mm/dd/yy に変更してください。
エラーメッセージ: ログファイルが見つかりません	DataDir アイテムで指定されたパスに要求された日付のログファイルが存在しません。

エラー メッセージまたは状況	原因/解決方法
エラー メッセージ: タグ変数 TAGX はデータベースにありません	要求されたタグ変数が、アプリケーションのタグ変数ディクショナリに存在しません。タグ変数名のスペルが正しいことを確認してください。
エラー メッセージ: C:\IT6.0B\HISTEST に tagname.x が 見つかりません	ファイル tagname.x が存在しないか、または破損しています。
.csv ファイルが出力されず、エラーは表示されません。	<ul style="list-style-type: none">• HistData が稼動していません。• Tags アイテムの一覧に、ログ用に指定されているタグ変数が含まれていません。• HDWWritefile の定義が正しくありません。タグ変数が整数型であること、DDE のアクセス名が正しいこと、アイテムが WriteFile であることを確認してください。また、MinEU=MinRaw および MaxEU=MaxRaw となるスケーリングがないことも確認してください。
出力された .csv ファイルには、日付と時刻は含まれていますが、要求されたタグ変数の記録日が入っていません。	要求された時間中、タグ変数の履歴ログにエントリがありません。履歴トレンドを表示して、要求された時間内にログ ファイルにデータが入るかどうかを確認してください。
WWLogger に以下のメッセージが表示されています。 DDE HistData Viewstream1 にエラーがあります。 WriteFile: Poke コマンドはサーバーから拒否されました。	HistData アイテムへの値の割り当てエラーのために、.csv ファイルの作成が失敗するたびに、WWLogger にエラー メッセージが書き込まれます。このエラーは、 WriteFile アイテムを 0 にセットしようとした場合、またはエラー アイテムに書き込みを行おうとした場合にも発生する可能性があります。
ログ ファイルに多数のレコードがあると、.csv ファイルには 1 つのレコードしか書き込まれません。	Interval アイテムが間違った値に指定されて、収集期間が非常に短くなっている可能性があります。 または、 Duration アイテムの形式が、HDWDuration=1- などのように無効である可能性があります（増分を指定することはできません）。

IEEE 小数ユニット

Historian HMI では、32 ビットの 2 進数の値を浮動小数点数に変換するために、IEEE (Institute of Electrical and Electronics Engineers) 754 規格を使用しています。

IEEE 754 形式の 32 ビット数は、16 ビットの 2 ワードとして 16 ビット PLC (Programming Logic Controllers) に保存されます。一般的に、PLC の浮動小数点レジスタの番号付け体系では、16 進数の下位ワードと上位ワードが連続しています。現在の 32 ビット パーソナル コンピュータでは、32 ビット単一レジスタが使用されています。レジスタのビット番号付け体系は、2 つの連続する 16 ビット レジスタと同じ形式です。

Historian アプリケーションで浮動小数点を使用するには、Historian が 2 つの 16 ビット PLC レジスタに保存された値を変換できなければなりません。Historian は常に PLC レジスタの生の値を個々の整数とみなすため、ビット変換を行う必要があります。2 つのレジスタ整数でブール値 **AND** 演算を行い、それを実数に変換することはできません。Historian は 2 つの整数レジスタのタイプ変換を行うことはできません。

Historian HMI における浮動小数点数の表示

Historian HMI では、アプリケーション内の実数を表示するために IEEE 32 ビット浮動小数点形式を使用しています。IEEE 浮動小数点形式は、実際の実数の近似に過ぎません。IEEE 32 ビット浮動小数点形式では、実数が 2 の累乗でない限り、正確に表現することはできません。IEEE 32 ビット浮動小数点数の精度は、小数点以下約 8 桁です。

実数を Historian アプリケーションで表示する場合は、数値が 8 桁を超えないよう注意してください。以下に、Historian アプリケーションで有効な実数を表記する浮動小数点数形式を示します。

```
#
###
###
0#
###.##
#####
###.#####
#####.##
#####.###
ABCDEF
###.####
```

浮動小数点数が 8 桁を超えると、切り上げの誤差が生じることがあります。

テキストのフォーマットに小数点を含めない場合、数値は小数点と共に表示されます。実際のフォーマットの小数点の位置は WindowViewer の [詳細な書式設定] プロパティで設定されます。

注記: 小数点の左側に「#」を追加した場合、または小数点がない場合、表示する桁数を制限しないでください。

例 1

Historian アプリケーションで実数 2.3 を表示します。ただし、2.3 は 2 の累乗ではないため、IEEE 32 ビット浮動小数点形式で 8 桁以上の数を正確に表記することはできません。

アプリケーションが値 2.3 を ASCII 文字の 2.3 と認識できるようにするには、この数が 8 桁以下である必要があります。上限の 8 桁を超えた場合、この数は 2.29999999 または 2.30000001 と表記されます。

例 2

2 つの実数型タグが比較されると、2 つの実数型タグの差分は FLT_EPSILON（値 1.19209290E-07F、10 進数では 0.0000001192092896）よりも大きくなる必要があります。しかし、数値が 8 桁を超えると、結果の値は正しくなくなることがあります。この問題を修正するには、値を 1000 で乗算するか、それより大きい 10 の倍数で乗算します。この処理を行うことにより、値が 1e-7 よりも大きくなります。必要な比較演算を実行し、1000（または乗算した数）で除算します。

InTouch OPC UA サーバーの設定および使用

InTouch HMI は、マシン ツー マシン通信の OPC UA（Unified Architecture）プロトコルをサポートします。

この OPC UA サーバー機能を使用すると、サードパーティ クライアントが InTouch HMI と対話して、OPC UA などの業界標準を活用できます。InTouch HMI で OPC UA サーバー機能を有効にすると、クライアントは組み込みの Gateway 通信ドライバまたはサードパーティ クライアントを利用して InTouch HMI に接続し、Gateway 通信ドライバまたはクライアントを使用して OPC UA ネームスペースを安全に参照して InTouch HMI と対話できます。

OPC UA の設定とチェックリスト

OPC UA サーバーおよび OPC UA クライアントのエンドツーエンドの設定に必要なタスク

設定タスクは、完了する必要がある順序で示されています。

1. **System Management Server の設定:** System Management Server は、マシン間での信頼関係を構築するために使用するので、ノード間の安全な通信を保証するように設定する必要があります。System Management Server は、通常、最初の System Platform をインストールするときに設定します。詳細については、『System Platform インストールガイド』の「System Management Server の設定」を参照してください。

注記: セキュアな通信を可能にする暗号化証明書にアクセスするために、InTouch HMI は管理特権を持つユーザーのコンテキストで実行する必要があります。

2. **OPC UA サーバーの設定:** 構成オプションの設定、OPC UA サーバー接続のテスト、および Gateway 通信ドライバのアクティブ化を行います。
3. **IT コンプライアンス/ファイアウォールの検証:** 構成のこの時点でファイアウォールの設定と検証を完了する必要があります。OPC UA サーバーが配置されているノードには、ファイアウォールの受信規則が構成され、検証されている必要があります。

重要! 残りの設定タスクを行う前に、ファイアウォールテストを正常に完了する必要があります。

4. **OPC UA クライアントの設定:** クライアント設定には以下が含まれることがあります。
 - OPC UA サーバー アドレスの定義（形式は `opc.tcp://<ServerName>:<ポート番号>`）。
 - 正しい OPC UA サーバー セキュリティ ポリシーの選択（Basic256Sha256）。

- ユーザーを InTouchHMIOPCUAWriteUsers ユーザー グループに追加します。
 - 設定済みの OPC UA ユーザー資格情報（ユーザー名およびパスワード）を入力します。
 - 匿名接続は、InTouch タグの読み取りでのみサポートされます。セキュリティ リスクを回避するために、認証済みの資格情報を使用してデータにアクセスすることが推奨されます。
5. **セキュリティ証明書:** ランタイム ノードに OPC UA セキュリティ証明書をダウンロードして設定します。
 6. **接続の検証:** OPC UA クライアントを開き、OPC UA サーバーに接続できること、およびネームスペースのアイテムを表示できることを確認します。

InTouch OPC UA サーバーの設定

InTouch OPC UA サーバーは、ゲートウェイやその他のプロトコル変換メカニズムなしで OPC UA クライアントから InTouch HMI データへのアクセスを提供します。

1. **AVEVA InTouch HMI アプリケーションマネージャ**を起動します。
2. [ツール] タブで **[OPC UA 設定]** をクリックします。
[OPC UA サーバー] ダイアログ ボックスが表示されます。
3. OPC UA サーバーはデフォルトで無効になります。サーバーを設定するには、**[OPC UA を有効にする]** をクリックします。

OPC UA server

Choose this option to enable InTouch as OPC UA server

☒ Enable OPCUA

Endpoint configuration

Configure the endpoint for this OPC UA server instance. This determines which URI the OPC UA clients will use to connect.

Port number: 48032

Resulting endpoint for OPC UA clients : opc.tcp://<deployment hostname>:portnumber

Security configuration

☒ Require encrypted communication between OPC UA clients and this server instance (Recommended)

Choose this option to require that all clients must use encryption (Basic256SHA256 and SignAndEncrypt) when establishing communication with this server instance. If enabled, unencrypted communications will not be supported.

Client access rules

Choose the type of access clients will have to InTouch data from this server instance

☒ Allow anonymous client connection (no username/password)

☒ Allow authenticated InTouch users to write to attributes, depending on their security role.

Additional information on the end-to-end process of configuring and using OPC UA can be found in the InTouch help. Click [here](#) for help.

Cancel Ok

4. ポート番号を編集します。デフォルトで、ポート番号は 48032 に設定されます。

注記: ポート番号は、ユーザーおよび RDS セッションに対して一意です。追加の RDS セッションには代替のポート番号を割り当てます。

5. **セキュリティ設定:** 通信でペイロードが暗号化されるので、このオプションを有効化することが強く推奨されます。クライアント側も、この設定に一致する必要があります。
6. [クライアントアクセスルール] オプションを使用すると、InTouch データに対するクライアントのアクセスのタイプを決定できます。
 - a. 匿名アクセスを許可するには、[匿名クライアント接続を許可（ユーザー名とパスワードの入力は不要です）] チェック ボックスをオンにします。
 - b. 認証されたユーザーだけを許可するには、[セキュリティの役割に応じて、認証済みの InTouch ユーザーによる属性への書き込みを許可する] チェック ボックスをオンにします。
7. [OK] をクリックします。
8. 設定後、WindowViewer を起動して OPC UA サーバーを起動します。これで OPC UA クライアントから InTouch HMI データにアクセスできるようになります。

OPC UA サービス用のファイアウォールの設定

InTouch HMI の OPC UA 通信では、ランタイム ノードのファイアウォールで OPC UA クライアント ノードとの通信を許可する必要があります。しかし、ファイアウォール設定を変更する前に[ファイアウォールのテスト](#)を実行することが推奨されます。

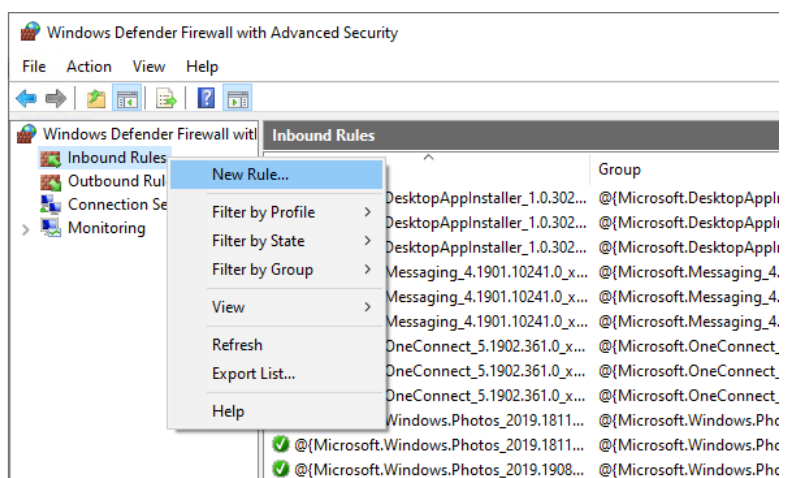
ランタイム ノードのファイアウォールの設定

重要: OPC UA サーバー サービスを配置するノードにファイアウォールルールを追加する必要があります。

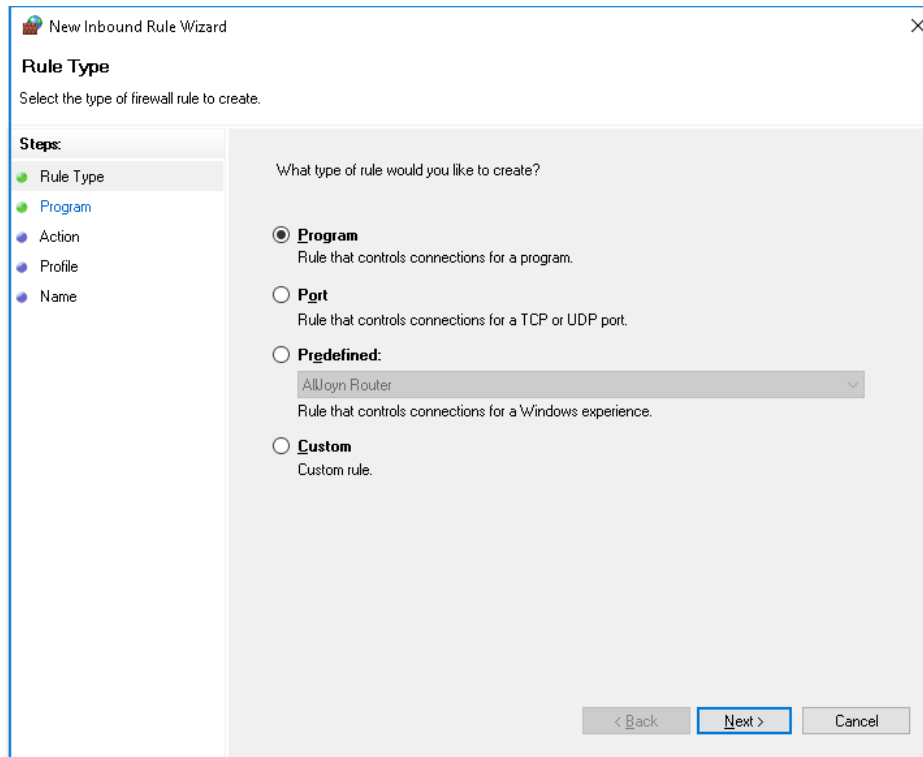
ランタイム ノードのファイアウォールを設定するには:

OPC UA サーバー サービスを配置したランタイム ノードで Windows ファイアウォールを開き、次のように設定します。

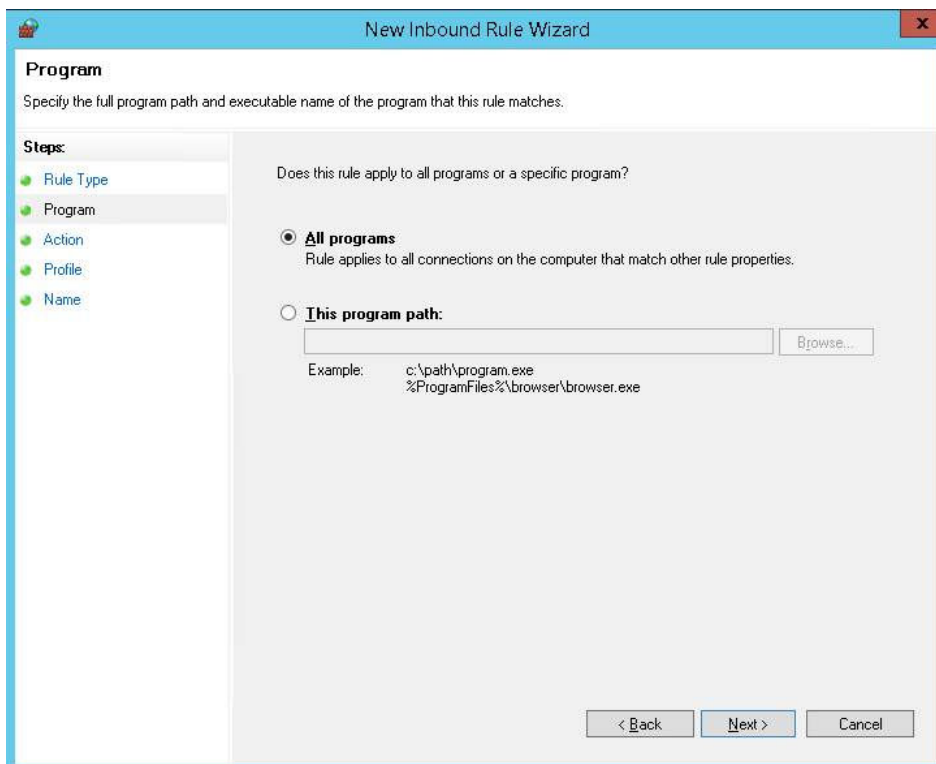
1. Windows の検索バーを使用して **Windows ファイアウォール**を開きます。
2. [詳細設定] を選択し、**受信規則**を作成します。



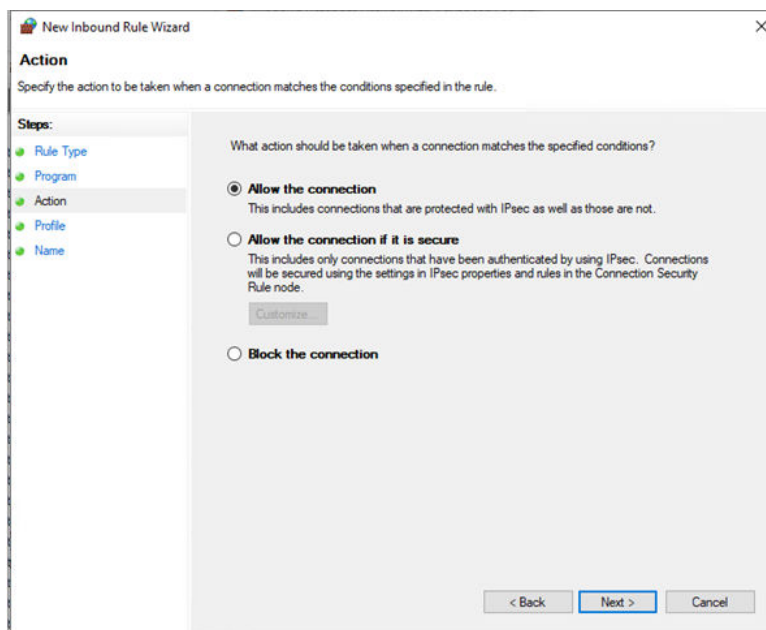
3. [新しい規則] をクリックします。
新規の受信の規則ウィザードが開きます。をクリックします。
4. 規則のタイプに [プログラム] を選択して [次へ] をクリックします。



5. 選択したルールของโปรแกรมパスを指定します。

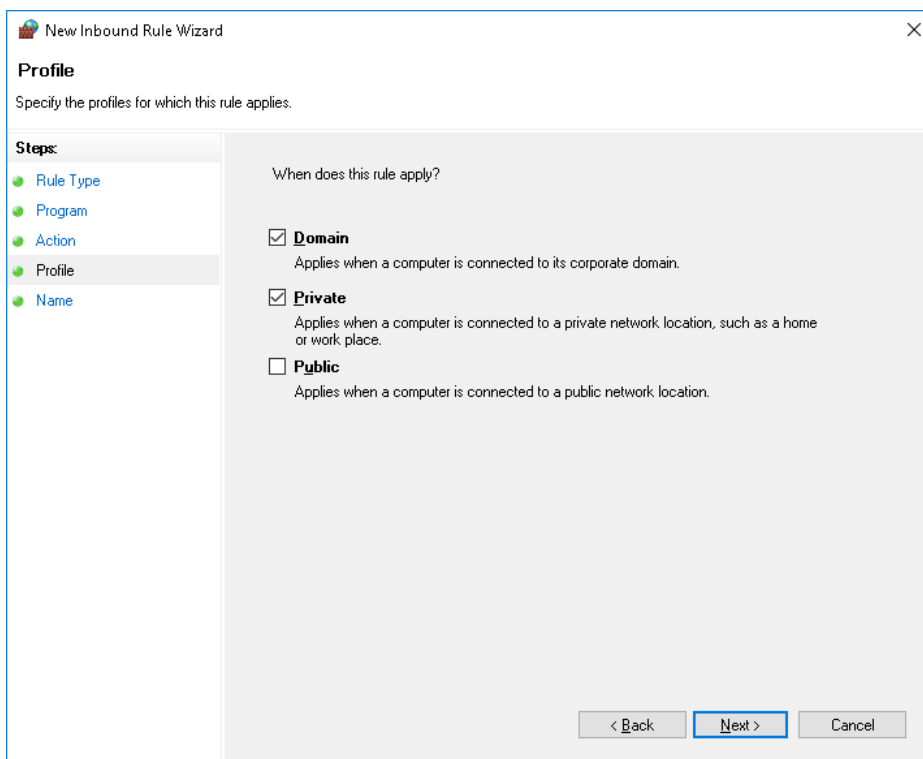


6. 次の画面で「接続を許可する」オプションを選択します。「次へ」をクリックします。

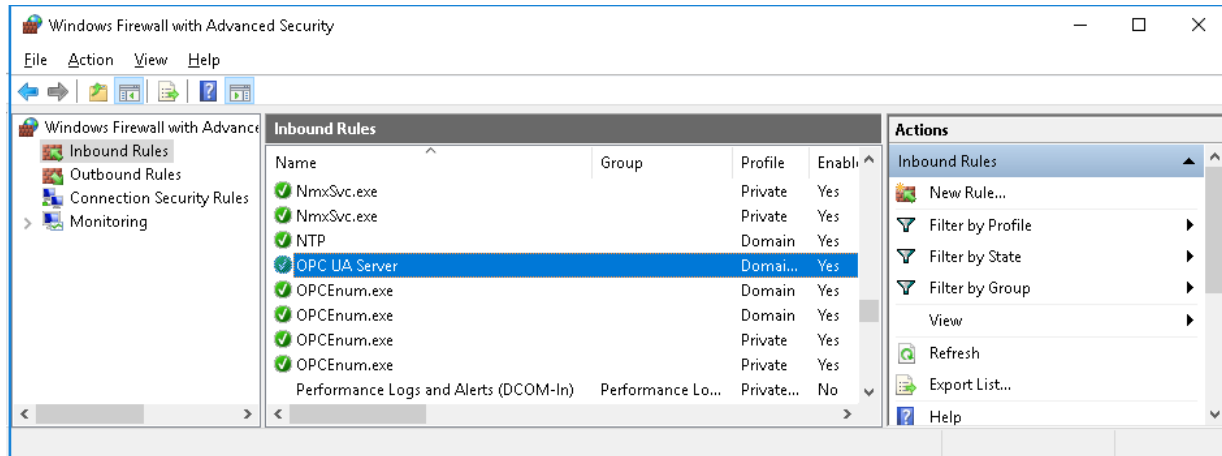


7. 規則を適用するときを尋ねる画面が表示されます。

- ドメイン環境の場合: [ドメイン] および [プライベート] を選択します。[パブリック] の選択を解除することをお勧めします。
- ワークグループ環境の場合: [パブリック] を選択します。[ドメイン] および [プライベート] の設定は、ワークグループ環境では何の影響も及ぼしません。



8. 最後に、このルールの名前を入力します（「OPC UA サーバー」など）。複数の OPC UA サービスを設定する場合、各サービスを区別できるような名前を使用してください。
9. 新しい規則が **Windows** ファイアウォールの受信規則のリストに追加され、有効になっていることを確認します。



10. ファイアウォールのテストを繰り返して OPC UA クライアント ノードからランタイム ノードに接続できることを確認します。

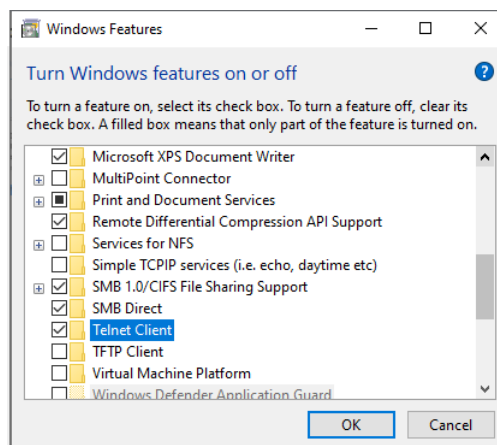
ファイアウォールのテスト

Telnet でファイアウォールのテストを実行するには

1. OPC UA サーバーが実行している別のノードから、Windows の **Telnet Client** 機能を有効化して **Telnet** を有効にします。

注記: このテストは、OPC UA クライアント ノードから実行します。

- a. Windows のコントロールパネルを開きます。
- b. [プログラムと機能] を開き、[Windows の機能の有効化または無効化] を選択します。
- c. 機能のリストを下にスクロールして [Telnet Client] を有効にします。Telnet はデフォルトで無効化されています。



2. このテストを実行する前に、WindowViewer が実行していることを確認してください。OPC UA サービス ホストが設定されている場合、WindowViewer で InTouch OPC UA サービスが起動します。詳細については、「[InTouch OPC UA サーバーの設定](#)」を参照してください。
3. OPC UA クライアント ノードのコマンド ウィンドウで以下を入力して Telnet を実行します。

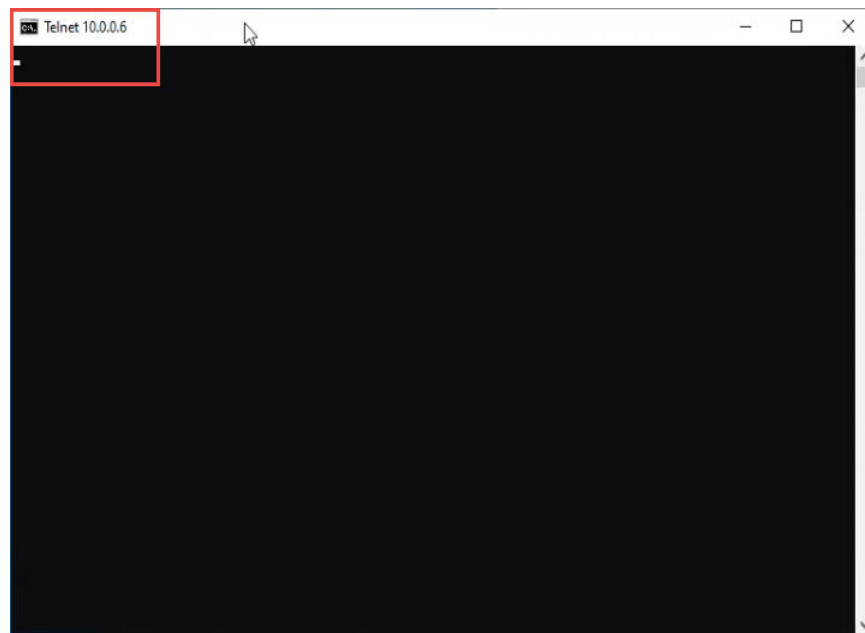
telnet <nodeName または ipAddress> <portNumber>

各項目の説明

- **nodeName** は、InTouch HMI ランタイム ノードのマシン名です。nodeName か ipAddress のいずれかを使用してください（両方使用することはできません）。
- **IPAddress** は、InTouch HMI ランタイム ノードの IP アドレスです。nodeName か ipAddress のいずれかを使用してください（両方使用することはできません）。
- **portNumber** は OPC UA サービス用に InTouch HMI で設定したポート番号です。デフォルトのポート番号は 48032 です。

例: telnet 10.10.10.06 48031

- コマンドが正常に処理されない場合はタイムアウトが発生して、接続に失敗したことを示すメッセージが表示されます。その場合、「[ランタイム ノードのファイアウォールの設定](#)」に進んでください。
- Telnet コマンドが正常に処理された場合、コマンドプロンプトが Telnet プロンプトに変わります。



- ファイアウォールのテストが正常に完了したら、OPC UA クライアントおよび OPC UA サーバー証明書を設定します。OPC UA 接続のセットアップの次のステップは、サードパーティ製 OPC UA クライアント アプリケーションを使用するか、Gateway 通信ドライバで使用可能な OPC UA 接続を使用するかに応じて異なります。

サードパーティ製 OPC UA クライアント アプリケーション用のサーバーおよびクライアント証明書の設定

重要! これらの手順は、サードパーティ製 OPC UA クライアントを使用する場合にのみ適用されます。OPC UA クライアントとして Gateway 通信ドライバを使用する場合は、[「OI Gateway を使用したクライアントセキュリティ証明書の設定」](#)を参照してください。

InTouch OPC UA サーバーとサードパーティ製 OPC UA クライアントの間の通信を暗号化するには、両方のコンピュータに以下の証明書へのアクセスが必要です。

- <コンピュータ名> ASB OPC UA サーバー
- OPC UA クライアントからのクライアント証明書。

このセットアップを完了するには、3 つの手順を実行する必要があります。

1. OPC UA サーバー ノードから証明書を OPC UA クライアント ノードにコピーします。この手順には以下の操作が含まれます。
 - OPC UA サーバー証明書のエクスポート。
 - 証明書のクライアントノードへのインストール。
2. OPC UA クライアント ノードの証明書の OPC UA サーバー ノードへのコピー。
3. ファイアウォールが InTouch.OPCUA.ServiceHost.exe アプリケーションを許可するように設定されていることを確認してください。

OPC UA クライアント ノードへの OPC UA サーバー証明書のエクスポート

OPC UA サーバー証明書をエクスポートするには

1. OPC UA サーバー ノードで Windows の証明書マネージャを開きます。

証明書マネージャを開くには、Windows の検索ボックスに「コンピュータ証明書の管理」と入力するか、コマンドプロンプトを開いて `certlm.msc` を実行します。
2. ツリー ビューで [個人] ノードを展開して、[証明書] をクリックします。
3. 「<コンピュータ名> ASB OPC UA サーバー」証明書を選択します。
4. 証明書を右クリックして、[すべてのタスク] > [エクスポート] を選択します。

[証明書のエクスポート ウィザード] が開きます。
5. クライアントアプリケーションが使用する証明書のタイプに応じて、証明書を 1 つまたは 2 つの証明書ファイルタイプとしてエクスポートする必要があります。
 - **.cer ファイル:** クライアントが Windows 証明書ストアを使用する場合
 - **.der ファイル:** クライアントがファイルベースの証明書を使用する場合

ファイル拡張子が異なってもファイル形式は同じです。
6. [証明書のエクスポート ウィザード] で以下のオプションを選択します。
 - **秘密キーのエクスポート:** [いいえ、秘密キーをエクスポートしません] (デフォルト) を選択し、[次へ] をクリックします。
 - **エクスポートファイルの形式:** クライアントアプリケーションの要件に応じて [DER エンコードされたバイナリ X.509 (.CER)] または [Base-64 でエンコードされた X.509 (.CER)] のいずれかを選択します。選択を行って [次へ] をクリックします。
 - **エクスポートするファイル:** ルート CA をエクスポートするファイル名 (`c:\temp\<マシン名> OPC UA Server.cer` など) を入力し、[次へ] をクリックします。

7. [証明書のエクスポート ウィザード] の処理が完了した後、クライアントアプリケーションの要件に応じて、証明書のファイル拡張子を ".cer" から ".der" に変更する必要がある場合があります。この操作は、OPC UA クライアントアプリケーションで証明書が Windows 証明書ストアではなく、特定のフォルダに格納される場合に行う必要があります。

クライアントコンピュータへの OPC UA サーバー証明書のインポート

OPC UA サーバー証明書のコピーと OPC UA クライアント ノードへのインストールの手順を完了するには、OPC UA サーバー証明書を OPC UA クライアント コンピュータにインポートする必要があります。

各 OPC UA クライアントアプリケーションには、証明書を管理するための独自のメカニズムがあります。一般的に、OPC UA クライアントは、次の 2 つのメカニズムのいずれかを使用して証明書を管理します。

- Windows 証明書ストアの利用
- OPC UA クライアントアプリケーションで定義された特定のフォルダへの証明書の格納。

サーバー証明書をインポートする方法の詳細については、使用する OPC UA クライアントアプリケーションのドキュメントを参照してください。

証明書を OPC UA クライアントの Windows 証明書ストアにインポートするには

1. 証明書ファイル (<マシン名> OPC UA Server.cer) を OPC UA クライアント ノードにコピーします。ファイルをコピーする場所は重要ではありません。
2. 証明書ファイルを右クリックし、コンテキストメニューから [証明書のインストール] を選択します。

証明書のインポート ウィザードが開きます。

注記:証明書をインストールするには管理者特権が必要です。

3. 証明書のインポート ウィザードで以下のオプションを選択します。
 - **保存場所:** [ローカル コンピュータ] を選択して [次へ] をクリックします。
 - **証明書ストア:** 「個人」を参照して [次へ] をクリックします。
 - **証明書のインポート ウィザードの完了:** 設定を確認して [完了] をクリックします。

証明書を OPC UA クライアント上の特定の場所にコピーするには

1. 証明書ファイル (<マシン名> OPC UA Server.cer) を OPC UA クライアントアプリケーションで指定されているフォルダにコピーします。以下の表には、一般的な OPC UA クライアントの証明書フォルダの場所が示されています。

OPC UA クライアント	製造業者	フォルダ
dataFEED OPC UA クライアント	Softing	C:\ProgramData\Softing\OpcClient\pki\trusted\certs
UaExpert	UnifiedAutomation	C:\Users\Admin\AppData\Roaming\unifiedautomation\uaexpert\PKI\trusted\certs
UA Client Getting Started	UnifiedAutomation	C:\ProgramData\unifiedautomation\UaSdkNetBundleEval\pkiclient\trusted\certs

OPC UA クライアント	製造業者	フォルダ
Matrikon	Matrikon	C:\Users\Admin\AppData\Local\Matrikon\OPCUAExplorer\pki\DefaultApplicationGroup\trusted\certs
KEPServer	Kepware	C:\ProgramData\Kepware\KEPServerEX\V6\UA\Client Driver\cert
Top Server	Software Toolbox	C:\ProgramData\Software Toolbox\TOP Server\V6\UA\Client Driver\cert

証明書の管理の詳細については、該当する OPC UA クライアント アプリケーションのドキュメントを参照してください。

2. 以下に示すように OPC UA 証明書を設定します。

OPC UA サーバー上での OPC UA クライアント証明書の設定

OPC UA サーバーおよびクライアント証明書の設定の次のステップは、OPC UA サーバー ノードにインストールされている OPC UA クライアント証明書を信頼することです。

最も簡単な方法は、OPC UA クライアントをサーバーに接続してみることです。クライアント証明書の信頼はまだ確立されていないので、接続は失敗することが予想されます。

接続が失敗すると、OPC UA サーバーにインストールされていない OPC UA クライアント証明書は OPC UA サーバーの “Rejected Certificate” (拒否された証明書) フォルダに配置されます。

フォルダのデフォルトの場所を以下に示します。

C:\ProgramData\AVEVA\PCS\OPC UA Rejected Client Certificates\certs

注記: このフォルダにアクセスするには管理者権限が必要です。このフォルダはデフォルトで非表示になっています。

Rejected Certificate フォルダに配置された証明書をインポートするには

1. OPC UA クライアント証明書をインポートするには、Rejected Certificate フォルダを参照します。
2. 信頼する OPC UA クライアントの証明書を右クリックして、[証明書のインストール] を選択します。
証明書のインポート ウィザードが開きます。
3. ウィザードで以下のオプションを選択します。
 - **保存場所:** [ローカル コンピュータ] を選択して [次へ] をクリックします。
 - **証明書ストア:** [信頼されたユーザー] を選択して [次へ] をクリックします。
 - 証明書のインポート ウィザードの完了: 設定を確認して [完了] をクリックします。

ポートの使用

OPC UA は単一の TCP ポートを介して通信します。これは、OPC UA サーバーの **エンドポイント接続設定** で指定されています。デフォルトでは、ポートは 48032 に設定されています。詳細については、「[InTouch OPC UA サーバーの設定](#)」を参照してください。

RDS セッションを使用して同一のコンピューター上で複数の OPC UA サーバーを実行する場合、後続の各サーバーに対して別々のポート番号を指定する必要があります。

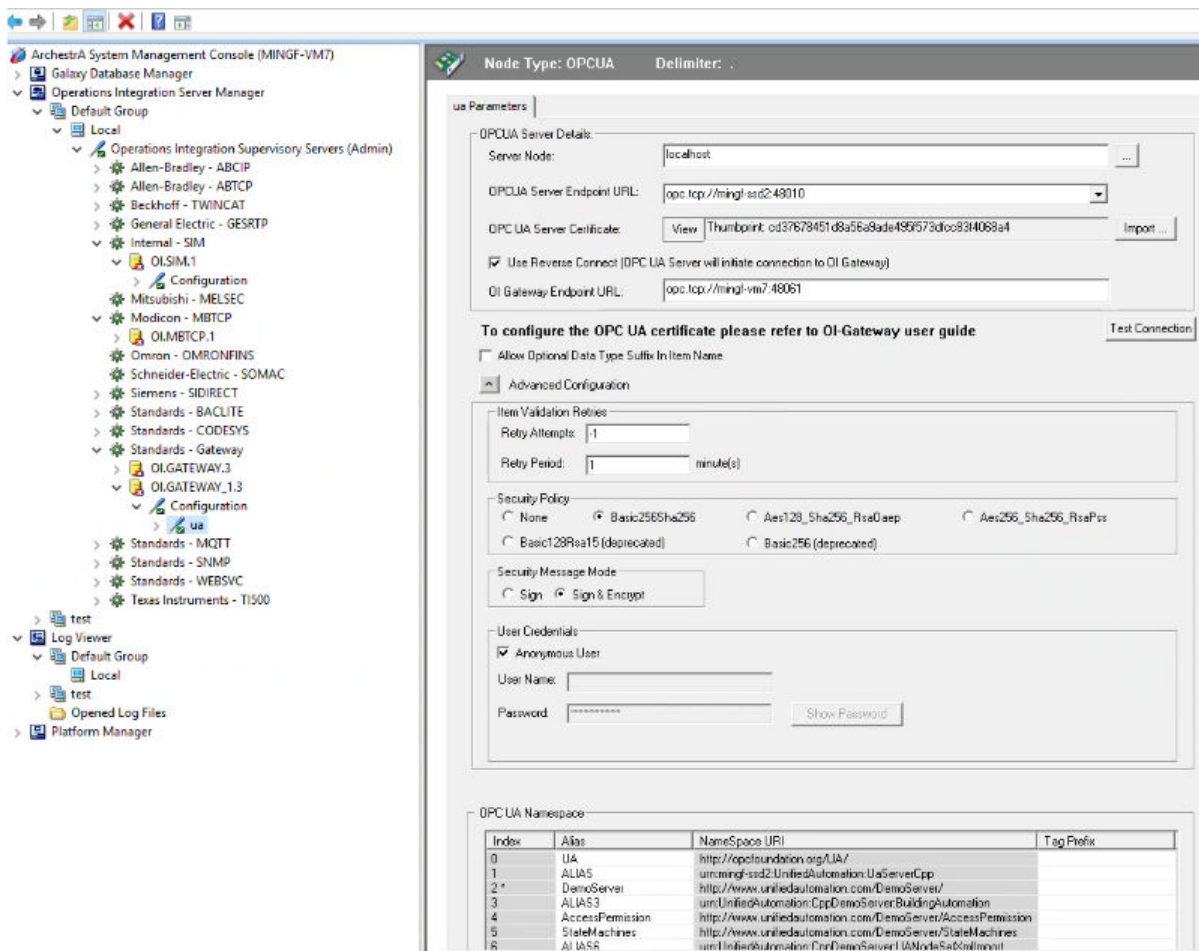
このポートがコンピュータにインストールされているファイアウォールソフトウェアによってブロックされないことを確認してください。ファイアウォールを介した通信のテストと設定については、「[OPC UA サービス用のファイアウォールの設定](#)」を参照してください。

OI Gateway を使用したクライアント セキュリティ証明書の設定

Gateway 通信ドライバを使用すると、サーバーおよびクライアント OPC UA ノードでセキュリティ証明書を容易に設定できます。

Gateway 通信ドライバを介してセキュリティ証明書を設定するには

- ランタイムノードの「スタート」メニューから Operations Control Management Console を開きます。
（「スタート」 > 「AVEVA」 > 「Operations Control Management Console」）



- コンソールツリーで Operations Integration Supervisory Servers の下にある **OI.GATEWAY.3** ノードに移動します。
- OPC UA 接続を作成します。
- OPC UA サーバーの詳細を設定します。
 - サーバー ノード: ランタイム ノードのマシン名を入力します。

- **OPC UA サーバー:** これは OPC UA サーバー（ランタイム ノード）の URI（Uniform Resource Identifier）です。アドレスは、この時点で探索できないので手動で入力する必要があります。「opc.tcp://<マシン名>:<OPC UA ポート番号>」の形式で入力します。

InTouch OPC UA サーバーを設定するときに InTouch HMI アプリケーションマネージャで入力した OPC UA ポート番号を使用します。デフォルトのポート番号は 48032 です。

5. 承認および認証の資格情報を入力します。

OPC UA サーバー ダイアログで設定した承認設定と一致する必要があります。[セキュリティ認証が必要] チェックボックスがオンになっている場合、以下の設定を選択する必要があります。

- **セキュリティ ポリシー:** Basic256Sha256。
- **セキュリティ メッセージ モード:** 署名と暗号化。
- 匿名アクセスを許可するには、ユーザー資格情報の下で [匿名ユーザー] を選択します。OPC UA 設定時に該当するオプションを選択した場合は、認証済みユーザーのユーザー資格情報を入力することもできます。入力するユーザー資格情報は InTouchHMIOPCUAWriteUsers ユーザー グループのものである必要があります。

6. [テスト] ボタンをクリックします。テストは失敗しますが、OPC UA 証明書がダウンロードされます。

重要! この最初のテストが失敗する理由は、クライアントおよびサーバー アプリケーションの間で証明書が信頼される必要があるからです。この問題は証明書をインストールすることで解決します。

7. 次のセクションに進みます。

証明書が信頼されたら、OPC UA クライアント設定を検証する必要があります。

OPC UA サーバーおよび OPC UA クライアントの間での証明書の信頼

この OPC UA サーバー サービスのリリースでは、OPC UA サーバーと OPC UA クライアントの間で信頼を作成する操作は手動で行う必要があります。テスト操作を実行すると、Gateway 通信ドライバによって独自の証明書が OPC UA サーバー ノードに送信されます。この証明書は信頼できます。以下の手順は、OPC UA サーバー ノードからクライアント証明書を信頼する方法を示します。Gateway 通信ドライバを使用する場合は、[「サードパーティ製 OPC UA クライアントアプリケーション用のサーバーおよびクライアント証明書の設定」](#)を参照してください。

1. C:\ProgramData\AVEVA\PCS\OPC UA Rejected Client Certificates フォルダにアクセスします。;

これは、OPC UA サーバーへの接続を試行する際にクライアントからの証明書がデフォルトで最初に配置される場所です。

注記: ProgramData フォルダはデフォルトで非表示になっています。非表示の項目を表示するには、Windows エクスプローラーで非表示の項目のオプションを有効にする必要があります。

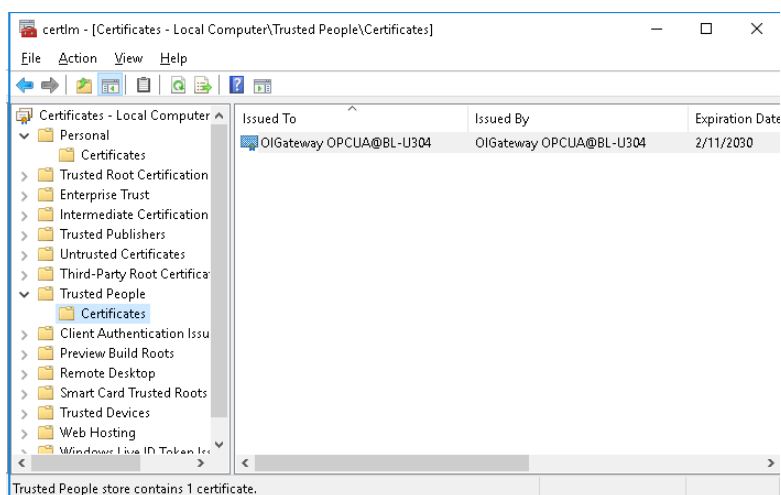
2. 証明書名（OIGatewayOPC UA@OPCUA client node{long hex ID}.der など）を右クリックします。
3. コンテキストメニューから [証明書のインストール] を選択します。証明書のインポート ウィザードが開きます。
4. 保存場所に [ローカル マシン] を選択し、[次へ] をクリックします。
5. [証明書ストアの選択] リストで証明書ストアとして [信頼されたユーザー] を選択します。これは OPC UA 証明書で機能する唯一のオプションです。

6. ウィザードを閉じてインストールを完了します。
7. 最後に、証明書がインストールされたので、**OPC UA Rejected Client Certificates** フォルダから証明書を削除します。

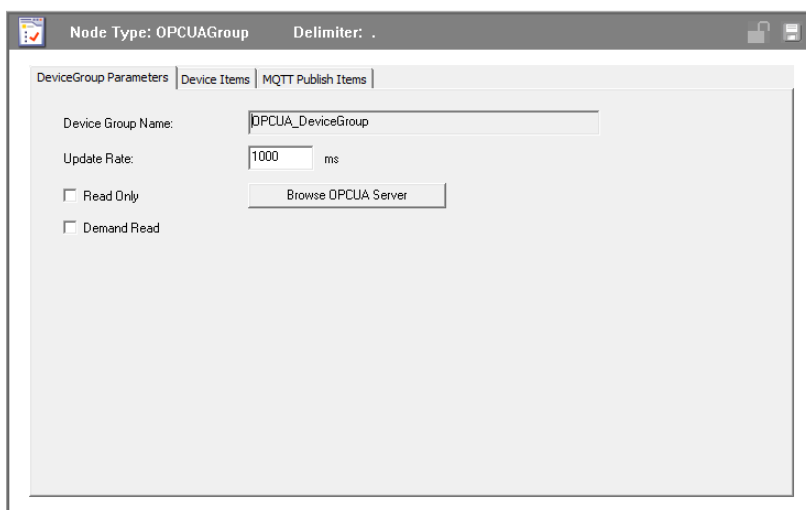
OPC UA 証明書のインストールの検証

証明書のインストールを検証するには

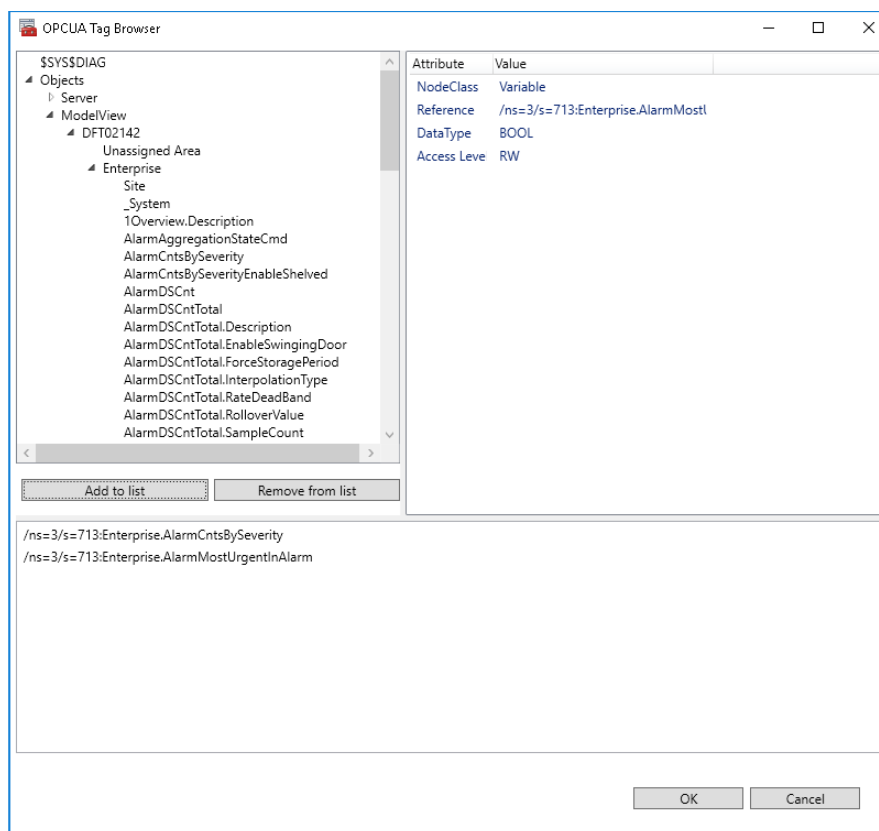
1. Windows の検索ウィンドウに「証明書」と入力し、検索結果から [コンピュータ証明書の管理] を選択して証明書マネージャを開きます。証明書マネージャが開きます。
2. [信頼されたユーザー] フォルダに移動して、OPC UA 証明書がインストールされていることを確認します。



3. OPC UA クライアントノードから **Operations Control Management Console** を再度開きます。OPC UA パラメータ ウィンドウで [接続テスト] ボタンをクリックします。ウィンドウの下部にある **OPC UA** ネームスペースのエイリアスリストが自動的に更新され、接続が正常に確立されたことが示されます。
4. OPC UA 接続の下にグループを追加します。入力したグループ名にデバイス グループ名の接頭辞 **OPCUA_** が付けられます。



5. **[OPC UA サーバーを参照]** ボタンをクリックして OPC UA 会葬を参照します。
6. オプション: OPC UA タグを追加してエイリアス リストを監視すると、ネームスペースを確実に参照できます。



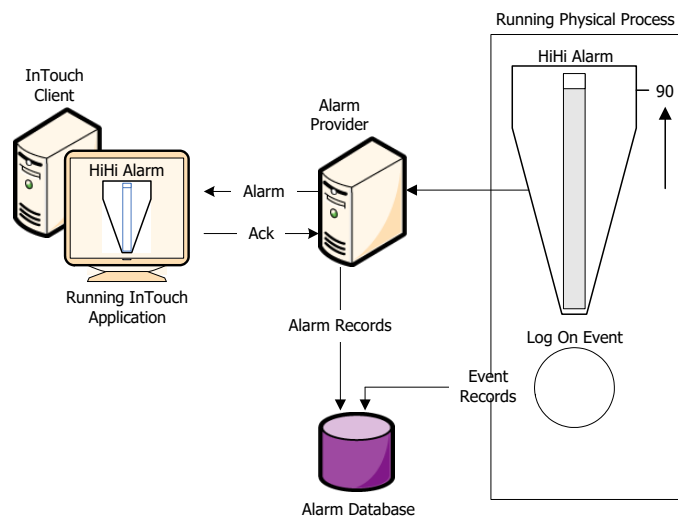
7. **Operations Control Management Console** で OI Gateway の設定に戻り、OPC UA タグが **[デバイス アイテム]** ウィンドウに一覧表示されていることを確認します。
8. デフォルトでは、リストのアイテム名は完全なアイテム参照パスと重複します。必要に応じてアイテムの名前を変更します。

章 4 アラーム

アラームとイベントを生成してオペレータにプロセスの動作状況を通知する InTouch アプリケーションを作成できます。

- 問題を引き起こす可能性のあるプロセス状態に関する警告が、アラームによってランタイム オペレータに警告されます。通常は、プロセス値が定義済みのしきい値を超えた場合にトリガするようにアラームを設定します。オペレータは、アラームを確認する必要があります。
- イベントは、通常のシステム ステータス メッセージを表します。オペレータが InTouch アプリケーションにログオンするなど、システム条件が発生するときに典型的なイベントとなります。オペレータは、イベントを確認する必要はありません。

以下の図は、アプリケーションが実行されている間、InTouch HMI でアラームとイベントが処理される方法を示しています。アラームとイベントのデータは、アラーム データベースに保存されます。



イベント監視用のタグを設定できます。イベントメッセージは、タグの値が変更するたびにアラーム システムにログ記録されます。イベントメッセージには、値がどのように変更されたかに加えて、オペレータ、I/O、スクリプト、またはシステムのいずれによってその変更が開始されたのかという情報が含まれます。

InTouch アラームの概要

アラームは、問題の原因となるプロセス状態が発生したためにオペレータの応答を要求する警告です。アラームは通常、アナログ値がしきい値の上限を超えた場合など、プロセス値がユーザー定義しきい値の範囲を超えた場合に発生します。この状況によってアラームがトリガされ、オペレータに問題の発生が通知されます。オペレータがアラームを確認した後で、InTouch HMI はアラームが確認されたことを認識します。

アラームの原因となっている条件が解決された場合でも、InTouch HMI でアラームの確認を要求するように設定できます。これによって、一時的なアラーム状態の原因となったものの通常状態に戻ったイベントについてオペレータが認識していることを確認できます。

主なアラーム状態を以下の表で説明します。

アラーム状態	条件
ACK	アラームが確認されました。
ALM	アラームが発生しました。
RTN	タグがアラーム状態から通常状態に戻りました。
LATCHED	アラームが "UNACK_RTN" 状態から確認されたか、アラームが "ACK" 状態から通常状態に戻りました。

アラーム優先度

アラームに優先度（重要度）を割り当てます。たとえば、ボイラー温度のしきい値には高優先度のアラームが要求され、迅速な対応を必要とします。作業交代を通知するアラームの優先度はかなり低くなります。アラームの優先度は通常、工場アプリケーション、設備の性質、安全性、バックアップシステムの利用状況、考えられる損害費用、またはダウンタイムなど、さまざまな状況によって異なります。

アラームの優先度は、タグ変数を定義する際に割り当てます。優先度は 1 ～ 999 の範囲で設定でき、1 が最も重要度が高くなります。

アラーム優先度の範囲を指定して、アラームの分類を表すことができます。たとえば、プロセスで 4 つのレベルの重要度を必要としている場合、4 つの優先度の範囲を作成できます。

アラームの重要度	優先度の範囲
最高	1 ～ 249
高	250 ～ 499
低	500 ～ 749
情報	750 ～ 999

範囲を設定すると、アラームをフィルタする際に役立ちます。たとえば、重大アラーム以外のすべてのアラームをフィルタするようにアラーム表示を設定できます。すべてアラーム優先度の範囲に基づく、アニメーションリンク、確認スクリプト、およびフィルタされた表示や印刷を作成できます。

アラームのサブステート

マルチ状況アラームには、アラームのサブステートの範囲が含まれています。たとえば、アナログアラームには通常、いくつかのしきい値があります。

- Hi と Lo のしきい値で、正常な動作範囲の境界が設定されます。
- HiHi と LoLo のしきい値は、値の正常範囲からの極端な偏差をマーク付けします。

ボイラー温度レベルでは、これらのサブステートのいずれが発生した場合でもアラーム状況となります。ボイラー温度は、全体的なアラーム状況を維持したまま、2 つのサブステート間で切り替わることもできます。

アラームの確認

アラームの発生時には、ランタイム オペレータ（またはシステム）はアラームを確認する必要があります。確認とは、単にアラームを認識していることを意味します。修正操作を行うこととは別で、修正操作はすぐに行われるとは限りません。確認は、アラーム条件が通常状態に戻ることも異なります。外部からの干渉がない場合でも自然にアラーム条件が通常状態に戻ることもあります。

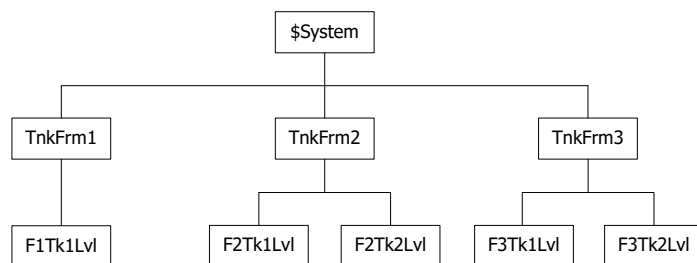
高い優先度または中間の優先度アラームでは、通常は迅速な確認が必要ですが、優先度の非常に低いアラームでは迅速な確認は必要とされません。アラームが生成される原因となった状況が自然に解消される場合（温度が高くなりすぎた後に温度が下がった場合など）であっても、確認されるまでそのアラームは解決されたとは見なされません。

アラームを確認すると、オペレータはアラームを解除できます。ラッチが有効な場合、通常状態に戻った確認済みアラームは、**LATCHED** アラーム状態になります。オペレータは、**LATCHED** アラームを解除して、アラーム クライアント コントロール グリッドの現在のモードから **LATCHED** アラームを削除できますが、アラームはアラーム クライアント コントロールの最近のモードに引き続き表示されます。

アラーム グループ

アラームをグループ化すると、追跡と管理を簡単に行えるようになります。アラーム グループは、工場の異なる領域、設備、オペレータの責任、または製造工程などを論理的に表したものです。

たとえば、以下の図はタンク ファーム アプリケーションの 3 層のアラーム グループ階層を示しています。



アラーム グループは、アラーム表示、アラーム プリンタ、および確認スクリプトでのフィルタに役立ちます。

各タグ変数は 1 つのアラーム グループに関連付けられています。デフォルトでは、タグ変数はメインの **\$System** グループに割り当てられます。**\$System** グループの下に、最大 32 レベルまでの追加のアラーム グループ階層を作成できます。

タグ変数ディクショナリでタグ変数を定義している間、アラーム グループを作成して、タグ変数をそのアラーム グループに関連付けることができます。

アラーム グループおよびグループ変数には **SmartSymbol** との互換性がありません。アラーム グループへの参照またはグループ変数を **SmartSymbol** 内で使用することはできません。

InTouch イベントの概要

イベントはシステム内で発生している検出可能な出来事で、必ずしもアラームに関連付けられているとは限りません。アラーム状況が発生するという移行やアラーム状況が解決されるという移行は、イベントの一種です。オペレータの操作、システム設定に対する変更、またはある種のシステム エラーなどもイベントと考えられます。

イベントは状況とは異なります。状況は数分、数時間、数日間、または数週間続くこともあります。イベントは一瞬だけのもので、発生した直後に終了します。アラームは状況であり、アラーム通知はイベントです。

イベントは正常のシステム ステータス メッセージであり、オペレータの応答は必要ありません。

イベント監視を行うようにタグを定義する場合、タグ値が変更されるたびにイベント メッセージがアラーム システムに印刷またはログ記録されるように選択できます。イベント メッセージには、値がどのように変更されたかに加えて、オペレータ、I/O、スクリプト、またはシステムのいずれによってその変更が開始されたのかという情報が含まれます。

イベントのタイプは以下のいずれかとなります。

イベント	状況
OPR	オペレータが値入力を使用してタグ値を修正しました。
LGC	QuickScript 関数でタグ値が修正されました。
DDE	DDE クライアントからタグ値がポークされました。
PROT	産業用グラフィック（カスタム プロパティまたは産業用グラフィックのプッシュボタン） からタグ値の変更が開始されました
SYS	システム イベントが発生しました。
USER	\$Operator が変更されました。

SYS および USER イベントは、イベント ログが有効になっているかどうかに関係なく、システムによってすべてのタグに対して生成されます。DDE、OPR、および LGC イベントはタグ値に関係があり、イベント ログが有効になっているタグに対してのみ生成されます。

InTouch アラームのタイプ

InTouch HMI 内では、アラームはその特性に基づいて一般的なカテゴリに分類されます。「クラス」と「タイプ」という一般カテゴリに分類されます。すべてのアラームは、分散アラーム システムにより、論理値、値、偏差、変化率、および SPC という 5 つの一般的な状態に分類されます。

アラーム状況	分散クラス	分散タイプ
論理型	DSC	DSC
値 - LoLo	VALUE	LOLO
値 - Lo	VALUE	LO
値 - Hi	VALUE	HI
値 - HiHi	VALUE	HiHi
偏差 - 大偏差	DEV	MAJDEV
偏差 - 小偏差	DEV	MINDEV

アラーム状況	分散クラス	分散タイプ
変化率	ROC	ROC
SPC	SPC	SPC

各 InTouch タグ変数を定義する際、アラーム状況に関連付けます。タグ変数のタイプに基づいて、1つまたは複数のアラームクラスやタイプを定義できます。

論理値アラーム

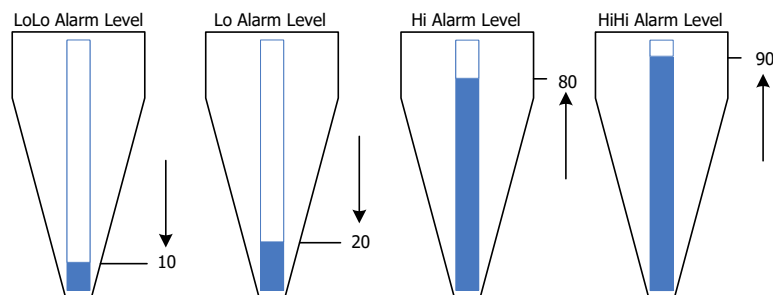
論理値アラームは、2つのうちいずれかの状況となる論理型タグ変数に対応します。論理型タグ変数を作成する際、そのアラーム状況がタグ変数の **true** または **false** のどちらの状況に対応するのかが設定します。

アナログ アラーム

アナログアラームは、整数値か実数値に関連付けられるアナログ型タグ変数に対応します。アナログアラームのタイプの中で、値、偏差、および変化率といういくつかのサブタイプがあります。

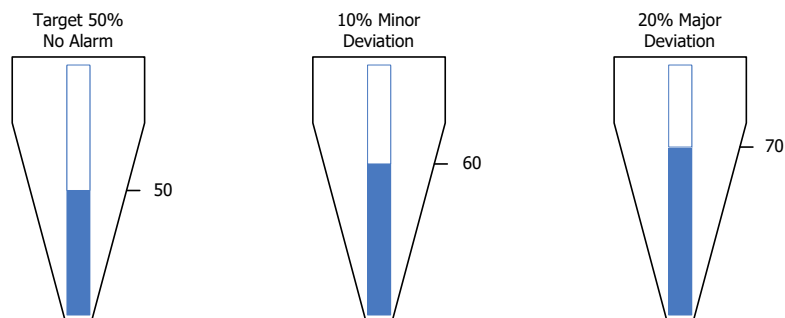
値アラーム

現在のタグ変数値は、事前に定義されている1つまたは複数のしきい値と比較されます。値がしきい値を超えると、アラーム状況が宣言されます。「LoLo」、「Lo」、「Hi」、および「HiHi」のしきい値に対して値および優先度を個別に設定し、各しきい値が使用されるどうかを指定できます。



偏差アラーム

現在のタグ変数値が目標値と比較され、差異の絶対値がタグ変数値の範囲のパーセントとして表される1つまたは複数のしきい値と比較されます。



小偏差と大偏差のしきい値に対する値および優先度を個別に設定して、各しきい値が使用されるかどうかを指定できます。また、タグ変数の範囲のパーセントとして表される偏差デッドバンドの値を設定することもできます。これにより、アラーム状況であると評価される前にタグ変数値が変化する必要のある（合計範囲の）パーセントを制御できます。

たとえば、しきい値は以下のように設定します。

- 0 ～ 100 の範囲
- 基準値 50
- 小偏差 10 パーセント。小偏差のしきい値の範囲を 40 ～ 60 に設定します。
- 大偏差 20 パーセント。大偏差のしきい値の範囲を 30 ～ 70 に設定します。
- デッドバンド 10 パーセント

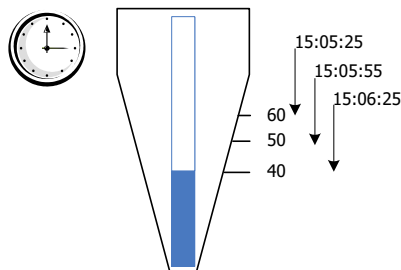
タグ変数の値が 39 の場合、小偏差アラームが発生します。ただし、アラームが評価されて消去されるには、値が少なくとも 50（40 とデッドバンド 10）まで変化する必要があります。

タグ変数の値が 72 の場合、大偏差アラームが発生します。アラームが消去されるには、この値が少なくとも 61 まで低下する必要があります。

変化率アラーム

タグ変数の現在の値と以前の値が、一定の測定期間で比較されます。タグの値の変化率は、以前のタグ値、以前の変更が行われた時間、現在のタグ値、および現在の時間を使用して計算されます。

タグの値が変更されると変化率アラームのテストが行われます。1 つの例外は、WindowViewer 内でアプリケーションの実行が開始するときの初期タグ値です。その場合、初期値は無視され、最初の変更率の比較はタグの値の 2 番目と 3 番目の変更の間で行われます。その後、変更率の測定は、連続するタグ値の変更の間で行われます。



連続するタグの変更の間における値の絶対的な差分が指定された制限を超えた場合、変更率アラームが発生します。変更率の制限は、一定の時間間隔（秒、分、または時間単位）におけるタグの値のパーセンテージとして表されます。変化率（ROC）のしきい値の値および優先度や、そのしきい値が使用されるかどうかも設定できます。

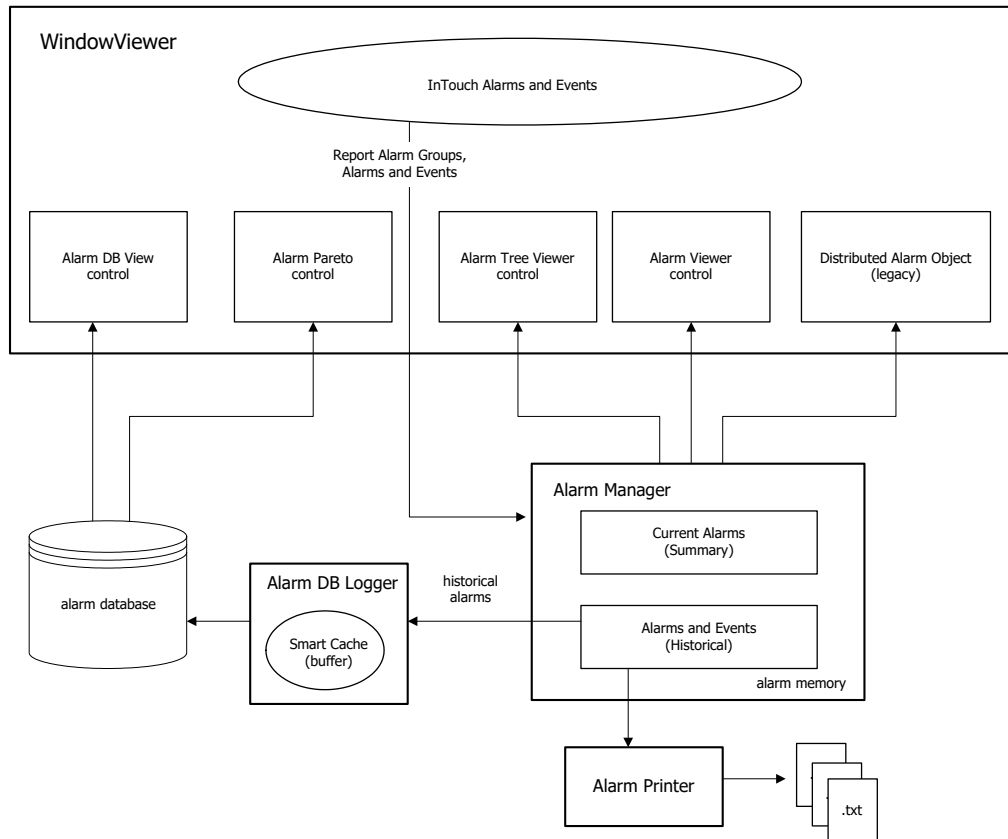
InTouch 分散アラート システム

分散アラームシステムは、以下の要素で構成されています。

- アラーム マネージャ。現在アクティブなアラーム（サマリ アラーム）および履歴アラームとイベントを管理します。サマリ アラームおよび履歴アラームは、InTouch の内部アラーム メモリに保持されます。
- Alarm DB Logger。履歴アラームとイベントをアラーム データベースに保存します。アラーム データベースは、SQL Server データベースです。

- **Alarm Printer。**履歴アラームとイベントを印刷します。
- 一組の **ActiveX** コントロール。ランタイムで内部アラーム メモリまたはアラーム データベースのいずれかからアラームとイベントを取得します。

以下の図は、システムの概要を示しています。



重要：分散アラーム システムは **Windows** サービスのセットとして実行します。管理者権限で分散アラーム システムを実行する場合にセキュリティの脅威にさらされる可能性を削減するために、ユーザー アカウントの許可は、これらのサービスに対して非対話型に設定されています。

ランタイム オペレータは、分散アラーム システムを使用して、以下の操作を実行できます。

- ローカルの **InTouch** アプリケーションとその他のネットワーク アプリケーションのアラーム システムによって生成されたアラームやイベントを表示、ログ記録、および印刷します。
- ローカルまたはリモート ネットワーク ノードから、アラームを確認します。
- **InTouch** アプリケーションでアラーム **ActiveX** コントロールを使用して、事前設定したアラーム表示を表示します。
- 個別のアラーム コメント フィールドを使用して、アラームに関する詳細なフィードバックを提供します。

アプリケーション開発者は、以下の操作を実行できます。

- ドットフィールドを使用して、アラームを制御します。
- アプリケーションのフルコントロール下で、アラームを直接または間接的に有効にしたり、無効にしたりできるように、アラームを設定します。特定の表示ノードでアラーム情報の表示を防ぐため

に、単一のアラーム クラス、タグ変数、またはグループにアラーム抑止を適用できます。システム全体で無効にすると、アラーム動作をソースでブロックできます。

- アラーム情報が履歴にログ記録されるように設定します。**Alarm DB Logger** は、**Windows** サービスとして実行することも、必要に応じて手動で起動することもできます。アラーム ロギングでは **UCT (GMT)** のタイムスタンプを使用しているため、**DST** やさまざまなタイムゾーンとの互換性を提供しています。
- フェイルオーバー アラーム プロバイダを設定します。プライマリ アラーム プロバイダが失敗した場合でも、分散アラーム システムはバックアップ システムからアラーム情報をシームレスに取得します。プライマリ ノードが再接続されると、分散アラーム システムは回復しているプライマリ システムがライブ状態になる前に、アラーム確認が再同期化されることを確認します。

分散アラーム システムでは、以下の処理が行われます。

- **SuiteLink** プロトコルを通じてデータを送信して、最小限の **CPU** リソースやネットワーク リソースを使用します。
- タイムスタンプは、コンシューマがアラームを受け取る時ではなく、アラームが発生した時に設定されます。タイムスタンプにはミリ秒も含まれます。

アラーム プロバイダとコンシューマ

どのノードでも、アラーム プロバイダ（発行側）とアラーム コンシューマ（要求側）を持つことができます。**InTouch** 分散アラーム システムでは、通信リンクを提供して、ノードやソフトウェア コンポーネント間でアラーム情報を渡すことができます。

アラーム プロバイダ

アラーム プロバイダは、以下の処理を実行します。

- アラーム可能なアイテム、つまりアラーム状況に移行する可能性のあるアイテムを追跡して、これらのアイテムの階層グループ化に関する情報を含む、アイテムのリストを分散アラーム システムに提供します。
- アラーム アイテムのステータスが変更されるとき、分散アラーム システムに通知します。ステータスの変更には、アイテムがアラーム状態に移行したかどうか、アラーム状態から戻ったかどうか、また最新のアラームが確認されたかどうかなどが含まれます。
- アラーム アイテムが無効になったかどうかを追跡します。

InTouch HMI は、**QI Analyst**、**Application Server Galaxy**、およびアラーム **API** ツールキットで作成されたその他のソフトウェアなどの外部アラーム プロバイダをサポートしています。これらのアラーム レコードの日付／時間スタンプはアラーム プロバイダによって提供され、分散アラーム システムによっては生成されません。

アラーム コンシューマ

アラーム コンシューマは、以下の処理を実行します。

- 通知を受け取ることを希望するアラーム可能アイテムを識別するクエリー セットを分散アラーム システムに提供します。クエリーはアラーム コンシューマによって変更または削除されるまでアクティブ状態を保ち、アラーム プロバイダやアラームのグループを指定します。これは、「ワイルドカード」を持つ **SQL** クエリーとよく似ています。アラーム プロバイダが変更の通知を発行するたびに、分散アラーム システムは登録されているすべてのクエリーとの一致についてアラームを確認し、対応するアラーム コンシューマにアップデートを渡します。

- アップデートを受け取ると、アイテムのステータスまたは移行に関連する情報を表示するか、ログ記録します。
- アラームを確認します。アラーム コンシューマは、アラームとアラーム プロバイダを識別する確認通知を分散アラーム システムに送信します。この通知はアラーム プロバイダに渡され、そのプロバイダによってアイテムのステータスが「確認」に更新され（適切な場合）、それを分散アラーム システムに通知することによって、その更新がそのアラームに関連を持つすべてのコンシューマに配布されることを確認します。

注: 分散アラーム システムの通信の大部分は、ノード間でのアラーム クエリーとアラーム レコードの送信により構成されています。ノード内では、ネットワーク トラフィックを最小限に抑えるために、内部アラーム メモリによってアラーム クエリーとアラーム レコードが追跡され、バッファされています。

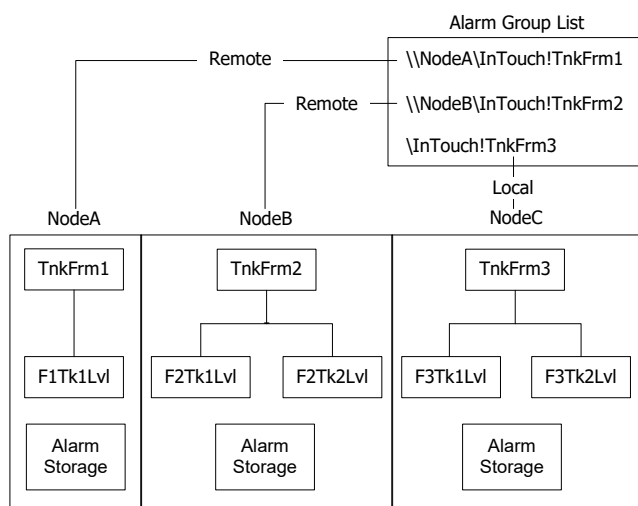
分散アラーム グループ リスト

分散アラーム システムでは、アラーム グループを使用して、ローカル ツリー ビューを構成します。分散アラーム 表示では、このツリービューを使用して、アラームをフィルタします。これらのアラーム グループは、ネットワークの複数のノードから表示できます。

分散アラーム システムでは、アラーム グループ リストを使用して、ローカル ノードとリモート ノードのアラーム グループを結合します。アラーム グループ リストは、InTouch ノードとこれらの各ノードで定義されるアラーム グループで構成される名前付きのリストです。このリストには、その他のアラーム グループ リストの名前やローカル アラーム グループも含めることができます。Alarm Viewer コントロールなどのアラーム コンシューマは、このリストを使用してアラームのクエリーを実行します。

クエリー エイリアスで、Alarm Viewer コントロールではリストに所属するグループからの組み合わせられたアラームを表示できます。アラームはローカルの InTouch ノードまたはネットワークのリモート ノードから確認できます。

以下の図は、3 つのノードからのアラームグループを結合するアラーム グループ リストを示しています。アラーム グループ リストは NodeC でローカルに定義されます。残りのアラーム グループは、リモート ノードからのグループです。



分散アラーム 表示では、このリストに所属するすべてのアラーム グループに対するクエリーから結果として生じるアラームが表示されます。たとえば、いくつかの InTouch ノードにあるタンク ファーム アラームをすべて表示する場合は、**TankFarmAlarms** という名前のリストを作成できます。このリストに、

タンク ファームの InTouch アプリケーションを実行するすべてのノードからのアラーム グループを追加します。

アラーム グループの作成の詳細については、「[アラーム グループの作成](#)」を参照してください。

サマリ アラームと履歴アラーム

サマリ アラームは、現在アクティブなアラームです。履歴アラームは、現在アクティブではなく、通常はアラームデータベースに保存されているアラームです。

たとえば、確認を待機している現在のアラームすべてのサマリを表示して、その他のアラーム情報は緊急性が低いため履歴として記録することができます。

アラームの無効化、抑止、非表示

アラームは「オフ」にするか、アラーム設定を実際に削除することなく無視することができます。アラームは無効にするか、抑止するか、非表示にすることができます。

アラームの無効化と抑止は、アラーム プロバイダで制御されます。非表示は、アラーム コンシューマで制御されます。プロバイダとコンシューマの詳細については、「[アラーム プロバイダとコンシューマ](#)」を参照してください。

- **無効化。** アラームは、アラーム プロバイダで無効化をマーク付けするフラグを設定することによって無効にします。どのようなアラーム状況が発生しても、このアイテムはアラーム状況にはなりません。アラームを無効化するために使用するドットフィールドの詳細については、「[タグ変数またはアラームグループのアラームの有効化と無効化](#)」を参照してください。

タグ変数のアラームをすべて一度に無効化または有効化することができます。また、サブステートを持つアラームの場合は、各サブステートを個別に無効にできます。

- **抑止。** アラームは、以下の方法で抑止します。
 - a. **WindowMaker** のアラーム設定に「抑止」タグ変数を追加します。抑止タグ変数は、アラームを「抑止」とマーク付けするためにランタイムで使用されます。
 - b. ランタイムで抑止タグ変数を **True** または **False** に設定します。抑止タグ変数が **False** の場合、アラームは通常通りに処理されます。抑止タグ変数が **True** の場合、アイテムはアラームにはなりません。

各サブステートは異なるタグ変数によって抑止できます。また、一部のサブステートに抑止タグ変数を割り当てないまま残すこともできます。

アラームのタグ変数を抑止タグ変数として割り当てると、そのクロスリファレンス使用数は増加します。

- **非表示。** 非表示では、アラーム コンシューマでは特定のアラームが無視されます。アラームが例外基準に一致する場合、非表示となります。つまり、特定のアラーム コンシューマでそのアラームが表示、印刷、またはログ記録されなくなります。

実際のアラーム生成は、非表示によってまったく影響されません。アラーム レコードは、アラーム履歴に引き続きログ記録されます。

アイテムがアラーム状況の間、アラームが無効となるかアクティブに抑止される場合、そのアイテムは強制的に別の（有効な）状態となります。どの状態となるべきかは、どの状態が使用可能であるかどうか、またそれらの状態が無効になっていないかどうかにより異なります。この処理は、アラームのタイプ、しきい値の値などに基づいて、アラーム プロバイダによって処理されます。

無効であるかアクティブに抑止されているアラームは確認を待機していません。アラームがサブステートを持つ場合、引き続き使用可能であるサブステートでのみ確認を待機している状態となります。

ターミナル サービス、アラームのサポート

分散アラーム システムを **Terminal Services for InTouch** と併用することで、異なるターミナルセッションで実行しているアラーム クライアント上でどのアラーム データをどのように表示するかを選択できます。

アラーム プロバイダは、アプリケーションに固有な名前とアプリケーションのインスタンスで識別できます。この情報は、アラーム プロバイダまたはアラーム コンシューマを分散アラーム システムに登録するときに分散アラーム システムに対して利用可能になります。

アラーム プロバイダを実行しているノードは、システム内のコンピュータ ノードを識別する固有な名前前で識別されます。ターミナル サービス エディション (TSE) クライアントセッションによって生成されるアラーム レコードには、「<ノード>:<IP アドレス>」の形式の拡張ノード名が含まれます (serverAlarm:192.168.1.23 など)。この情報は、そのコンピュータ ノードでそのインスタンスが起動したときに分散アラーム システムで利用可能になります。

アラーム イベントがログ記録されると、ノードと完全なアラーム プロバイダの名前によってアラームのソースが識別されます。

ターミナル サービス環境でアラームが確認されると、記録されるオペレータ ノードは、対応するオペレータがターミナル サービス セッションを確立しているクライアント マシンの名前になります。ノード名を取得できない場合、ノードの IP アドレスが代わりに使用されます。

ターミナル サーバー クライアントセッションが **InTouch** アラーム プロバイダとなることはできません。

分散アラーム システム データの保存

分散アラーム システムでは、いくつかの形式でデータが保存されます。

- **内部アラーム メモリ (バッファ)**

現在のアラームと最近のアラームに関するほとんどの情報は、さまざまなコンピュータ ノードのメモリに保持されます。**InTouch** では 2 種類のメモリ ロケーションを使用しています。1 つは (現在の) サマリ アラーム用で、もう 1 つは履歴アラームとイベント用です。このモデルは分散アラーム システムでも使用されています。

サマリ アラーム用のメモリは、現在のアラームをすべて格納できるように、必要に応じて使用可能なメモリサイズの限界まで拡大されます。履歴アラーム用のメモリは、事前定義されたしきい値までしか拡大することはできません。履歴メモリがこのしきい値に達すると、最も古いアラーム レコードが破棄され、新しいレコードが追加されます。マルチノード環境では、複数のノードのアラーム メモリが単一の集合アラーム メモリを構成します。

制限の設定の詳細については、「[アラーム バッファ サイズの設定](#)」を参照してください。

- **アラーム データベース**

Alarm DB Logger は、データベースを作成し、アラームが発生したり、アラームがサブステート間の移行を行ったり、アラームが確認されたり、アラームが平常に戻る時間を追跡します。実質的に、これらのレコードはシステムのアラーム履歴を形成します。

分散アラーム システムはクエリーの使用に基づいているため、1 つのコンピュータ ノードを使用してその他の複数のノードに対してアラームをログ記録することがサポートされています。

Build

章 5 InTouch HMI アプリケーションの管理

InTouch HMI アプリケーションの管理では、以下の作業を行います。

- InTouch アプリケーションを作成または削除する。「[InTouch アプリケーションの作成](#)」および「[アプリケーション マネージャからの InTouch アプリケーションの削除](#)」を参照してください。
- WindowMaker または WindowViewer のどちらかでアプリケーションを開く。「[WindowMaker と WindowViewer でアプリケーションを開く](#)」を参照してください。
- アプリケーションを検索する。「[InTouch アプリケーションの検索](#)」を参照してください。
- アプリケーションを別のコンピュータに移動する。「[リモート ノードへのアプリケーションのパブリッシュ](#)」を参照してください。
- 複数のコンピュータにアプリケーションを分散する。『AVEVA™ InTouch HMI アプリケーション配置ガイド』の「アプリケーションの分散」を参照してください。
- InTouch サービスを管理する。「[InTouch サービスの管理の概要](#)」を参照してください。
- タグの定義、ウィンドウ、およびスクリプトをインポートまたはエクスポートする。「[InTouch コンポーネントのエクスポートとインポート](#)」を参照してください。
- セキュリティを設定する。『AVEVA™ InTouch HMI 管理ガイド』の「[InTouch のセキュリティ保護](#)」を参照してください。

アプリケーションは以下の方法で拡張できます。

- テキスト文字列およびアラーム コメントを別の言語に翻訳する。『AVEVA™ InTouch HMI アプリケーション ランタイム ガイド』の「[ランタイムでの言語の切り替え](#)」を参照してください。
- Application Server とアプリケーションの統合。「マネージド InTouch アプリケーション」および「[ランタイム時のアプリケーションの表示の概要](#)」を参照してください。
- 複数のモニタにアプリケーションを表示する。「マルチ モニタ システムの設定の概要」を参照してください。
- タブレット PC でアプリケーションを使用する。「タブレット PC での InTouch の使用の概要」を参照してください。

Windows Vista、Windows 7、および Windows Server 2008 オペレーティング システムの場合、標準のユーザーは InTouch アプリケーション マネージャを使用してアプリケーションを検索し、WindowViewer を開くことができます。標準のユーザーの場合、アプリケーションのプロパティは読み取り専用になります。アプリケーション マネージャからすべての読み取り/書き込みまたは設定操作を行うには、管理特権が必要です。

アプリケーション マネージャの起動

アプリケーション マネージャは、[スタート] メニューから、またはコンピュータのデスクトップにあるショートカットから起動できます。

アプリケーション マネージャを初めて起動するには

1. タスクバーで、[スタート] をクリックし、[プログラム] をポイントして、[AVEVA InTouch HMI] をポイントし、次に [InTouch HMI アプリケーション マネージャ] をクリックします。

AVEVA アプリケーション マネージャが開きます。

コンフィグレータで「**Operations Control 接続エクスペリエンス**」モードが有効化されている場合、アプリケーションマネージャの初回起動時に AVEVA Identity Manager での認証を受けるよう求めるメッセージが表示されます。

2. AVEVA Identity Manager を使用した認証の詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

ウィンドウには、アプリケーションマネージャを使用して作成または検索した、コンピュータ上にある InTouch アプリケーションが表示されます。

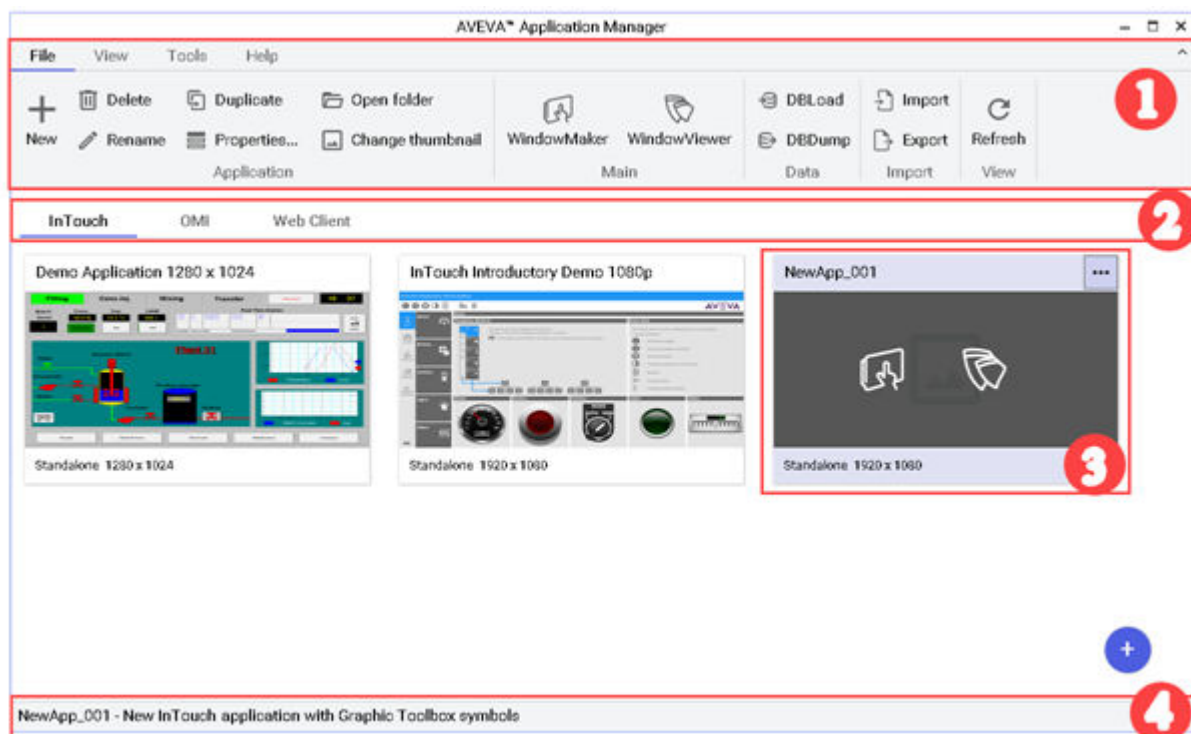
アプリケーションを最後に保存した InTouch のバージョンを確認するには

1. アプリケーションマネージャを開きます。
2. [表示] メニューの [表示] グループで [詳細] をクリックします。
[詳細] グリッドビューにアプリケーションリストが表示されます。
3. アプリケーションの [バージョン] カラムを確認します。

アプリケーションマネージャの使用

InTouch HMI アプリケーションの作成に使用するアプリケーションマネージャは、アプリケーション管理に関する多くのオプションを提供します。

インターフェイスは、リボンと作業エリア/キャンバスに分割されます。



識別子	要素	説明
1	リボン バー	メニュー項目および各メニュー項目のコマンドが画面上部で機能に基づいてグループ化されています。

識別子	要素	説明
2	アプリケーションバー	InTouch、OMI、および Web Client のタブを表示します。 リボン内の各メニュー項目のコマンドは、選択したアプリケーションタイプに応じて異なります。
3	アプリケーションタイトル	各アプリケーションを表します
4	ステータスバー	選択した項目の説明メッセージを表示します。

リボンには、以下のメニュー項目が含まれます。

- **ファイル:** [ファイル] メニューはデフォルトのリボンメニューです。このメニューには、5 つの関連コマンド ([アプリケーション]、[メイン]、[データ]、[インポート]、および [表示]) があります。アプリケーション設定は、作成、複製、削除、WindowMaker または WindowViewer で開く、インポート、エクスポート、およびその他の多くの一般的な機能などのコマンドを使用して変更できます。
- **表示:** [表示] メニューには、キャンバスのリストの表示を変更するオプション ([一覧表示]、[すべて表示]、[詳細] など) があります。
- **ツール:** [ツール] メニューには、重要なアプリケーションマネージャ ツール ([ノードのプロパティ]、[OPC UA 設定]、[検索]、[Insight Publisher]) が含まれています。
- **ヘルプ:** [ヘルプ] メニューからは、InTouch HMI ヘルプ ドキュメント、アプリケーションマネージャのバージョン情報、および著作権情報にアクセスできます。

作業エリア/キャンバスはタブで分割され、各タブはアプリケーションタイプ (InTouch、OMI、および Web Client) を表します。


InTouch アプリケーションの作成

アプリケーションマネージャを使用して、新規 InTouch アプリケーションを作成できます。アプリケーションパスは、ネットワーク ドライブ文字、コロン、およびすべての円記号を含め、114 文字を超えないようにしてください。制限を超えると、WindowMaker でアプリケーションを開くことができません。アプリケーション名は 32 文字以内である必要があります。

アプリケーションを作成すると、INTOUCH.ini ファイルが作成されます。INTOUCH.ini ファイルには、アプリケーションのデフォルト設定内容が含まれています。アプリケーションの設定をユーザーが変更すると、新しい設定が INTOUCH.ini ファイルに書き込まれます。

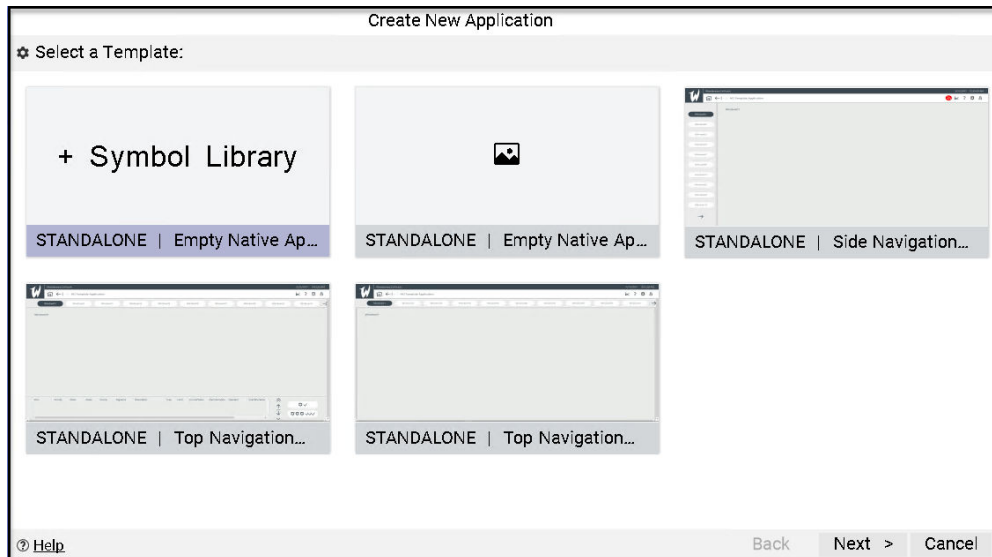
アプリケーションを作成した後、既存の INTOUCH.ini ファイルをアプリケーションフォルダにコピーできます。このようにすると、新規アプリケーションを作成するたびに、カスタマイズされた InTouch パラメータを設定する必要がありません。

新規アプリケーションを作成するには

1. アプリケーションマネージャで別のオプションを使用して、新規アプリケーションを作成できます。
 - [ファイル] タブの [アプリケーション] グループで [新規] を選択します。
 - 右下の  をクリックします。

- Ctrl + N を押します。
- 空白のスペースを右クリックして、[新規...] をクリックします。

[新規アプリケーションの作成 - テンプレートを選択] ページが表示されます。



2. 新しいアプリケーションのテンプレートを選択して [次へ] をクリックします。

空白のテンプレートには、デフォルトのグラフィック ツールボックス シンボルと SAL シンボルは含まれません。すべてのテンプレートで、InTouch シンボルおよび産業用グラフィックの両方をアプリケーションで使用できます。作成したアプリケーションテンプレートが正しいフォルダで使用可能である場合、そのアプリケーションも表示されます。

[新規アプリケーションの作成 - アプリケーションの詳細を入力] ページが表示されます。

3. 以下の詳細を入力します。
 - **タイプ:** 選択したアプリケーションのタイプが表示されます。
 - **アプリケーション名:** アプリケーションの名前を入力します。
 - **ディレクトリ名:** アプリケーションフォルダ名を入力します。
 - **アプリケーションパス:** デフォルトの場所以外のフォルダを参照するには、省略記号ボタン ([...]) をクリックします。
 - **デフォルトディレクトリとして設定:** トグルを使用して、アプリケーションパスをデフォルトディレクトリとして設定します。
 - **解像度:** ドロップダウンリストからアプリケーションのターゲット解像度を選択します。

使用可能な解像度オプションを以下に示します。

- 画面解像度 (デフォルト)
- 1024 x 768
- 1280 x 1024
- 1366 x 768
- 1440 x 900

幅と高さを編集するには、[解像度] ドロップダウン リストから [**カスタム解像度**] を選択します。

- **幅:** アプリケーションの幅を指定します。
- **高さ:** アプリケーションの高さを指定します。
- **説明:** 255 文字までのオプションの説明を入力します。

4. [完了] をクリックします。

作成されたアプリケーションは、アプリケーション マネージャのアプリケーションのリストに表示されます。



新規 InTouchView アプリケーションの作成

アプリケーションを作成するとき、アプリケーション マネージャのオプションを設定することにより、InTouchView アプリケーションを識別します。InTouch アプリケーションを InTouchView アプリケーションに設定することや、その反対の設定を行うことができます。InTouchView アプリケーションから変換された InTouch アプリケーションを実行するには、InTouch のフルライセンスが必要です。アプリケーションを InTouchView アプリケーションに変更するした場合、必要な取得ライセンスは、選択したデータソースに応じて異なります。詳細については、「[InTouchView アプリケーション](#)」を参照してください。

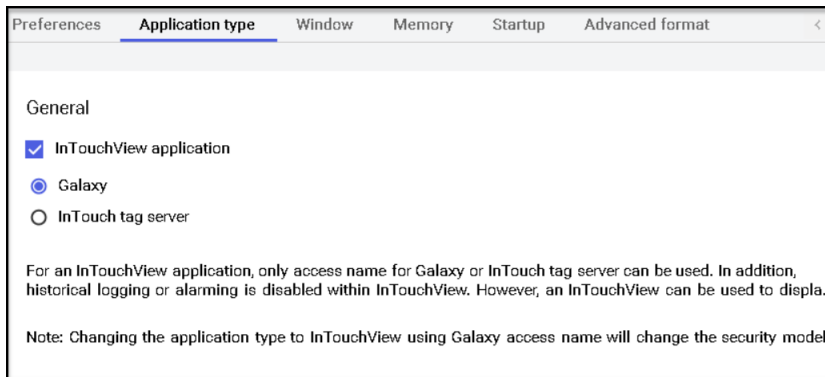
InTouch アプリケーションから InTouchView アプリケーションへの変更

アプリケーションに必要な機能が Application Server または InTouch タグ サーバーのデータへの接続だけである場合、InTouch アプリケーションを InTouchView アプリケーションに変更できます。

WindowMaker の一部の機能は InTouchView アプリケーションで使用できません。

InTouch アプリケーションを InTouchView アプリケーションに変換するには

1. WindowMaker で InTouch アプリケーションを開きます。
2. [ファイル] メニューの [**設定**] をクリックし、[**WindowViewer**] をクリックします。
WindowViewer の設定画面が表示されます。
3. [アプリケーション タイプ] タブで [**InTouchView アプリケーション**] チェック ボックスをオンにします。



- Galaxy のデータに接続するには、[Galaxy] を選択します。
- InTouch タグ サーバーのデータに接続するには、[InTouch タグ サーバー] を選択します。

4. [ホーム] メニューの [タグ] グループで [アクセス名] をクリックします。

- a. [Galaxy] を選択した場合、InTouch アプリケーションを InTouchView に変換する前に、Galaxy 以外のすべてのアクセス名を削除する必要があります。削除しない場合、変換の試行中にメッセージが表示されます。
- b. [InTouch タグ サーバー] を選択した場合、新しいアクセス名を設定できます。[追加] をクリックしてアクセス名とノード名を入力します。

アプリケーション名とトピック名はグレーアウトされます。このアプリケーションを InTouchView に変更する前に、Galaxy 以外のアクセス名や InTouch タグ サーバーを参照するアクセス名をすべて削除する必要があります。アプリケーションが表示として設定されているアクセス名からの I/O 参照だけがバインドされます。アクセス名 **Galaxy** からの I/O 参照はバインドされません。

- c. [OK] をクリックします。

アクセス名の設定の詳細については、「[アクセス名の設定](#)」を参照してください。

5. [閉じる] をクリックします。

6. [OK] をクリックします。メッセージが表示されたら、[OK] をクリックします。

WindowMaker と WindowViewer でアプリケーションを開く

新規アプリケーションを WindowViewer で開く前に、WindowMaker で開く必要があります。

WindowMaker でアプリケーションを開くには

1. アプリケーション マネージャ ウィンドウでアプリケーションを選択します。
2. [ファイル] タブの [メイン] グループで [WindowMaker] を選択します。

アプリケーション タイルの上にカーソルを置いて、[WindowMaker] をクリックすることもできます。



アプリケーションが WindowMaker で開きます。

ヒント: アプリケーションをダブルクリックして、WindowMaker で開くこともできます。

WindowViewer でアプリケーションを開くには

1. アプリケーション マネージャ ウィンドウでアプリケーションを選択します。
2. [ファイル] メニューの [メイン] グループで [WindowViewer] を選択します。

アプリケーション タイルの上にカーソルを置いて、[WindowViewer] をクリックすることもできます。



アプリケーションが WindowMaker で開きます。

アプリケーション マネージャ ウィンドウのカスタマイズ

ツールバーおよびステータス バーを非表示にできます。アプリケーション マネージャ ウィンドウにアプリケーションが一覧表示される方法を変更することもできます。アプリケーションは、詳細、リスト、およびアイコンとして表示できます。

- **詳細** – このビューでは、アプリケーションが表形式で表示され、アプリケーションに関連するすべてのプロパティ（パスや解像度など）が表示されます。
- **リスト** – このビューでは、アプリケーションの名前、タイプ、サムネイル、およびリボンが表示されます。
- **アイコン** – このビューでは、アプリケーションの名前、タイプ、解像度、説明、サムネイル、およびリボンが表示されます。

ツールバーの表示と非表示を切り替えるには

- **[表示]** メニューで **[ツールバー]** をクリックして、選択したアプリケーションのステータスの表示と非表示を切り替えます。


ステータス バーの表示と非表示を切り替えるには

- **[表示]** メニューで **[ステータス バー]** をクリックして、選択したアプリケーションのステータスの表示と非表示を切り替えます。

アプリケーション リストの表示を変更するには

- **[表示]** メニューで適切なコマンドをクリックするか、ツールバーの対応するボタンをクリックします。

ツールバー オプションの配置を変更するには

1. この  アイコンをクリックして **[カスタマイズ...]** を選択します。
2. **[アイテム]** タブでオプション（**[WindowMaker]** など）を選択し、**[上へ移動]** または **[下へ移動]** ボタンをクリックして、ツールバーに表示されるオプションの順序を変更します。

ツールバーに表示されるツールを削除するには

- **[表示]** メニューで、目的のオプションのチェック ボックスをクリックします。

たとえば、**[WindowMaker]** オプションのチェック ボックスをオフにして、**[閉じる]** をクリックします。**[Window Maker]** オプションがツールバーに表示されなくなります。

アプリケーション マネージャにテーマを適用するには

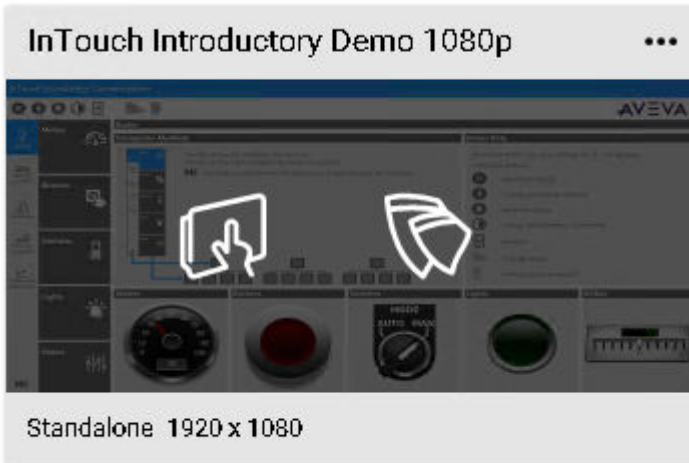
- **[表示]** メニューの **[テーマ]** グループで **[ライトテーマ]** / **[ダークテーマ]** を選択して2つのテーマを切り替えます。

アプリケーション タイルの使用

アイコンまたは一覧ビューでは、各アプリケーションは、該当する特定のオプションを含むタイルとして表示されます。

WindowMaker または WindowViewer の起動

アプリケーション タイルの上にカーソルを置くと、アプリケーションを **WindowMaker** または **WindowViewer** で起動するオプションが表示されます。



追加設定

省略記号アイコンをクリックすると、追加設定が表示されます。

- **WindowMaker:** クリックすると WindowMaker が起動します。

注記: WindowMaker を初めて起動するときに AIM で認証する必要があることがあります。詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

- **WindowViewer:** クリックすると WindowViewer が起動します。

注記: WindowViewer を初めて起動するときに AIM で認証する必要があることがあります。詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

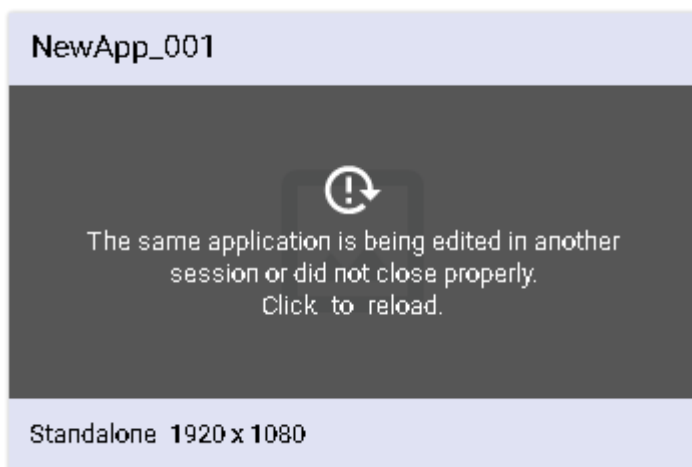
- **アプリケーションフォルダ:** クリックしてアプリケーションフォルダに移動します。
- **プロパティ:** クリックするとアプリケーションのプロパティ ウィンドウを開きます。
- **サムネイルの変更:** アイコンをクリックして、[参照] ダイアログ ボックスから画像を選択します。画像がアイコンおよび一覧ビューに表示されます。
- **名前の変更:** クリックして、アプリケーションの新しい名前を入力します。
- **DBLoad:** クリックして DBLoad ユーティリティを起動します。
- **DBDump:** クリックして DBDump ユーティリティを起動します。
- **テンプレートとしてエクスポートする:** クリックして InTouch アプリケーションをエクスポートします。「[テンプレートとして使用するための InTouch アプリケーションのエクスポート](#)」を参照してください。
- **削除:** クリックしてアプリケーションを削除します。

InTouch アプリケーションのロックおよびロック解除

アプリケーションが別のセッションで編集集中の場合、または正しく終了しなかった場合、アプリケーションはロックされることがあります。その場合、アプリケーションマネージャを使用してアプリケーションのロックを解除する必要があります。

InTouch アプリケーションのロックを解除するには

- アプリケーションマネージャを起動します。
- 現在ロックされているアプリケーションを選択します。
- アプリケーションタイルをクリックしてロックを解除し、アプリケーションを再ロードします。



アプリケーションのロックが解除され、アプリケーションを使用できるようになります。詳細については、「[アプリケーション編集のロック](#)」を参照してください。

InTouch アプリケーションの変更

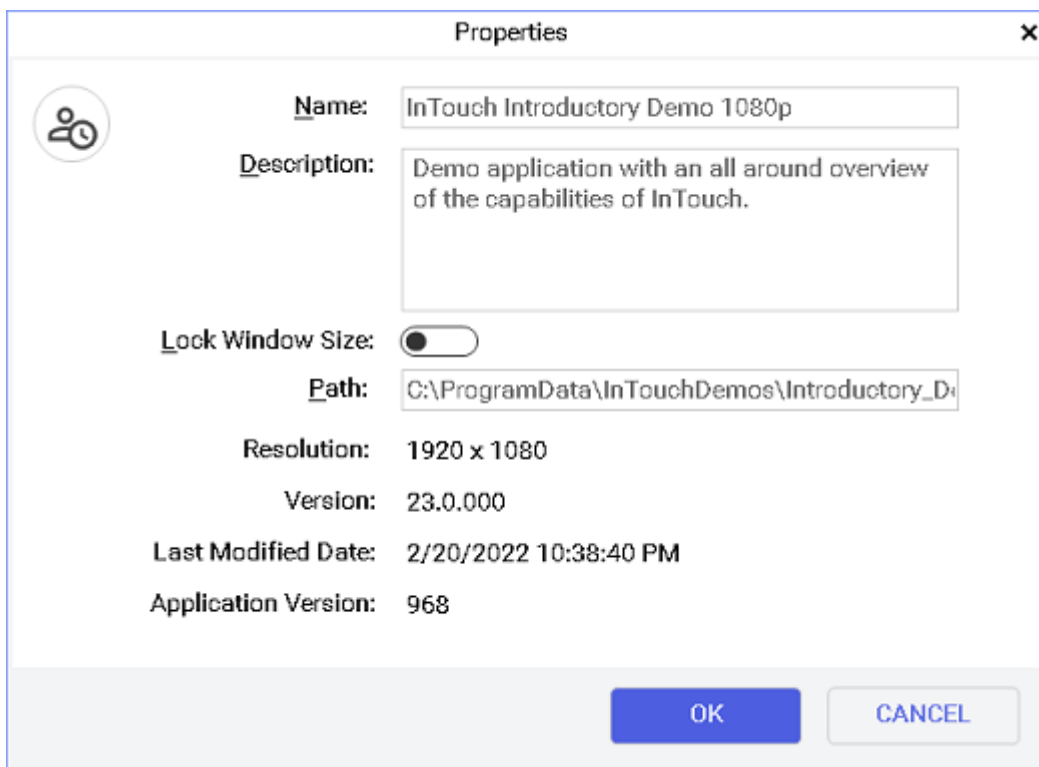
アプリケーションの名前を変更したり、アプリケーションのプロパティを変更することができます。

アプリケーションの名前を変更するには

1. 一覧からアプリケーションを選択します。
2. [ファイル] タブの [アプリケーション] グループで [名前の変更] を選択します。

アプリケーションのプロパティを変更するには

1. 一覧からアプリケーションを選択します。
2. [ファイル] タブの [アプリケーション] グループで [プロパティ] を選択します。
[プロパティ] ダイアログ ボックスが表示されます。



Properties

Name: InTouch Introductory Demo 1080p

Description: Demo application with an all around overview of the capabilities of InTouch.

Lock Window Size: ☒

Path: C:\ProgramData\InTouchDemos\Introductory_Di...

Resolution: 1920 x 1080

Version: 23.0.000

Last Modified Date: 2/20/2022 10:38:40 PM

Application Version: 968

OK **CANCEL**

3. アプリケーションのプロパティを設定します。

- **[名前]** ボックスで、アプリケーションの新規名を入力します。
- **[説明]** ボックスに、アプリケーションの別の説明を入力します。

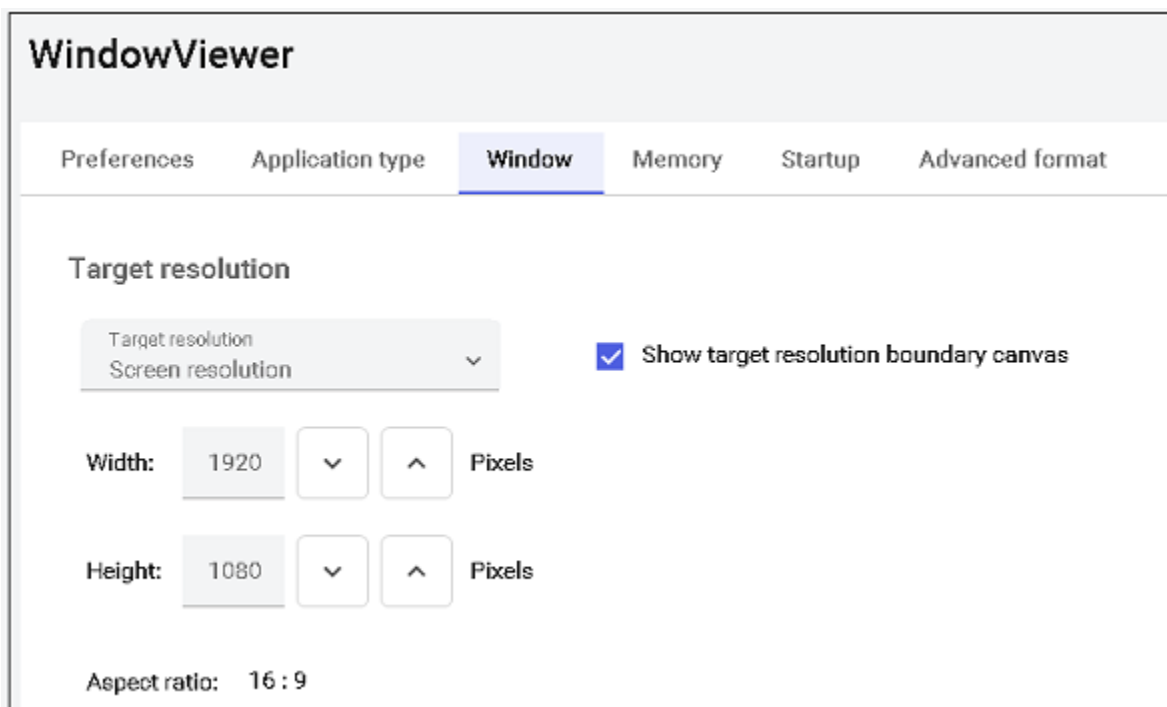
4. **[OK]** をクリックします。

ターゲット解像度を変更するには

WindowMaker でアプリケーションを編集するとき、指定したターゲット解像度を変更できます。この機能はコマンドラインでは使用できません。

以下の手順を実行します。

1. WindowMaker を開きます。
2. **[ファイル]** メニューの **[設定]** をクリックし、**[WindowViewer]** をクリックします。
WindowViewer の設定画面が表示されます。
3. **[ウィンドウ]** タブを選択します。
4. **[ターゲット解像度]** セクションで、必要に応じてターゲット解像度を編集します。



5. [OK] をクリックします。

境界キャンバスが変更され、変更が反映されます。グラフィック、ウィンドウ サイズ、およびウィンドウ コントロールは変更されずに維持されます。

注記: [ターゲット解像度の境界キャンバスを表示] はデフォルトで選択されます。

アプリケーション マネージャからの InTouch アプリケーションの削除

アプリケーション マネージャからアプリケーションを削除しても、アプリケーション ファイルはコンピュータ上に残ります。

アプリケーションを削除するには

1. 一覧からアプリケーションを選択します。
2. [ファイル] タブの [アプリケーション] グループで [削除] をクリックします。
3. メッセージが表示されたら、[はい] をクリックします。
4. アプリケーションは以下の方法でも削除できます。
 - アプリケーションを右クリックして、コンテキストメニューから [削除] をクリックします。
 - アプリケーション リボンから [削除] アイコンを選択します。
 - アプリケーションを選択して、キーボードの **delete** キーを押します。

複数のアプリケーションを選択して削除することができます。

以前のモダン InTouch アプリケーションの削除

古いモダンアプリケーションを含むノードを現在の製品バージョンにアップグレードした場合、古いモダンアプリケーションを削除できます。この操作を行うと、モダンアプリケーションのフォルダとレポジトリが恒久的に削除されます。

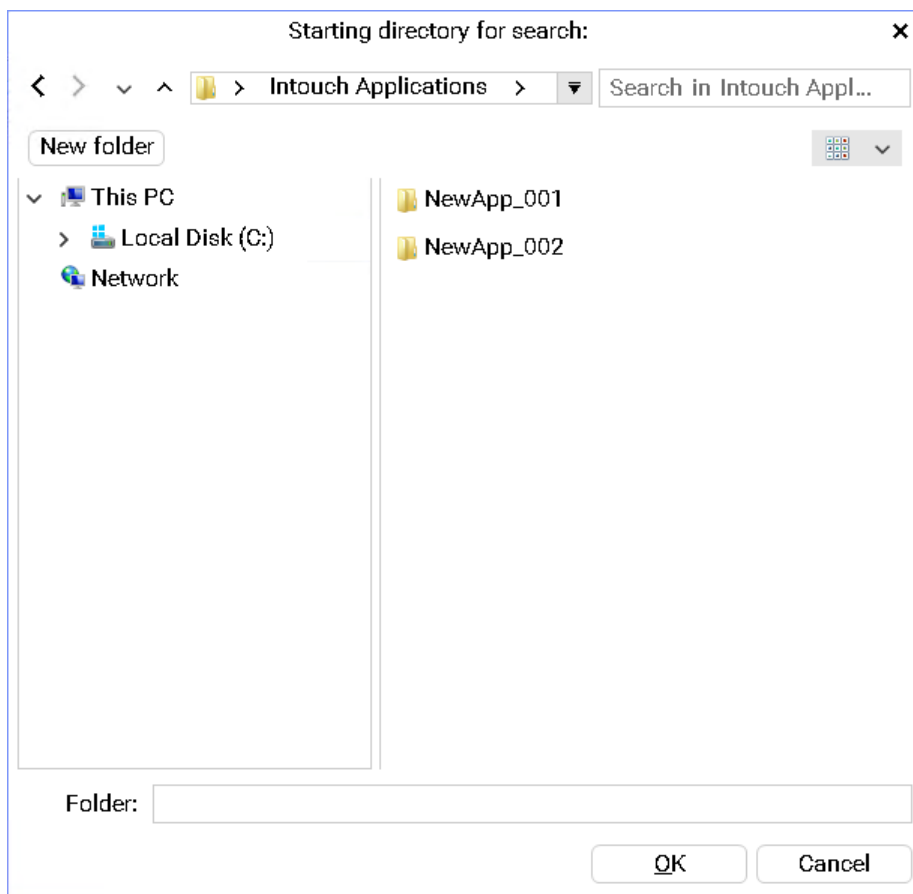
InTouch アプリケーションの検索

既存の InTouch アプリケーションを検索できます。アプリケーション マネージャには、見つかったすべてのアプリケーションが表示されます。

(ネットワーク ドライブ文字、コロン、およびすべての円記号を含めて) フルパスが **114** 文字を超える場合、**WindowMaker** でアプリケーションを開くことができません。文字数の制限をアプリケーションが超えている場合、パスのフォルダの名前を変更するか、またはアプリケーションを別の場所に移動してください。

アプリケーションを検索するには

1. [ツール] タブで [検索] をクリックします。[検索を開始するディレクトリ] ダイアログボックスが表示されます。



2. アプリケーションを検索するフォルダを選択します。
3. [OK] をクリックします。

アプリケーション テンプレートでの作業

スタンドアロンアプリケーションから独自のアプリケーション テンプレートを開発できます。アプリケーション テンプレートの開発は、3 ステップのプロセスで構成されます。

1. アプリケーションを作成して、テンプレート ウィンドウに対する適切なアプリケーション ウィンドウを設定します。
2. アプリケーション テンプレート ブラウザのプレビューに使用するアプリケーション サムネイルを作成します。
3. アプリケーションを .aaPKG ファイルとしてエクスポートして、ブラウザでテンプレートとして使用できるようにします。

アプリケーション ウィンドウをテンプレート ウィンドウとして設定するには

1. アプリケーションを作成します。
 - a. グラフィック、スクリプト、およびウィンドウを使用してアプリケーションを開発します。
2. アプリケーション内の全てのウィンドウに対して、以下の手順を実行します。
 - a. 各ウィンドウを右クリックして、[プロパティ] を選択します。
 - b. [ウィンドウのプロパティ] パネルで、[テンプレート] チェックボックスをオンにします。

Name	Window_001
Comment	
Window type	Overlay
Location	4, 4
X	4
Y	4
Size	632, 278
Width	632
Height	278
Window color	<input type="checkbox"/> 250, 250, 250
Titlebar	<input checked="" type="checkbox"/>
Frame style	Single
Close button	<input checked="" type="checkbox"/>
Size controls	<input checked="" type="checkbox"/>
Template	<input checked="" type="checkbox"/>

Name
Determines the name of the window

[OK] をクリックすると、各ウィンドウは [ウィンドウ] ペインの [テンプレート ウィンドウ] フォルダに自動的に配置されます。

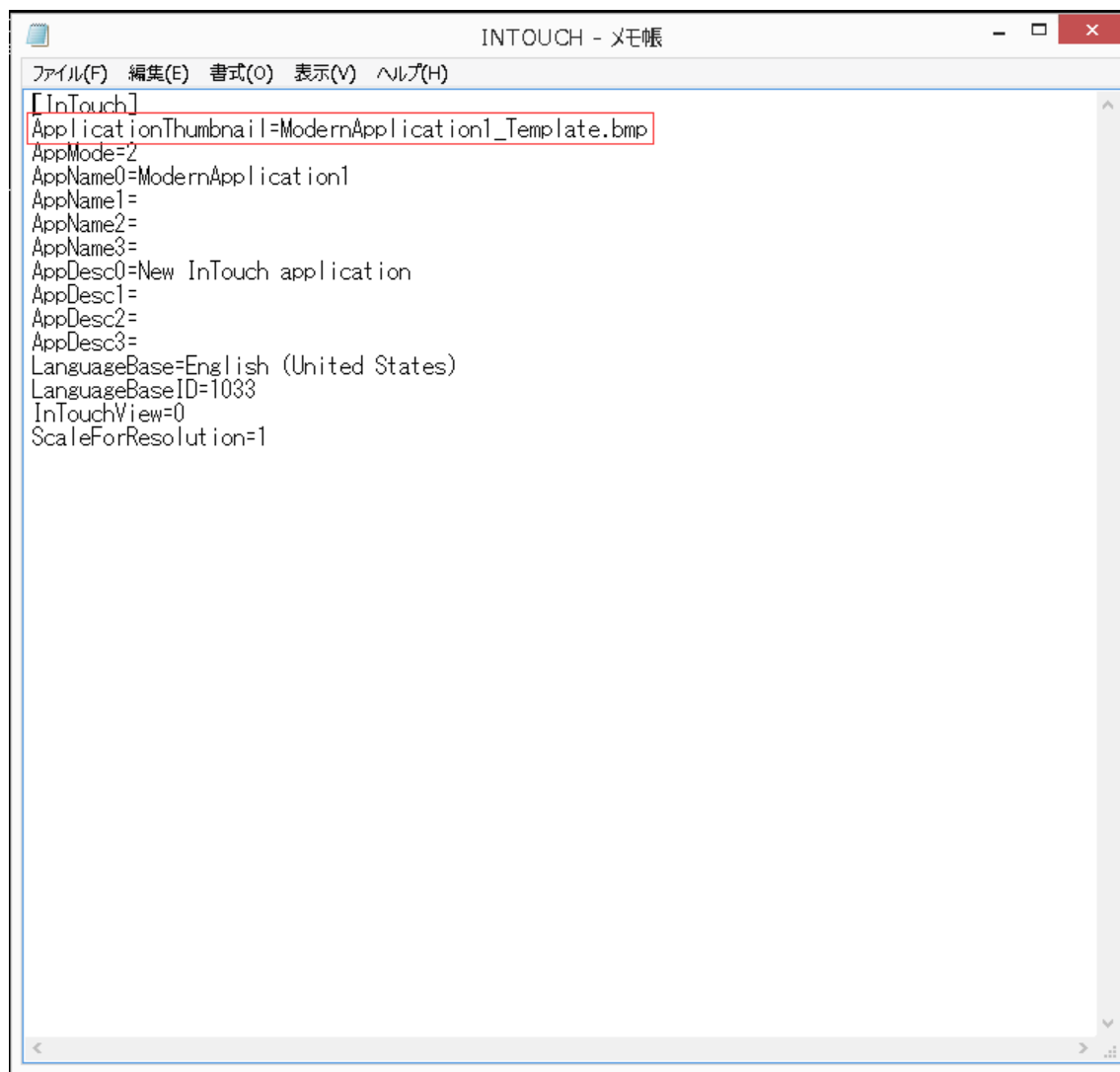
サムネイルを作成して、テンプレートを作成するアプリケーションに割り当てる必要があります。

アプリケーション サムネイルを作成して割り当てるには

1. 任意のスクリーン キャプチャ プログラムを使用して、設定時または実行時のいずれかのアプリケーションのスクリーン キャプチャを取得します。

2. 任意のピクチャ ファイル形式 (.bmp や .png ファイルなど) で画像を保存し、アプリケーション フォルダにコピーします。
3. アプリケーション フォルダから、メモ帳などの任意のテキスト エディタを使用して INTOUCH.INI ファイルを開きます。
4. INTOUCH.INI ファイルを編集して、[ApplicationThumbnail] フィールドの画像のファイル名を編集します。

注記: [ApplicationThumbnail] フィールドでは大文字と小文字が区別され、サムネイル画像の名前と拡張子に正確に一致する必要があります。



5. 変更を保存してファイルを閉じます。

[アプリケーション テンプレート ブラウザ] に表示される .aaPKG ファイルとしてアプリケーションをエクスポートする必要があります。

アプリケーションをエクスポートしてテンプレートを作成するには

1. アプリケーション マネージャでアプリケーションをエクスポートして .aaPKG ファイルを作成します。

アプリケーションのエクスポートの詳細については、「[テンプレートとして使用するための InTouch アプリケーションのエクスポート](#)」を参照してください。

2. エクスポートした .aaPKG ファイルを次のディレクトリにコピーします。

C:\Program Files (x86)\Wonderware\InTouch\ApplicationTemplates

アプリケーションがアプリケーションテンプレートブラウザ内でテンプレートとして使用できるようになります。

注記: 上記の手順で作成したアプリケーションテンプレートサムネイルは、エクスポートされた aaPKG ファイルから抽出されます。有効な画像が抽出できない場合、[アプリケーションテンプレートブラウザ]に表示されるアプリケーションテンプレートのサムネイルが空白になります。有効な画像ファイル形式を使用してサムネイルを保存したこと、および正しいファイル名が ITOUCH.INI ファイルに入力されていることを確認してください。

InTouch アプリケーション フォルダの表示

WindowMaker を使用すると、InTouch アプリケーション フォルダに配置されていてランタイム時に使用可能なファイルを表示できます。

アプリケーションフォルダを開くには

1. WindowMaker で InTouch アプリケーションを開きます。
2. [表示] メニューの [アプリケーション] グループで [フォルダを開く] をクリックします。

標準の Windows エクスプローラ ウィンドウが開き、InTouch アプリケーション フォルダが表示されます。マネージドアプリケーションの場合、Galaxy リポジトリ ノードのアプリケーションフォルダが表示されます。

テンプレートとして使用するための InTouch アプリケーションのエクスポート

[テンプレートとしてエクスポート] オプションでは、グラフィック関連のすべてのコンテンツ（シンボル、クライアント コントロール、スクリプトライブラリ、言語、スタイル）を含む .aapkg ファイルをエクスポートして、テンプレートとして保存できます。このテンプレートファイルを使用して、他のスタンドアロンまたはマネージドアプリケーションを作成できます。

アプリケーションをテンプレートとしてエクスポートするには

1. アプリケーションマネージャの [ファイル] タブで、[メイン] グループの [テンプレートとしてエクスポート] を選択します。

[InTouch アプリケーションのエクスポート] ダイアログ ボックスが表示されます。

2. .Aapkg ファイルを保存する場所を指定します。

[InTouch アプリケーションのエクスポート] 進行状況ウィンドウが表示されます。

3. エクスポートが完了したら [閉じる] をクリックします。

4. [詳細の表示] をクリックしてエラーをチェックします。

指定した場所に保存された .aapkg ファイルは新しいアプリケーションを作成するアプリケーションテンプレートとして使用できます。

アプリケーション マネージャを使用した Web Client 設定の更新

AVEVA アプリケーション マネージャの [Web Client] タブには、Web Client 関連の設定を構成するオプションがあります。設定は InTouch HMI と OMI アプリケーションの両方に適用できます。

Web Client 設定の更新:

1. アプリケーション マネージャを起動し、[Web Client] タブをクリックします。
[Web Client の設定] 画面が表示されます。
2. 以下の設定を構成します。
 - **現在のアプリケーション: Web Client** の設定を変更するアプリケーションをリストから選択します。
 - **グラフィックのリフレッシュ レート (ms)** : Web ブラウザが Web Server にグラフィック データをクエリーする頻度を設定します。デフォルトは 1 秒です。詳細については、『System Platform インストール ガイド』を参照してください。
 - **アラームのリフレッシュ レート (ms)** : Web ブラウザが Web Server にアラーム データをクエリーする頻度を設定します。デフォルトは 1 秒です。詳細については、『System Platform インストール ガイド』を参照してください。
 - **ヘッダーを表示**: チェック ボックスをオンにするとタイトルバーが表示されます。
 - **ナビゲーションバーを有効化**: チェック ボックスをオンにするとナビゲーションバーが表示されます。

[ヘッダーを表示] 設定が選択されていない場合、この設定は無効化されます。

 - **匿名アクセスを許可**: ユーザーが認証なしで Web Client にアクセスすることを許可する場合はチェック ボックスをオンにします。
 - **作業領域を有効化**: 作業領域を有効にするには、このチェック ボックスをオンにします。
3. [詳細設定] セクションで [カスタマイズ] をクリックして以下の設定を構成します。
 - **Web サイト名**: 標準の URL を置き換える文字列を入力します。
 - **アプリケーションタイトル**: アプリ バーのアプリケーションのタイトルとして表示する文字列を入力します。
 - **Web サイトタイトル**: ブラウザのタイトルバーのタイトルとして表示する文字列を入力します。
 - **Web サイトアイコン**: ブラウザのタイトルバーのアイコンを置換する画像ファイルを入力します。
 - i. [ファイルを選択...] をクリックしてアイコン画像を選択します。
 - ii. ファイルを選択します。
 - iii. [開く] をクリックします。
4. [保存] をクリックして詳細設定を保存します。
5. 設定を変更したら [適用] をクリックします。
6. Web Client を表示するには、[起動] をクリックします。

注記: Web サイトの名前またはアドレスが変更された場合、AVEVA Identity Manager を設定して新しい Web サイトを登録してください。詳細については、「[AVEVA Identity Manager での登録](#)」を参照してください。

これらの設定の詳細については、「[Web ブラウザでのアプリケーショングラフィックの表示](#)」を参照してください。

InTouch OMI ViewApp の起動

アプリケーションマネージャの [OMI] タブを使用して Viewer for InTouch OMI ViewApps を起動します。

1. InTouch OMI アプリを開発および配置します。
配置した ViewApp は、アプリケーションマネージャの [OMI] タブに表示されます。
2. ViewApp を選択します。
3. OMI の [ファイル] メニューで [OMI Viewer] をクリックします。
Viewer が起動して ViewApp が表示されます。

AVEVA Identity Manager での登録

AVEVA Identity Manager (AIM) を使用すると、デフォルトの Windows OS ベースの認証ではなく、シングルサインオンを使用するように Web Client を設定できます。

注記: Web Client は、AIM 設定を含む ArchestrA ベースのセキュリティのみをサポートします。

Identity Server を設定するには

1. System Platform コンフィグレータで [共通プラットフォーム] > [System Management Server] を設定します。
2. AVEVA アプリケーション マネージャでユーザー資格情報を使用して AIM サーバーを登録します。
[AIM 登録] ダイアログ ボックスを使用して、リバース プロキシサーバーを設定することもできます。
1. リバース プロキシサーバーをセットアップします。
2. System Platform コンフィグレータで [Identity Server] > [System Management Server] を設定します。
3. [AIM 登録] ダイアログ ボックスにセキュア ゲートウェイのアドレスを入力します。

リモートまたはローカルの System Management Server を選択できます。System Platform Server の設定の詳細については、『*System Platform インストールガイド*』を参照してください。

Identity Server に登録するには

1. アプリケーション マネージャで [Web Client] タブをクリックします。
2. [ツール]、[ID マネージャ] の順にクリックします。
[Identity Server 設定] 画面が表示されます。
3. [AIM サーバーを認証サーバーとして使用する] チェック ボックスをクリックして、AVEVA Identity Manager の使用を有効にします。
[Identity Server] フィールドにコンフィグレータで設定した Identity Server が表示されます。
4. 以下の設定を更新します。

- a. **ユーザー名:** Identity Server に接続するユーザー名を入力します。ユーザーは Administrators ユーザー グループに属する必要があります。
- b. **パスワード:** 入力したユーザー名のパスワードを入力します。
5. リバース プロキシのセットアップを完了するには、リバース プロキシまたは DMZ サーバーの URL を **[セキュア ゲートウェイ]** フィールドに入力します。
これはオプションの設定で、Web Client がリバース プロキシサーバーの背後でホストされる場合にのみ設定する必要があります。
6. **[Web サイトへの産業用グラフィックの埋め込みを許可する]** チェック ボックスをクリックして、ランタイム時に HTML iframe 内に Web Client を表示することもできます (オプション)。
7. **[OK]** をクリックします。
8. **[適用]** をクリックします。

注記: Web サイトの名前またはアドレスが変更された場合、AVEVA Identity Manager を設定して新しい Web サイトを登録する必要があります。

資格情報マネージャの操作

資格情報マネージャでは、資格情報をセキュアな場所に保存できます。保存された資格情報は、WindowMaker、WindowViewer、およびアラーム ユーティリティのその他のコンポーネントへのアクセスの認証に使用できます。

資格情報マネージャを起動するには

1. 管理者としてアプリケーション マネージャを起動します。
2. **[ツール]** メニューの **[ツール]** グループで **[資格情報マネージャ]** をクリックします。

資格情報マネージャ画面が表示されます。

注記: 資格情報マネージャにアクセスするには、管理者としてアプリケーション マネージャを開く必要があります。標準ユーザーが資格情報マネージャを開こうとすると、管理者モードで起動するよう求めるメッセージが表示されます。

資格情報マネージャは 2 つのセクションで構成されています。

- 名前付き資格情報
- 資格情報の詳細

[名前付き資格情報] セクションには、資格情報が以下の列を含むグリッド形式で一覧表示されます。

- **名前:** 資格情報の名前を示します。
- **タイプ:** 資格情報のタイプ (ユーザー名とパスワード、またはドメインのユーザー名とパスワード) を示します。
- **グループ:** 資格情報が属する Windows ユーザー グループを示します。[グループ] 列の省略記号アイコン (...) をクリックすると、1 つまたは複数のグループを選択できます。詳細については、「[名前付き資格情報の管理](#)」を参照してください。

[資格情報の詳細] のセクションには、選択した資格情報の以下のフィールドが表示されます。

- ドメイン
- ユーザー名

- パスワード

名前付き資格情報の管理

資格情報マネージャ画面を使用して資格情報を追加または削除できます。グループを各資格情報に割り当てる必要があります。グループのユーザーは、WindowMaker、WindowViewer、アラームユーティリティなどのさまざまな InTouch コンポーネントから資格情報にアクセスできるようになります。

資格情報を追加するには

1. 資格情報マネージャの [名前付き資格情報] セクションで以下の手順を実行します。

- a. [追加 (+)] をクリックします。

[名前付き資格情報] グリッドに新しい行が追加されます。

- b. [名前] 列に資格情報の名前を入力します。

- c. [タイプ] 列で、ドロップダウンリストから資格情報のタイプを選択します。

- ユーザー名とパスワード: SQL Server アカウントを資格情報に割り当てることができます。
- ドメインのユーザー名とパスワード: Windows アカウントを資格情報に割り当てることができます。

- d. [グループ] 列の末尾にある省略記号をクリックします。

[グループを選択] 画面が表示されます。

- e. [リストから選択] ドロップダウンリストからドメインを選択します。

- f. [使用可能な OS グループ] リストをスクロールして目的のグループを見つけて [+] アイコンをクリックします。

選択した OS グループが [選択されたグループ] セクションに表示されます。

デフォルトでは、すべての名前付き資格情報は Administrators グループに追加されます。さらに、アプリケーションを実行しているユーザーが属するグループを追加することが推奨されます。そうすることで、WindowViewer などのアプリケーションを管理者以外のモードで実行しているときでも、ユーザーは資格情報にアクセスできるようになります。

注記: グループを削除するには、[選択されたグループ] セクションでグループを選択して [削除 (-)] をクリックします。

- g. [OK] をクリックします。

2. [詳細] セクションで以下の手順を実行します。

- a. [ドメイン] フィールドにユーザー ドメインを入力します。

注記: [ドメイン] フィールドは、[ドメインのユーザー名とパスワード] タイプの場合にのみ有効になります。資格情報の [タイプ] で [ユーザー名とパスワード] を選択した場合、[ドメイン] フィールドは無効になります。

- b. [ユーザー名] フィールドに選択した資格情報のユーザー名を入力します。

- c. [パスワード] フィールドに選択したユーザー名のパスワードを入力します。

- d. [パスワードの確認] フィールドにパスワードを再度入力して確認します。

3. [保存] をクリックします。

資格情報を削除するには

1. グリッドから資格情報を選択します。
2. [削除 (-)] をクリックします。
選択した資格情報が削除されます。

名前付き資格情報のエクスポートとインポート

選択したアプリケーションの資格情報をエクスポートして、アプリケーションをインポートするノードにインポートできます。

資格情報のエクスポート

次のいずれかの方法で資格情報をエクスポートできます。

- 資格情報のみをエクスポートする
- アプリケーションをエクスポートするときに資格情報をエクスポートする

資格情報をエクスポートするとき、セキュリティ キーを設定する必要があります。セキュリティ キーは資格情報を暗号化し、資格情報をインポートするときの認証の際に必要になります。

資格情報マネージャを使用して資格情報をエクスポートするには

1. 資格情報マネージャ画面で [エクスポート] をクリックします。
2. [資格情報をエクスポート] ウィンドウが表示されます。
3. エクスポートする資格情報をグリッドから選択します。
4. 次のセクションでセキュリティ キーを設定します。

注記: セキュリティ キーは 12 文字以上の長さで、少なくとも 1 つの大文字、小文字、数字、および特殊文字を含む必要があります。

5. [エクスポート] をクリックします。
[資格情報の詳細をエクスポート] ファイル ブラウザが表示されます。
6. 資格情報を保存するフォルダを選択して [保存] をクリックします。
資格情報は .bin ファイル形式でエクスポートされます。

アプリケーションをエクスポートするときに資格情報をエクスポートするには

1. アプリケーションマネージャで、エクスポートするアプリケーションを右クリックして [テンプレートとしてエクスポートする] をクリックします。
2. [資格情報をエクスポート] チェック ボックスをオンにします。
アプリケーションが資格情報リストと共にエクスポートされます。

資格情報のインポート

アプリケーションをインポートした特定のノードで使用する資格情報をインポートできます。

資格情報をインポートするには

1. 資格情報マネージャ画面で [インポート] をクリックします。
2. ファイル ブラウザを使用して .bin ファイルを選択し、[開く] をクリックします。

3. セキュリティ キーを入力します。

ヒント: アプリケーションをエクスポートしたときに使用したものと同じセキュリティ キーを使用する必要があります。

資格情報が資格情報マネージャにインポートされます。

NAD の資格情報をインポートするには

このリリースの時点では、NAD マスターの名前付き資格情報は NAD クライアントに自動的に反映されません。したがって、NAD で実行するアプリケーションを（[アプリケーションの検索] アクションを使用して）選択しても名前付き資格情報はインポートされません。

NAD マスターの名前付き資格情報をインポートするには、各 NAD クライアントのアプリケーションマネージャから名前付き資格情報をインポートします。

資格情報のインポート シナリオ

シナリオ 1: インポートした資格情報とマシンに存在する資格情報が異なる。

資格情報が資格情報マネージャにインポートされ、資格情報グリッドに新しいエントリが追加されます。

シナリオ 2: インポートした資格情報と既存の資格情報が同じ。

資格情報が資格情報マネージャにインポートされ、資格情報グリッドに新しいエントリが追加されます。新しくインポートされた資格情報の名前が変更されます。

シナリオ 3: インポートした資格情報の名前と既存の資格情報の名前が同じで、詳細が異なる。

資格情報が資格情報マネージャにインポートされ、資格情報グリッドに新しいエントリが追加されます。既存の資格情報の名前に増分値が付けられた名前で新しくインポートした資格情報の名前が変更されます。

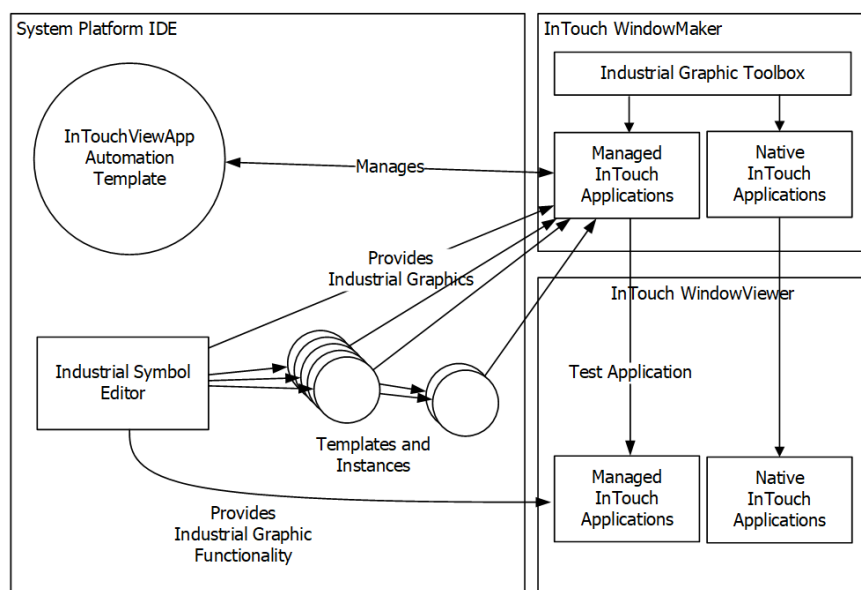
章 6 IDE を使用した InTouch アプリケーションの管理

System Platform IDE を使用して、以下を実行できます。

- 新しいマネージド InTouch アプリケーションを作成する。
- マネージド InTouch アプリケーションとして使用するために、既存の InTouch アプリケーションをインポートする。
- WindowMaker を起動する。
- WindowMaker で行った変更をマネージド InTouch アプリケーションに送信する。
- マネージド InTouch アプリケーションをその InTouchViewApp オブジェクトと共にエクスポートおよびインポートする。
- マネージド InTouch アプリケーションをパブリッシュする。
- マネージド InTouch アプリケーションを削除する。
- マネージド InTouch アプリケーションで使用されているタグデータをインポートおよびエクスポートする。
- .csv ファイルへのタグデータのエクスポートおよび .csv ファイルからのタグデータのインポートを行う。
- マネージド InTouch アプリケーションの言語を切り替える。
- マネージド InTouch アプリケーションにファイルを追加する。
- すべての Galaxy グラフィックを InTouchViewApp に関連付ける。
- アプリケーションテンプレートを使用してマネージド InTouch アプリケーションを作成する。

System Platform IDE は InTouch HMI アプリケーション マネージャから起動できます。

以下の図は、アプリケーションがどのようにインポート、エクスポート、管理、およびパブリッシュされるのかを示しています。



マネージド InTouch アプリケーションの作成

InTouchViewApp オブジェクトを作成および設定することにより、マネージド InTouch アプリケーションを作成します。

また、アプリケーションバージョン、解像度、および説明の情報を InTouchViewApp オブジェクトから直接表示することもできます。

次の InTouch アプリケーションディレクトリが共有ディレクトリとして作成されます。

\\GRNodeName\GalaxyName-\$InTouchViewAppObjectName

このディレクトリは、InTouch HMI アプリケーションマネージャではなく IDE によって管理されます。

マネージド InTouch アプリケーションを作成するには

1. System Platform IDE を開きます。
2. [テンプレートツールボックス] で、[システム] ツールセットを展開します。
3. **\$InTouchViewApp** ベース テンプレートからテンプレートを派生させます。以下の手順を実行します。
 - a. **\$InTouchViewApp** ベース テンプレートを右クリックし、[新規] をポイントし、次に [派生テンプレート] をクリックします。デフォルトの名前で新しい派生テンプレートが表示されます。
 - b. 必要に応じて、新しいテンプレートの名前を変更します。
4. 派生テンプレートをダブルクリックします。

[InTouchViewApp 初期化] ダイアログ ボックスが表示されます。

InTouchViewApp initialization

Please select the source of the associated InTouch application

☐ Import existing InTouch application

☒ Create new InTouch application

☒ Overwrite the contents and settings of this galaxy with those from the application template

☒ Overwrite Industrial graphics

☒ Overwrite galaxy style library

☒ Overwrite script function library

☒ Overwrite client controls

☒ Overwrite languages

Backing up your galaxy is highly recommended

☒ Backup the current galaxy

Blank Template

Select application template

Cancel Next

5. [新規 InTouch アプリケーションの作成] を選択し、[次へ] をクリックします。

次のパネルが表示されます。

Application name:
\$InTouchViewApp_002

☐ InTouchView application

Description:

Custom resolution
Screen resolution ▼

Width : 1920 ^ ▼

Height : 1080 ^ ▼

Aspect Ratio:
16 : 9

Cancel Back Next

6. InTouch アプリケーションの名前および説明を入力します。
7. 外部データ ソースとして Application Server 参照のみを使用する InTouch アプリケーションを作成するには、[InTouchView アプリケーション] を選択します。
8. アプリケーションのターゲット解像度がデフォルトの画面解像度オプションと異なる場合、ターゲット解像度を選択します。このフィールドのオプションを以下に示します。
- [ターゲット解像度の選択] ドロップダウンメニューをクリックして、事前定義されたターゲット解像度のリストを表示します。
 - [ターゲット解像度の選択] ドロップダウンメニューをクリックして、[カスタム] を選択します。ピクセルの幅と高さのフィールドが編集可能になります。境界の制限は 150x150 および 10000x10000 です。
9. [次へ] をクリックします。

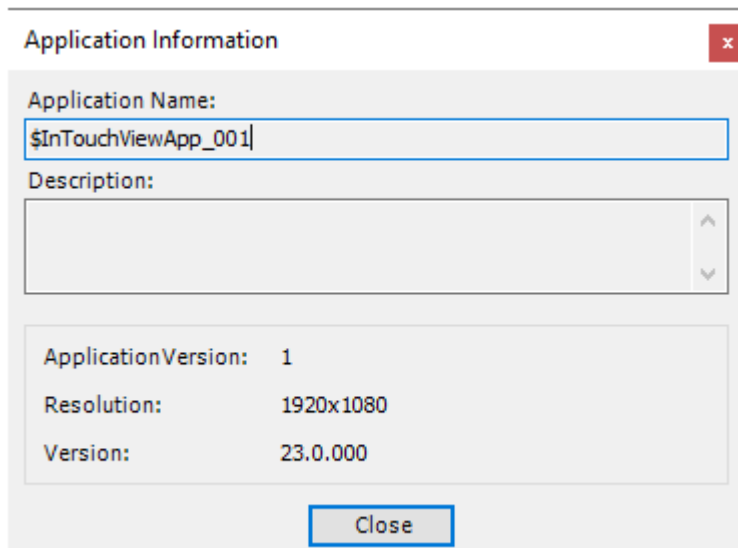
WindowMaker が起動します。

注記: 画面解像度と異なるターゲット解像度を選択した場合、WindowMaker でアプリケーションを編集する際に画面解像度を変換するよう求めるプロンプトは表示されません。

アプリケーションバージョン、解像度、および説明を表示するには

1. テンプレート ツールボックスを開きます。

2. InTouchViewApp テンプレートを右クリックし、次に **[アプリケーション情報]** をクリックします。
[アプリケーション情報] ダイアログボックスが表示されます。



Application Information

Application Name:
\$InTouchViewApp_001

Description:

ApplicationVersion: 1
Resolution: 1920x1080
Version: 23.0.000

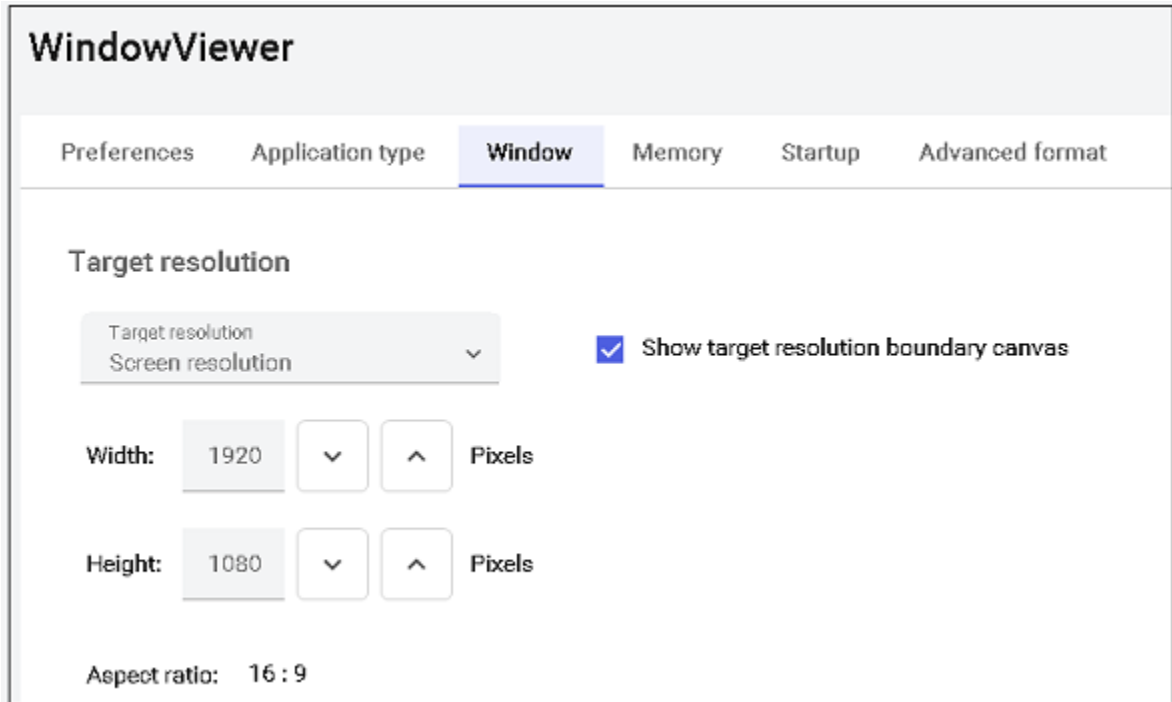
Close

ターゲット解像度を変更するには

WindowMaker でアプリケーションを編集するとき、指定したターゲット解像度を変更できます。この機能はコマンドラインでは使用できません。

以下の手順を実行します。

1. WindowMaker を開きます。
2. **[ファイル]** メニューの **[設定]** をクリックし、**[WindowViewer]** をクリックします。
WindowViewer の設定画面が表示されます。
3. **[ウィンドウ]** タブを選択します。
4. **[ターゲット解像度]** セクションで、必要に応じてターゲット解像度を編集します



5. [OK] をクリックします。

境界キャンバスが更新され、変更が反映されます。ウィンドウ、ウィンドウ サイズ、およびウィンドウ コントロールは変更されずに維持されます。

注記: [ターゲット解像度の境界キャンバスを表示] はデフォルトで選択されます。

アプリケーション テンプレートからのマネージド アプリケーションの作成

アプリケーション テンプレートに基づいてマネージド InTouch アプリケーションを作成すると、デザイン時間を大幅に削減し、新規アプリケーションを既存の標準に基づいて作成することができます。

作成するマネージド InTouch アプリケーションのベースとなるテンプレートを選択するには、アプリケーション テンプレート ブラウザを使用します。アプリケーション テンプレート ブラウザで使用できるテンプレートは、エクスポートされた .aaPKG ファイルです。このファイルは、InTouch インストール ディレクトリの Application Templates フォルダから読み込まれます。次に例を示します。

C:\Program Files(x86)\Wonderware\InTouch\ApplicationTemplates

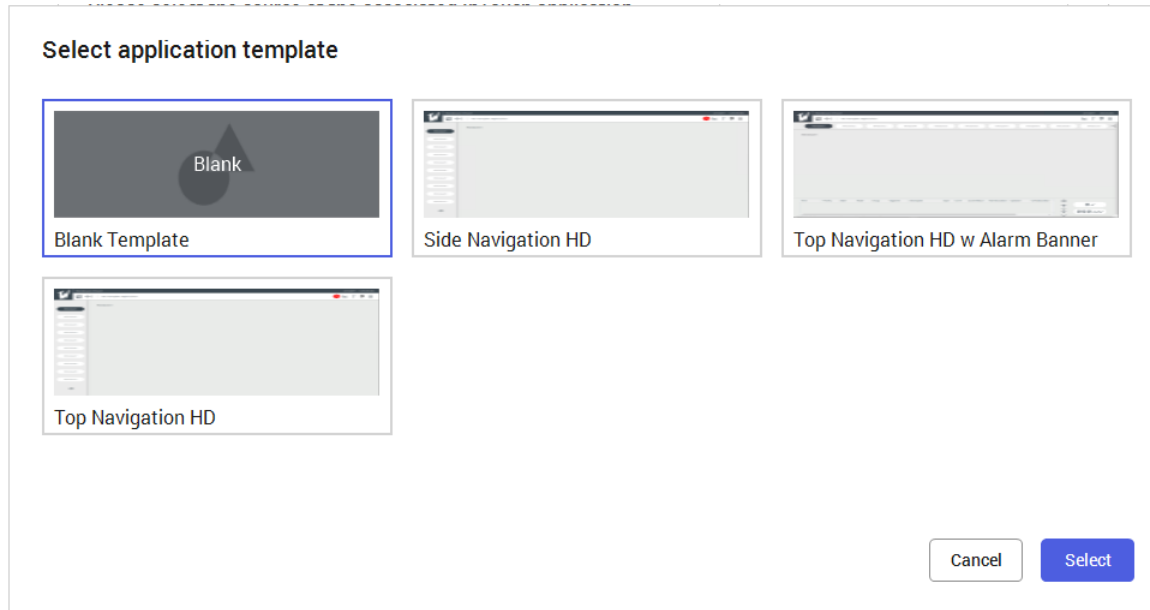
[アプリケーション テンプレート ブラウザ] のフォルダ階層には、このディレクトリの階層が反映されます。

アプリケーション テンプレートに基づいてマネージド アプリケーションを作成するには

1. IDE を開き、\$InTouchViewApp オブジェクトから派生テンプレートを作成します。
派生テンプレートの名前は変更できます (オプション)。
2. 派生テンプレートをダブルクリックします。

[InTouchViewApp 初期化] ダイアログ ボックスが表示されます。デフォルトでは、[新規 InTouch アプリケーションの作成] オプションが選択されています。

3. [アプリケーションテンプレートを選択] をクリックします。
[アプリケーションテンプレートを選択] ダイアログが表示されます。



ブラウザに表示される各サムネイルでは、縦横比が維持されます。

4. 目的のアプリケーションテンプレートを選択して、[OK] をクリックします。
選択したテンプレートが [InTouchViewApp 初期化] ダイアログ ボックスに表示されます。
5. 目的の Galaxy 設定の上書きオプションを注意深くレビューして選択します。

注記: デフォルトでは、[この Galaxy の内容と設定をアプリケーションテンプレートの内容と設定で上書き] オプションが選択されています。上書きされる特定の Galaxy 設定は、子オプションに分割されます。これらのオプションは、この時点で選択を解除できます。

上書きオプションを以下に示します。

- 産業用グラフィックの上書き

このオプションを選択すると、競合が発生した場合、Galaxy シンボルがアプリケーションテンプレートシンボルで置換されます。

- Galaxy スタイルライブラリの上書き

このオプションを選択すると、アプリケーションテンプレートスタイルライブラリによって Galaxy スタイルライブラリ全体が置換されます。選択を解除すると、この手順がスキップされます。スタイルライブラリはマージされません。

- スクリプト関数ライブラリの上書き

このオプションを選択すると、競合が発生した場合、Galaxy スクリプト関数がアプリケーションテンプレートスクリプトライブラリによって置換されます。

- クライアントコントロールの上書き

このオプションを選択すると、競合が発生した場合、Galaxy クライアントコントロールがアプリケーションテンプレートクライアントコントロールによって置換されます。

- 言語の上書き

このオプションを選択すると、Galaxy で定義されている言語設定がアプリケーションテンプレートで定義されている言語設定によって置換されます。

選択を解除すると、競合が発生した場合、Galaxy の言語設定が維持されます。選択を解除したときに競合が発生しなかった場合、アプリケーションテンプレートの言語設定が Galaxy の設定に追加されます。

6. [次へ] をクリックします。

[現在の Galaxy のバックアップ] オプションを選択した場合、バックアップ .cab ファイルの場所およびファイル名を尋ねるプロンプトが表示されます。

重要: Galaxy をバックアップすることが強く推奨されます。

7. アプリケーション名を確認して、[InTouchView アプリケーション] を選択します。

ターゲット解像度は、デフォルトでテンプレートの解像度に設定されます。[ピクセル] フィールドは無効化されています。この設定はアプリケーション開発中に調整できます。

8. [次へ] をクリックします。

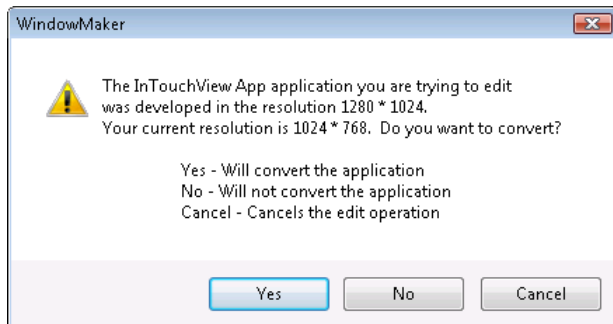
初期化が完了すると、アプリケーションを配置できます。

IDE からの WindowMaker の起動

IDE 内から WindowMaker を起動することにより、マネージド InTouch アプリケーションを編集できます。

InTouchViewApp のテンプレートまたはインスタンスから WindowMaker を起動できます。

現在のシステムの解像度とは異なる画面解像度で作成されたマネージド InTouch アプリケーションを開くとき、メッセージが表示されます。



- InTouch アプリケーションを現在のシステムの解像度に変換して開くには、[はい] をクリックします。
- 元の解像度で InTouch アプリケーションを開いて編集するには、[いいえ] をクリックします。

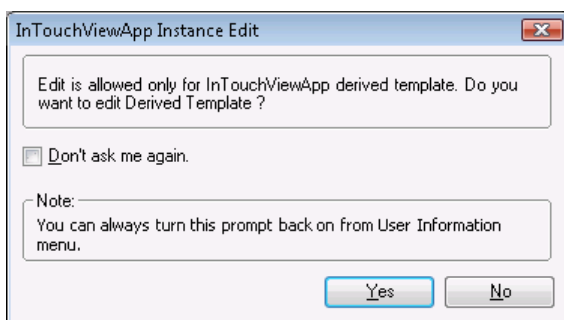
InTouchViewApp テンプレートから WindowMaker を起動するには

1. IDE を開きます。
2. 変更するマネージド InTouch アプリケーションが含まれている InTouchViewApp テンプレートを検索します。

3. InTouchViewApp テンプレートをダブルクリックします。オブジェクトのデフォルトのエディタとして WindowMaker が起動し、InTouch アプリケーションが開きます。マネージドアプリケーションを編集する準備ができました。

InTouchViewApp インスタンスから WindowMaker を起動するには

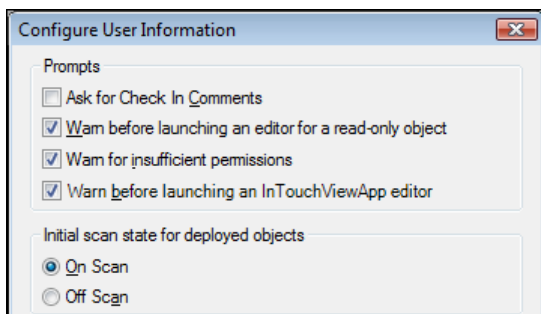
1. IDE を開きます。
2. 変更するマネージド InTouch アプリケーションをホストする親を持つ、InTouchViewApp オブジェクトを検索します。
3. InTouchViewApp オブジェクトをダブルクリックします。[InTouchViewApp インスタンスの編集] ダイアログボックスが表示されます。



4. 以下のいずれかを実行します。
 - テンプレートを開かない場合は、[いいえ] をクリックします。
 - 関連付けられている InTouchViewApp テンプレートを WindowMaker で開くには、[はい] をクリックします。

[再度表示しない] チェック ボックスをオンにし、[はい] をクリックした場合、次回 InTouchViewApp インスタンスを開いたときに、関連付けられている InTouchViewApp テンプレートからマネージド InTouch アプリケーションが自動的に開きます。

この設定は、[編集] メニューから開くことができる [ユーザー情報の編集] ダイアログボックスを使用して変更できます。



マネージド InTouch アプリケーションの言語の切り替え

マネージド InTouch アプリケーションでは、特定の言語の切り替え機能が InTouch HMI ではなく IDE によって処理されます。

IDE を使用して、以下を行うことができます。

- 翻訳用にグラフィック テキストをエクスポートおよびインポートする。
- マネージドアプリケーションによって使用される言語を構成する。

詳細については、**Application Server** ユーザー ヘルプの「言語の操作」の章を参照してください。

InTouch アプリケーションの変更の送信

マネージド InTouch アプリケーションの変更後、対象ノードに配置する変更を送信できます。

マネージド InTouch アプリケーションに変更を加えた後、派生したすべての InTouchViewApp オブジェクトが [変更の保留] アイコンと共に表示されます。

これは、対象ノード上にある WindowViewer が変更を反映するためには、対象ノードに変更を再配置する必要がありますことを示します。

変更をオペレータ ノードに配置する方法の詳細については、「[マネージド InTouch アプリケーションの配置](#)」および「[オペレータ ノードでの新しいアプリケーションバージョンの受け入れ](#)」を参照してください。

InTouch アプリケーション用に変更を送信するには

1. スタンドアロンの InTouch アプリケーションと同じ方法で、WindowMaker でマネージド InTouch アプリケーションを変更します。
2. InTouch ウィンドウを保存します。
3. [ファイル] メニューで、[削除] をクリックします。WindowMaker が閉じ、フォーカスが IDE に戻ります。[チェック イン] ダイアログ ボックスが表示されます。
4. 必要に応じてチェック インのコメントを入力し、[OK] をクリックします。[チェック イン] 進捗ダイアログ ボックスが表示されます。
5. [閉じる] をクリックします。

InTouch アプリケーションのインポート

マネージド InTouch アプリケーションとして使用するために、既存の InTouch アプリケーションをインポートできます。これを行うには、以下の 2 つの手順を実行します。

- InTouchViewApp オブジェクトを作成し、インポートされた InTouch アプリケーションに関連付けます。
- 既存の InTouch アプリケーションをインポートします。

インポートされた InTouch アプリケーションは、マネージド InTouch アプリケーションになります。

IDE が管理しているフォルダに既存の InTouch アプリケーションのコピーが作成されます。IDE によって既存の InTouch アプリケーションがそのまま残され、場所も変わりません。

インポートする InTouch アプリケーションのバージョンが 6.0 以降である場合、変換に関するメッセージが表示されます。アプリケーションは WindowMaker で開く前に変換されます。6.0 以前のバージョンの InTouch アプリケーションは変換できません。

Galaxy に InTouch アプリケーションをインポートするには

1. IDE を開きます。

2. [テンプレート ツールボックス] で、\$InTouchViewApp ベース テンプレートからテンプレートを派生させます。
3. 派生テンプレートを開きます。[InTouchViewApp 初期化] ダイアログ ボックスが表示されます。
4. [既存のアプリケーションのインポート] をクリックし、[次へ] をクリックします。
5. 既存の InTouch アプリケーションを見つけます。以下のいずれかを実行します。
 - 省略記号ボタンをクリックし、マネージド InTouch アプリケーションが含まれているフォルダを参照します。
 - アプリケーションを検索するには、[アプリケーションの検索] チェック ボックスをオンにします。検索を開始する検索ルートを指定し、[検索] をクリックします。リストから InTouch アプリケーションを選択します。
6. [次へ] をクリックします。
7. 必要に応じて、[アプリケーション名] ボックスに新しい名前を入力し、[説明] ボックスに説明を入力します。マネージド InTouch アプリケーションが配置されるときに、アプリケーション マネージャに名前と説明が表示されます。
8. アプリケーションのターゲット解像度がデフォルトの画面解像度オプションと異なる場合、アプリケーションのターゲット解像度を選択します。このフィールドのオプションは、次のように表示されます。
 - a. [ターゲット解像度の選択] ドロップダウン メニューをクリックして、事前定義されたターゲット解像度のリストを表示します。
 - b. [ターゲット解像度の選択] ドロップダウン メニューをクリックして、[カスタム] を選択します。ピクセルの幅と高さのフィールドが編集可能になります。

注記: 既存のアプリケーションをインポートするときに別のターゲット解像度を選択した場合、ソース アプリケーション解像度からターゲット解像度へのアプリケーション解像度の変換は行われません。

9. [次へ] をクリックします。次のパネルが表示され、インポートの進捗が表示されます。
10. [閉じる] をクリックします。[終了] をクリックします。WindowMaker が起動し、InTouch アプリケーションをマネージド InTouch アプリケーションとして編集できます。

グラフィックを含むスタンドアロン InTouch アプリケーションのインポート

マネージド アプリケーションに変換されたスタンドアロン InTouch アプリケーションにグラフィックが含まれる場合、アプリケーションはインポートされますが、グラフィックはインポートされません。

スタンドアロン アプリケーション内に含まれるグラフィックをインポートするには

1. スタンドアロン アプリケーションをテンプレートとしてエクスポートします。
2. テンプレートをアプリケーション テンプレートの場所に保存します。
3. IDE で、この作成したアプリケーション テンプレートを選択して新しい \$InTouchViewApp 派生テンプレートを作成します。

InTouchViewApp オブジェクトのインポートとエクスポート

IDE で InTouchViewApp オブジェクトをインポートおよびエクスポートできます。InTouchViewApp オブジェクトには、マネージド InTouch アプリケーションをホストするためのすべての情報が含まれており、Galaxy 間でマネージド InTouch アプリケーションを交換するために使用できます。

自動オブジェクトのインポートとエクスポートの詳細については、Application Server のマニュアルを参照してください。

InTouchViewApp オブジェクトをインポートするには

1. **[Galaxy]** メニューで、**[インポート]** をポイントし、次に **[オブジェクト]** をクリックします。
[Import Automation Object(s)] ダイアログ ボックスが表示されます。
2. インポートする InTouchViewApp オブジェクトが含まれている .aaPKG ファイルを選択し、**[開く]** をクリックします。オブジェクトおよびそのマネージド InTouch アプリケーションがインポートされます。
3. **[閉じる]** をクリックします。

InTouchViewApp オブジェクトをエクスポートするには

1. **[Galaxy]** メニューで、**[エクスポート]** をポイントし、次に **[オブジェクト]** をクリックします。
[自動オブジェクトのエクスポート] ダイアログ ボックスが表示されます。
2. エクスポート先のフォルダを選択し、.aaPKG ファイル名を指定して **[保存]** をクリックします。オブジェクトおよびそのマネージド InTouch アプリケーションがエクスポートされます。
3. **[閉じる]** をクリックします。

タグ データのエクスポートとインポート

マネージド InTouch アプリケーションのタグ データを、.csv ファイルにエクスポートできます。このデータは、他のマネージド InTouch アプリケーションまたはスタンドアロンの InTouch アプリケーションにインポートできます。

マネージド InTouch アプリケーションからタグ データをエクスポートするには

1. IDE を開きます。
2. タグ データのエクスポート元のマネージド InTouch アプリケーションが含まれている、派生した InTouchViewApp テンプレートを選択します。
3. **[Galaxy]** メニューで、**[エクスポート]** をポイントし、次に **[DB ダンプ]** をクリックします。
[ダンプ先 CSV ファイル名] ダイアログ ボックスが表示されます。
4. .csv ファイルの場所およびファイル名を指定し、**[保存]** をクリックします。
確認ダイアログ ボックスが表示されます。
5. .csv ファイル内でデータ タイプごとにタグ データをグループ化する場合、**[タグ タイプ毎にグループ出力]** を選択します。
6. **[OK]** をクリックします。正常に完了したことを示すメッセージが表示されたら、**[OK]** をクリックします。

.csv ファイルからマネージド InTouch アプリケーションにタグデータをインポートするには

1. IDE を開きます。
2. タグデータのインポート先のマネージド InTouch アプリケーションが含まれている、派生した InTouchViewApp テンプレートを選択します。
3. [Galaxy] メニューで、[インポート] をポイントし、次に [DB ロード] をクリックします。
[ロード元 CSV ファイル名] ダイアログ ボックスが表示されます。
4. .csv ファイルを選択し、[開く] をクリックします。
インポート中、重複した名前があることを警告する、1 つまたは複数のメッセージが表示される場合があります。重複した各タグについて、適切なオプションを選択します。
5. 正常に完了したことを示すメッセージが表示されたら、[OK] をクリックします。

タグの値およびパラメータの保持

1 つのノードで複数のマネージド InTouch アプリケーションを使用している場合、すべてのアプリケーションで同じディレクトリが使用されます。タグデータまたはパラメータを保持するように設定し、アプリケーションを切り替えた場合、ランタイム データが失われます。

マネージド InTouch アプリケーションで保持を設定するには

1. IDE から WindowMaker を使用してマネージド InTouch アプリケーションを開きます。
2. [ファイル] メニューで、[設定] をポイントし、[WindowViewer] をクリックします。
[WindowViewer のプロパティ] 画面が表示されます。
3. [マネージドアプリケーション] タブをクリックします。
4. [ローカルディレクトリ] ボックスで、独自の既存のディレクトリの名前を入力します。これは、対象となるランタイム ノード上にあるディレクトリの名前です。特殊文字を使用して、ローカルの作業ディレクトリ パスを指定することもできます。以下の表には、使用できる特殊文字が示されています。

特殊文字	ローカル作業ディレクトリ パス
%SystemDrive%	システム ドライブ。"C:" など。
%USER%	ログオン ユーザーの名前。WindowViewer がサービスとして実行している場合、アプリケーションは "All Users" にコピーされます。
%APPDATA%	ログオンしているユーザーのアプリケーション データ。WindowViewer がサービスとして実行している場合、アプリケーションは "All Users" アプリケーション データにコピーされます。

特殊文字	ローカル作業ディレクトリパス
%ITUSER%	ログオンユーザー。WindowViewer がサービスとして実行している場合またはコンソールセッションで実行している場合、アプリケーションは "All Users" にコピーされます。
%ITAPPDATA%	ログオンしているユーザーのアプリケーションデータ。WindowViewer がサービスとして実行している場合またはコンソールセッションで実行している場合、アプリケーションは "All Users" アプリケーションデータにコピーされます。

マネージド InTouch アプリケーションの削除

関連付けられている InTouchViewApp テンプレートを削除することにより、IDE で InTouch アプリケーションを削除できます。

これを行うと、テンプレートに関連付けられているテンプレートおよび InTouch アプリケーションディレクトリは完全に削除されます。

関連付けられている InTouchViewApp オブジェクトが派生したインスタンスを持たない場合、InTouch アプリケーションのみ削除できます。

InTouchViewApp インスタンスを削除しても、関連付けられている InTouch アプリケーションは削除されません。

InTouchViewApp テンプレートを削除するには

1. IDE を開きます。
2. 削除するマネージド InTouch アプリケーションが含まれている InTouchViewApp テンプレートを選択します。
3. [ホーム] メニューの [編集] メニューから [削除] をクリックします。
[削除] ダイアログボックスが表示されます。
4. [はい] をクリックします。

InTouchViewApp テンプレートおよび関連付けられている InTouch アプリケーションフォルダが削除されます。

InTouchViewApp 内のすべての Galaxy グラフィックの関連付け

アプリケーションに埋め込むことなく、すべての Galaxy グラフィックを InTouchViewApp テンプレートに関連付けると、配置およびパブリッシュ済み InTouch アプリケーションが、Galaxy 内の任意のグラフィックで作成された「グラフィックの表示」要求を実行できるようになります。

- ShowGraphic() 関数は、グラフィックをパラメータとして使用します。すべての Galaxy グラフィックを関連付ければ、それが InTouchViewApp から参照されているかどうかには関わりなく、ランタイム時に配置され、利用できるようにすることができます。
- すべての Galaxy グラフィックを関連付ければ、ShowGraphic() 関数から参照されているテンプレートシンボルがランタイム時に配置され、利用できるようにすることができます。

注記:「グラフィック」という用語には、グラフィック ツールボックス内に存在するすべてのシンボルまたはクライアント コントロール、およびテンプレートとインスタンスが所有している、または継承されているすべてのシンボルが含まれます。

すべての Galaxy グラフィックと InTouchViewApp との関連付けに関する詳細については、『Application Server ユーザー ガイド』の「アプリケーションの配置と実行」を参照してください。

InTouchViewApp の右クリック コンテキスト メニューで [すべての Galaxy グラフィックを含める] オプションにアクセスします。

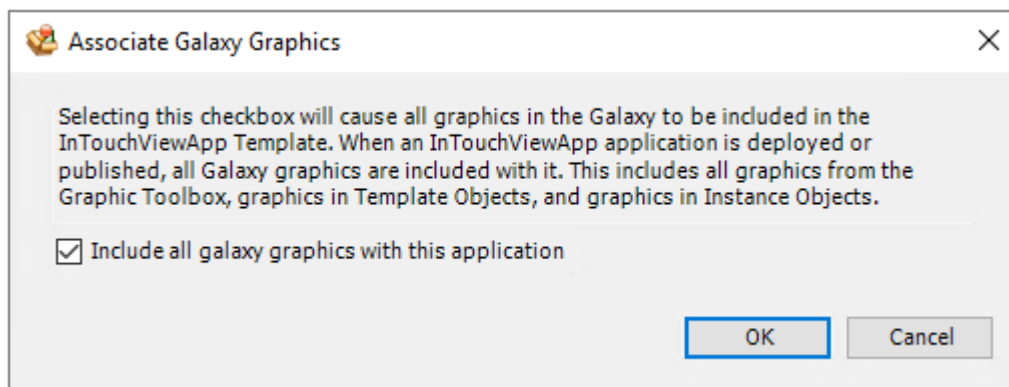
すべての Galaxy グラフィックを InTouchViewApp テンプレートに含めるには

1. 設定する InTouchViewApp テンプレートを右クリックします。

InTouchViewApp のコンテキスト メニューが表示されます。

2. [Galaxy グラフィックの関連付け] メニュー項目を選択します。

[Galaxy グラフィックの関連付け] ダイアログ ボックスが表示されます。



[Galaxy グラフィックの関連付け] ダイアログ ボックスが表示されると、\$InTouchViewApp テンプレートがチェックアウトされます。

\$InTouchViewApp テンプレートがまだ初期化されていない、またはチェックアウトされていない場合には、[すべての Galaxy グラフィックをこのアプリケーションに含める] チェック ボックスは利用できません。

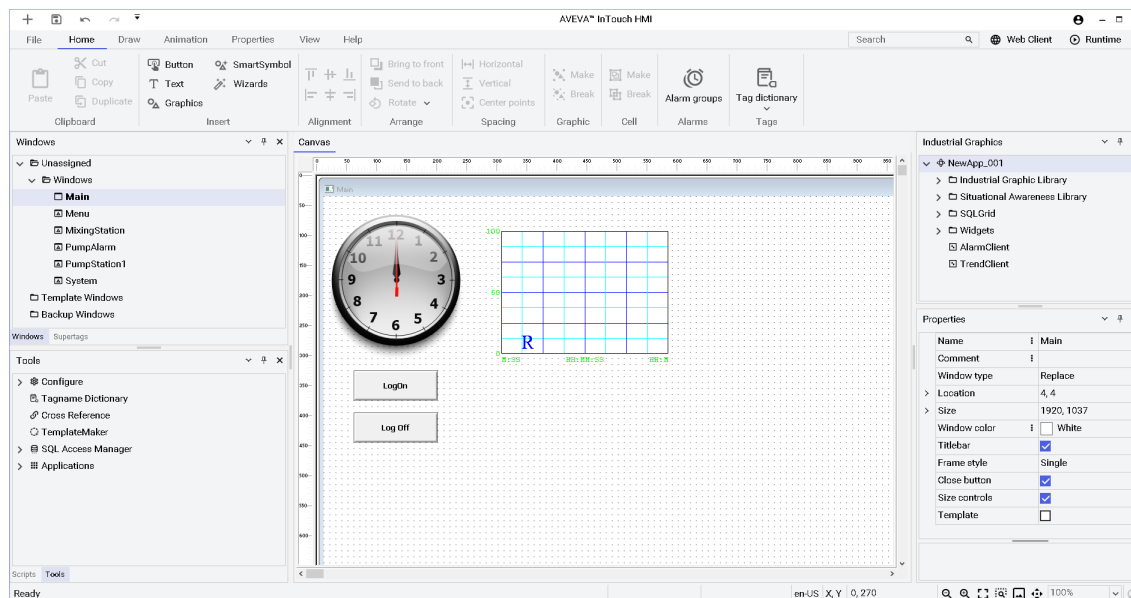
3. チェック ボックスをクリックして、[OK] をクリックします。

ステータス ボックスにはチェックインの進行状況が表示されます。

[すべての Galaxy グラフィックを含める] 設定を変更しようとしたときに InTouchViewApp インスタンスが配置されていた場合には、配置されているすべてのインスタンスの配置が解除されることを示す情報メッセージが表示されます。

章 7 ウィンドウ

アプリケーションウィンドウは、製造プロセスをモデル化する 1 つまたは複数のグラフィックのコンテナです。たとえば、ユニット内の装置を表示するウィンドウを設定できます。別のウィンドウには、そのユニットに関連するアラーム情報のグリッドを表示することができます。



ウィンドウはいくつでも作成でき、背景色、画面位置、ウィンドウタイトルなどのウィンドウプロパティを定義できます。

アプリケーションウィンドウの作成

新しいアプリケーションウィンドウを作成する場合、デフォルト設定では以前に作成されたウィンドウまたは現在アクティブな InTouch ウィンドウの設定が反映されます。以下の設定には、以前に作成したウィンドウの設定が反映されます。

- ウィンドウ名: 最大 32 文字。使用できる特殊文字はドル記号 (\$)、シャープ記号 (#)、およびアンダースコア (_) だけです。ウィンドウを作成する場合、ウィンドウ名を指定することだけが必要とされます。その他のすべてのアイテムはオプションです。

注記: ウィンドウ名にサポートされない文字が含まれる場合、スペース文字などのサポートされない各文字はアンダースコア (_) で置き換えられます。アプリケーションを移行した場合、ウィンドウの編集、またはウィンドウプロパティの変更を行うと、ウィンドウ名に含まれる特殊文字はアンダースコア (_) で置き換えられます。この処理は、アプリケーション、枠、およびテンプレートを始めとするすべてのタイプのウィンドウに適用されます。

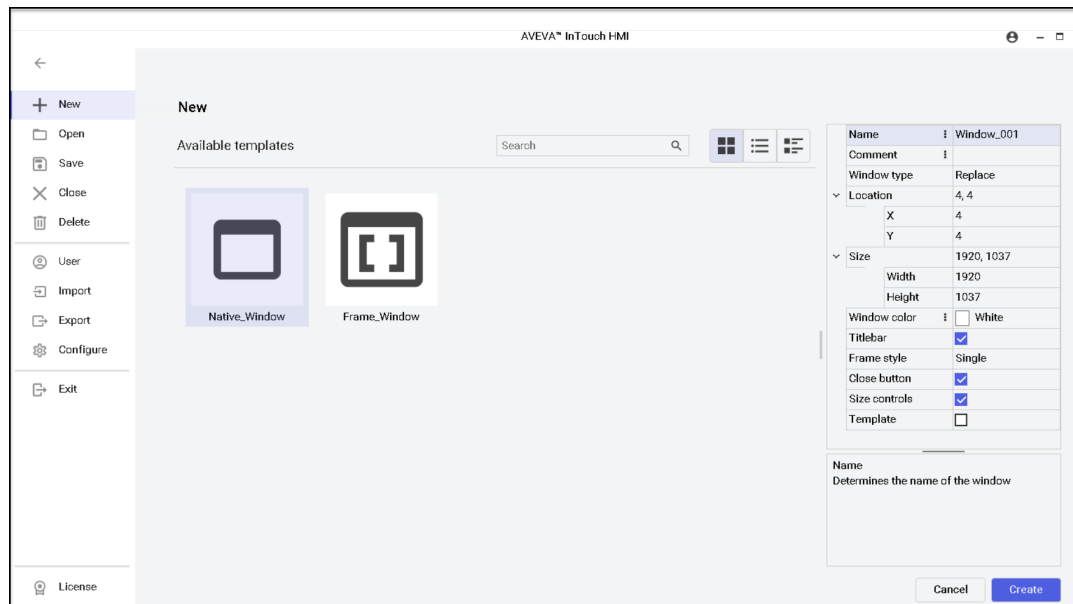
- コメント: ウィンドウに関連付けるコメントは入力できますが、情報目的でのみ使用できます。コメントはファイルリストに表示されますが、アプリケーションでは使用されません。
- ウィンドウの寸法値: デフォルトでは、ウィンドウの寸法値は以前作成したウィンドウの寸法に設定されます。これらの値は、ウィンドウの境界をドラッグすることによってウィンドウサイズを手動で変更した場合にも自動的に修正されます。

注記: 画面解像度と異なるアプリケーション ターゲット解像度を指定した場合、アプリケーション ウィンドウはキャンバス境界を超えることがあります。しかし、ランタイム時に表示されるのは、ターゲット解像度のキャンバス境界内のウィンドウだけです。

新しいウィンドウを作成するには

1. [ファイル] メニューで、[新規] をクリックします。

[新規] ウィンドウが開き、新しいネイティブ ウィンドウおよび枠ウィンドウを作成するために使用できるテンプレートが表示されます。



2. 使用可能なテンプレートから作成するウィンドウのタイプを選択します。
ウィンドウの作成に使用するテンプレートの名前を [検索] ボックスに入力して **Enter** キーを押すこともできます。
3. テンプレートの表示レイアウトを設定するには、[アイコン]、[リスト]、または [詳細] を選択します。
4. [ウィンドウのプロパティ] パネルを使用して、ウィンドウの基本的なプロパティを設定します。
ウィンドウのプロパティは 3 つのカテゴリ（外観、レイアウト、およびウィンドウ スタイル）に分類されます。
5. 新規ウィンドウの外観を設定するには:
 - [名前] テキスト ボックスに、ウィンドウを識別する一意の名前を入力します。
 - [コメント] テキスト ボックスに、ウィンドウに関連付けるコメントを入力します。コメントは 60 文字以内である必要があります。
 - [ウィンドウの色] テキスト ボックスをクリックして、ウィンドウの背景色を選択します。
 - ウィンドウの [枠スタイル] を選択すると、ウィンドウの周囲に境界線が表示されます。
 - 一本の実線の枠にするには、[一重線] をクリックします。
 - 3D 枠線にするには、[二重枠] をクリックします。

- ウィンドウに枠線を付けない場合には、[なし] をクリックします。

[名前]、[コメント]、または [ウィンドウの色] フィールドの横にある省略記号をクリックすると、次の操作を行うことができます。

- リセット - フィールドの値をデフォルト値に戻します。
- 編集 - フィールドの詳細を編集します。
- ソート - ウィンドウのプロパティ フィールドをアルファベット順、カテゴリ、カテゴリとアルファベット順でソートします。
- 説明の表示 - 説明の表示と非表示を切り替えます。
- ツールバーの表示 - ツールバーの表示と非表示を切り替えます。

6. 新規ウィンドウのレイアウトを設定するには:

- **[ウィンドウ タイプ]** ドロップダウンからオプションを選択すると、ランタイム時にウィンドウがどのように開くかを確認できます。
 - [ウィンドウのタイプ] で **[リプレース]** を選択すると、ウィンドウが画面に表示されたときに、そのウィンドウが交差するウィンドウが自動的に閉じます。
 - [ウィンドウのタイプ] で **[オーバーレイ]** を選択すると、ウィンドウは現在開いているウィンドウの上に表示されます。重なっている他のウィンドウよりも大きくなる可能性があります。オーバーレイ ウィンドウが閉じると、その背後にあったウィンドウが再び表示されます。オーバーレイ ウィンドウの背後にあるウィンドウの表示部分をクリックすると、そのウィンドウがアクティブ ウィンドウとして前面に表示されます。
 - [ウィンドウのタイプ] で **[ポップアップ]** を選択すると、ウィンドウは常に他のすべてのウィンドウの一番上に表示されます。ポップアップ ウィンドウの削除は通常、ユーザーからの応答を必要とします。
- **[位置]** フィールドを展開すると、ウィンドウの位置と寸法を指定できます。
 - **[X]** ボックスに設計領域の左端と定義中のウィンドウの左端の間隔をピクセル数で入力します。
 - **[Y]** ボックスに設計領域の上端と定義中のウィンドウの上端の間隔をピクセル数で入力します。
- **[サイズ]** フィールドを展開すると、ウィンドウの高さと幅をピクセル単位で指定できます。

7. 新規ウィンドウのウィンドウ スタイルを設定するには:

- **[閉じるボタン]** チェック ボックスをオンにすると、デザイン時またはランタイム時にウィンドウを閉じるためのボタンがタイトルバーに表示されます。これらのウィンドウは別のアプリケーションにエクスポートした後でも、ウィンドウを閉じるボタンを保持します。
- **[最大化ボタン]** チェック ボックスをオンにすると、デザイン時またはランタイム時にウィンドウを最大化するためのボタンがタイトルバーに表示されます。
- **[最小化ボタン]** チェック ボックスをオンにすると、デザイン時またはランタイム時にウィンドウを最小化するためのボタンがタイトルバーに表示されます。
- **[サイズ コントロール]** チェック ボックスをオンにすると、ユーザーが WindowMaker でウィンドウのサイズを変更できるようになります。
- **[タイトルバー]** チェック ボックスをオンにして、タイトルバーを表示します。

- [テンプレート] チェック ボックスをオンにすると、設定をテンプレートとして保存し、後で使用することができます。
- [ウィンドウの状態] ドロップダウンからウィンドウのデフォルト状態（通常、最小化、または最大化）を選択します。

8. [OK] をクリックします。

テンプレート ウィンドウとしてのアプリケーション ウィンドウの設定

ウィンドウのプロパティを使用して、アプリケーション ウィンドウをテンプレート ウィンドウとして設定できます。このように設定すると、ウィンドウを再使用して設定時間を短縮できます。

アプリケーション ウィンドウをテンプレート ウィンドウとして設定するには

1. 次のいずれかを実行して、[ウィンドウのプロパティ] ペインを開きます。
 - a. 新しいウィンドウをテンプレート ウィンドウとして設定するには、[ファイル] メニューで [新規] をクリックします。
 - b. 既存のウィンドウをテンプレート ウィンドウとして設定するには、[ウィンドウ] ペインで既存のウィンドウをクリックします。

画面の右側にウィンドウのプロパティ ペインが表示されます。

2. [ウィンドウのプロパティ] ペインで [テンプレート] チェックボックスをオンにします。

Name	Window_001
Comment	
Window type	Overlay
Location	4, 4
X	4
Y	4
Size	632, 278
Width	632
Height	278
Window color	250, 250, 250
Titlebar	<input checked="" type="checkbox"/>
Frame style	Single
Close button	<input checked="" type="checkbox"/>
Size controls	<input checked="" type="checkbox"/>
Template	<input checked="" type="checkbox"/>

Name
Determines the name of the window

3. [OK] をクリックします。

[ウィンドウ] ペインの下にある [テンプレート ウィンドウ] フォルダにウィンドウが表示されます。新しく作成したウィンドウは、テンプレート ウィンドウ ブラウザでも使用できます。

注記: [ウィンドウのプロパティ] ペインの [テンプレート] チェックボックスの選択を解除すれば、テンプレート ウィンドウをアプリケーション ウィンドウに戻すことができます。このプロパティは、複数の [テンプレート ウィンドウ] フォルダの間でウィンドウをドラッグアンドドロップすることによっても変更できます。

テンプレート ウィンドウからのアプリケーション ウィンドウの作成

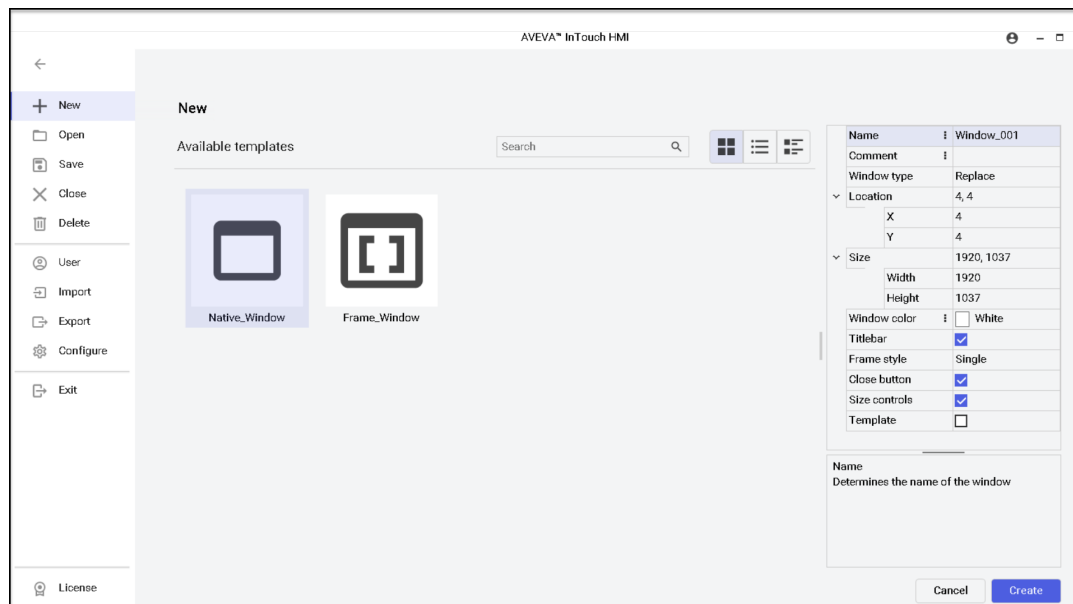
テンプレート ウィンドウ ブラウザで使用可能なテンプレート ウィンドウから InTouch アプリケーション ウィンドウを作成できます。

注記: テンプレート ウィンドウ ブラウザで使用できるのは、[テンプレート] プロパティが選択されているウィンドウだけです。詳細な手順については、「[テンプレート ウィンドウとしてのアプリケーション ウィンドウの設定](#)」を参照してください。

テンプレート ウィンドウからアプリケーション ウィンドウを作成するには

1. [ファイル] メニューで、[新規] をクリックします。

[使用可能なテンプレート] が表示されます、



2. テンプレート ウィンドウのサムネイルを参照して、目的のウィンドウを選択します。

注記: デフォルトでは、各テンプレート ウィンドウがサムネイル ビューに表示されます。ウィンドウ サムネイルは、アプリケーション全体に対してデザイン時の実際のウィンドウの位置に相対的に表示されます。

3. [OK] をクリックします。

注記: 移行したアプリケーションのテンプレート ウィンドウのウィンドウ名にサポートされない文字が含まれる場合、新しいアプリケーション ウィンドウ内のスペース文字などのサポートされない各文字はアンダースコア (_) で置き換えられます。

枠ウィンドウの操作

スタンドアロンまたはマネージド InTouch アプリケーションで作業している場合、アプリケーション ウィンドウに加えて、またはアプリケーション ウィンドウの代わりに枠ウィンドウを作成および開発できます。枠ウィンドウを使用すると、パン、ズーム、およびタッチ機能をサポートできる産業用グラフィックをホストできます。

枠ウィンドウは、アプリケーション ウィンドウと同じプロパティの大半をサポートします。しかし、いくつかの制限が適用されます。

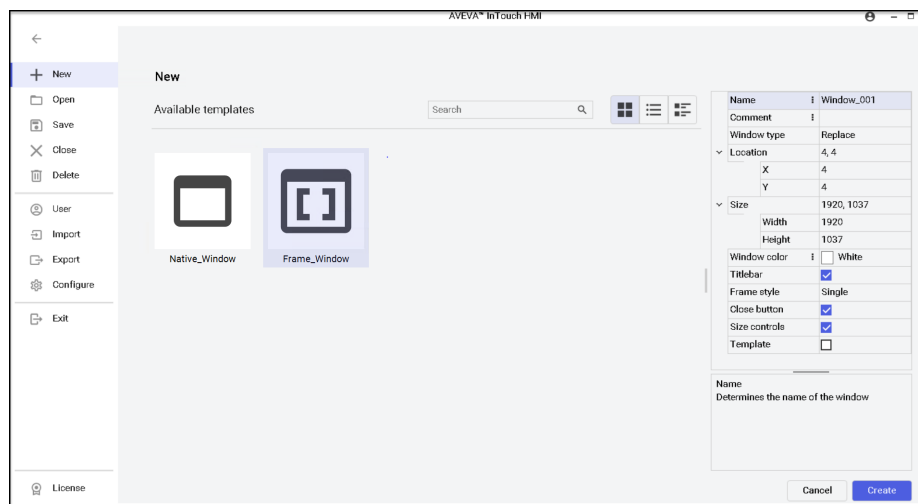
枠ウィンドウには、以下の制限があります。

- 複数の産業用グラフィックはサポートされません。
- ネイティブの InTouch コントロールはホストされません。
- ActiveX コントロールはホストされません。
- SmartSymbol はサポートされません。
- ウィンドウ コンポーネントの「元に戻す」/「やり直す」操作はサポートされません。
- 埋め込まれたシンボルはフリップおよび回転しません。
- 切り取り、コピー、貼り付け、および重複操作はサポートされません。
- 枠ウィンドウに埋め込まれた産業用グラフィックのクロス参照のサポートは提供されません。
- 産業用グラフィックへのウィンドウ変換はサポートされません。
- 新規インスタンスの作成、インスタンスの編集、および埋め込まれたオートメーション オブジェクト グラフィックの代替インスタンスの選択はサポートされません。

枠を作成するには

1. [ファイル] メニューで、[新規] をクリックします。

[使用可能なテンプレート] のリストを含む [新規] ウィンドウが表示されます。



2. [枠ウィンドウ] を選択します。
3. ウィンドウのプロパティを設定して、[OK] をクリックします。

プロパティ パネルの使用

枠のプロパティは、WindowMaker キャンバスの [プロパティ] パネルから直接編集できます。枠のプロパティは、ウィンドウの作成時にプロパティ グリッドに入力されます。

Name	:	Window_001
Comment	:	
Window type		Replace
Location		4, 4
X		4
Y		4
Size		1920, 1037
Width		1920
Height		1037
Window color	:	<input type="checkbox"/> White
Titlebar		<input checked="" type="checkbox"/>
Frame style		Single
Close button		<input checked="" type="checkbox"/>
Size controls		<input checked="" type="checkbox"/>
Template		<input type="checkbox"/>

デフォルトでは、産業用グラフィック ツールボックスおよびプロパティ グリッドがキャンバスの右側に表示されます。グラフィック ツールボックスまたはプロパティ グリッドは、キャンバスの左側に固定できます。WindowMaker の設定をデフォルトに戻すには、[表示]、[レイアウトのリストア] をクリックします。

注記: アプリケーション ウィンドウのプロパティは、**プロパティ グリッド**に表示されません。

枠ウィンドウは、アプリケーション ウィンドウと同じプロパティに加えて、3 つの追加プロパティをサポートします。詳細については、以下を参照してください。

各ウィンドウ プロパティの詳細な説明については、「[アプリケーション ウィンドウの作成](#)」を参照してください。

枠のプロパティ

機能

MaximizeButton

枠の右上の最大化ボタンを有効にします。
デフォルトは **[False]** です。

MinimizeButton

枠の右上の最小化ボタンを有効にします。
デフォルトは **[False]** です。

WindowState ウィンドウの初期状態（**[Normal]**、**[Minimized]**、または **[Maximized]**）を表示します。

デフォルトは **[Normal]** です。

FrameStyle 以外の枠のプロパティに対して行った変更は、直ちに枠に反映されます。このプロパティを設定するには、最初に **SizeControl** および **TitleBar** プロパティを **False** に設定する必要があります。




[FrameStyle] ドロップダウン メニューが有効になります。

枠に埋め込まれたグラフィックの操作

産業用グラフィックは、アプリケーション ウィンドウと同じ方法で枠ウィンドウに埋め込むことができます。産業用グラフィック ツールボックスは、**System Platform IDE** と **InTouch WindowMaker** の両方に統合されています。グラフィック ツールボックスには、定義済みシンボルの産業用グラフィック ライブラリと **Situational Awareness Library** を含む個別のフォルダが含まれます。産業用グラフィック ライブラリには標準的な産業用オブジェクトの写実的なシンボルが含まれています。

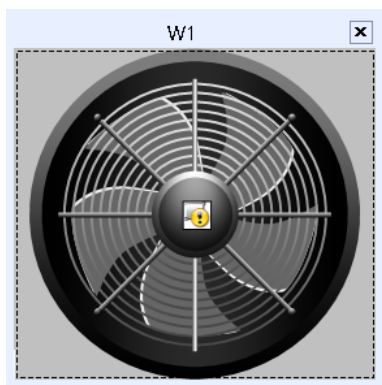
Situational Awareness Library のシンボルは、簡略化された外観で、最低限の視覚的詳細を提供することにより、オペレータに無関係の情報を表示することなく、機能の目的とステータスを効率的に伝達します。ほとんどの **Situational Awareness Library** のシンボルは、各シンボルに複数の視覚的および機能的な設定が組み込まれたシンボル ウィザードとして設計されています。シンボルの機能設定は、シンボルの **[ウィザード オプション]** からオプションを選択するだけで簡単に選択できます。

シンボルを枠に埋め込むには、次のいずれかの手順を実行します。

- 枠ウィンドウを作成した後、キャンバスにある **[グラフィックを埋め込む]** ボタン  をクリックしてグラフィックを参照します。
- 枠ウィンドウを作成した後、キャンバスにある **[グラフィックを作成]** ボタン  をクリックして産業用グラフィック エディタでグラフィックを作成し、グラフィックを保存します。作成したグラフィックはウィンドウに埋め込まれます。
- 産業用グラフィックをグラフィック ツールバーから枠にドラッグアンド ドロップします。
- **[産業用グラフィックの埋め込み]** ツールバー ボタン  を選択して、グラフィックを参照します。
- 枠ウィンドウのコンテキスト メニューを使用してグラフィックを埋め込みます。

枠のサイズを埋め込まれた産業用グラフィックに自動的に合わせることができます。埋め込まれたシンボルに枠のサイズを合わせるには、ウィンドウを右クリックして、**[シンボルに合わせる]** を選択します。

枠のサイズがシンボルに合わせて変更されます。



注記: [シンボルに合わせる] の操作は元に戻すことができません。枠ウィンドウの寸法または Size プロパティを手動で変更した場合、[シンボルに合わせる] 操作を再度実行する必要があります。

産業用グラフィックを枠に埋め込むと、以下に示すように、シンボルが [プロパティ グリッド] ドロップダウンメニューに含まれるようになります。

Name	:	Window_001
Comment	:	
Window type	:	Overlay
Location	:	4, 4
X	:	4
Y	:	4
Size	:	632, 278
Width	:	632
Height	:	278
Window color	:	250, 250, 250
Titlebar	:	<input checked="" type="checkbox"/>
Frame style	:	Single
Close button	:	<input checked="" type="checkbox"/>
Size controls	:	<input checked="" type="checkbox"/>
Template	:	<input checked="" type="checkbox"/>

Name
Determines the name of the window

枠ウィンドウとシンボル プロパティを切り替えることができます。以下のいずれかを実行します。

- ドロップダウン メニューからシンボルまたは枠ウィンドウ名を選択します。
- 枠ウィンドウを右クリックして、[ウィンドウのプロパティ] を選択します。
- タイトルバーまたは枠の境界をクリックして、枠ウィンドウのプロパティをグリッドに表示します。

埋め込まれたシンボルをクリックして、シンボルのプロパティを表示します。

シンボルのプロパティを設定して、キーボード、マウス、およびタッチ ジェスチャを使用したパンとズーム機能を有効にする必要があります。設定可能なシンボル プロパティの一覧を以下に示します。

シンボルのプロパティ 機能

MaintainAspectRatio	モダン枠のサイズを変更する際にシンボルのアスペクト比を維持します。デフォルトは True です。
SymbolName	モダン枠によってホストされるシンボル名を設定します。
InteractionMode	操作モードを設定します。以下のオプションがあります。 <ul style="list-style-type: none">• None: パンとズームを無効にします。• PanZoom (デフォルト) : パンとズームを有効にします。
ShowZoomControl	シンボルとズーム コントロールを表示します。以下のオプションがあります。 <ul style="list-style-type: none">• Auto: コントロールは必要に応じて表示されます。• Visible: コントロールは常に表示されます。

ランタイム時のパンとズーム機能の使用の詳細については、『AVEVA™ InTouch HMI コンポーネントの標準の作成』ガイドを参照してください。

枠ウィンドウのリボンの使用

枠フレームワークを作成する際、ウィンドウのリボンで以下のオプションが使用可能になります。

- シンボルの編集 - シンボルを編集できるグラフィック エディタを起動します。詳細については、『産業用グラフィック エディタ ユーザー ガイド』を参照してください。
- ウィンドウを保存 - ウィンドウを保存します。「[ウィンドウを開く、保存する、閉じる](#)」を参照してください。
- ウィンドウに名前を付けて保存 - ウィンドウの複製を作成します。「[ウィンドウの複製](#)」を参照してください。
- ウィンドウの削除 - ウィンドウを削除します。「[ウィンドウの削除](#)」を参照してください。



枠ウィンドウのキャンバスを右クリックすると、使用可能ないくつかのオプションがリボンでハイライトされます。

アプリケーション ウィンドウの変更

アプリケーションを開発するときに、いつでもウィンドウのプロパティを変更できます。

アプリケーション ウィンドウのプロパティを変更するには

1. ウィンドウ名を右クリックして、[プロパティ] を選択します。

2. [ウィンドウのプロパティ] パネルで必要な変更を行います。

3. [OK] をクリックします。

ウィンドウ オプションの詳細については、「[アプリケーション ウィンドウの作成](#)」を参照してください。

ウィンドウを開く、保存する、閉じる

ウィンドウを開く

アプリケーションの開発中、コンピュータのメモリがサポートする範囲で複数のウィンドウを開くことができます。

ウィンドウを開くには

1. [ファイル] メニューで、[開く] をクリックします。

[開くウィンドウを選択] ダイアログ ボックスが表示され、アプリケーション内のすべてのウィンドウ名が一覧表示されます。

2. 以下のいずれかを実行します。

- ウィンドウを1つ開くには、ウィンドウ名をダブルクリックします。
- 複数のウィンドウを開くには、開くウィンドウのチェック ボックスをオンにして、[OK] をクリックします。

ウィンドウの保存

ウィンドウを保存するときには、ウィンドウに関連付けられているすべてのグラフィック、QuickScript、プロパティなども保存されます。

ウィンドウを保存するには

1. [ファイル] メニューで、[保存] をクリックします。

[保存するウィンドウを選択] ダイアログ ボックスが表示され、すべてのウィンドウの名前が一覧表示されます。

2. 保存する必要があるウィンドウを選択します。

3. [OK] をクリックします。

ウィンドウを閉じる

変更されたウィンドウを閉じると、変更内容を保存するよう求めるメッセージが表示されます。

ウィンドウを閉じるには

1. [ファイル] メニューの [閉じる] をクリックします。

[閉じるウィンドウを選択] ダイアログ ボックスが表示され、現在開いているすべてのウィンドウ名が一覧表示されます。

2. ウィンドウ名の横にあるチェック ボックスをオンにします。

3. [OK] をクリックします。

ウィンドウ サムネイル プレビューの表示

WindowMaker でウィンドウのサムネイルをプレビュー表示することができます。これは、特に多くのウィンドウを持つアプリケーションの場合に便利な機能です。ウィンドウを開く前に、サムネイルのプレビュー表示で正しいウィンドウかどうかを確認することができます。

ウィンドウのサムネイルをプレビュー表示することができます。サムネイルのプレビューの表示と更新は、ウィンドウを保存した後にのみ行うことができます。アプリケーションを移行またはウィンドウを XML ファイルからインポートする場合、ウィンドウを明示的に保存しなくてもサムネイルのプレビューを表示することができます。別のアプリケーションからウィンドウをインポートする場合、ウィンドウを明確に保存しなくてもサムネイルのプレビューを表示することができます。しかし、インポート後にウィンドウを変更した場合、ウィンドウを保存しなければサムネイルに変更は反映されません。

InTouch アプリケーションのすべてのウィンドウ サムネイルを更新することもできます。サムネイルを更新する前に、すべてのウィンドウを閉じる必要があります。

すべてのウィンドウ サムネイルを更新するには

- [表示] メニューの [更新] グループで [ウィンドウ サムネイル] をクリックします。

特定のウィンドウのサムネイルを更新するには

1. ウィンドウを右クリックして、[サムネイルの更新] をクリックします。

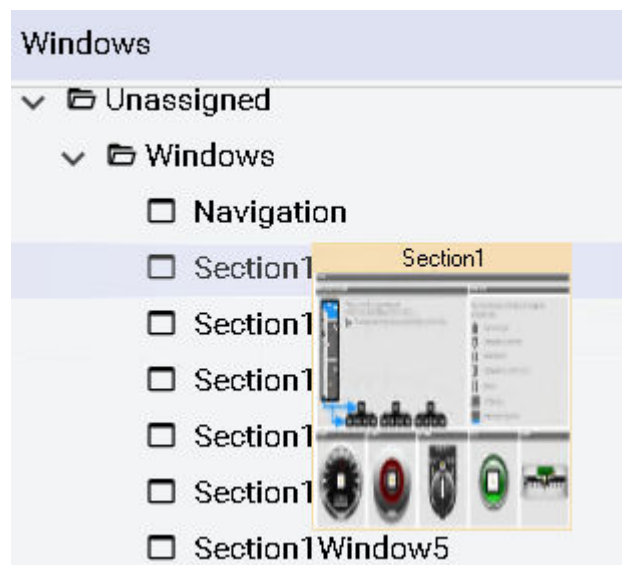
確認メッセージが表示されます。

2. [続行] をクリックします。InTouch アプリケーションのすべてのウィンドウ サムネイルが更新されます。

サムネイルを更新するために、システムによってすべてのウィンドウが開かれてから閉じられます。

ウィンドウのサムネイル プレビューを表示するには

1. WindowMaker を開きます。
2. アプリケーションを開いてウィンドウを保存します。
3. ウィンドウの名前と関連付けられたアイコンを表示している長方形領域の上にポインタを移動します。サムネイルのプレビューが表示されます。



ウィンドウの複製

シミュレーションおよびコントロールが類似した複数のプロセスがある場合、ウィンドウを複製して、2つ目のプロセスまたはユニット用にカスタマイズして使用することができます。

ウィンドウに関連付けられているすべてのグラフィック、QuickScript、プロパティなども含まれるウィンドウを複製できます。

開始する前に、複製するウィンドウを開いた状態にし、少なくとも1回保存する必要があります。一度に複製できるのは1つのウィンドウのみです。

ウィンドウを複製するには

1. [ウィンドウ] ペインで目的のウィンドウを右クリックし [名前を付けて保存] をクリックします。
[ウィンドウを保存] ダイアログ ボックスが開きます。
2. [新しい名前] ボックスに、新しいウィンドウの名前を入力します。
3. [OK] をクリックします。

注記: 元のウィンドウ名にサポートされない文字が含まれる場合、複製したウィンドウ名のサポートされない各文字はアンダースコア (_) 文字で置き換えられます。

ウィンドウの削除

コンピュータの保存領域を節約するため、またはアプリケーション エクスプローラのウィンドウ リストが長すぎて管理できない場合には、使用していないウィンドウを削除できます。

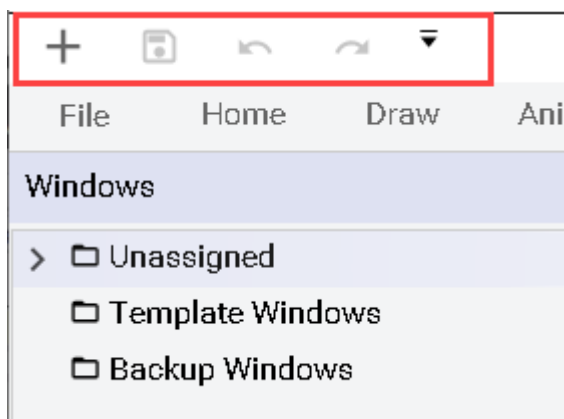
注意: 正しいウィンドウを削除することを確認してください。削除されたウィンドウは、[元に戻す] では復元できません。

ウィンドウを削除するには

1. [ファイル] メニューで、[削除] をクリックします。
[削除するウィンドウを選択] 画面が表示されます。
2. 削除するウィンドウ名を選択して、[OK] をクリックします。メッセージが表示されたら、[はい] をクリックします。
3. [OK] をクリックします。

クイック アクセス ツールバーの使用

クイック アクセス ツールバーはカスタマイズ可能なツールバーで、事前定義済みのコマンドまたは頻繁に使用するコマンドを含めて、すべての画面からすばやくアクセスすることができます。デフォルトでは、クイック アクセス ツールバーは、WindowMaker 画面の左上角に表示されます。



クイック アクセス ツールバーからのコマンドの実行

クイック アクセス ツールバーからは以下のコマンドを実行できます。

注記: クリック アクセス ツールバーのほとんどのコマンドは他のリボン グループからもアクセスできますが、いくつかのコマンドはクリック アクセス ツールバーからしかアクセスできません。

クイック アクセス ツールバーのコマンド	説明
新規	新しいウィンドウを作成します。
開く	既存のウィンドウを開きます
保存	ウィンドウを保存します
名前を付けて保存	ウィンドウのコピーを作成し、別の名前と場所を指定して保存します
すべて保存	開いているすべてのウィンドウを保存します
閉じる	開いているウィンドウを閉じます
削除	ウィンドウを削除します
インポート	ウィンドウをインポートします
エクスポート	ウィンドウをエクスポートします
印刷	ウィンドウを印刷します
WindowViewer	ウィンドウを WindowViewer で表示します
産業用グラフィックに変換	既存のグラフィックを産業用グラフィックに変換します
元に戻す	最後の操作をキャンセルします
やり直す	最後の操作を繰り返します
終了	WindowMaker を終了します

クイック アクセス ツールバーのカスタマイズ

デフォルトでは、クイック アクセス ツールバーには、[新規]、[保存]、[元に戻す]、および [やり直す] のコマンドが表示されます。

リストからクイック アクセス ツールバーに表示するコマンドを追加または削除できます。

クイック アクセス ツールバーにコマンドを追加するには:

1. クイック アクセス ツールバーの下向きの矢印をクリックします。
2. 追加するコマンドを選択します。

クイック アクセス ツールバーからコマンドを削除するには:

1. クイック アクセス ツールバーの下向きの矢印をクリックします。
2. 選択したコマンドをクリックします。

クイック アクセス ツールバーの配置

デフォルトでは、クイック アクセス ツールバーはリボンの上に表示されます。

- クイック アクセス ツールバーをリボンの下に配置するには、クイック アクセス ツールバーの下向きの矢印をクリックして **[リボンの下に表示]** を選択します。
- クイック アクセス ツールバーをリボンの上に配置するには、クイック アクセス ツールバーの下向きの矢印をクリックして **[リボンの上に表示]** を選択します。

クイック アクセス ツールバーを使用したリボンのサイズ変更

クイック アクセス ツールバーからリボンを最小化または最大化することができます。

- リボンを最小化: このビューではメニュー項目だけが表示され、グループは表示されません。リボンを最小化するには、クイック アクセス ツールバーの下向きの矢印をクリックして **[リボンを最小化]** を選択します。
- リボンを最大化: このビューではメニュー項目とグループが表示されます。リボンを最大化するには、クイック アクセス ツールバーの下向きの矢印をクリックして **[リボンを最大化]** を選択します。

InTouch ウィンドウに関する情報の印刷

InTouch ウィンドウに関する以下の情報を印刷できます。

- ウィンドウ内に配置されたグラフィカルオブジェクトの詳細。たとえば、すべてのタイプのオブジェクトのウィンドウの位置、テキストオブジェクトに使用されているフォント、産業用グラフィック用に定義されたカスタムプロパティなどです。
- ウィンドウで使用されているアニメーションリンクに関する詳細
- ウィンドウに関連するスクリプト。
- ウィンドウで使用されているタグ。

グラフィカルオブジェクトの詳細は、プリンタまたは .html ファイルに出力できます。各ウィンドウに対して、.html ファイルが作成されます。.html ファイルには、以下のものが含まれています。

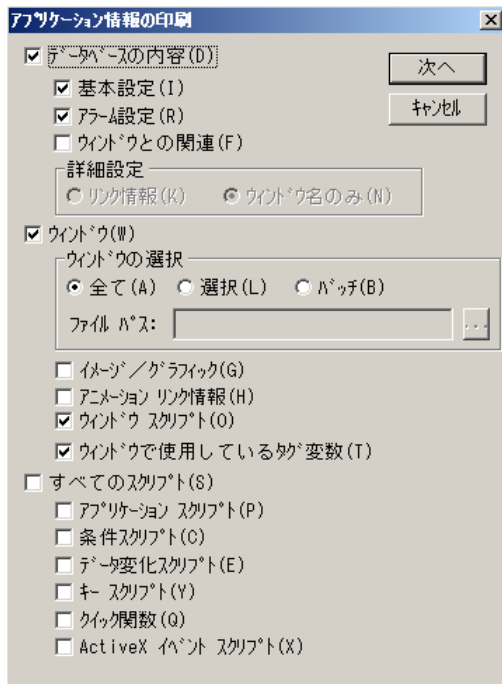
- 参照されている .png ファイルとしてのウィンドウのピクチャ。
- ウィンドウ内の各グラフィカルオブジェクトに関する詳細の一覧。

複数のウィンドウについてのグラフィカルオブジェクトの詳細を印刷した場合、ウィンドウ固有の .html ファイルにリンクしているサマリ .html ファイルが作成されます。

プリンタまたは .txt ファイルにアニメーションリンク、スクリプト、またはタグの詳細を出力できます。

InTouch ウィンドウに関する情報を印刷するには

1. WindowMaker で InTouch アプリケーションを開きます。
2. クイック アクセス ツールバーの [印刷] をクリックします。
[アプリケーション情報の印刷] ダイアログ ボックスが表示されます。



3. [ウィンドウ] を選択します。
4. 印刷するウィンドウを選択します。
 - [すべて] を選択すると、アプリケーションのすべてのウィンドウの情報（埋め込まれたグラフィックの名前を含む）が出力されます。
 - [選択] を選択すると、特定のウィンドウの情報（選択したウィンドウの埋め込まれたグラフィックの名前を含む）だけが出力されます。[ウィンドウの印刷] ダイアログ ボックスが開きます。アプリケーションで印刷するウィンドウを選択して [OK] をクリックします。
 - [バッチ] を選択すると、.csv ファイルで指定したウィンドウの情報（埋め込まれたグラフィックの名前を含む）が出力されます。

.csv 形式の詳細については、「[ウィンドウを印刷するための .CSV 形式](#)」を参照してください。

5. 選択したウィンドウ内の印刷対象を選択します。
 - [イメージ/グラフィック] を選択すると、ウィンドウに配置されているすべてのグラフィック オブジェクトに関する情報が出力されます。
 - [アニメーション リンク情報] を選択すると、ウィンドウのリンクの詳細が出力されます。

- [ウィンドウ スクリプト] を選択すると、ウィンドウに関連付けられているスクリプトが出力されます。
 - [ウィンドウで使用しているタグ] を選択すると、ウィンドウで使用しているタグが出力されます。
6. [次へ] をクリックします。[出力の送信先の選択] ダイアログ ボックスが表示されます。
7. 以下のいずれかを実行します。
- 情報を印刷するには、[プリンタに出力を送信する] をクリックします。
 - 単一の .txt ファイルを作成するには、[テキスト ファイルに出力を送信する] をクリックします。
 - 指定した各ウィンドウのサマリ .html ファイル、単一の .html ファイル、および .png ファイルを作成するには、[HTML ファイルに出力を送信する] をクリックします。 .html ファイルおよび .png ファイルが存在する場合、自動的に上書きされます。
8. [印刷] をクリックします。

コマンドプロンプトからのウィンドウ情報の印刷

コマンドプロンプトからウィンドウ情報を印刷できます。これを行うには、ウィンドウ名が含まれている .csv ファイルを作成し、次に印刷コマンドで .csv ファイルを参照します。

印刷コマンドが実行されると、デフォルトの InTouch アプリケーションが WindowMaker で自動的に開き、印刷操作が実行されてから閉じます。コマンドプロンプトからの印刷は、スタンドアロン InTouch アプリケーションに対してのみ機能します。

コマンドプロンプトからウィンドウ情報を印刷するには

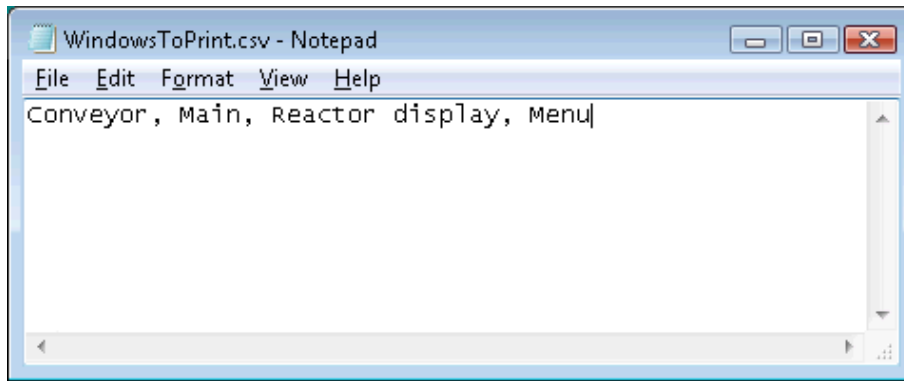
1. 印刷するウィンドウの名前が含まれている .csv ファイルを作成します。
 .csv 形式の詳細については、「[ウィンドウを印刷するための .CSV 形式](#)」を参照してください。
2. WindowMaker を終了し、.csv ファイルを閉じます。
3. コマンドプロンプトを開きます。
4. コマンドを入力して、情報を印刷します。
 コマンドの構文の詳細については、「[コマンドプロンプトから印刷するための構文](#)」を参照してください。
5. **Enter** キーを押します。WindowMaker が起動し、情報が印刷されます。

ウィンドウを印刷するための .CSV 形式

.csv ファイルを使用すれば、ウィンドウ情報を Microsoft Excel のような様々な形式で印刷することができます。 .csv ファイルで単一行を作成し、各ウィンドウ名をカンマで区切って入力します。

Window1,Window2,Window3,WindowN

次に例を示します。



Microsoft Excel を使用してファイルを作成する場合、1 行の各カラムセルに各ウィンドウの名前を入力します。

ウィンドウ名に円記号 (\) は使用できません。

カンマを使用する必要があります。その他の区切り文字は使用できません。

コマンドプロンプトから印刷するための構文

コマンドの構文は以下のとおりです。

"<WindowMaker のパス名>" COMMANDFILE="<CSV ファイル>" ALL OUTPUTTARGET=<target name>

構文の要素を以下に示します。

<WindowMaker のパス名> は、WindowMaker アプリケーション (WM.exe) のパスです。

- <CSV ファイル> は、印刷するウィンドウを指定する .csv ファイルの名前です。
- <ターゲット名> は、プリンタまたは .html ファイルへの出力先です。
- ALL は、すべてのリンク、データベース エントリ、およびスクリプト情報を印刷するコマンドです。ALL コマンドを含めない場合、グラフィカル オブジェクト情報のみが印刷されます。

以下の例では、すべてのリンク、データベース エントリ、およびスクリプト情報がデフォルトのプリンタに送信されます。

```
"C:\Program Files\Wonderware\InTouch\wm.exe" COMMANDFILE="D:\print.csv" ALL OUTPUTTARGET = PRINTER
```

以下の例では、グラフィカル オブジェクト情報が .html ファイルに送信されます。

```
"C:\Program Files\Wonderware\InTouch\wm.exe" COMMANDFILE="D:\print.csv" OUTPUTTARGET = HTML  
<DEMOAPP.html>
```

コンテキスト メニューの深さの設定

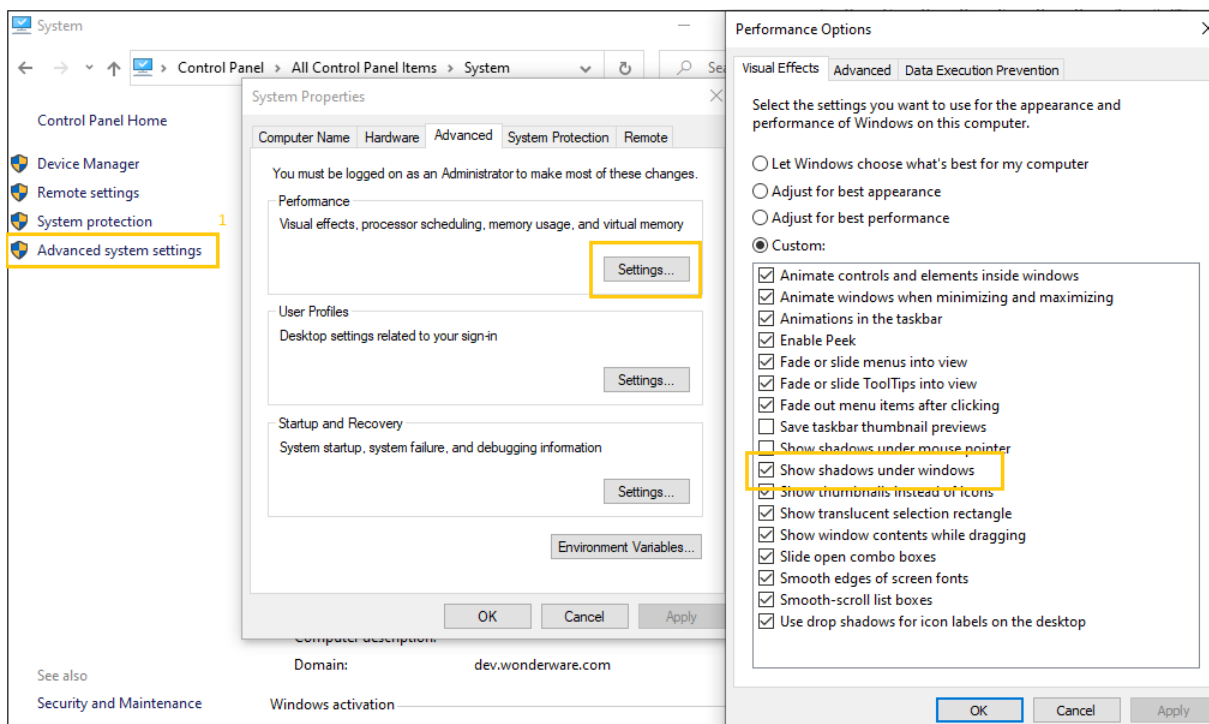
コンテキスト メニューの深さ（ウィンドウの影またはコンテキスト メニューの影）は、画面上で各フローティング ウィンドウの境界を作成します。これは、ウィンドウと重なる他のウィンドウから区別するために役立ちます。

InTouch アプリケーションのコンテキスト メニューの深さは 2 つの方法で設定できます。

- コントロール パネルでのシステム プロパティの変更
- レジストリ エディタでのレジストリ エントリの変更

コントロールパネルを使用してコンテキストメニューの深さを設定するには

1. コントロールパネルを開きます。
2. [システム]、[システムの詳細設定] をクリックします。
[システムのプロパティ] ダイアログボックスが表示されます。
3. [詳細設定] タブの [パフォーマンス] セクションで [設定] をクリックします。
[パフォーマンス オプション] ダイアログボックスが表示されます。
4. [ウィンドウの下に影を表示する] チェックボックスをオンにします。



レジストリ エディタを使用してコンテキストメニューの深さを設定するには

1. レジストリ エディタを開きます。
2. HKEY_CURRENT_USER\Software\Microsoft\Windows\DWM に移動します。
3. ColorPrevalence" -Type DWORD の値を 1 に設定します。
4. WindowMaker を再起動します。

章 8 グラフィック要素

このセクションには、以下の内容が含まれます。

[WindowMaker オブジェクト](#)

[産業用グラフィック エディタの操作](#)

[WindowMaker での産業用グラフィックの使用](#)

[InTouch ウィンドウから産業用グラフィックへの変換](#)

[シンボル ウィザード エディタでのシンボル ウィザードの作成](#)

[トレンド ペンの履歴データ検索について](#)

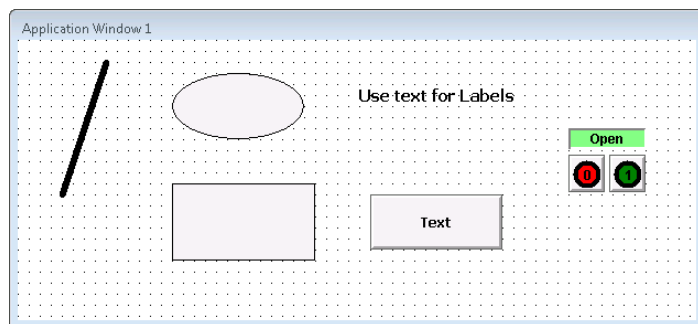
[ランタイム時のトレンド ペンプロパティの変更](#)

[マルチペントレンドの設定](#)

WindowMaker オブジェクト

グラフィカル オブジェクトは、作成するヒューマン マシン インターフェイス（HMI）アプリケーションの重要な部分です。

アプリケーションを作成する場合、シンプル オブジェクトを作成し、シンプル オブジェクトを組み合わせ、複合オブジェクトを作成して、定義済みの複合オブジェクトを使用します。



複合オブジェクトの個々の要素は、通常は次の目的のためにグループ化されています。

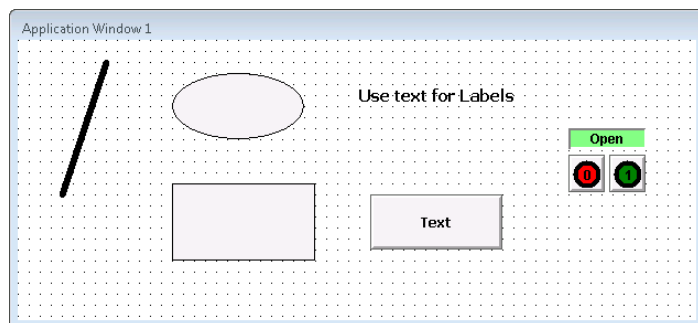
- 編集集中に複合オブジェクトが分離しないようにするため。
- オブジェクト全体で複製するため。
- 一般的な一連のプロパティを個々の要素に割り当てるため。

産業用グラフィック エディタで作成した産業用グラフィックは、マネージドまたは Advanced InTouch アプリケーションで使用できます。WindowMaker の産業用グラフィック ツールボックスから産業用グラフィックを直接追加することもできます。産業用シンボルと Situational Awareness Library シンボルの操作に関する詳細については、『産業用グラフィック エディタ ユーザー ガイド』または WindowMaker ヘルプを参照してください。

WindowMaker オブジェクト

グラフィカル オブジェクトは、作成するヒューマン マシン インターフェイス（HMI）アプリケーションの重要な部分です。

アプリケーションを作成する場合、シンプル オブジェクトを作成し、シンプル オブジェクトを組み合わせ、複合オブジェクトを作成して、定義済みの複合オブジェクトを使用します。



複合オブジェクトの個々の要素は、通常は次の目的のためにグループ化されています。

- 編集集中に複合オブジェクトが分離しないようにするため。
- オブジェクト全体で複製するため。
- 一般的な一連のプロパティを個々の要素に割り当てるため。

産業用グラフィック エディタで作成した産業用グラフィックは、マネージドまたは **Advanced InTouch** アプリケーションで使用できます。**WindowMaker** の産業用グラフィック ツールボックスから産業用グラフィックを直接追加することもできます。産業用シンボルと **Situational Awareness Library** シンボルの操作に関する詳細については、『**産業用グラフィック エディタ ユーザーズ ガイド**』または **WindowMaker** ヘルプを参照してください。

シンプル オブジェクト

以下のタイプのシンプル オブジェクトを作成できます。






- 線
- 図形
- テキスト
- ボタン

各シンプル オブジェクトは表示方法を制御する属性を持っています。

- 線色および太さ
- 塗りつぶし色
- 高さ
- 幅
- 角度

線と図形の作成

以下の表には、基本描画タスクを実行する方法が示されています。描画ボタンは、**[描画]** メニューにあります。

描画内容	クリックするボタン	ボタン
線	線ボタン	
水平または垂直線	H/V 線ボタン	
長方形	長方形ボタン	
角の丸い長方形	角丸長方形ボタン	
注記: 角丸長方形の角の丸みを調整するには、「 角丸長方形の角丸みの変更 」を参照してください。		
円または楕円形	楕円形ボタン。円を描画するには Shift キーを押しながら描画します。	

ボタンの作成

ボタンを使用して、アプリケーションと対話するポイントを作成できます。このプロセスはシンプル描画オブジェクトを作成する過程に非常に類似しています。

ポリゴンの作成については、「[折れ線と多角形の作成](#)」を参照してください。

ボタンを作成するには



1. [描画] メニューの [挿入] グループで [ボタン] をクリックします。
2. クリックしてドラッグし、ボタンを配置およびサイズ設定します。
3. デフォルトのボタンテキストを編集します。以下の手順を実行します。
 - a. ボタンを右クリックして、[文字列の変更] をクリックします。
 - b. [新しい文字列] ボックスに、ボタンのテキストを入力します。
 - c. [OK] をクリックします。

折れ線と多角形の作成

折れ線の描画は、線の描画とは少し異なります。

折れ線または多角形を作成するには

1. [描画] メニューの [図形] グループで [折れ線] または [多角形] をクリックします。
2. アプリケーション ウィンドウをクリックして、最初のポイントを設定します。
3. 再びアプリケーション ウィンドウをクリックして、折れ線または多角形を定義するポイントをさらに設定します。
4. 最後のポイントをダブルクリックします。

テキストの作成

テキストを使用して、アプリケーションのビジュアルアイテムにラベルを付けることができます。

テキストを作成するとき、テキストの書式設定は **WindowMaker** の設定画面で設定されている内容と一致します。選択したテキストの外観を変更できます。詳細については、「[テキスト外観の変更](#)」を参照してください。

複数行のテキストを入力すると、個別に移動および編集できるオブジェクトとなります。テキストオブジェクトをシンボルに組み込んで、グループとして編集することもできます。

テキストを作成するには

1. **【描画】** メニューの **【挿入】** グループで **【テキスト】** をクリックします。
2. テキストの開始場所をクリックします。
3. テキストを入力して、**Enter** キーを押します。新しいテキスト行が表示されます。

複合オブジェクト

複合オブジェクトは、シンプルオブジェクトよりも多くの機能を提供します。以下の表には、複合オブジェクトのタイプが示されています。

複合オブジェクト	説明
セル	シンボルまたはその他のセルを含む 2 つまたはそれ以上のオブジェクトのグループが結合され、単一のユニットを形成しています。セルを使用すると、スライドコントローラなどの仮想デバイスを作成できます。セルは、異なるタグに関連付けられる複数のデバイスを作成する場合に役立ちます。
シンボル	線、図形、およびテキストなどのシンプルオブジェクトのグループが結合されたもので、単一オブジェクトとして扱われます。シンボルに適用された属性の変更は、シンボルのすべてのコンポーネントオブジェクトに影響を与えます。シンボルには、ビットマップ、ボタン、セル、ウィザード、およびトレンドを含めることはできません。
SmartSymbol	再使用可能なグラフィックテンプレートに変換された InTouch セルです。1 つまたは複数の SmartSymbol テンプレートのインスタンスをアプリケーションウィンドウに配置できます。テンプレートへの変更はすべてインスタンスに反映されます。詳細については、「 SmartSymbol について 」を参照してください。
産業用グラフィック	Integrated Development Environment (IDE) で産業用グラフィックエディタを使用して作成される高多様性グラフィックです。
ビットマップ コンテナ	写真、描画、およびスクリーンショットなどの画像のインポートを可能にするオブジェクトです。ビットマップを回転させたり、透過背景を付けたりすることができます。詳細については、「 ビットマップ コンテナの操作 」を参照してください。

複合オブジェクト	説明
トレンドオブジェクト	時間経過による複数タグのリアルタイムまたは履歴データ値の変化を示すチャートです。詳細については、「 トレンドオブジェクト 」を参照してください。
ウィザード	選択、配置、そして設定するだけで使用できる事前に作成されたオブジェクトです。 詳細については、「 ウィザード 」を参照してください。
ActiveX コントロール	アプリケーション内で実行するソフトウェア コンポーネントです。WindowMaker は、AVEVA とサードパーティ製 ActiveX コントロールの両方をサポートしています。詳細については、「 ActiveX コントロールの使用 」を参照してください。

セルとシンボル

複数のオブジェクトを組み合わせ、セルとシンボルと呼ばれる 2 種類の単体を作成できます。セルには任意のオブジェクトを含めることができます。シンボルにはシンプル オブジェクトのみを含めることができます。シンボルにセルを含めることはできません。

特定のオブジェクトがセルまたはシンボルであるかどうかを確認するには、そのオブジェクトをダブルクリックします。

- セルが開き、[タグ変数の変更] ダイアログ ボックス、またはセルにタグ変数が含まれていない場合は [タグ変数の変更] 警告メッセージが表示されます。
- シンボルまたはシンプル グラフィック オブジェクトによって、[アニメーションリンクの選択] ダイアログ ボックスを開きます。

セルについて

セルを使用して、複数要素間の固定位置関係を組み合わせ、維持します。また、セルを使用して複数要素を移動したり、他のグラフィカル要素と整列させたりすることができます。

セルの要素を変更するには、セルを分解し、要素を変更して、要素を再びセルに組み合わせる必要があります。

セルの要素をアニメーション化することはできますが、セルをアニメーション化することはできません。セルのサイズを変更することもできません。

シンボルについて

シンボルおよびシンプル オブジェクトはアニメーション化できます。シンボルを使用して、複合グラフィックのパーツをアニメーション化することもできます。

選択したオブジェクトの 1 つ以上にリンクが設定されている場合は、シンボルを作成できません。

2 つのシンボルを組み合わせ、新しいシンボルを作成すると、元のシンボル構造は失われます。新しいシンボルを分解すると、元の各シンボルの構成要素に分解されます。元の 2 つのシンボルは失われます。

オブジェクトのセルへのグループ化

シンボル、ビットマップ、トレンド、ボタン、ウィザード、およびその他のセルを 1 つのセルに組み合わせることができます。セルにシンボルを含める場合、そのシンボルに関連付けられているすべてのアニメーションリンクはそのままの状態に残ります。

SmartSymbol でセルを使用した後で、SmartSymbol を分解した場合、セルのサイズを変更することはできません。

セルを作成するには

1. 含めるオブジェクトを選択します。
2. [アニメーション] メニューの [セル] グループで [セルの作成] をクリックします。

セルを分解するには

1. セルを選択します。
2. [アニメーション] メニューの [セル] グループで [分解] をクリックします。

オブジェクトのシンボルへのグループ化

シンボルにビットマップ、ボタン、セル、ウィザード、またはトレンドを含めることはできません。選択したオブジェクトの 1 つにアニメーションリンクが接続されている場合、リンクは新しいシンボルに接続されます。

シンボルを作成するには

1. 含めるオブジェクトを選択します。
2. [アニメーション] メニューの [グラフィック] グループで [グラフィックの作成] をクリックします。

シンボルを分解するには

1. シンボルを選択します。
2. [アニメーション] メニューの [グラフィック] グループで [分解] をクリックします。

共通の操作

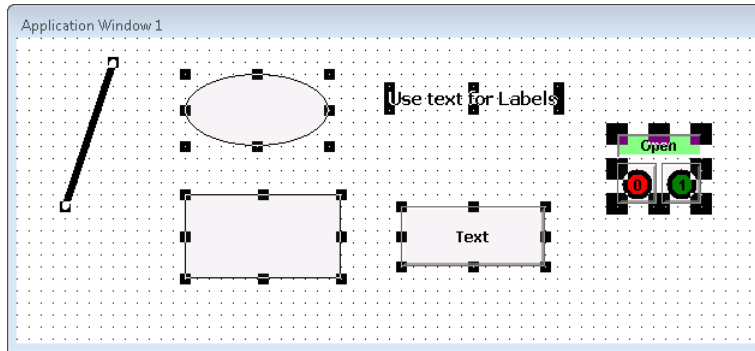
オブジェクトを右クリックすると、オブジェクトに適用できる有効なコマンドまたはアクションを表示するメニューが表示されます。以下の操作を実行できます。

- オブジェクトを選択する
- オブジェクトを移動する
- オブジェクトを整列する
- オブジェクトを配置する
- オブジェクトを重ねる
- 変更を元に戻す
- オブジェクトを反転させる
- シンボルを反転させる
- オブジェクトのサイズを変更する
- オブジェクトを回転させる
- フォントを変更する
- 線または輪郭線を変更する
- 塗りつぶしを変更する
- オブジェクトを削除する

- 左右および上下の間隔を制御する

オブジェクトの選択

オブジェクトを変更する前に、そのオブジェクトを選択する必要があります。オブジェクトを選択すると、オブジェクトの周囲にハンドルが表示されます。これらのハンドルを使用して、オブジェクトのサイズ変更や変形を行います。



アクティブ ウィンドウのオブジェクトをすべて選択するには

- [アニメーション] メニューの [オブジェクト] グループで [すべて選択] をクリックします。
または、**F2** キーをクリックします。

オブジェクトを選択するには

1. [アニメーション] メニューの [モード] グループで [選択] をクリックします。
選択モードが有効になります。

2. 選択するオブジェクトをクリックします。

オブジェクトの選択を解除するには

- ウィンドウの空白領域をクリックします。

複数のオブジェクトを選択するには

1. [アニメーション] メニューの [モード] グループで [選択] をクリックします。
選択モードが有効になります。
2. 最初のオブジェクトを選択し、Shift キーを押しながら他のオブジェクトをクリックします。

オブジェクト グループを選択するには

1. [アニメーション] メニューの [モード] グループで [選択] をクリックします。
選択モードが有効になります。
2. オブジェクトの周囲にボックスをドラッグします。長方形内のすべてのオブジェクトが選択されます。

特定のオブジェクトをオブジェクト グループから選択解除するには

- Shift キーを押しながらオブジェクトをクリックします。

オブジェクトの移動

オブジェクトは以下の方法で移動できます。

- ドラッグする
- キーボードの矢印キーを使用する
- ステータス バー内のボックスにウィンドウ座標を入力する

オブジェクトを移動するときには、ステータス バーの座標がどのように変化するかを確認してください。

オブジェクトをドラッグして移動するには

- オブジェクトを選択して、ドラッグします。

矢印キーを使用してオブジェクトを移動する場合、オブジェクトの移動距離はグリッドが表示されているかどうかによって異なります。

グリッドが表示されている場合、オブジェクトの移動ピクセル数は **WindowMaker** の設定画面で設定したグリッド間隔に依存します。デフォルト設定のグリッド間隔は **10** ピクセルです。

グリッドが表示されている場合:

- 矢印キーを押すと、オブジェクトが **1** グリッド ポイント移動します。
- **Shift** キーと矢印キーを押すと、オブジェクトが **2** グリッド ポイント移動します。
- **Ctrl** キーと矢印キーを押すと、オブジェクトが **4** グリッド ポイント移動します。

グリッドが表示されていない場合:

- 矢印キーを押すと、オブジェクトが **1** ピクセル移動します。
- **Shift** キーと矢印キーを押すと、オブジェクトが **10** ピクセル移動します。
- **Ctrl** キーと矢印キーを押すと、オブジェクトが **50** ピクセル移動します。



矢印キーを使用してオブジェクトを移動するには






- オブジェクトを選択して
 - 矢印キーを押します。
 - **Shift** キーを押しながら矢印キーを押します。
 - **Ctrl** キーを押しながら矢印キーを押します。

オブジェクトの整列

オブジェクトを左端、右端、中心線、中心点、上端、中央、または下端に揃えることができます。

メニュー コマンドまたはボタンを使用して、複数の方法で整列させることができます。

選択	クリック	目的
左寄せ		オブジェクトの左端をオブジェクトの一番左端に揃えます。
中心線揃え		オブジェクトをグループの垂直中心線に揃えます。

選択	クリック	目的
右寄せ		オブジェクトの右端をオブジェクトの一番右端に揃えます。
上端揃え		上端を一番高いオブジェクトの上端と揃えます。
中央揃え		グループの中央に水平中心に揃えます。
下端揃え		下端を一番低いオブジェクトの下端と揃えます。
中心点揃え		中心点をグループの中心点に揃えます。

オブジェクトを整列させるには

1. 複数のオブジェクトを選択します。
2. [ホーム] メニューの [整列] グループで適切な整列コマンドをクリックします。

オブジェクトを重ねる

オブジェクトは他のオブジェクトの前面または背面に配置できます。

オブジェクトを別のオブジェクトの背面に移動するには

1. オブジェクトを選択します。
2. [ホーム] メニューの [配置] グループで [最背面へ移動] をクリックします。

オブジェクトを別のオブジェクトの前面に移動するには

1. オブジェクトを選択します。
2. [ホーム] メニューの [配置] グループで [最前面へ移動] をクリックします。
または、SHIFT+F9 キーを押します。

オブジェクトの間隔の制御

左右両端の選択オブジェクトを均等間隔に水平に配置することができます。

また、上下両端の選択オブジェクトの間隔も制御できます。

オブジェクトの間隔を指定して、中心に揃えることもできます。

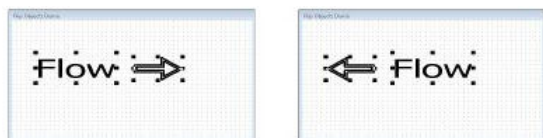
オブジェクトを左右または上下に均等間隔で配置するには

1. オブジェクトを選択します。
2. [描画] メニューの [間隔] グループで [水平、垂直] または [中心点] をクリックします。

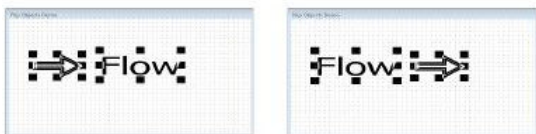
オブジェクトとセルの反転

ほとんどのオブジェクトは左右または上下に反転できます。オブジェクトは単一またはグループで反転できます。

オブジェクトを反転すると、ミラー画像として変換され、表示されます。テキストは反転できません。



セルを反転しても、ミラー イメージとしては表示されません。オブジェクト グループ内のセルの位置だけがミラー イメージとして表示されます。



反転前後のセルの位置を比較します。位置は反転していますが、内容はそのままです。

オブジェクトまたはセルを反転するには

1. オブジェクトを選択します。



2. [描画] メニューの [配置] グループで、[回転] の横にある下向きの矢印をクリックします。
3. [左右反転] または [上下反転] をクリックします。

オブジェクトのサイズ変更

オブジェクトは2つの方法でサイズ変更できます。ドラッグするか、正確な幅と高さを指定します。

グリッドに配置がオンになっている場合、オブジェクトは比例サイズ変更操作中にグリッドに配置されます。垂直から水平サイズ間の比率にわずかな偏差が生じます。この偏差を避けるには、グリッドに配置をオフにします。

オブジェクトをドラッグしてサイズ変更するには

1. オブジェクトを選択して、矢印形のカーソルの先をハンドルの真ん中に合わせます。
2. ハンドルをドラッグして、オブジェクトのサイズを変更します。

オブジェクトを比例サイズ変更するには

- オブジェクトを選択して、Shift キーを押しながらドラッグします。

オブジェクトを寸法でサイズ変更するには

1. オブジェクトを選択します。
2. [プロパティ] パネルに幅と高さの寸法を [W, H] ボックスにそれぞれ入力します。

オブジェクトの回転

シンボル、テキスト、およびビットマップを含むほとんどのオブジェクトを回転できます。セルは回転できません。

オブジェクトは時計回りまたは反時計回りに 90 度ずつ 360 度回転させることができます。

WindowViewer でのオブジェクトの回転は、ランタイム時にまたは WindowViewer でのオブジェクトの動的な回転とは関係ありません。WindowViewer でオブジェクトを回転させるには、オブジェクトを角度アニメーションにリンクします。

オブジェクトを回転するには

1. オブジェクトを選択します。



2. [描画] メニューの [配置] グループで、[回転] の横にある下向きの矢印をクリックします。
3. [時計回り] または [反時計回り] をクリックします。

テキスト外観の変更

フォントの外観は、事前にデフォルトを設定することによって、テキストを作成する前に設定できます。また、テキストを作成した後で変更することもできます。デフォルトのフォントを設定する詳細については、「[フォントのデフォルト設定](#)」を参照してください。

テキストの揃え方の属性設定は、動的な値を示すテキストオブジェクトに対して重要です。ランタイム時に長さの変化するフィールドがどのように表示されるかは、揃え方によって決まります。

たとえば、中央揃えか右揃えのテキスト文字列の最後に数値を表示している場合、表示される桁数が変更されるたびに、値を含む文字列全体が中央または右に揃え直されます。

メニュー コマンドまたはボタンを使用して、複数の方法でテキストを設定できます。

目的	クリック	ボタン
フォント、スタイル、色、またはテキストサイズを変更する	フォント	
テキストを太字にする	太字	
テキストを斜体にする	斜体	
テキストに下線を付ける	下線	
フォント サイズの縮小または拡大	フォントの縮小またはフォントの拡大	
揃え方を変更する	左揃え、中央揃え、または右揃え	

テキストの外観を設定するには

1. テキストオブジェクトを選択します。
2. [描画] メニューの [形式] グループで適切なテキスト コマンドをクリックします。

線と輪郭線の変更

輪郭線付きオブジェクトの線または輪郭線の色、およびパターンまたは幅のいずれかを変更できます。輪郭線付きオブジェクトには、楕円形、長方形、および多角形などの塗りつぶし図形とビットマップおよびその他のインポートされた画像が含まれます。

線が太くなるほど、ランタイムでの描写にかかる時間が長くなります。破線および点線の太さは 1 ピクセル幅に固定されています。

線の外観のデフォルトを設定するには

1. ウィンドウの空白領域をクリックします。
2. [描画] メニューの [形式] グループで線の幅またはパターンを選択します。
3. [線の色] ツールをクリックして、色を選択します。

線の色を変更するには

1. 線、線グループ、または輪郭線付きのオブジェクトを選択します。
2. [描画] メニューの [形式] グループで [線の色] ツールをクリックします。
3. 色を選択します。

線または輪郭線のスタイルまたは幅を変更するには

1. オブジェクトを選択します。
2. [描画] メニューの [形式] グループで線のスタイルまたは幅をクリックします。

輪郭線を削除するには

1. オブジェクトを選択します。
2. [描画] メニューの [形式] グループで [線なし] をクリックします。

塗りつぶしの変更

塗りつぶし図形には線で囲まれた図形が含まれます。塗りつぶし図形の例には、長方形、角丸長方形、円、楕円形、および多角形があります。

オブジェクトの塗りつぶし色を変更するには

1. オブジェクトを選択します。
2. [描画] メニューの [形式] グループで [塗りつぶしの色] ツールをクリックします。
3. 色を選択します。

塗りつぶし図形のデフォルト色を設定するには

1. ウィンドウの空白領域をクリックします。
2. [描画] メニューの [形式] グループで [塗りつぶしの色] ツールをクリックします。
3. 色を選択します。

オブジェクトの削除

1つまたは複数のオブジェクトを削除できます。

オブジェクトを削除するには

- 以下のいずれかを実行します。
 - オブジェクトを右クリックし、[削除] をクリックします。
 - オブジェクトを選択して **Delete** キーを押します。

変更を元に戻す

WindowMaker は、各ウィンドウの編集および形式の変更を記録します。デフォルトでは、WindowMaker は元に戻す/やり直しの 10 レベルをサポートしています。1 レベルとは 1 回のアクションを表します。

WindowMaker は最大で 25 レベルのアクションを保持するように設定できます。レベルを 0 に設定して、元に戻す/やり直しをオフにすることもできます。

ウィンドウを閉じると、記録されたすべてのアクションはすべて消去されます。

コマンドを元に戻すには

- クイック アクセス ツールバーの [元に戻す] をクリックします。

コマンドをやり直しするには

- クイック アクセス ツールバーの [やり直す] をクリックします。

元に戻す/やり直しレベルの数を設定するには

1. [ファイル] メニューの [設定] をクリックし、[WindowMaker] をクリックします。

WindowMaker の設定画面が表示されます。

2. [元に戻すメニューのレベル] ボックスに、レベル数を入力します。

すべてのオブジェクトの特殊操作

シンプル オブジェクトは変更したり、操作したりできます。以下の操作を実行できます。

- オブジェクトの切り取り、コピー、貼り付け
- オブジェクトリンクの切り取り、コピー、貼り付け
- オブジェクトの複製

オブジェクトの切り取り、コピー、貼り付け

WindowMaker での切り取り、コピー、および貼り付け操作は、他の Windows ベースのアプリケーションと同じですが、いくつか注意すべき相違点があります。

オブジェクトを切り取り、コピー、または貼り付けする際に、属性およびアニメーション リンクも切り取り、コピー、または貼り付けされます。

貼り付けられたオブジェクトはすべて、貼り付け後も選択されたままになるため、動かして位置を調整することができます。

オブジェクトを切り取るには

- オブジェクトを右クリックし、[切り取り] をクリックします。

オブジェクトをコピーするには

- オブジェクトを右クリックし、[コピー] をクリックします。

オブジェクトを貼り付けるには

1. ウィンドウの空白領域を右クリックし、[貼り付け] をクリックします。カーソルがコーナー シンボルに変わります。
2. 左マウス ボタンを押します。カーソルがコピーしたオブジェクトと同じ大きさの点線の長方形に変わります。
3. 長方形をドラッグして、オブジェクトを配置します。
4. マウス ボタンを離します。

オブジェクトリンクの切り取り、コピー、貼り付け

クリップボードは、切り取りまたはコピーしたリンクの一時保存領域です。

- クリップボードには、最後に行われた切り取りやコピー操作のリンクのみが保存されます。
- リンクは、クリップボードのリンクをサポートする任意のオブジェクトまたはシンボルに貼り付けることができます。
- たとえば、テキスト オブジェクトの線／輪郭色リンクなど、貼り付けられたリンクがオブジェクトでサポートされていない場合、リンクは貼り付けられません。
- 貼り付け用に複数オブジェクトを選択すると、すべてのオブジェクトにリンクが貼り付けられます。

リンクを切り取り、コピー、貼り付け、クリアするには

- オブジェクトを右クリックし、[リンク] をポイントして、適切なコマンドをクリックします。

オブジェクトの複製

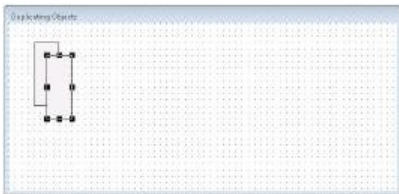
オブジェクトの複製はオブジェクトおよびそのアニメーションリンクのコピーと類似していますが、オブジェクトが2回以上複製されるとオフセット距離および方向も複製されるという利点があります。

複製オブジェクトを選択解除しないで移動し、再び複製すると、3番目の複製は最初の2つの複製との間で同じ距離と方向のオフセットとなります。

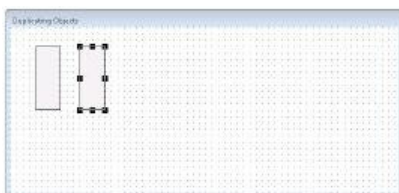
この操作は必要なだけ繰り返すことができます。

オブジェクトを複製するには

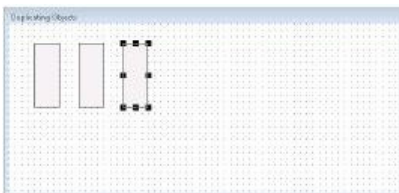
1. オブジェクトを右クリックし、[複製] をクリックします。オブジェクトは元のオブジェクトからのオフセット位置にコピーされ、貼り付けられます。



2. 複製オブジェクトを選択したままにして、別の位置にドラッグします。



3. 複製オブジェクトを選択解除しないで、オブジェクトを右クリックして、[複製] をクリックします。3回目のオブジェクト複製は最初の2つと同じ相対位置に表示されます。



特殊オブジェクトの特殊操作

以下のオブジェクト タイプは、編集できる固有の属性を備えています。

- 折れ線と多角形
- ビットマップ コンテナ
- ビットマップ透明色
- 角丸長方形
- オブジェクト テキスト

折れ線および多角形オブジェクトの形状変更

折れ線および多角形の形状は調整できます。

折れ線または多角形の形状を変更するには

1. オブジェクトを選択します。
2. 以下のいずれかを実行します。
 - [アニメーション] メニューの [オブジェクト] グループで [再形成] をクリックして、すべての再形成オプションを表示します。
 - オブジェクトを右クリックし、[オブジェクトの変形] をクリックします。それぞれの図形定義点がハンドルとなります。
3. ハンドルをドラッグして形状を変更します。

多角形に頂点を追加するには

1. オブジェクトを選択します。
2. 以下のいずれかを実行します。
 - [アニメーション] メニューの [オブジェクト] グループで矢印をクリックして非表示のコマンドを表示し、[ポイントを追加] をクリックします。
 - オブジェクトを右クリックし、[頂点の追加] をクリックします。
3. 多角形の端をクリックし、頂点をドラッグして多角形の形状を変更します。

多角形から頂点を削除するには

1. オブジェクトを選択します。
2. 以下のいずれかを実行します。
 - [アニメーション] メニューの [オブジェクト] グループで矢印をクリックして非表示のコマンドを表示し、[ポイントを削除] をクリックします。
 - オブジェクトを右クリックし、[ポイントを削除] をクリックします。
3. 多角形の頂点をクリックすると、頂点が削除され、多角形の形状が変更されます。

ビットマップ コンテナの操作

ビットマップ コンテナは、ピクチャ、画面キャプチャ、および描画などのグラフィック オブジェクトのアプリケーションへのインポートを可能にするオブジェクトです。

サポートされるビットマップ コンテナ ファイル タイプは、.bmp、.jpeg、.jpg、.pcx および .tga です。

ビットマップをインポートすると、ビットマップ コンテナに自動的に挿入されますが、元のサイズおよび比率にサイズを変更できます。

ビットマップは 90 度ずつ回転させることができます。

ビットマップをセルに含めることはできますが、シンボルに含めることはできません。

WindowMaker を使用して、WindowViewer にロード可能なビットマップよりも多くのビットマップをウィンドウに配置することができます。多数のビットマップをウィンドウに配置する必要がある場合は、アプリケーションをリリースする前に WindowViewer でウィンドウをテストしてください。

ビットマップ画像をインポートするには

1. **〔描画〕** メニューの **〔図形〕** グループで **〔ビットマップ〕** をクリックします。
カーソルが十字型に変わります。
2. カーソルをドラッグして、ビットマップ コンテナを描画します。
3. **〔アニメーション〕** メニューの **〔ビットマップ〕** グループで **〔インポート〕** を選択します。
〔画像ファイルの選択〕 ダイアログ ボックスが表示されます。
4. 画像のファイル名を選択し、**〔OK〕** をクリックします。

ビットマップを元のサイズにするには

1. 画像を選択します。
2. **〔アニメーション〕** メニューの **〔ビットマップ〕** グループで **〔元のサイズ〕** を選択します。

ビットマップ画像を貼り付けるには

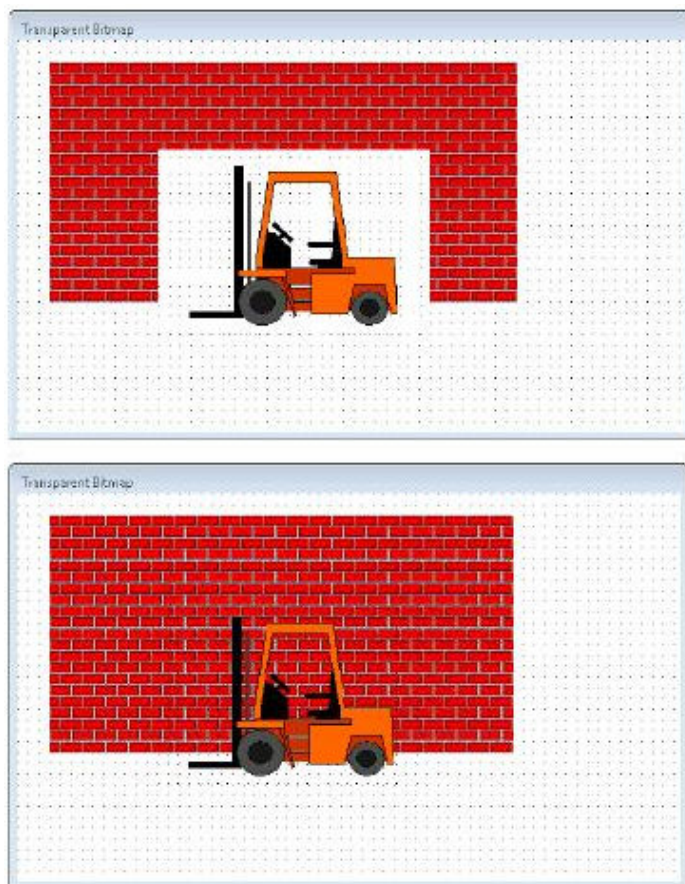
1. グラフィックを Windows のクリップボードにコピーします。
2. **〔ビットマップ〕** ツールをクリックして、ビットマップ コンテナをウィンドウに描画します。
3. **〔アニメーション〕** メニューの **〔ビットマップ〕** グループで **〔貼り付け〕** を選択します。
または、ビットマップ コンテナを右クリックして、**〔貼り付け〕** をクリックします。

ビットマップを編集するには

1. ビットマップを選択します。
2. **〔アニメーション〕** タブの **〔ビットマップ〕** グループで **〔編集〕** をクリックします。
Microsoft ペイントが開き、ビットマップが表示されます。
3. ペイントでビットマップを編集します。
4. 保存して、ペイントを閉じます。

ビットマップ透明色の定義

ビットマップに透明色を定義すると、ウィンドウの背景色やビットマップの背面にあるオブジェクトは、透明色を使用した部分すべてを通して表示されます。各画像には、1 つの透明色だけを定義できます。



透明ビットマップを作成するには

1. ビットマップが選択された状態で、[描画] メニューの [形式] グループにある [透明色] ボタンをクリックして、透明色パレットを開きます。
2. カスタムパレットの色付き四角形を右クリックします。[カスタムカラーの編集] ダイアログボックスが表示されます。
3. スポイトツールをクリックします。
4. 透明にするビットマップ内の色をクリックします。選択した色がカラーパレットの選択した色の四角にコピーされます。
5. 色の四角をクリックして、透明色をビットマップに適用します。画像内のその色のすべてのピクセルが透明になります。

角丸長方形の角丸みの変更

角丸長方形の角丸みは拡大または縮小できます。

オブジェクトの角丸みを拡大または縮小するには

1. オブジェクトを選択します。
2. [アニメーション] メニューの [オブジェクト] グループで非表示のコマンドを表示し、[半径を拡大] または [半径を縮小] をクリックします。

オブジェクトテキストの置換

シンボル、セル、またはボタンなどのテキストを持つオブジェクトのテキストを編集できます。

テキスト文字列を変更しても、元のすべての属性、フォント、スタイル、色などはすべて保持されます。テキスト書式設定も数値に適用できます。

オブジェクトのテキストを変更するには

1. テキストを含むオブジェクトまたはボタンを選択します。以下のいずれかを実行します。
 - [アニメーション] メニューの [置換] グループで [文字列] をクリックします。
 - テキストオブジェクトを右クリックして、[置換] をポイントし、[文字列の変更] をクリックします。
2. [新しい文字列] ボックスで、新しい文字列を入力して、[OK] をクリックします。

一連のテキストオブジェクトの一部のテキストを変更するには

1. テキストオブジェクトをすべて選択します。
2. [アニメーション] メニューの [置換] グループで [文字列] をクリックします。
3. [置換] をクリックします。
[文字列の置換] ボックスが表示されます。
4. [旧文字列] ボックスに、置換するテキストの一部を入力します。
5. [新文字列] ボックスに、新しいテキスト文字列を入力します。
6. [OK] をクリックします。

選択したすべてのオブジェクトで新しいテキスト文字列が旧テキスト文字列に置き換わります。

WindowMaker オブジェクトに関するウィンドウ情報の印刷

アプリケーションウィンドウに配置した InTouch グラフィカルオブジェクトに関する情報を印刷できます。InTouch グラフィカルオブジェクトには、線、ボタン、およびテキストなどのシンプルなオブジェクトや、セル、ActiveX コントロール、SmartSymbol、および産業用グラフィックなどの複雑なオブジェクトが含まれています。

グラフィカルオブジェクトの詳細は、プリンタまたは .html ファイルに出力できます。各ウィンドウに対して、.html ファイルが作成されます。.html ファイルには、以下のものが含まれています。

- 参照されている .png ファイルとしてのウィンドウのピクチャ。
- ウィンドウ内の各グラフィカルオブジェクトに関する詳細の一覧。

複数のウィンドウについてのグラフィカルオブジェクトの詳細を印刷した場合、ウィンドウ固有の .html ファイルにリンクしているサマリ .html ファイルが作成されます。

WindowMaker オブジェクトに関する情報を印刷するには

1. WindowMaker で InTouch アプリケーションを開きます。
2. クイックアクセスツールバーの [印刷] をクリックします。
[アプリケーション情報の印刷] ダイアログボックスが表示されます。
3. [ウィンドウ] チェックボックスをオンにします。

4. 印刷するウィンドウを指定します。

- **[全て]** を選択すると、アプリケーションにあるすべてのウィンドウのイメージ/グラフィックが印刷されます。

注記: ウィンドウに多くのグラフィックが含まれる場合、または **WindowMaker** をホストするコンピュータで使用可能なメモリを超えるサイズの大きいグラフィックが含まれる場合、アプリケーションのすべてのウィンドウを **XPS** ファイルに出力すると **WindowMaker** が応答しなくなります。そのような場合は、メモリ不足による問題を回避するために一度に **1** つのウィンドウを出力してください。

- **[選択]** を選ぶと、特定のウィンドウのイメージ/グラフィックのみが印刷されます。**[ウィンドウの印刷]** ダイアログボックスが開きます。アプリケーションで印刷するウィンドウを選択して **[OK]** をクリックします。
- **[バッチ]** を選択すると、**.csv** ファイルで指定したウィンドウの情報のみが印刷されます。
.csv 形式の詳細については、「[ウィンドウを印刷するための .CSV 形式](#)」を参照してください。

1. **[イメージ/グラフィック]** チェック ボックスをオンにします。

2. **[次へ]** をクリックします。**[出力の送信先の選択]** ダイアログボックスが表示されます。

3. 以下のいずれかを実行します。

- 情報を印刷するには、**[プリンタに出力を送信する]** をクリックします。
- 指定した各ウィンドウに対して、概要の **.html** ファイル、単一の **.html** ファイル、および **.png** ファイルを作成するには、**[HTML ファイルに出力を送信する]** をクリックします。**.html** ファイルおよび **.png** ファイルが存在する場合、自動的に上書きされます。

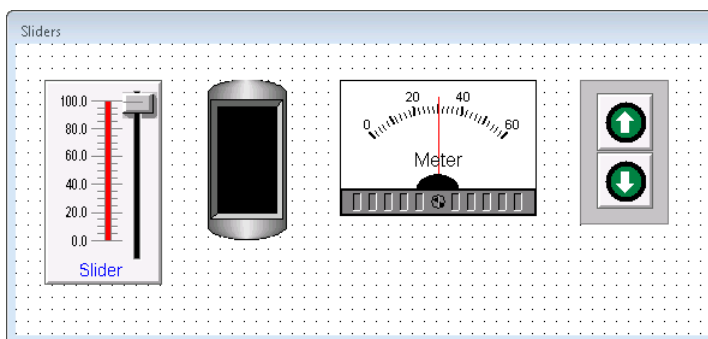
1. **[印刷]** をクリックします。

オブジェクトのアニメーション化

オブジェクトまたはシンボルはアニメーションリンクによってアニメーション化できます。アニメーションリンクはタグまたは式の値をオブジェクトまたはシンボルに結合できます。

たとえば、以下の操作を実行できます。

- タンクの液体レベルを示すスライダまたはタンク シンボルを作成する
- 値の範囲を表示するメーターを作成する
- オペレータ コントロール用のタッチ スクリーン シンボルを作成する



アニメーションリンクの2つのタイプ

アニメーションリンクには、表示リンクおよびタッチリンクの2つの基本タイプがあります。

- 表示リンクはオペレータに情報を表示します。表示リンクの例は、色の変更、塗りつぶしレベルの変更、水平または垂直移動、オブジェクトの点滅などがあります。
- タッチリンクは、システムへのオペレータ入力を許可します。タッチリンクの例には、オペレータ入力に応答するスライダまたは押しボタンなどがあります。

オブジェクトまたはシンボルには、複数のリンクを定義できます。さまざまなリンクを組み合わせることによって、ほとんどの画面アニメーション効果を作成できます。

データ表示アニメーション

データ表示アニメーションは、オペレータに情報のみを表示します。これらのアニメーションはオペレータ入力をサポートしません。

値表示の作成

値表示テキストオブジェクトを使用して、タグ変数の値を表示します。これにより、塗りつぶしレベル、オン/オフステータス、またはアラームメッセージを表示できます。

3つのタイプの値表示リンクのいずれか1つを使用して、ランタイムにメッセージを表示できます。

値表示タイプ	内容
論理型	オンまたはオフなどの論理値
アナログ型	塗りつぶしレベルまたは速度などのアナログ式の値
文字列型	"Fill Level = 100" などの文字列式の値

式には最大 1023 文字までを使用できます。これより長い式が必要な場合、クイック関数を作成して式を呼び出します。

メッセージは、そのオブジェクトに設定されているフォント、サイズ、色、配置、およびリンクされている属性を使用して、元のテキストオブジェクトの位置に表示されます。フィールドの元の内容は、ランタイムのメッセージには影響されません。

論理値表示リンクを作成するには

- テキストオブジェクトを右クリックして、[アニメーションリンク] をクリックします。[アニメーションリンク] ダイアログボックスが表示されます。
- [値の表示] 領域で、[論理値] をクリックします。[論理値表示リンク] ダイアログボックスが表示されます。

- [タグ変数/式] ボックスに、論理型タグ変数か論理値に等しい式を入力します。以下に例を示します。

Cooling_Pump

- [オンメッセージ] ボックスに、式の値が 1、true、on、または yes のときに表示するメッセージを入力します。以下に例を示します。

Pump is ON

5. [オフメッセージ] ボックスに、式の値が 0、false、off、または no のときに表示するメッセージを入力します。例：

Pump is OFF

6. [OK] をクリックします。

アナログ値表示リンクを作成するには

1. テキストオブジェクトを右クリックして、[アニメーションリンク] をクリックします。[アニメーションリンク] ダイアログボックスが表示されます。
2. [値の表示] 領域で、[アナログ] をクリックします。[アナログ値出力] ダイアログボックスが表示されます。

3. [タグ変数/式] ボックスに、アナログ型（整数型または実数型）タグ変数またはアナログ値に等しい式を入力します。以下に例を示します。

Tank_CV*0.06

4. [書式設定] 領域のリストで、ランタイム時の詳細な書式設定を行う、それぞれのデータタイプをクリックします。選択したデータタイプに基づいて、[固定幅] チェックボックス、[精度] ボックス、および [ビット数から] および [まで] ボックスが有効になります。これらのオプションの設定の詳細については、「[テキストの詳細な書式設定](#)」セクションを参照してください。

注記: ランタイム時、アナログ値入力リンク フィールドのサイズは、クリックしてポインタとマウスでドラッグすることによって変更できます。

1. [OK] をクリックします。

文字列表示リンクを作成するには

1. テキストオブジェクトを右クリックして、[アニメーションリンク] をクリックします。[アニメーションリンク] ダイアログボックスが表示されます。
2. [値の表示] 領域で、[文字列] をクリックします。[文字列表示リンク] ダイアログボックスが表示されます。

3. [タグ変数/式] ボックスに、メッセージ型タグ変数またはメッセージ型タグ変数に等しい式を入力します。以下に例を示します。

"The Tank Level is:" + Text(TankLevel, "#")

4. [OK] をクリックします。

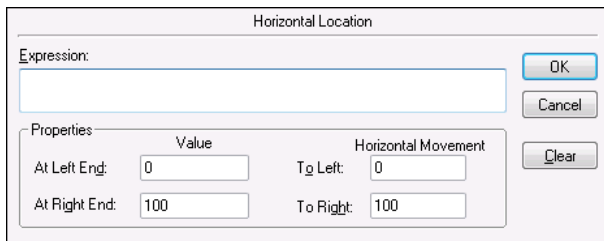
注記: ランタイム時、文字列値入力リンク フィールドのサイズは、クリックしてポインタとマウスでドラッグすることによって変更できます。

移動の作成

ランタイム時のオブジェクトの移動は、位置リンクを使用することによって実行できます。オブジェクトを水平、垂直、またはその両方向に、アナログ型タグの値または式の変化に合わせて移動できます。たとえば、タンク レベルが増減すると、インジケータが上下に移動します。

水平移動を作成するには

1. オブジェクトを画面上の開始位置に配置します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーション リンク] を選択します。
[アニメーション リンク] ダイアログ ボックスが表示されます。
3. [位置] セクションで [水平] をクリックします。
[水平位置変化リンク] ダイアログ ボックスが表示されます。



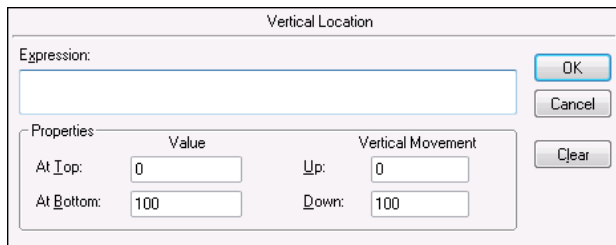
The image shows the 'Horizontal Location' dialog box. It has a title bar 'Horizontal Location'. Inside, there is an 'Expression:' label above a text input field. To the right of the input field are 'OK', 'Cancel', and 'Clear' buttons. Below the input field is a 'Properties' section. It contains two columns: 'Value' and 'Horizontal Movement'. Under 'Value', there are two rows: 'At Left End:' with a value of '0' and 'At Right End:' with a value of '100'. Under 'Horizontal Movement', there are two rows: 'To Left:' with a value of '0' and 'To Right:' with a value of '100'.

4. [式] ボックスに、アナログ型タグの名前、またはアナログ値に等しい式を入力します。
5. [プロパティ] セクションでオブジェクトの移動距離を設定します。以下の手順を実行します。
 - a. [左端の値] ボックスに、オブジェクトが左端に位置する場合のアナログ型タグの値を入力します。
 - b. [右端の値] ボックスに、オブジェクトが右端に位置する場合のアナログ型タグの値を入力します。
 - c. [左への移動幅] ボックスに、オブジェクトを開始位置から左に動かすことができる幅をピクセル数で入力します。
 - d. [右への移動幅] ボックスに、オブジェクトを開始位置から右に動かすことができる幅をピクセル数で入力します。
6. [OK] をクリックします。

垂直移動を作成するには

1. オブジェクトを画面上の開始位置に配置します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーション リンク] を選択します。
[アニメーション リンク] ダイアログ ボックスが表示されます。
3. [位置] セクションで [垂直] をクリックします。

[垂直位置変化リンク] ダイアログ ボックスが表示されます。

The image shows a dialog box titled "Vertical Location". It has a section labeled "Expression:" with a text input field. To the right of this field are three buttons: "OK", "Cancel", and "Clear". Below the "Expression:" section is a section labeled "Properties". Inside "Properties", there are two columns: "Value" and "Vertical Movement". Under "Value", there are two rows: "At Top:" with a value of "0" and "At Bottom:" with a value of "100". Under "Vertical Movement", there are two rows: "Up:" with a value of "0" and "Down:" with a value of "100".

4. [式] ボックスに、アナログ型タグの名前、またはアナログ値に等しい式を入力します。
5. [プロパティ] セクションで以下の手順を実行します。
 - a. [上端の値] ボックスに、オブジェクトが上端に位置する場合のアナログ型タグの値を入力します。
 - b. [下端の値] ボックスに、オブジェクトが下端に位置する場合のアナログ型タグの値を入力します。
 - c. [上への移動幅] ボックスに、オブジェクトを開始位置から上に動かすことができる幅をピクセル数で入力します。
 - d. [下への移動幅] ボックスに、オブジェクトを開始位置から下に動かすことができる幅をピクセル数で入力します。
6. [OK] をクリックします。

回転の作成

向きリンクを使用して、オブジェクトを中心点の周囲で、アナログ型タグの値の変化に合わせて移動するようにできます。たとえば、圧力が増加または低下するにつれてポインタがダイヤルの周囲を移動します。

向きリンクは、オブジェクトやシンボルの中心を回転のデフォルト中心として使用します。回転の中心をオフセットすることができます。

産業用グラフィックでは向きリンクはサポートされていません。

ヒント: オブジェクトの中心から回転中心点まで一時的な長方形を描画します。XおよびYオフセットの寸法をステータスバーの [W, H] ボックスで読み取ることができます。

向きリンクを作成するには

1. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] をクリックします。
[アニメーションリンク] ダイアログ ボックスが表示されます。
2. [その他] セクションで [向き] をクリックします。
[向き -> アナログ値] ダイアログ ボックスが表示されます。

3. **[式]** ボックスに、アナログ型タグの名前、またはアナログ値に等しい式を入力します。
4. **[プロパティ]** セクションで以下の手順を実行します。
 - a. **[左回り最大値]** ボックスに、オブジェクトを反時計回りの最大角度まで回転させる式の値を入力します。
 - b. **[右回り最大値]** ボックスに、オブジェクトを時計回りの最大角度まで回転させる式の値を入力します。
 - c. **[左回り角度]** ボックスに、**[左回り最大値]** に入力された式の値に対して、オブジェクトが反時計回りに回転する角度を入力します。
 - d. **[右回り角度]** ボックスに、**[右回り最大値]** に入力された式の値に対して、オブジェクトが時計回りに回転する角度を入力します。
5. **[回転中心設定 (オブジェクトの中心点からのオフセット値)]** セクションで以下の手順を実行します。
 - a. **[水平方向]** ボックスに、回転中心点の水平方向のオフセットを入力します。オブジェクトの中心点からのオフセットをピクセルで入力します。
 - b. **[垂直方向]** ボックスに、回転中心点の垂直方向のオフセットを入力します。オブジェクトの中心点からのオフセットをピクセルで入力します。
6. **[OK]** をクリックします。

サイズのアニメーション化

オブジェクトサイズリンクを使用して、アナログ型タグや式の値に応じてオブジェクトの高さや幅を変更できます。

たとえば、圧力インジケータは圧力が増加すると大きくなり、コンベヤ上のオブジェクトは大きくなることによってビューアに向かって移動します。

オブジェクトサイズリンクはオブジェクトのサイズを制御するだけでなく、オブジェクトがアニメーションの基準点を使用してサイズを変更する方向を制御します。

オブジェクトサイズ高さリンクを作成するには

1. **[描画]** メニューの **[モード]** グループで **[アニメーション化]** をクリックします。
または、オブジェクトを右クリックして、**[アニメーションリンク]** をクリックします。
[アニメーションリンク] ダイアログ ボックスが表示されます。
2. **[オブジェクトサイズ]** セクションで **[高さ]** をクリックします。
[オブジェクトの高さ -> アナログ値] ダイアログ ボックスが表示されます。

3. **[式]** ボックスに、アナログ型タグの名前、またはアナログ値に等しい式を入力します。
4. **[プロパティ]** セクションで以下の手順を実行します。
 - a. **[最大高さの値]** ボックスに、オブジェクトが最大の高さに達したときのタグまたは式の値を入力します。
 - b. **[最小高さの値]** ボックスに、オブジェクトが最小の高さに達したときのタグまたは式の値を入力します。
 - c. **[最大高さ %]** ボックスに、タグ名または式の値が **[最大高さの値]** ボックスで設定した値に達したときのオブジェクトの高さを元の高さのパーセントで入力します。パーセント値は、オブジェクトの描画サイズの割合で表します。描画サイズは常に **100%** サイズです。
 - d. **[最小高さ %]** ボックスに、タグ名または式の値が **[最小高さの値]** ボックスで設定した値に達したときのオブジェクトの高さを元の高さのパーセントで入力します。パーセント値は、オブジェクトの描画サイズの割合で表します。描画サイズは常に **100%** サイズです。
5. オブジェクトを拡大する基点となる**変化基準点**ポイントを選択します。
 - **[上端]** を選択すると、オブジェクトは上端を基点に下方に拡大されます。
 - **[中心点]** を選択すると、オブジェクトは中心点を基点に上下両方向に拡大されます。
 - **[下端]** を選択すると、オブジェクトは下端を基点に上方に拡大されます。

6. **[OK]** をクリックします。

オブジェクトサイズ幅リンクを作成するには

1. **[描画]** メニューの **[モード]** グループで **[アニメーション化]** をクリックします。
または、オブジェクトを右クリックして、**[アニメーションリンク]** をクリックします。
[アニメーションリンク] ダイアログ ボックスが表示されます。
2. **[オブジェクトサイズ]** セクションで **[幅]** をクリックします。
[オブジェクトの幅 -> アナログ値] ダイアログ ボックスが表示されます。

3. **[式]** ボックスに、アナログ型タグの名前、またはアナログ値に等しい式を入力します。
4. **[プロパティ]** セクションで以下の手順を実行します。
 - a. **[最大幅の値]** ボックスに、オブジェクトが最大の幅に達したときのタグまたは式の値を入力します。
 - b. **[最小幅の値]** ボックスに、オブジェクトが最小の幅に達したときのタグまたは式の値を入力します。
 - c. **[最大幅 %]** ボックスに、タグ名または式の値が **[最大幅の値]** ボックスで設定した値に達したときのオブジェクトの幅を元の幅のパーセントで入力します。パーセント値は、オブジェクトの描画サイズの割合で表します。描画サイズは常に **100%** サイズです。
 - d. **[最小幅 %]** ボックスに、タグ名または式の値が **[最小幅の値]** ボックスで設定した値に達したときのオブジェクトの幅を元の幅のパーセントで入力します。パーセント値は、オブジェクトの描画サイズの割合で表します。描画サイズは常に **100%** サイズです。
5. オブジェクトの幅を拡大する基点となる**変化基準点**ポイントを選択します。
 - **[左端]** を選択すると、オブジェクトは左端を基点に拡大されます。
 - **[中心点]** を選択すると、オブジェクトは中心点を基点に上下両方向に拡大されます。
 - **[右端]** を選択すると、オブジェクトは右端を基点に拡大されます。
6. **[OK]** をクリックします。

色のアニメーション化

色リンクを使用して、任意のオブジェクトに対して色の変更をアニメーション化できます。変化はアナログ型または論理型タグの値、アナログ型または論理型式の値、または論理型またはアナログ型アラームステータスに基づきます。

3 種類の色リンクを使用して、オブジェクトをアニメーション化できます。

- 線/輪郭色
- 塗りつぶしの色
- テキスト色

これら 3 つの色リンクの種類に対して、4 種類の式で色の変化を制御できます。

式タイプ	色を変更する基準
論理型	論理型タグまたは式の値
アナログ型	アナログ型タグまたは式の値。異なる値を表すために 10 色を定義できます。
論理型アラーム	タグ、アラーム グループ、またはグループ変数のアラーム状態
アナログ型アラーム	アナログ型タグ、アラーム グループ、またはグループ変数のアラーム状態。5 つのアラーム条件を表すために 5 色を定義できます。

警告! リンクが InTouch バージョン 7.11 より前に作成された未変換アプリケーションからリモートタグに設定されている場合、アナログ型アラーム リンクを使用しているときにはオブジェクトはアラーム状態にはなりません。

すべての論理値色リンクは同じ方法で作成されます。以下の手順では、塗りつぶし色リンクの作成について説明します。

論理型塗りつぶし色リンクを作成するには

1. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] をクリックします。
[アニメーションリンク] ダイアログ ボックスが表示されます。
2. [塗りつぶしの色] セクションで [論理型] をクリックします。
[塗りつぶしの色 -> 論理型式] ダイアログ ボックスが表示されます。

3. [式] ボックスに、True または False に等しい論理型タグまたは論理型式の名前を入力します。
論理型式には、アナログ型タグを含めることができます。たとえば、TankLevel >= 75 の場合、変数 "TankLevel" の値が 75 以上の場合にオブジェクトの塗りつぶし色が変化します。
4. [色] セクションでカラー ボックスをクリックして、カラー パレットを開きます。各状態に使用する色を選択します。
5. [OK] をクリックします。

アナログ型式の色リンクを作成するには

1. オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ダイアログ ボックスが表示されます。
2. [塗りつぶしの色] セクションで [アナログ型] をクリックします。
[塗りつぶしの色 -> アナログ型式] ダイアログ ボックスが表示されます。

3. [式] ボックスに、アナログ型タグの名前、またはアナログ値に等しい式を入力します。
4. [変化境界値] セクションで以下の手順を実行します。
 - オブジェクトの色が変化するブレイクポイント値を指定します。

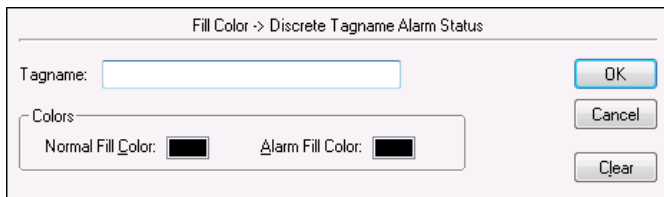
ヒント: 10 の異なる色を使用する必要はありません。たとえば、オブジェクトの色を 3 回だけ変化させる場合、3 種類の値を入力して残りの値に同じ色を使用します。より多様な範囲が必要な場合は、産業用グラフィックのアナログ塗りつぶし機能を参照してください。

- [色] セクションで各ブレイクポイント値の色を選択します。

1. [OK] をクリックします。

論理型アラーム ステータスの色リンクを作成するには

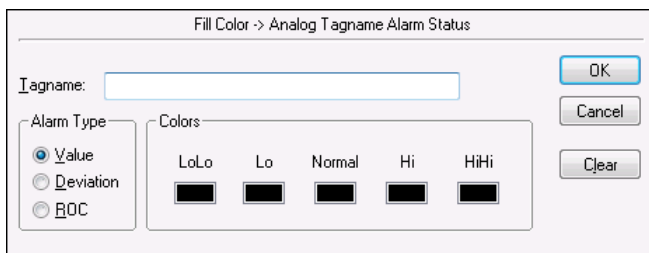
1. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] をクリックします。
[アニメーションリンク] ダイアログ ボックスが表示されます。
2. [塗りつぶしの色] セクションで [論理型アラーム] をクリックします。
[塗りつぶしの色 -> 論理型タグ名アラーム ステータス] ダイアログ ボックスが表示されます。



3. [タグ] ボックスに、オブジェクトに関連付けられている論理型タグの名前を入力します。
4. [色] セクションで各アラーム状態の色を選択します。
5. [OK] をクリックします。

アナログ型アラーム ステータスの色リンクを作成するには

1. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] をクリックします。
[アニメーションリンク] ダイアログ ボックスが表示されます。
2. [塗りつぶしの色] セクションで [アナログ型アラーム] をクリックします。
[塗りつぶしの色 -> アナログ型タグ名アラーム ステータス] ダイアログ ボックスが表示されます。



3. [タグ] ボックスに、オブジェクトに関連付けられているアナログ型タグの名前を入力します。
4. [アラーム タイプ] セクションで、オブジェクトに関連付けるアラームの 3 つのタイプから 1 つを選択します。

アラーム タイプ	最大使用数
値	値アラームの状態を表す 5 色
偏差	偏差アラームの状態を表す 3 色
変化率	変化率アラームの状態を表す 2 色

5. [色] セクションで各アラーム状態の色を選択します。

6. [OK] をクリックします。

塗りつぶしレベルのアニメーション化

塗りつぶしパーセント リンクを使用して、オブジェクトの塗りつぶしレベルを変更できます。塗りつぶしパーセントは、アナログ型タグまたは式の値に基づきます。水平塗りつぶし、垂直塗りつぶし、または両方を作成できます。

たとえば、垂直塗りつぶしリンクを使用してタンク内の液体のレベルを表したり、水平塗りつぶしリンクを使用してプロセスの進捗状況を表すことができます。

水平塗りつぶしパーセントリンクと垂直塗りつぶしパーセントリンクは同じ方法で作成します。

塗りつぶしパーセント リンクを作成するには

1. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。

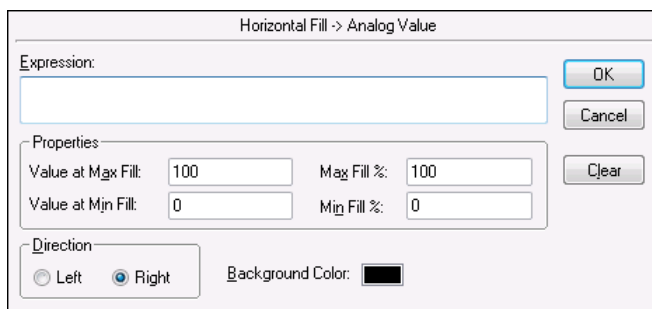
または、オブジェクトを右クリックして、[アニメーションリンク] をクリックします。

[アニメーションリンク] ダイアログ ボックスが表示されます。

2. [塗りつぶしパーセント] セクションで以下のいずれかを実行します。

- [垂直] をクリックすると、[垂直塗りつぶし->アナログ値] ダイアログ ボックスが表示されます。

- [水平] をクリックすると、[水平塗りつぶし->アナログ値] ダイアログ ボックスが表示されます。



3. [式] ボックスに、アナログ型タグの名前、またはアナログ値に等しい式を入力します。
4. [プロパティ] セクションで以下の手順を実行します。
 - a. [最大レベル値] ボックスに、オブジェクトが最大塗りつぶしレベルとなる式の値を入力します。
 - b. [最小レベル値] ボックスに、オブジェクトが最小塗りつぶしレベルとなる式の値を入力します。
 - c. [最大値] ボックスには、式の値が [最大レベル値] ボックスで設定したレベルに達したときにオブジェクトが塗りつぶされる値を 0～100 パーセントで入力します。
 - d. [最小値] ボックスには、式の値が [最小レベル値] ボックスで設定したレベルに達したときにオブジェクトが塗りつぶされる値を 0～100 パーセントで入力します。
5. [方向] セクションで塗りつぶしの方向を選択します。
6. [背景色] ボックスに、オブジェクトが塗りつぶされない部分の色を選択します。
 - 実際の塗りつぶし色は、オブジェクトを描画する場合にオブジェクトに対して選択する色です。
 - 垂直塗りつぶしパーセントリンクと水平塗りつぶしパーセントリンクの両方を同じオブジェクトにリンクした場合、最後に選択した色が背景色となります。
7. [OK] をクリックします。

オブジェクトを点滅させる

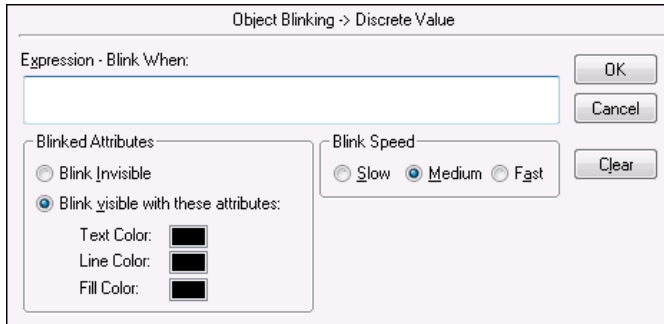
点滅リンクを使用して、タグの値に基づいて点滅するアニメーションオブジェクトを作成できます。たとえば、特定の装置がオンになったとき、またはアラーム設定ポイントに達したときに赤く点滅するオブジェクトを作成できます。

点滅アニメーションの同期は、開いているすべてのウィンドウのアクティブな点滅アニメーションすべてを、共通の点滅速度（低速、中速、または高速）でいっせいに点滅させることができます。

ヒント: 論理型式には、アナログ型タグ名を含めることができます。たとえば、`TankLevel > 75` の場合、`TankLevel` タグの値が 75 より大きい場合にオブジェクトの点滅が開始します。

点滅リンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. [その他] セクションで [点滅] をクリックします。
[オブジェクト点滅->論理値] ダイアログボックスが表示されます。



4. [タグ/式- 点滅条件] ボックスに、論理型タグまたは論理値に等しい式の名前を入力します。
5. [オブジェクト点滅プロパティ] セクションで以下の手順を実行します。
 - [目に見えない点滅] をクリックして、ウィンドウ内で表示のオン/オフを切り替えてオブジェクトを点滅させます。
 - [これらの属性を備えた目に見える点滅] をクリックして、オブジェクトが表示されたままになるようにし、アクティブになったときに色が変わります。
 - [テキストの色]、[線の色]、または [塗りつぶしの色] ボックスをクリックして、オブジェクトのパーツの色を選択します。カラーパレットが表示されます。

注記: オブジェクトの塗りつぶし色と点滅の塗りつぶし色を同じにすると、オブジェクトは点滅しているように見えません。

6. [点滅速度] セクションで、オブジェクトの点滅速度を設定します。[低速]、[中速]、または [高速] のいずれかをクリックします。
7. [OK] をクリックします。

WindowMaker の点滅速度を設定するには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[WindowViewer] をクリックします。
WindowViewer の設定画面が表示されます。
3. [環境設定] タブの [点滅速度 (ミリ秒)] セクションに、3 つの速度で使用する時間をミリ秒で入力します。

注記: 点滅速度の変更はグローバル設定で、アプリケーションのすべての点滅リンクに影響します。

4. [OK] をクリックします。

表示オン/オフの有効化

表示オン/オフリンクを使用して、さまざまなタグの値に基づいてオブジェクトを非表示にするリンクを作成できます。表示オン/オフリンクを使用すると、以下の操作を行えます。

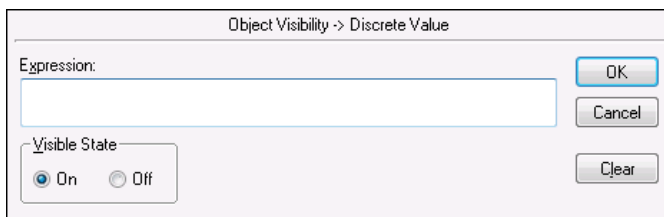
- 間違った方向に移動したときに非表示にすることによって、移動オブジェクトが一方向にのみ移動するという印象を作成します。
- 移動オブジェクトが停止したという印象を作成します。
- アラームやエラー メッセージなどのオブジェクトは、アクティブになっているときにのみ表示されるようにします。

ヒント: 論理型式には、TankLevel >= 75 など

アナログ型タグを含めることができます。この例では、タグ TankLevel の値が 75 以上の場合に、オブジェクトがウィンドウで表示されるようになります。

表示オフ/オン リンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. [その他] セクションで [表示オン/オフ] をクリックします。
[オブジェクト表示オフ/オン->論理値] ダイアログボックスが表示されます。



4. [式] ボックスに、論理型タグの名前か論理値に等しい式を入力します。
5. オブジェクトの [表示状態 (値が真 1 のとき)] を選択します。[オフ] を選択すると、式の値が True の場合オブジェクトは表示されません。[オン] を選択すると、式の値が True の場合にオブジェクトが表示されます。
6. [OK] をクリックします。

オブジェクトの無効化

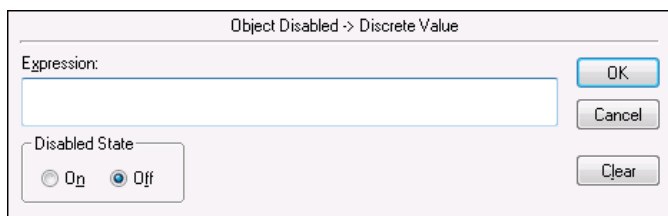
タッチアクション無効化リンクを持つアプリケーションにセキュリティ レベルを設定できます。たとえば、オペレータのアクセス レベルや名前に基づいて、タッチオブジェクトを無効にできます。または、誰もログオンしていない場合にボタンが間違って操作されないように設定できます。

無効化状態がオンの場合、式の値が True である限り、オブジェクトやボタンのタッチ機能がオフとなり、アクティブにはなりません。

ヒント: 論理型式には、アナログ型タグ名を含めることができます。たとえば、TankLevel >= 75 の場合、変数 "TankLevel" の値が 75 以上の場合にオブジェクトが無効になります。

無効化リンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. [その他] セクションで [無効] をクリックします。
[オブジェクト無効化->論理値] ダイアログボックスが表示されます。



4. [式] ボックスに、論理型タグの名前か論理値に等しい式を入力します。

5. [無効化状態] セクションで、以下のいずれかを実行します。

- [オン] を選択すると、無効化状態が設定され、オブジェクトは論理型タグまたは式が True の間はアクティブになりません。
- [オフ] を選択すると、無効化状態が削除され、オブジェクトは論理型タグまたは式が True の間は機能します。

6. [OK] をクリックします。

ツールヒントの設定

ツールヒントリンクを使用して、ユーザーに画面上でオブジェクトに関する情報を提供するツールヒントを作成できます。ツールヒントはポインタがオブジェクト上に置かれたときに表示され、ポインタが離れると非表示となります。ツールヒントが表示される持続時間とツールヒントの配置は、オペレーティングシステムによって決定されます。

ツールヒントには式または静的テキストを設定できます。

- 静的ツールヒントを作成して、ツールヒントが表示されるたびに同じメッセージを表示します。
- 式ツールヒントを作成して、ツールヒントが表示されるたびに式が評価されてツールヒントテキストとして表示されるようにします。

以下の式の例では、テキストが msgTooltipTag01 メッセージ型タグの現在の値として表示されます。

msgTooltipTag01

この例では、リテラル文字列の次に iTemp タグの現在の値が表示され、結果がツールヒントテキストとして表示されます。

"Current temp. is " + StringFromIntg (iTemp,10)

ツールヒントウィンドウの幅は、Windows 7 およびそれ以降のバージョンの Windows で変更できます。InTouch.INI ファイルを編集して、ツールヒントの幅を指定するエントリを追加できます。

例：

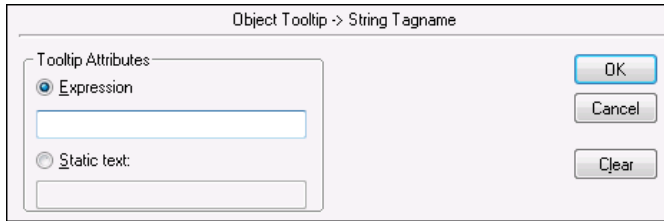
- Tooltipwidth=200
- -1 の値 (Tooltipwidth=-1) を指定すると、ツールヒントは改行なしの単一行に表示されます。
- ツールヒントの幅を指定しない場合、ツールヒントウィンドウのデフォルトの幅は 88 です。

ツールヒントリンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。

3. [その他] セクションで [ツールヒント] をクリックします。

[オブジェクト ツールヒント -> 文字列タグ名] ダイアログ ボックスが表示されます。



4. [ツールヒント属性] セクションで、[式] または [静的テキスト] のいずれかを選択します。

- [式] を選択した場合、メッセージ値を評価する式を入力します。シンプルなメッセージ型タグ名またはより複雑な式を使用できます。
- [静的テキスト] を選択した場合は、ツールヒントテキストとして静的メッセージ（最大 131 文字）を入力します。

5. [OK] をクリックします。

タッチスクリーン ウィンドウの配置

ランタイムでタッチスクリーン オブジェクトに関連した正確な位置にウィンドウを表示させることができます。たとえば、オペレータがオブジェクトを選択して、ステータス、名前、またはそのオブジェクトにリンクされているその他のデータを参照できます。オペレータがクリックまたはマウスオーバーでオブジェクトを選択すると、指定した位置にウィンドウが表示されます。

スクリプト関数 `ShowAt()` または `ShowTopLeftAt()` をシステム読み取り専用タグ `$ObjHor` および `$ObjVer` と共に使用して、オブジェクトに関連するウィンドウを検索できます。これらの関数で固定位置を使用することもできます。

Windows の [画面のプロパティ] が [Windows XP] のテーマに設定されていると、この機能は場合によっては不安定になります。

構文は以下のとおりです。

```
ShowTopLeftAt (windowname, $ObjHor, $ObjVer);
```

各項目の説明

windowname : 開かれるウィンドウの名前です。

\$ObjHor : 選択したオブジェクトの中心の水平位置

\$ObjVer : 選択したオブジェクトの中心の垂直位置。

左上隅が選択したオブジェクトの中心に配置された状態で、新しいウィンドウが表示されます。

同様のスクリプト関数は、中心が選択したオブジェクトの中心に配置された状態でウィンドウを開きます。構文は以下のとおりです。

```
ShowAt (windowname, $ObjHor, $ObjVer);
```

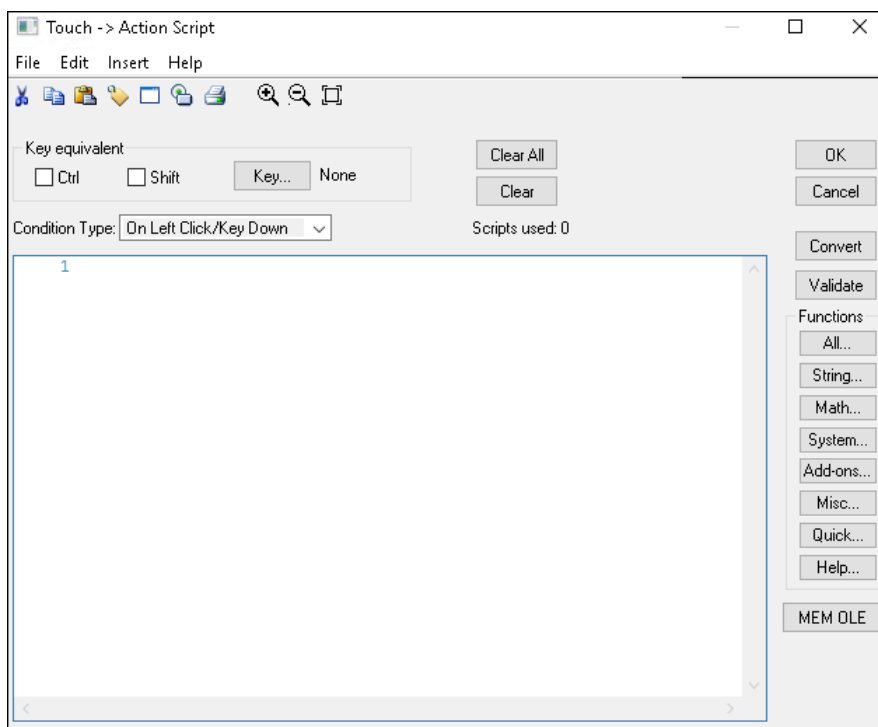
選択したオブジェクトでウィンドウを開くには

1. 表示するウィンドウを設計、命名、および作成します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。

[アニメーション リンク] ウィンドウが表示されます。

3. [タッチ押しボタン] セクションで [アクション] を選択します。

[タッチ->アクションスクリプト] ダイアログ ボックスが表示されます。



4. 前に定義した構文に従って、以下のスクリプトの 1 つを入力します。

```
ShowTopLeftAt (windowname, $ObjHor, $ObjVer);
```

または

```
ShowAt (windowname, $ObjHor, $ObjVer);
```

5. [実行条件] ボックスで、ウィンドウを開くマウス アクションをクリックします。

6. [OK] をクリックします。

\$ObjHor システム タグ変数

対象オブジェクトの中心の水平ピクセル位置が含まれます。

カテゴリ

システム

使用法

\$ObjHor

データ タイプ

整数型（読み取り専用）

参照項目

\$ObjVer

\$ObjVer システム タグ変数

対象オブジェクトの中心の垂直ピクセル位置が含まれます。

カテゴリ

システム

使用法

\$ObjVer

データタイプ

整数型（読み取り専用）

参照項目

\$ObjHor

データ入力アニメーション

オペレータ操作を有効にするオブジェクトを作成するには、タッチ リンクを使用します。オペレータは、タッチ リンクを使用してシステムにデータを入力できます。たとえば、オペレータはキーボードを使用したログオン、バルブのオン／オフ切り替え、新しいアラーム限界値の入力、プロセスの開始と停止を実行できます。

タッチ オブジェクトにフォーカスされると周囲に枠が表示されます。オブジェクトはユーザーがカーソルをオブジェクトの上に動かしたとき、または **Tab** キーまたは矢印キーを押してオブジェクトにフォーカスを移動したときにフォーカスされます。

ユーザーが **Tab** キーを使用してタッチ オブジェクトを選択するようにする場合は、水平パターンに配置するようにしてください。**Tab** キーを押すと、フォーカスが 1 つのオブジェクトから別のオブジェクトへ左から右にウィンドウの上部から下へ順番に移動します。

タッチ オブジェクトにそれぞれが重なり合ったテキスト オブジェクトを含む場合は、一番上のテキスト オブジェクトのみが表示されます。

オペレータは、タッチ オブジェクトをクリックしたり、割り当てられているキーを押したり、オブジェクトの枠が表示されたときに **Enter** キーを押したり、またはタッチ スクリーン表示装置を使用している場合は実際にタッチすることによって、オブジェクトをアクティブにすることができます。

9 つのタイプのユーザー入力タッチ リンクを作成することができます。

タッチ リンク	アクション
データ入力	<ul style="list-style-type: none"> 論理型 アナログ型 文字列型
スライダ入力	<ul style="list-style-type: none"> 垂直方向 水平方向
押しボタン	<ul style="list-style-type: none"> 論理値 アクション ウィンドウ表示 ウィンドウ消去

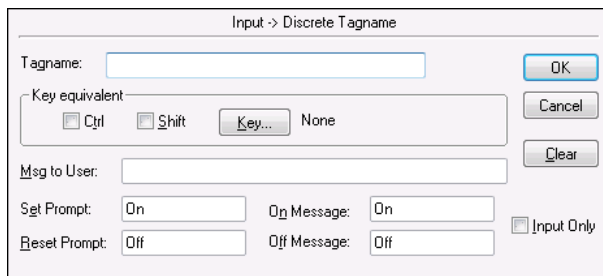
テキスト フィールドが入力に使用されている場合、キーが押されるとテキストが画面に表示されます。入力しているときにテキストを表示しない場合は、リンクの設定パネルの **【入力のみ】** オプションを選択します。

論理値入力の有効化

論理型タグの値をある状態から別の状態に変更するためのオペレータ入力タッチ リンクを設計できます。たとえば、ポンプをオンまたはオフにするには、論理型リンクを使用します。

論理値入力リンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. **【描画】** メニューの **【モード】** グループで **【アニメーション化】** をクリックします。
または、オブジェクトを右クリックして、**【アニメーションリンク】** を選択します。
【アニメーションリンク】 ウィンドウが表示されます。
3. **【タッチリンク】** 領域で、**【データ入力】** の下にある **【論理値】** を選択します。
【論理値入力リンク】 ダイアログ ボックスが表示されます。



4. **【タグ名】** ボックスに、論理型タグの名前を入力します。
5. オプションで、**【キー アクションの割り当て】** 領域で、リンクに割り当てるキーを設定します。詳細については、「[キーボードショートカットの作成](#)」を参照してください。
6. 論理値入力詳細を設定します。以下の手順を実行します。
 - **【プロンプト メッセージ】** ボックスに、**【入力】** ダイアログ ボックスに表示するメッセージを入力します。
 - **【オンボタン タイトル】** ボックスおよび **【オフボタン タイトル】** ボックスに、オペレータが論理値のオン／オフを切り替えるときにクリックするボタンに表示するメッセージを入力します。
 - **【オンメッセージ】** ボックスおよび **【オフメッセージ】** ボックスには、オブジェクトに関連付けられているテキスト フィールドに表示するオン／オフ メッセージを入力します。
7. オブジェクトに関連付けられているテキスト フィールドに入力値が表示されないようにするには、**【入力のみ】** チェック ボックスをオンにします。
8. **【OK】** をクリックします。

アナログ入力の有効化

入力リンクを使用して、オペレータがアラーム限界点またはコンベヤ速度などの実数値を入力できるオブジェクトを作成できます。WindowViewer のプロパティで **【10 進法入力のみを許可】** オプションを選択すると、1E2 などの数値以外の値を入力フィールドに入力できなくなります。ランタイム設定の詳細

については、『AVEVA™ InTouch HMI コンポーネントの標準の作成』ガイドの「[一般的な WindowViewer プロパティの設定](#)」を参照してください。

アナログデータ入力リンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. [タッチリンク] 領域で、[データ入力] の下にある [アナログ] を選択します。
[アナログデータ入力リンク] ダイアログボックスが表示されます。

4. [タグ名] ボックスに、アナログ型タグの名前を入力します。
5. [キーアクションの割り当て] 領域で、リンクに割り当てるキーを設定します。詳細については、「[キーボードショートカットの作成](#)」を参照してください。
6. アナログ値入力詳細を設定します。以下の手順を実行します。
 - 値を入力するために画面上に数値キーパッドを表示するには、[キーパッドの使用] 領域で、[する] をクリックします。[キーパッドのタイトル] ボックスに、キーパッドに表示するプロンプトメッセージを入力します。
 - [最小値] ボックスおよび [最大値] ボックスに、タグに対する最小および最大の入力値を入力します。これらの設定は、ランタイム時のアニメーションに対するユーザー入力を制限します。定数アナログ値（整数または実数）、および参照もサポートされています。デフォルト値はそれぞれ 1 および 100 です。最大値は最小値より大きい必要があります。
参照を使用する場合の詳細については、「アナログ入力アニメーションの最小と最大の制限で参照を使用する」セクションを参照してください。
7. オブジェクトに関連付けられているテキストフィールドに入力値が表示されないようにするには、[入力のみ] チェックボックスをオンにします。
8. [書式設定] 領域のリストで、ランタイム時の詳細な書式設定を行う、それぞれのデータタイプをクリックします。選択したデータタイプに基づいて、[固定幅] チェックボックス、[精度] ボックス、および [ビット数から] および [まで] ボックスが有効になります。これらのオプションの設定の詳細については、「テキストの詳細な書式設定」セクションを参照してください。
9. [OK] をクリックします。

アナログ入力アニメーションの最小と最大の制限に対する参照の使用

アナログユーザー入力アニメーションの最小と最大の制限では、定数のアナログ値と参照がサポートされています。これらの制限は、**[アナログデータ入力リンク]** ダイアログ ボックスで設定できます。

参照はメモリ タグ変数または I/O タグ変数などの、リモート アナログ タグ変数またはローカル アナログ タグ変数のいずれかになります。

- InTouch I/O タグ変数を最小または最大値として使用した場合、InTouch I/O タグ変数だけが評価されます。I/O タグ変数が文字列値の割り当てられたリモート タグ リファレンスを参照している場合、I/O タグ変数の値は 0 として評価されます。
- 最小値と最大値でリモート タグ リファレンスが設定されている場合、そのリファレンスは文字列として評価されます。リモート タグ リファレンスが文字列値に割り当てられており、アナログ値への変換に失敗した場合、警告メッセージがロガーに記録されます。**[最小値]** ボックスと **[最大値]** ボックスは、ユーザー入力が設定されたリファレンスのデータ タイプに基づいた値を表示します。
- **[最小値]** フィールドまたは **[最大値]** フィールドのユーザー入力が、「e」の文字を含んでいる場合、入力は文字列値または指数値のいずれかになります。ユーザー入力に数字、「e」の文字、数字（つまり $\pm Ne \pm N$ の形式、この場合の N は数字）を含んでいる場合、ユーザー入力は指数値としてみなされます。ユーザー入力に「e」とアルファベット（「e」を含む）が含まれている場合、ユーザー入力はリファレンス名としてみなされます。

参照値がランタイム時に不適切である場合、次の状況が発生します。

[最小値] ボックスの参照については、

- アニメーションのプライマリ参照が InTouch タグ変数の場合、設定されたタグ変数の最小値がタグ変数データベースから取得され、最小値に使用されます。
- アニメーションのプライマリ参照が外部参照の場合、データ タイプの最小値が使用されます。設定されたデータ ポイントが文字列の場合、最後の既知の値によってデータ タイプが決まります。
 - 最後の既知の値に小数点がある場合、データ タイプは実数になります。
 - 最後の既知の値に小数点がある場合、データ タイプは整数になります。
 - 最後の既知の値がない場合、整数が使用されます。

[最大値] ボックスの参照については、

- アニメーションのプライマリ参照が InTouch タグ変数の場合、設定されたタグ変数の最大値がタグ変数データベースから取得され、最大値に使用されます。
- アニメーションのプライマリ参照が外部参照の場合、データ タイプの最大値が使用されます。設定されたデータ ポイントが文字列の場合、最後の既知の値によってデータ タイプが決まります。
 - 最後の既知の値に小数点がある場合、データ タイプは実数になります。
 - 最後の既知の値に小数点がある場合、データ タイプは整数になります。
 - 最後の既知の値がない場合、整数が使用されます。

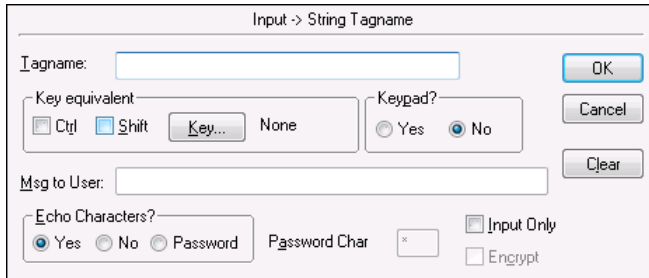
文字列入力の有効化

文字列入力リンクを使用して、ユーザーがバッチ名、オペレータ ID、またはパスワードなどのテキスト文字列を入力できるようにするオブジェクトを作成できます。

文字列入力リンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。

2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. [タッチリンク] 領域で、[データ入力] の下にある [文字列] を選択します。
[文字列入力リンク] ダイアログボックスが表示されます。



4. [タグ名] ボックスに、メッセージ型タグの名前を入力します。
5. オプションで、割り当てるキーまたはキーボードを設定します。
 - [キーアクションの割り当て] 領域で、リンクに割り当てるキーを設定します。詳細については、「[キーボードショートカットの作成](#)」を参照してください。
 - 新しい値を入力するためのキーボードを表示するには、[キーパッドの使用] 領域で、[する] をクリックします。[キーパッドのタイトル] ボックスに、キーボードに表示するメッセージを入力します。
6. [入力文字列の表示] 領域で、ユーザーが文字列を入力したときに入力ボックスに文字を表示するかどうかを選択します。
 - 入力したテキストを入力ボックスに表示するには、[する] をクリックします。
 - 入力したテキストを表示しない場合は、[しない] をクリックします。
 - 入力したテキストの代わりに「マスク」文字を使用するには、[パスワード] をクリックします。
[パスワードのエコー文字] ボックスに、マスク文字を入力します。[暗号化] チェックボックスをオンにして、パスワードを暗号化します。

重要: パスワード暗号化は、InTouch HMI コンテキスト内でのみ動作します。オペレーティングシステムや SQL Server データベースなどの外部のセキュリティシステムに渡す場合は文字列を暗号化しないでください。外部セキュリティシステムは暗号化パスワード文字列を読み取ることができず、アクセスは失敗します。

1. オブジェクトに関連付けられているテキストフィールドに入力値が表示されないようにするには、[入力のみ] チェックボックスをオンにします。
2. [OK] をクリックします。

スライダ入力の有効化

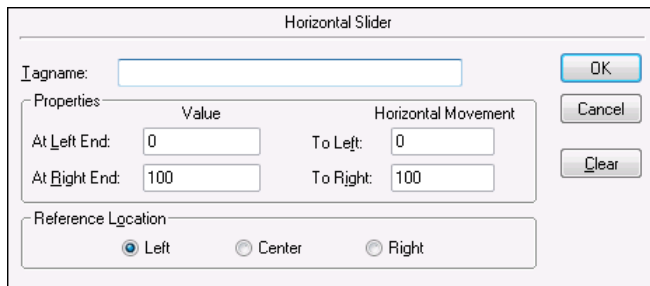
ユーザーがスライダタッチリンクを使用して、前後にドラッグできるオブジェクトを作成できます。ユーザーがオブジェクトを動かすと、それにリンクされているタグの値が変わります。オブジェクトを水平または垂直スライダにリンクできます。

オブジェクトをスライダにする際、カーソルがロックするために使用するオブジェクト上のポイントとなる参照位置を設定します。

単一オブジェクトに水平リンクおよび垂直リンクの両方を使用して、2つのアナログ型タグの値が同時に変わるようにします。

水平スライダリンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. [スライダ入力] 領域で、[水平方向] をクリックします。
[水平スライダ] ダイアログボックスが表示されます。



The image shows a 'Horizontal Slider' dialog box. It has a title bar 'Horizontal Slider'. Inside, there is a 'Tagname:' field with a text input box. Below it is a 'Properties' section with two columns: 'Value' and 'Horizontal Movement'. Under 'Value', there are 'At Left End:' (0) and 'At Right End:' (100). Under 'Horizontal Movement', there are 'To Left:' (0) and 'To Right:' (100). To the right of these fields are 'OK', 'Cancel', and 'Clear' buttons. At the bottom is a 'Reference Location' section with three radio buttons: 'Left' (selected), 'Center', and 'Right'.

4. [タグ名] ボックスに、アナログ型タグの名前を入力します。
5. [プロパティ] 領域で、以下の手順を実行します。
 - a. [左端の値] ボックスに、スライダが左端にあるときのタグの値を入力します。
 - b. [右端の値] ボックスに、スライダが右端にあるときのタグの値を入力します。
 - c. [左への移動幅] ボックスに、スライダを左に動かすことができる幅をピクセル数で入力します。
 - d. [右への移動幅] ボックスに、スライダを右に動かすことができる幅をピクセル数で入力します。
6. [カーソルの基準点] 領域で、カーソルがオブジェクトにロックする際に使用するオブジェクト上の位置をクリックします。
7. [OK] をクリックします。

垂直スライダリンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. [スライダ入力] 領域で、[垂直方向] をクリックします。
[垂直スライダ入力タッチリンク] ダイアログボックスが表示されます。

The 'Vertical Slider' dialog box contains the following elements:

- Tagname:** A text input field.
- Properties:**
 - Value:** Fields for 'At Top' (0) and 'At Bottom' (100).
 - Vertical Movement:** Fields for 'Up' (0) and 'Down' (100).
- Reference Location:** Radio buttons for 'Top', 'Middle', and 'Bottom' (selected).
- Buttons:** 'OK', 'Cancel', and 'Clear'.

4. **[タグ名]** ボックスに、アナログ型タグの名前を入力します。
5. **[プロパティ]** 領域で、以下の手順を実行します。
 - a. **[上端の値]** ボックスに、スライダが上端にあるときのタグの値を入力します。
 - b. **[下端の値]** ボックスに、スライダが下端にあるときのタグの値を入力します。
 - c. **[上への移動幅]** ボックスに、スライダを上を動かすことができる幅をピクセル数で入力します。
 - d. **[下への移動幅]** ボックスに、スライダを下を動かすことができる幅をピクセル数で入力します。
6. **[カーソルの基準点]** 領域で、カーソルがオブジェクトにロックする際に使用するオブジェクト上の位置をクリックします。
7. **[OK]** をクリックします。

押しボタンの有効化

タッチ押しボタンリンクを使用して、アクション スクリプトを開始するタッチ オブジェクトを作成できます。

アクション スクリプトは、タグを特定の値に設定したり、他のアプリケーションの起動や制御、機能の実行などに使用できます。

論理値タッチ リンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. **[描画]** メニューの **[モード]** グループで **[アニメーション化]** をクリックします。
または、オブジェクトを右クリックして、**[アニメーションリンク]** を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. **[タッチ押しボタン]** 領域で、**[論理値]** をクリックします。
[押しボタン - 論理値] ダイアログ ボックスが表示されます。

The 'Pushbutton -> Discrete Value' dialog box contains the following elements:

- Tagname:** A text input field.
- Key equivalent:** Checkboxes for 'Ctrl' and 'Shift', and a 'Key...' button followed by 'None'.
- Action:** Radio buttons for 'Direct' (selected), 'Reverse', 'Toggle', 'Reset', and 'Set'.
- Buttons:** 'OK', 'Cancel', and 'Clear'.

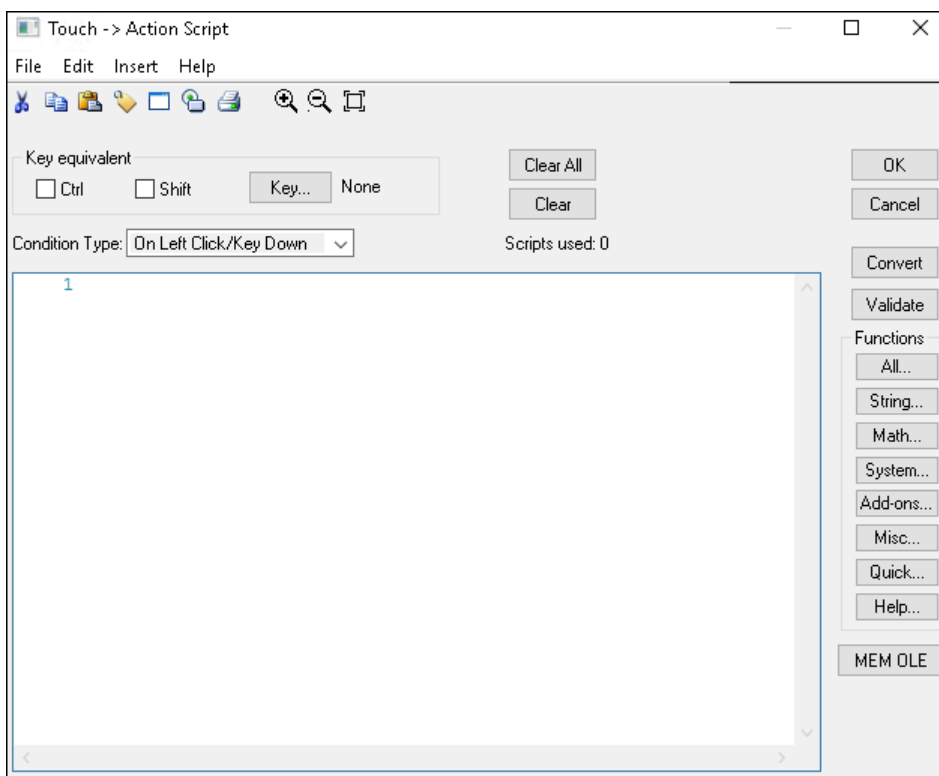
4. **[タグ名]** ボックスに、論理型タグ名を入力します。
5. **[キー]** ボタンをクリックして、リンクにキーを割り当てます。
6. **[ボタンアクションプロパティ]** 領域で、以下のいずれかのタイプをクリックします。

- [ダイレクト] をクリックすると、押しボタンが押されている間、値が **1** に設定されます。ボタンを離すと、値は自動的に **0** にリセットされます。
- [リバーズ] をクリックすると、押しボタンが押されている間、値が **0** に設定されます。ボタンを離すと、値は自動的に **1** にリセットされます。
- [トグル] をクリックすると、論理型タグの状態をリバーズします。
- [リセット] をクリックすると、値が **0** に設定されます。
- [セット] をクリックすると、値が **1** に設定されます。

7. [OK] をクリックします。

アクションスクリプト リンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. [タッチ押しボタン] 領域で、[スクリプト ボタン] をクリックします。
[タッチアクションスクリプト] ダイアログが表示されます。



4. [実行条件] リストで、スクリプト タイプを選択します。アクションスクリプトのタイプの詳細については、「[スクリプト トリガ](#)」を参照してください。

注記: アクション押しボタンにキーを割り当てて、同じキーがキー スクリプトに使用されている場合、割り当てたキーがキー スクリプトよりも優先されます。

5. [スクリプト エディタ] ウィンドウで、オブジェクトが起動されたときに実行するスクリプトを入力します。
6. [OK] をクリックします。

ウィンドウを開く／閉じる

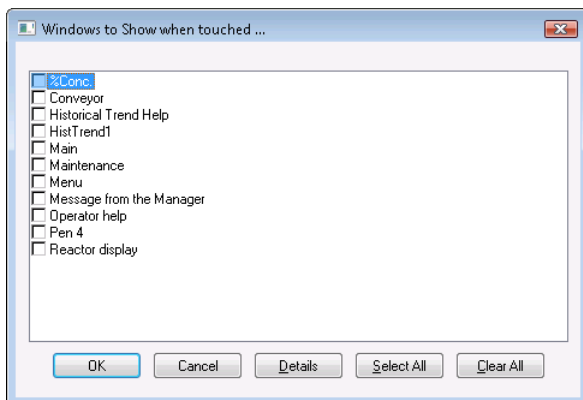
ウィンドウの表示／消去リンクを使用して、他の InTouch ウィンドウを開いたり閉じたりするタッチ リンクを作成できます。

オブジェクトをプログラムして、一度に複数のウィンドウを開くように設定できます。ただし、複数のウィンドウを開くようにリンクをプログラムした場合、重なりとウィンドウのタイプに注意してください。

開いているウィンドウの1つが置換タイプのウィンドウで別に開いているウィンドウと交差している場合は、開く前に他のウィンドウが閉じます。

ウィンドウ表示／消去リンクを作成するには

1. アニメーションを適用するオブジェクトを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
3. [タッチ押しボタン] 領域で、[ウィンドウ表示] または [ウィンドウ消去] をクリックします。
[タッチ時に表示するウィンドウ] または [タッチ時に非表示するウィンドウ] ダイアログ ボックスが表示されます。



4. 表示または非表示にするウィンドウを選択します。
5. [OK] をクリックします。

画面上キーボードの設定

画面上キーボードを使用すると、コンピュータにキーボードが接続されていない場合にオペレータ入力が可能になります。

3つの画面上キーボードタイプのうちの1つを指定できます。

- 標準 InTouch キーボードまたはキーパッド。これがデフォルトのキーボードです。

- **Windows** システムのキーボード。**Windows** キーボードは、完全に機能する QWERTY タイプのキーボードで、ファンクションキー、プリントスクリーンキー、数字ロックキー、方向矢印などを備えています。
- リサイズ可能なキーボードまたはキーパッド。このキーボードは、ランタイムにサイズを変更できます。

キーボードは `DialogStringEntry()` 関数および `DialogValueEntry()` 関数をスクリプトで使用して開くこともできます。詳細については、「[DialogStringEntry\(\) 関数](#)」および「[DialogValueEntry\(\) 関数](#)」を参照してください。

画面上キーボードタイプを設定するには

1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[WindowViewer] をクリックします。
WindowViewer の設定画面が表示されます。
3. [環境設定] タブの [キーボード] 領域で、使用するキーボードのタイプを選択します。
4. [リサイズ可能なキーボード] を選択した場合は、[オプション] をクリックして、キーボードのフォント、位置、およびサイズプロパティを選択します。
5. [OK] をクリックします。

画面上キーボードを表示するには

1. 画面上キーボードタイプを設定します。
2. アニメーションを適用するオブジェクトを選択します。
3. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーションリンク] を選択します。
[アニメーションリンク] ウィンドウが表示されます。
4. [タッチリンク] 領域で、[データ入力] の下にある [文字列] を選択します。
[文字列入力リンク] ダイアログボックスが表示されます。

The screenshot shows a dialog box titled "Input -> String Tagname". It has several sections: "Tagname:" with a text input field; "Key equivalent" with checkboxes for "Ctrl" and "Shift", a "Key..." button, and a "None" option; "Keypad?" with radio buttons for "Yes" and "No"; "Msg to User:" with a text input field; "Echo Characters?" with radio buttons for "Yes", "No", and "Password"; "Password Char" with a text input field and a "*" character; "Input Only" with a checkbox; and "Encrypt" with a checkbox. There are three buttons on the right: "OK", "Cancel", and "Clear".

5. [キーパッドの使用] 領域で、[する] を選択します。
6. [OK] をクリックします。

DialogStringEntry() 関数

この関数は、画面に英数字キーボードを表示し、オペレータがタグ変数ディクショナリにあるメッセージ型タグ変数の現在の文字列の値を画面上で変更できるようにします。

カテゴリ

その他

構文

```
[Result=]DialogStringEntry(MessageTag_Text, UserPrompt_Text);
```

パラメータ

MessageTag_Text

変更するメッセージ型タグ変数の名前です。この値は文字列値です。タグ変数を引用符で囲んで指定するか、引用符なしの .Name ドットフィールドを使用します。メッセージ型タグ変数をポインタとして使用することもできます。

UserPrompt_Text

キーボード上部に表示するユーザー メッセージです。

戻り値

以下の整数値のいずれかを返します。

0 = [キャンセル] がクリックされました。

1 = [OK] がクリックされました。

-1 = 内部エラーです。

-2 = 開始できませんでした。

-3 = タグ変数が定義されていません。

-4 = タグ変数がメッセージ型ではありません。

-5 = 書き込めません。

備考

この関数は、タッチ スクリーンを含むアプリケーションで主に使用されます。

例

```
Errmsg=DialogStringEntry(MyMessageTag.Name, "Enter a new string...");  
Errmsg=DialogStringEntry("MyMessageTag","Enter a new string...");
```

たとえば、以下のスクリプトは英数字キーボードを開き、その上部に「新しい文字列を入力してください...」というメッセージを表示する一方、そのキーボードから **MyMessageTag** を変更できるようにします。

```
MessageTagX="MyMessageTag"; {文字列 MyMessageTag (変更するタグ変数) をメモリ メッセージ型タグ変数 MessageTagX に割り当てます}  
MessageDisplay="新しい文字列を入力してください..."; {新しいメッセージ文字列をメモリ メッセージ型タグ変数 MessageDisplay に割り当てます}  
Errmsg=DialogStringEntry(MessageTagX, MessageDisplay); {MessageTagX はメッセージ型タグ変数として定義されているので引用符は必要ありません}
```

参照項目

DialogValueEntry()

DialogValueEntry() 関数

画面上に数字キーパッドを表示し、ユーザーが論理型、整数型、または実数型タグ変数の現在の値を変更できるようにします。

カテゴリ

その他

構文

```
[Result=] DialogValueEntry(ValueTag_Text, LowLimit, HighLimit, UserPrompt_Text);
```

パラメータ

ValueTag_Text

変更する論理型、整数型、または実数型タグ変数の名前です。この値は文字列値です。タグ変数名を引用符で囲んで指定するか、引用符なしの **.Name** フィールドを使用します。メッセージ型タグ変数をポインタとして使用することもできます。

LowLimit

タグ変数の下限値です（この値は、最小値、最小生データ、または最小工学値に対するタグ変数の定義値と同じかそれ以上です）。

HighLimit

タグ変数の上限値です（この値は、最大値、最大生データ、または最大工学値に対するタグ変数の定義値と同じかそれ以下です）。

UserPrompt_Text

キーパッド上部に表示するユーザー メッセージです。

戻り値

以下の整数値のいずれかを返します。

0 = [キャンセル] がクリックされました。

1 = [OK] がクリックされました。

-1 = 上限 <= 下限

-2 = 開始できませんでした。

-3 = タグ変数が定義されていません。

-4 = タグ変数が論理型、整数型、または実数型ではありません。

-5 = 書き込みに失敗しました。

備考

この関数は、タッチ スクリーンを含むアプリケーションで主に使用されます。

例

```
Errmsg=DialogValueEntry(MyIntegerTag.Name, MyIntegerTag.MinEU, MyIntegerTag.MaxEU, "新しい値を入力してください...");
```

```
Errmsg=DialogValueEntry("MyIntegerTag", -100, 100, "新しい値を入力してください...");
```

たとえば、以下のスクリプトは数字キーパッドを表示し、その上部に「新しい文字列を入力してください」というメッセージを表示する一方、そのキーパッドから **-100 ~ 100** の最小および最大制限を使用して、**MyIntegerTag** を変更できるようにします。

```
TagnameX="MyIntegerTag"; {文字列 MyIntegerTag (変更するタグ変数) をメモリ メッセージ型タグ変数 TagnameX に割り当てます}
```

```
Min=-100; {タグ変数に許可される最小値をメモリ実数/整数型タグ変数 Min に割り当てます}
```

```
Max=100; {タグ変数に許可される最大値をメモリ実数/整数型タグ変数 Max に割り当てます}
```

```
MessageDisplay="新しい値を入力してください..."; {新しいメッセージ文字列をメモリ メッセージ型タグ変数 MessageDisplay に割り当てます}  
Errmsg=DialogValueEntry(TagnameX, Min, Max, MessageDisplay); {TagnameX はメッセージ型タグ変数として定義されているため、引用符は必要ありません。論理型、整数型、または実数型タグ変数を TagnameX に割り当てることによって割り当てられたタグ変数を変更します}
```

参照項目

DialogStringEntry()

共通のアニメーションタスク

共通アニメーションタスクには、タグ変数の選択、キーボードショートカットの作成、タグ変数参照の変更、プレースホルダタグ変数の置換などが含まれます。

タグまたは属性の選択

[タグの選択] ダイアログ ボックス（タグブラウザ）を使用して、以下の内容を選択します。

- ローカルまたはリモート InTouch アプリケーションに定義されているタグ。
- Galaxy ブラウザに関連付けられた Application Server オブジェクト属性。

[タグの選択] ダイアログ ボックスを開くには、入力用のタグ名を必要とするテキスト ボックス内をダブルクリックします。

InTouch タグの選択

ローカルまたはリモート InTouch アプリケーションに定義されているタグを選択できます。タグ名ディクショナリ インターフェイスをサポートするタグ ソース内にタグへの参照を作成できます。

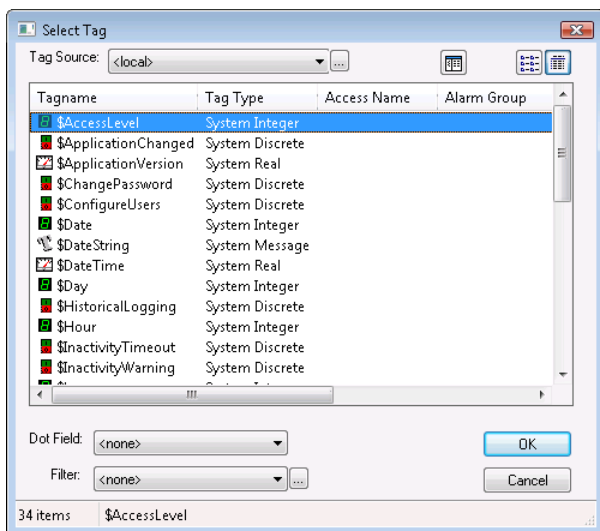
たとえば、リモート タグ 参照を使用すると、ローカルのタグ名ディクショナリでタグを作成することなく、I/O サーバーからデータにアクセスできます。

選択した各 InTouch タグ用のドットフィールドを設定できます。ドットフィールドは、タグプロパティへのアクセス、監視、および変更を行えます。ドットフィールドを選択しない場合、.Value ドットフィールドが使用されます。ドットフィールドの詳細については、「[タグ ドットフィールドを使用したタグ プロパティの表示または変更](#)」を参照してください。

InTouch タグを選択するには

1. 入力にタグ名が必要なテキスト ボックスをダブルクリックします。

[タグの選択] ダイアログ ボックスが表示されます。



2. [タグ ソース] リストで、タグ ソースの名前をクリックするか、参照ボタンをクリックして、新しいタグ ソースを定義します。

タグ ソースの定義の詳細については、「[I/O を使用したデータ アクセス](#)」を参照してください。

3. [フィルタ] リストで、ウィンドウに表示されるタグ名の数減らすためのフィルタをクリックします。フィルタを定義するには、省略記号ボタンをクリックします。詳細については、「[タグ変数フィルタの作成](#)」を参照してください。

4. ウィンドウでタグ名を選択します。

[タグの選択] ダイアログ ボックスに表示されるタグ名の表示方法は変更できます。詳細については、「[\[タグの選択\] ダイアログ ボックスの表示の変更](#)」を参照してください。

5. [ドット フィールド] リストで、選択したタグ名に付加するドットフィールドをクリックします。

ドットフィールドは、タグ プロパティへのアクセス、監視、および変更を行えます。ドットフィールドを選択しない場合、.Value ドットフィールドが使用されます。

6. [OK] をクリックします。

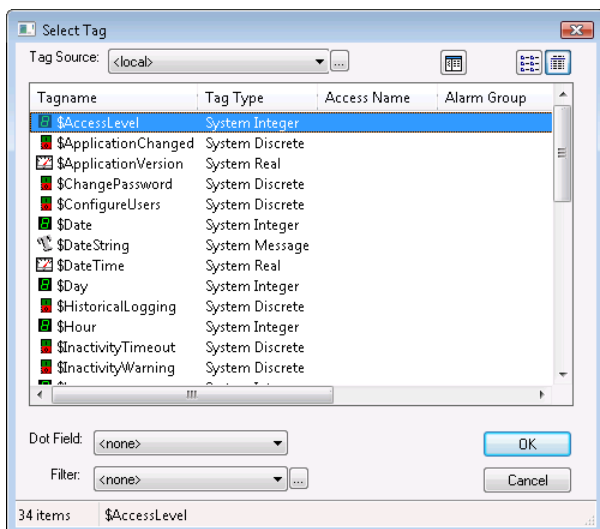
Application Server オブジェクト属性の選択

Application Server オブジェクトに関連付けられている属性を選択できます。これを行うには、最初にオブジェクトを InTouch アプリケーション用のデータ ソースとして含む **Galaxy** を追加します。データ ソースの詳細については、「[I/O を使用したデータ アクセス](#)」を参照してください。

オブジェクト属性を選択するには

1. 入力にタグ名が必要なテキスト ボックスをダブルクリックします。

[タグの選択] ダイアログ ボックスが表示されます。



2. [タグ ソース] リストで、Galaxy を使用するタグ ソースの名前をクリックするか、参照ボタンをクリックして、新しいタグ ソースを定義します。

[Galaxy ブラウザ] ダイアログ ボックスが表示されます。

3. [Galaxy ブラウザ] を使用し、オブジェクトの属性を検索して選択します。詳細については、Application Server のマニュアルを参照してください。
4. [OK] をクリックして、[Galaxy ブラウザ] を閉じます。

タグ名を必要とする属性参照がテキスト ボックス内に表示されます。

注記: [Galaxy ブラウザ] から [タグの選択] ダイアログ ボックスに戻るには、[Galaxy ブラウザ] の右下隅にある [戻る] ボタンをクリックします。

タグ変数フィルタの作成

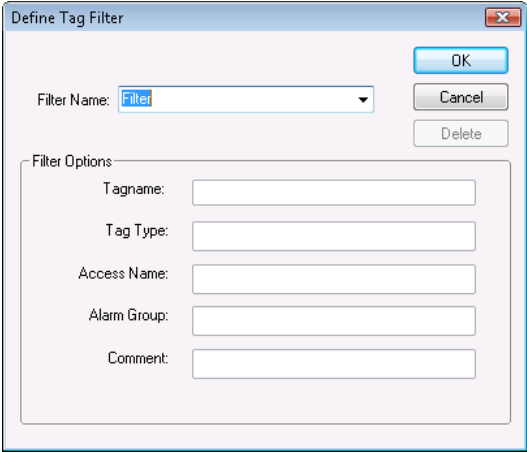
タグ変数ディクショナリ内に多数のタグ変数がある場合、アニメーション用のタグ変数を探すのは困難です。たとえば、特定のアクセス名またはアラーム グループに関連付けられているタグ変数のみを参照したい場合、フィルタを設定してタグ変数のサブセットのみが [タグ変数を選択してください] ダイアログ ボックスに表示されるようにすることができます。

フィルタには次のワイルドカード式を使用できます。

- 複数文字を表すワイルドカードには、アスタリスク (*) を使用します。たとえば、「Asyn*」と指定すると、「Asyn」の文字で始まるタグ変数がすべて検索されます。
- 単一文字を表すワイルドカードには疑問符 (?) を使用します。たとえば、「Tag?」というフィルタは、「Tag」で始まる 4 文字のタグ変数をすべて検索します。「Tag*」というフィルタは、「Tag」で始まるタグ変数をすべて検索します。
- フィルタには、タグ変数に有効な文字に加えて 2 つのワイルドカードを使用できます。有効なタグ変数文字は、A ~ Z、a ~ z、0 ~ 9、!、@、-、#、\$、%、_ および & です。

検索フィルタを定義するには

1. [タグ変数を選択してください] ダイアログ ボックスで、[フィルタ] リストの横にある参照ボタンをクリックします。[タグフィルタの定義] ダイアログ ボックスが表示されます。



- 2. [フィルタ名] ボックスに、フィルタ名を入力します。
- 3. [オプション] 領域で、フィルタ条件を設定します。以下のいずれかを実行します。
 - [タグ変数] ボックスに、タグ変数を入力します。
 - [タグ変数タイプ] ボックスに、タグ変数タイプを入力します。
 - [アクセス名] ボックスに、ローカルアクセス名を入力します。
 - [アラーム グループ] ボックスに、アラーム グループ名を入力します。
 - [コメント] ボックスに、コメント式を入力します。
- 4. [OK] をクリックします。フィルタ名が [タグ変数を選択してください] ダイアログ ボックスの [フィルタ] リストに表示されます。フィルタ条件に一致するタグ変数を表示するフィルタを選択できます。


検索フィルタを削除するには

- 1. [フィルタ名] ボックスで、フィルタをクリックします。
- 2. [削除] をクリックします。

[タグの選択] ダイアログ ボックスの表示の変更

[タグの選択] ダイアログ ボックスには、一覧ビュー、詳細ビュー、ツリー ビューという 3 つの異なるビューがあります。

表示するビュー	クリックするボタン	説明
一覧ビュー	一覧ビュー ボタン 	各タグ名のタイプに従って小さいアイコンがタグ名の横に表示されます。
詳細ビュー	詳細ビュー ボタン 	同じ小さいアイコンおよびタグ名とタグ タイプ、アクセス名、アラーム グループ、およびコメントが表示されます。リストをソートするにはカラム ヘッダーをクリックします。

表示するビュー	クリックするボタン	説明
ツリー ビュー	ツリー ビュー アイコン 	ツリー ビューは、タグ名を 2 つのビューで表示します。 スーパータグ テンプレート内のメンバー タグ名にアクセスできます。

キーボードショートカットの作成

キー割り当てリンクを使用して、特定のアニメーション リンクを有効にするには、キーボード上の特定のキーを割り当てることができます。割り当てたキーは、そのリンクを持つオブジェクトが表示または選択されている場合のみ機能します。オブジェクトに表示オン／オフや使用オン／オフのリンクを関連付けた場合、そのオブジェクトがオフの状態では割り当てたキーを使用することはできません。

同じキーは複数のウィンドウで定義できます。ただし、最後に開いたウィンドウでの定義がアクティブになります。オーバーレイ ウィンドウの場合には、一番上のウィンドウのキーがアクティブになります。

注記: アクティブ ウィンドウのオブジェクトや操作ボタンにキー アクション スクリプトで使用しているものと同じキーを割り当てる場合、アクティブ ウィンドウのキーに対するキー アクションの割り当てリンクがキー アクション スクリプトよりも優先されます。

割り当てられたキーをサポートしているアニメーション リンクの設定ダイアログには、**[キー アクションの割り当て]** 領域が表示されます。

キー リンクは、1 ～ 16 のファンクション キーでのみ利用できます。16 以上のファンクション キーを持つカスタム キーボードを使用している場合は、システムがその拡張ファンクション キーへアクセスするためのデバイス ドライバをキーボードの製造元から入手する必要があります。

リンクにキーを割り当てるには

1. 設定しているリンクのタイプに対する **[アニメーション リンク]** ダイアログ ボックスを開きます。
2. Ctrl キーか Shift キーを押しながらキーを押すように指定する場合は、**[Ctrl]** か **[Shift]** チェック ボックスをオンにします。
3. **[キー]** をクリックします。**[キーの選択]** ダイアログ ボックスが表示されます。
4. リンクに割り当てるキーをクリックします。

[アニメーション リンク] ダイアログ ボックスが再び表示され、選択されたキーの名前が **[キー]** ボタンの横に表示されます。

5. **[OK]** をクリックします。

タグ名参照の変更

オブジェクトを複製すると、リンク、アニメーション、スクリプトなどを含む元のオブジェクトと全く同じものが作成されます。ただし、複製したオブジェクトに別のタグを使用する必要がある場合には、タグを置換できます。

オブジェクトのタグは選択して置換できます。複数のオブジェクトを選択し、そのすべてのタグを一度に置換することもできます。

システムのライセンスによりタグ名の数制限されている場合には、ローカルタグをリモートタグ参照に変換して、ローカルタグ名ディクショナリで定義されているタグの数を減らすことができます。

タグの置換はすべてのタグおよび参照に有効です。

タグを別のタグと置換するには

1. 別のタグに変更するタグに関連付けられているオブジェクトを選択します。
2. [アニメーション] メニューの [置換] グループで [タグ] をクリックします。
[タグ名の置換] ダイアログボックスが表示されます。
3. [新しい名前] ボックスに、新しいタグ名を入力します。
 - [新しい名前] ボックスのタグをダブルクリックすると、タグ名ディクショナリにそのタグの定義が表示されます。
 - タグ名を消去した後に空白のボックス内をダブルクリックすると、[タグの選択] ダイアログボックスが表示されます。
4. [OK] をクリックします。

オブジェクトに関連付けられているタグが自動的に変更されます。

プレースホルダタグの変換

ウィンドウまたは QuickScript を現在のアプリケーション間でインポートまたはエクスポートする場合、そのウィンドウまたは QuickScript に関連付けられているすべてのタグがウィンドウと一緒に転送されます。

ローカルタグは、自動的にアプリケーションデータベースに追加されません。その代わりに、これらのタグは自動的にプレースホルダタグとしてマーク付けされます。

リモートタグ参照は影響を受けず、プレースホルダタグとしてマークされません。

プレースホルダタグを既存のローカルタグに変換して、新しいアプリケーションのタグ名ディクショナリに定義するか、またはリモートタグ参照にする必要があります。

タグの前には、?d:、?i:、?m:、?r: などのプレースホルダが付いていることに注意してください。これらのプレースホルダは、最初に定義した際のタグのタイプを以下のように表しています。

プレースホルダ シンボル	タグ名 タイプ
d	論理型
i	整数型
m	メッセージ型
r	実数型

注記: リモートタグ参照は、プレースホルダとして表示されませんが、**PLC2:Temperature** のようにリモートタグ参照として表示されます。

[タグ名の置換] ダイアログボックスでは、複数の方法を使用してプレースホルダタグをローカルタグに変換できます。詳細については、『AVEVA™ InTouch HMI アプリケーションランタイムガイド』の「[InTouch コンポーネントのエクスポートとインポート](#)」を参照してください。

ヒント: タグ名を手動で変換し、元のタグ名ディクショナリで定義されている元のタグを必要がなくなった場合は、タグ名の使用カウントを更新して、使用されていないタグを削除できます。

ウィンドウまたは **QuickScript** を別のアプリケーションからインポートし、アニメーションリンクまたは **QuickScript** に関連付けられているタグ名をすべてリモートタグ名参照に変換すると、ローカルのタグ名ディクショナリでタグを全く定義しない場合でも数多くのリモートタグ名からデータをすぐに受け取ることができるようになります。

テキストの詳細な書式設定

InTouch WindowMaker を使用して、InTouch WindowViewer がランタイム時に使用するインテリジェントな書式設定または詳細な書式設定を設定することができます。これらのオプションは、アナログ値の表示またはアナログユーザー入力アニメーションに対して詳細な書式設定を使用している場合にのみ適用されます。詳細な書式設定オプションを設定していない場合、InTouch WindowViewer の現在値が保持されます。

詳細な書式設定オプションは、[アナログデータ入力リンク] ダイアログボックスと [アナログ値表示リンク] ダイアログボックスに表示されます。

数値の詳細な書式設定により、データのタイプと品質に基づいて数値データをネイティブな方法で書式設定することができます。詳細な書式設定オプションでは、ランタイム時に InTouch WindowViewer でアナログ値の書式設定を行うことができます。

個々のアプリケーションに対してテキスト文字列を使用するか、または詳細な書式設定オプションを有効にすることができます。旧バージョンの InTouch からアニメーションを移行するか、またはアニメーションを初めて作成する場合、デフォルトとしてテキスト文字列が設定されます。

詳細な書式設定オプションは次のとおりです。

- テキスト文字列。既存の InTouch 書式設定を使用します。
- 実数。ランタイム時の値のサイズに基づいた実数として書式設定されます。グローバルな小数点の精度の設定は、小数点以下の桁数を設定します。
- 固定小数点の書式は、値のタイプによって決まります。
 - 実数値: 値は [精度] コントロールを使用して、ユーザーが指定した小数点の桁数で書式設定されます。
 - 整数値: 値は小数点のない整数として書式設定され、小数点のあるべき場所に合わせて右揃えされます。
 - 論理値: 値は小数点のない 0 または 1 で書式設定され、小数点のあるべき場所に合わせて右揃えされます。
- 整数。常に少数点のない整数として書式設定されます。
- 指数。常に固定少数点を配慮した指数として書式設定されます。
- 16 進数値。設定したブール値の範囲に基づいた 16 進数で書式設定されます。
- バイナリ。常に設定したブール値の範囲を配慮したバイナリ文字列として書式設定されます。

[固定幅] ボックスでは、デザイン時に指定したテキストサイズを大きくすることができます。このオプションは、書式設定のスタイルに [テキスト文字列] が選択されている場合には利用できません。

デザイン時に、アニメーションの設定で [固定幅] チェックボックスをオンにした場合、ランタイム時に表示される値はテキストフィールドの長さより長くなりません。値がテキストフィールドの長さを超えた場合、グローバルの [長すぎる値の固定フィールド幅] の設定がテキストフィールドの長さを決

めます。固定幅の書式設定は、ランタイム時に値のサイズに合わせて、テキスト要素の幅が広がることを防ぎます。

デザイン時にテキストフィールドにテキストを何も指定しなかった場合、固定幅の設定は無視され、値全体が表示されます。たとえば、デザイン時に "" (テキストなし) と入力した場合、テキストフィールドの幅は 0 に相当します。この状態で **[固定幅]** オプションが有効になっていると、テキストは何も表示されなくなります。この場合、**[固定フィールドの幅]** のランタイム時のデフォルトは、「なし」に設定されます。

[クリア] をクリックすると、詳細な書式設定オプションは、次のようにリセットされます。

- **[書式設定]** オプションには、**[テキスト文字列]** が選択されます。
- **[固定フィールドの幅]** オプションはクリアされ、無効になります。**[テキスト文字列]** の書式設定が選択されているときには、このオプションは利用できないからです。

デフォルトにより、次の値が設定されます。

- 固定少数点の精度には 0 が設定されます。
- 指数の精度には 0 が設定されます。
- 16 進数のビット範囲には 0 ～ 31 が設定されます。
- バイナリのビット範囲には 0 ～ 31 が設定されます。

書式設定のコントロール:

- 精度コントロールは整数値を受け付けます。**[固定少数点]** または **[指数]** のオプションを選択した場合にのみ有効になります。このコントロールの整数値の範囲は 0 ～ 8 で、デフォルト値は 0 です。ビット範囲は 2 つのスピン コントロールで構成され、それぞれが受け入れる入力値は整数です。これらのコントロールは、**[16 進数]** または **[バイナリ]** オプションを選択した場合にのみ使用可能になります。これら 2 つのコントロールには、0 ～ 31 の範囲のビット値を指定する必要があります。ビット範囲は、正順または逆順で指定することができます。範囲と同じビット仕様を使用して、単一のビットを指定することもできます。ビット仕様に対するデフォルトの数値範囲は 0 ～ 31 です。範囲は 0 に基づいており、InTouch でビットが指定されているほかの場所でも一致します。

設定済みのアナログ ユーザー入力アニメーションまたはアナログ値の表示アニメーションを含むバージョン 3.5 以前の InTouch の場合、移行したアニメーションには **[テキスト文字列]** 書式設定が設定され、詳細な書式設定はオフになります。

最新バージョンの場合、詳細な書式設定オプションは移行されます。

数字としてのテキスト オブジェクトの書式設定

テキスト オブジェクトを使用して、静的または動的な数値を表示できます。**[値の表示]** - **[アナログアニメーション]** リンクをテキスト オブジェクトに関連付けることによってアナログ タグの整数値または実数値を表示できます。

テキスト オブジェクトのアニメーションによって表示されるアナログ タグ値の書式を指定するには、以下の文字を使用します。

文字	説明
#	アナログ値の表示を整数に強制します。
0	実数の指定した各場所の桁を表します。数字の整数部分に先行ゼロを適用します。

文字	説明
,	実数の指定した各場所にカンマを挿入します。
.	実数の指定した場所に小数点を配置します。

フォント、サイズ、および色などのその他のテキスト書式設定プロパティは、文字列内に表示されている数値に適用されます。

数字としてのテキスト オブジェクトの書式設定の例

アニメーション化された書式設定文字がテキスト オブジェクト内で使用された場合のアナログ値の表示例を次の表に示します。これらの例では、温度 I/O タグの現在の値は **123.45 C°** です。このタグは、アナログ アニメーション リンクを使用して、テキスト オブジェクトの参照値として割り当てられています。

書式設定されたテキスト オブジェクト	テキスト オブジェクトの表示値
Temp = # C	124 C 温度を整数として表示します。テキスト オブジェクトに入力する必要のある # は 1 つだけです。
Temp = #.# C	123.5 C 「#」 文字を使用して、温度の小数部分が 1 桁の数字に丸められます。温度の整数部分は変更されません。
Temp = 00000 C	00124 C 温度の整数部分に先行ゼロを追加します。
Temp = 000.0 C	123.5 C 書式内の 1 つのゼロによって温度の小数部分が 1 桁の数字に丸められます。
Temp = 00.00 C	123.45 C 元の温度値は変更されません。
Temp = 0,000.#	0,123.5 C 温度参照値の整数部分に先行ゼロとカンマが強制されます。「#」 文字を使用して、小数部分が 1 桁の数字に丸められます。

数字としてのテキスト オブジェクトの書式設定を指定するには

1. ウィンドウでテキスト オブジェクトを作成して、テキスト オブジェクトに関連付けられた参照値の書式を設定する文字を挿入します。
2. テキスト オブジェクトをダブルクリックして、[アニメーション リンク] 選択ダイアログ ボックスを表示します。

3. [値の表示] セクションで [アナログ] をクリックします。[アナログ値出力] ダイアログボックスが表示されます。

4. [タグ変数/式] フィールドにアナログ タグ名または式を入力します。
5. [OK] をクリックします。
6. ウィンドウの変更を保存します。
7. アプリケーションを実行して、テキストオブジェクト内の数字が正しく表示されることを確認します。

産業用グラフィック エディタの操作

産業用グラフィック エディタは、産業用グラフィックを作成するために使用するツールです。エディタの使用方法については、『産業用グラフィック エディタ ユーザー ガイド』を参照してください。次のセクションでは、産業用グラフィック エディタと共に使用する InTouch HMI 固有の構成または設定を紹介します。

産業用グラフィックの管理

産業用グラフィックは、HMI/SCADA システムのデータを視覚化するために作成できるグラフィックです。

産業用グラフィック エディタを使用して、長方形、線、テキスト要素などの基本要素から産業用グラフィックを作成します。また、産業用グラフィック エディタを使用して、グラフィックのグラフィック ツールボックス ライブラリから産業用グラフィックを埋め込んで設定することもできます。

産業用グラフィック エディタは、産業用グラフィックとしてのスケーラブルベクターグラフィックス (SVG) のインポートをサポートしています。以下の操作を行うことができます。

- SVG を産業用グラフィック エディタにインポートする。グラフィック要素は、多くのプリミティブを含むことのできる産業用グラフィックに自動的に変換されます。
- 産業用グラフィック エディタの [ツール] パネルの [画像] アイコンを使用して SVG を挿入する。
- ボタンの **UplImage** と **DownImage** に SVG を使用する。
- [スタイル] 設定のクォリティとステータスの表示アイコンに SVG を使用する。

SVG および SVG のインポートに関する制限の詳細については、産業用グラフィック エディタ ヘルプの「産業用グラフィックとしての SVG のインポート」セクションを参照してください。

作成した産業用グラフィックは、別のグラフィックまたは HMI システム ウィンドウに埋め込んで、ランタイム時に使用できます。

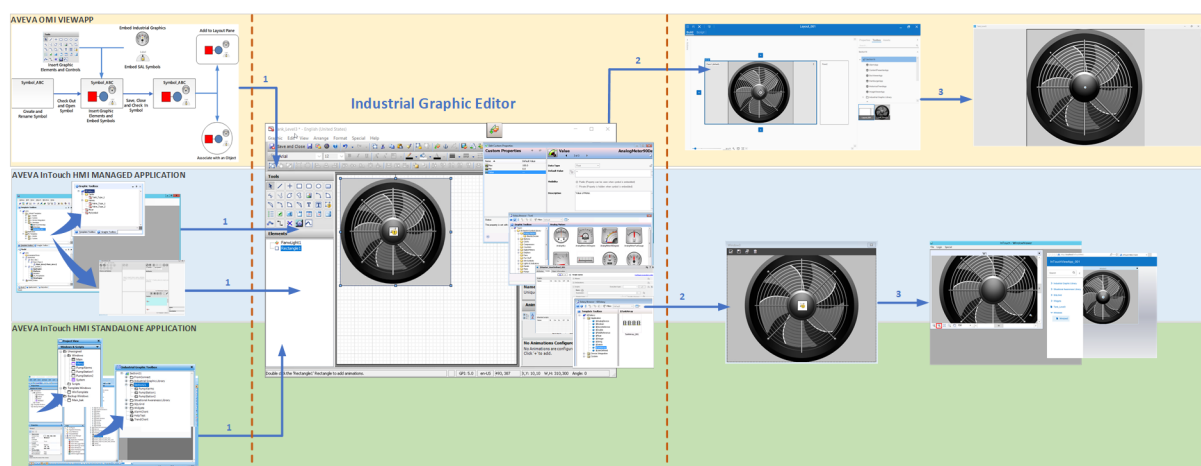
産業用グラフィックをテンプレートまたはオブジェクトのインスタンスに埋め込むことによって、オブジェクト固有の情報をすばやく簡単に視覚化することが可能です。テンプレートにグラフィックを埋め込むと、1つのグラフィックを更新するだけで、その変更をアプリケーション全体に適用できます。

開発要件に応じて、産業用グラフィックを保存する場所と方法を選択できます。

- 作成したグラフィックを再使用可能な標準セットとして保存します（汎用バルブのグラフィックなど）。
- ランタイム時にグラフィックを複数のインスタンスで使用する場合、グラフィックをテンプレートとして保存します。
- 特定のアプリケーションで使用する目的でグラフィックを保存します。

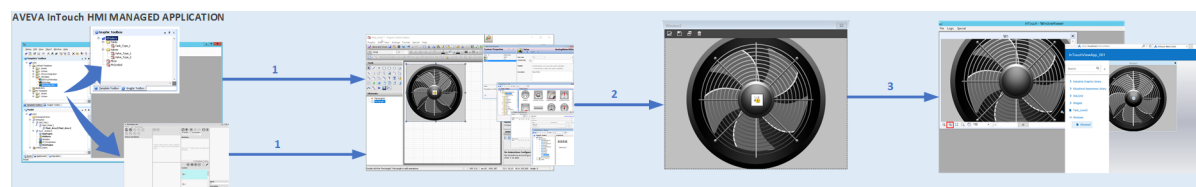
HMI/SCADA アプリケーションと共に使用する産業用グラフィック エディタ ワークフロー

産業用グラフィック エディタは、さまざまな HMI/SCADA アプリケーションで使用する標準のグラフィック エディタです。産業用グラフィック エディタを使用すると、工場作業場のアセットを表すグラフィックを作成して HMI/SCADA アプリケーションに埋め込むことができます。以下のセクションでは、全体的なワークフローについて説明します。

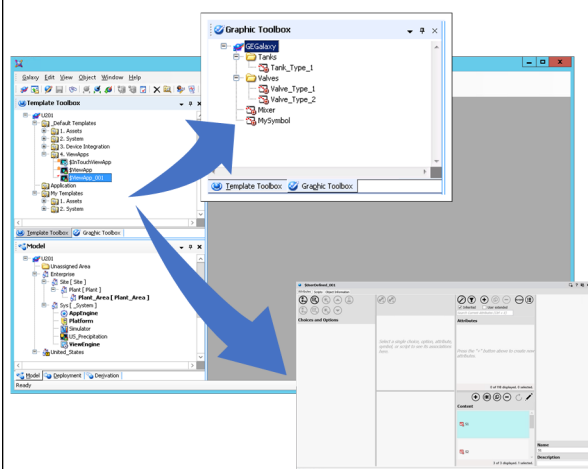


InTouch Managed Applications

The workflow for InTouch HMI managed applications starts with the IDE.



Step 1



In the IDE:

From the Visualization folder, create a new graphic or select an existing graphic.

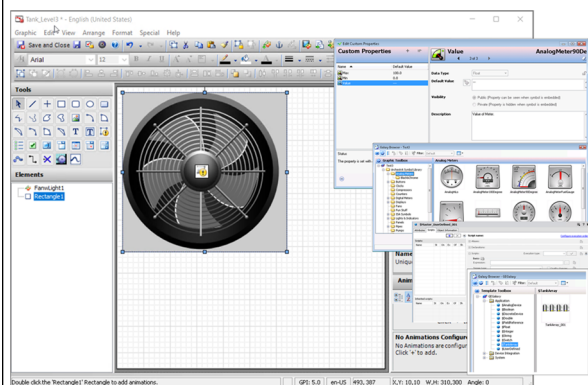
- Open the graphic to launch the Industrial Graphic Editor.

- or -

Open a \$UserDefined object in the Template folder.

1. From the Content pane, click the + button to add a graphic to the object.
2. Click the **Edit Content** button to launch the Industrial Graphic Editor.

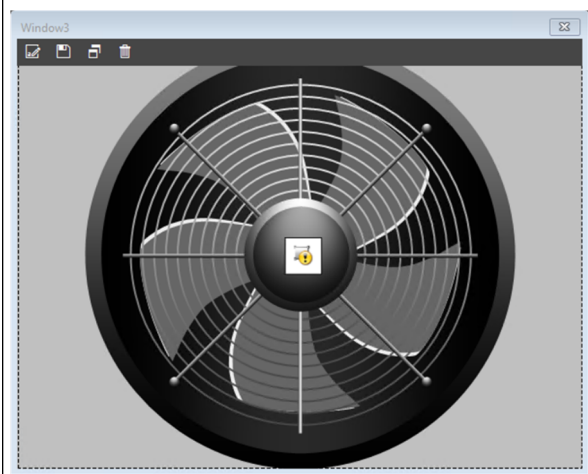
Step 2



Edit the graphic in the Industrial Graphic Editor. Modify graphical elements as needed.

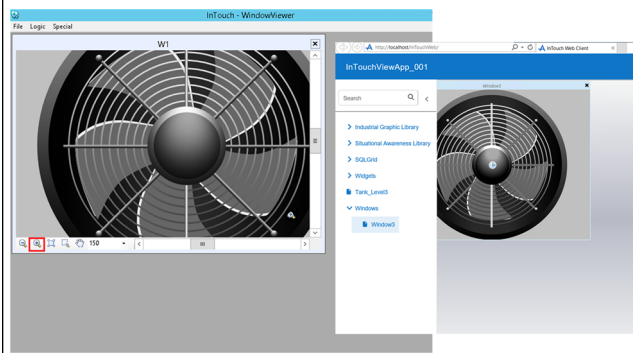
- Insert graphic elements and embed graphics.
- Add/modify scripts, references, custom properties, and animations to enable the graphic to represent the asset, and to respond to commands and data changes at run time.

Step 3



Embed the graphic in a Window.

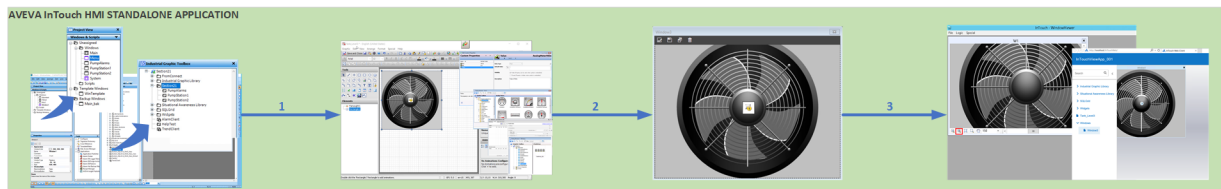
Step 4



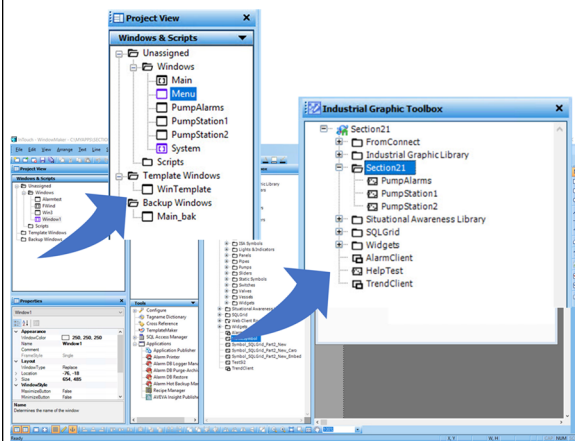
To view the graphic in runtime, launch WindowViewer or the Web Client.

InTouch スタンドアロンアプリケーション

InTouch HMI スタンドアロンアプリケーションのワークフローは WindowMaker から始まります。



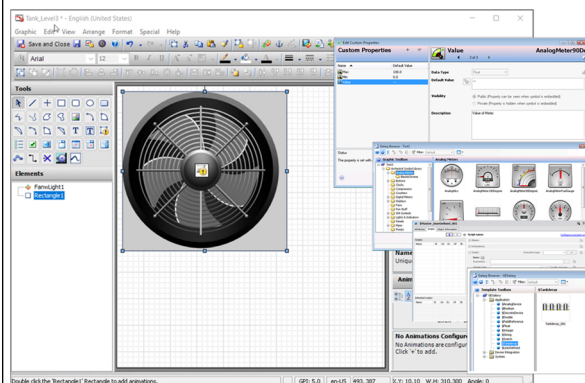
手順 1



InTouch HMI WindowMaker で産業用グラフィック エディタを使用してシンボルを作成します。

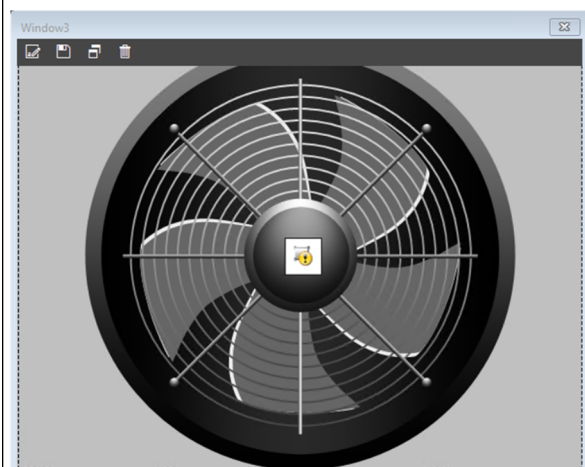
グラフィックを編集して産業用グラフィック エディタを起動します。

手順 2



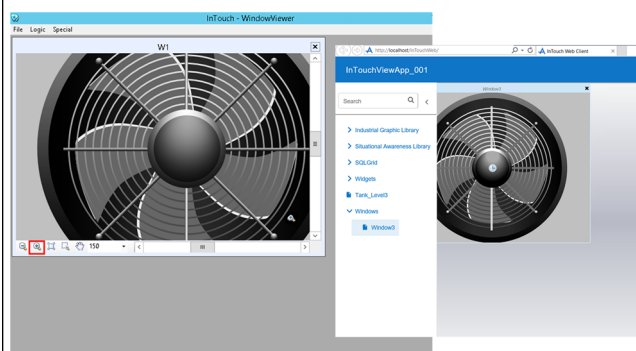
産業用グラフィック エディタでスクリプト、カスタムプロパティ、タグとオブジェクト参照、スタイル、ツール、アニメーション、および書式設定を使用してグラフィックを構築します。

手順 3



グラフィックをウィンドウに埋め込みます。

手順 4



ランタイム時にグラフィックを表示するには、WindowViewer または Web Client を起動します。

グラフィックを開いて編集する

グラフィックから産業用グラフィック エディタを起動できます。

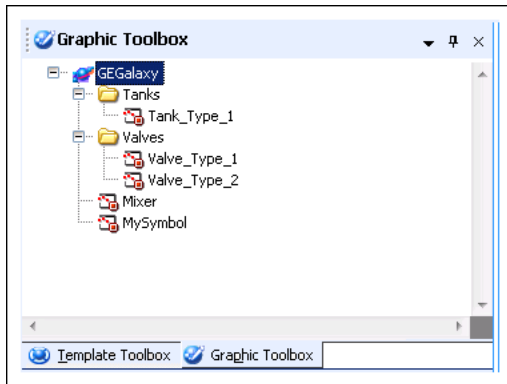
- グラフィック ツールボックス内のシンボル。
- 自動オブジェクト テンプレートまたはインスタンスに含まれるシンボル。
- InTouch ウィンドウに埋め込まれたシンボル。

産業用グラフィック エディタで編集目的で開いたグラフィックはチェックアウトされます。他のユーザーはチェックアウトされているグラフィックを編集できません。

産業用グラフィック エディタの複数のインスタンスを同時に開くことができます。しかし、産業用グラフィック エディタの複数のインスタンス中で同じグラフィックを編集することはできません。

グラフィック ツールボックス内のグラフィックを編集するには

1. グラフィック ツールボックスを開きます。
2. 編集するグラフィックを参照します。



3. グラフィックをダブルクリックします。産業用グラフィック エディタが表示されます。
4. グラフィックを編集します。
5. [保存して閉じる] をクリックします。産業用グラフィック エディタが閉じ、更新されたシンボルがチェック インされます。

自動オブジェクトに含まれるシンボルを編集する方法

1. 自動オブジェクトを開きます。
2. [グラフィック] タブをクリックします。
3. 編集するシンボルを選択し、[開く] をクリックします。産業用グラフィック エディタが表示されます。
4. シンボルを編集します。
5. [保存して閉じる] をクリックします。産業用グラフィック エディタが閉じ、更新されたグラフィックがチェック インされます。オートメーションオブジェクトによっては、確認メッセージが表示されることがあります。[はい] をクリックすると、保存されます。

InTouch ウィンドウに埋め込まれているグラフィックを編集するには

1. 埋め込まれたグラフィックを含む InTouch ウィンドウを WindowMaker で開きます。
2. 編集するグラフィックを右クリックし、[産業用グラフィック "グラフィック名"] をポイントし、[シンボルの編集] をクリックします。産業用グラフィック エディタが表示されます。
3. グラフィックを編集します。
4. [保存して閉じる] をクリックします。産業用グラフィック エディタが閉じ、更新されたグラフィックがチェック インされます。

注記: グラフィックをチェックアウトしたままにするには、産業用グラフィック エディタで **[チェックアウト状態を維持]** をクリックします。このオプションを使用すると、他のユーザーはそのグラフィックを編集用にチェックアウトできなくなります。

シンボル変更の反映

ソースの産業用グラフィックへのすべての変更は、埋め込まれたすべての産業用グラフィック **ArchestraA** シンボルに反映されます。これは、**WindowMaker** の産業用グラフィック、およびオートメーション オブジェクトによって継承された産業用グラフィックに影響します。

産業用グラフィックに変更が加えられ、開いている **InTouch** ウィンドウでそれが使用されている場合、ステータス バーの右下隅に **[シンボルの変更]** アイコンが表示されます。

このアイコンをダブルクリックすると、埋め込まれた産業用グラフィックで変更が更新されます。

以下の例は、シンボルの変更の反映がどのように行われるのかを示しています。

シンボルの変更を反映するには

1. [「新しい Application Server オブジェクトインスタンスの自動作成」](#) の例に従います。
2. **WindowMaker** で、バルブのシンボルが表示されているウィンドウを開きます。
3. **Application Server** オブジェクト テンプレート **\$Valve1** によってホストされている産業用グラフィック **ValveSymbol** を開きます。
4. いくつかの変更を加え、**[閉じて保存]** をクリックします。**Application Server** オブジェクト インスタンス **Valve1_E22** に変更が反映されます。**WindowMaker** で、**[シンボルの変更]** アイコンが表示されます。
5. アイコンをダブルクリックして、変更を受け入れます。埋め込まれた産業用グラフィックが更新されます。

シンボルのダイナミック サイズの反映

埋め込まれたシンボルにソース シンボルのサイズの変更が反映される方法を制御できます。たとえば、サイズの変更には以下のようなものがあります。

- シンボルの境界が変更されるように、ソース シンボルの要素の 1 つをサイズ変更する。
- シンボルの境界が変更されるように、ソース シンボルに要素を追加するか、ソース シンボルから要素を削除する。

この機能はダイナミック サイズ変更の反映と呼ばれ、有効化または無効化できます。

ダイナミック サイズの反映の詳細については、『産業用グラフィック エディタ ユーザー ガイド』を参照してください。

WindowMaker と産業用グラフィック エディタの比較

産業用グラフィック エディタを使用すると、**InTouch WindowMaker** での大部分のタスクを実行できます。また、多くのショートカット キーも使用できます。

産業用グラフィック エディタには、**InTouch WindowMaker** で使用できない次のような機能が備わっています。

- 追加の要素。

- 追加および改良された要素の外観。
- 追加および改良されたデザイン時の機能。

外観

産業用グラフィック エディタは、InTouch グラフィック設定を拡張します。たとえば、以下を使用できます。

- 線、塗りつぶし、テキストの色のグラデーション。
- 線、塗りつぶし、テキストの色のパターン。
- 線、塗りつぶし、テキストの色のテクスチャ。
- 調節可能な透明度
- シンボルまたは画面に相対的な塗りつぶし動作

要素

要素は産業用グラフィックを作成するために使用するグラフィック オブジェクトです。産業用グラフィック エディタには、InTouch WindowMaker で使用できない次のような要素が備わっています。

- 曲線および閉曲線。
- 2、3 個の点で定義される円弧、パイ、弦。
- 属性データのクォリティとステータスに応じてアイコンを表示するステータス要素。
- 線ベースの要素を組み合わせで新しい閉じた要素を形作ることで作成するパス グラフィック。
- カレンダー コントロール、日付時間ピッカー コントロールなどの Windows 共通コントロール。

改良された機能

産業用グラフィック エディタは、製造環境を表示することで、生活がより便利になるような、さまざまな改良を提供します。

ユーザビリティの向上

産業用グラフィック エディタを使用すれば、要素を容易に選択および設定できます。以下の操作を実行できます。

- リストとキャンバスから要素を選択できます。これにより、他の要素を移動することなく、他の要素の下にある要素を選択できます。
- キャンバスで選択するだけで、要素のプロパティとアニメーションを表示および変更できます。
- グループまたはパス グラフィックを解除することなく、グループとパス グラフィック中の要素を編集できます。これをインライン編集といいます。

スタイルの複製

書式のコピーを利用すれば、1 回のクリックで、要素のスタイルを他の要素に適用できます。異なるタイプの要素に対しても可能です。

アニメーションの複製

産業用グラフィック エディタを使用すれば、要素のアニメーションのコピー、切り取り、他の要素への貼り付けを行うことができます。これらの操作は、別のタイプの要素に対して行うこともできます。

要素の配置

産業用グラフィック エディタは、InTouch WindowMaker の配置機能を拡張し、以下の操作が可能になります。

- 水平方向または垂直方向に要素を等間隔で分布する。
- 要素の水平方向および/または垂直方向のサイズを等しくする。
- 水平方向または垂直方向のスペースを増加または減少する。
- 要素間の水平方向または垂直方向のスペースを削除する。
- 要素を誤って移動または編集しないようにロックする。
- デザイン時に回転中心の周りで角度を指定して要素を回転する。
- 同時に複数要素に対してサイズ変更と回転を適用する。
- 要素の Z 順序を、前後方向に 1 段階変更する。
- テキスト ボックスまたはボタン内のテキストを整列する。

グループ機能

産業用グラフィック エディタは、InTouch WindowMaker のセルとシンボルの概念の代わりにグループの概念を使用します。以下の操作を実行できます。

- 要素をグループに埋め込む。
- グループを解除することなく、グループ中の各要素 (または埋め込まれたグループ) を編集する。
- 既存のグループにおいて要素を容易に削除または追加する。

カスタム プロパティの拡張性

シンボルまたは埋め込まれたシンボルに対し、カスタム プロパティを追加できます。カスタム プロパティを **AutomationObject** 属性、要素プロパティ、InTouch タグに関連付けできます。デザイン時とランタイム時に定義済みプロパティを使用すると同じように、カスタム プロパティを使用できます。

注記: ハイフンを含む名前の InTouch タグを参照している **Application Server** カスタム プロパティは、ランタイム時に動作しません。たとえば、"InTouch:TAG-1" はランタイム時に動作しません。

要素スタイル

要素スタイルはグラフィック要素における 1 つまたは複数の塗りつぶし、線、テキスト、点滅、輪郭、ステータスのプロパティを定義します。要素スタイルをグラフィック要素に適用し、要素スタイルで定義された事前設定プロパティに要素を設定します。要素スタイルで定義された要素のローカルプロパティは無効になります。

要素スタイルは、スクリーン作成に関するドライブ標準を確立する場合、およびシンボルを作成する場合に役立ちます。

その他の改良点

産業用グラフィック エディタを使用すると、以下のことが可能になります。

- スクリプトによって、要素のプロパティとシンボルのカスタム プロパティにアクセスする。
- 要素のタブ順序を設定する。
- 矢印など、線の端点スタイルを使用する。

- 設定情報を失うことなく、要素の固有のアニメーションをダイナミックに無効にする。
- 画像のメタファイルと他の画像形式を使用する。
- シンボルの表示方法を向上させるためにアンチエイリアス機能を利用する。

一般的な WindowMaker のタスクと技術の手順

InTouch WindowMaker で実行する大部分の設定は、産業用グラフィック エディタで容易に実行することができます。グラフィック、アニメーション、スクリプトにはいくつかの相違点があります。

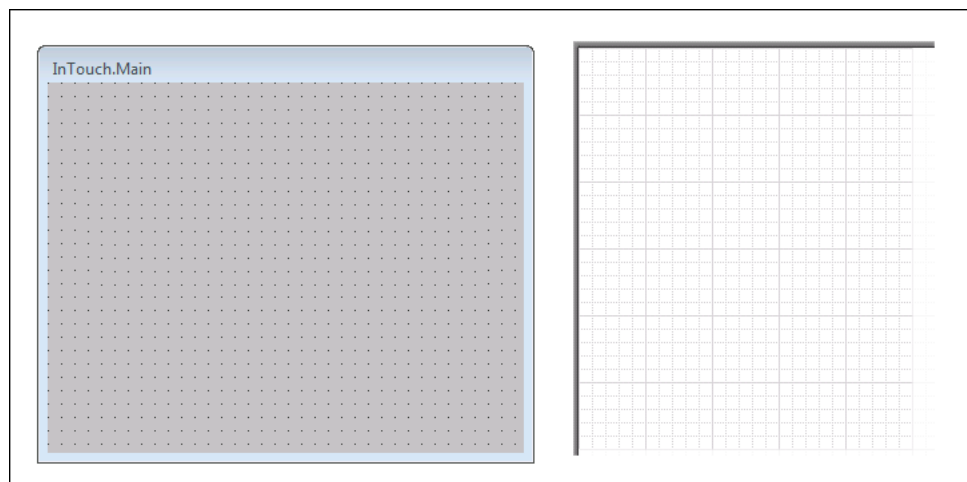
グラフィックの使用

基本的に、産業用グラフィック エディタは InTouch WindowMaker と同じように使用できます。産業用グラフィック エディタに含まれる描画領域では、グラフィック オブジェクトを配置して、生産プロセスの視覚的な表現を作成し、人と機械とのインターフェースを作成できます。

InTouch で使用するオブジェクト（ActiveX コントロールやいくつかのウィザードなど）には、産業用グラフィック エディタに存在しないものがあります。それらの機能は、より強力で密接に統合できる他のコントロールに置き換えられています。

描画領域の使用

産業用グラフィック エディタの描画領域はキャンバスと呼ばれます。キャンバスは InTouch のウィンドウのように使用できます。そのサイズは 2,000 × 2,000 ピクセルです。



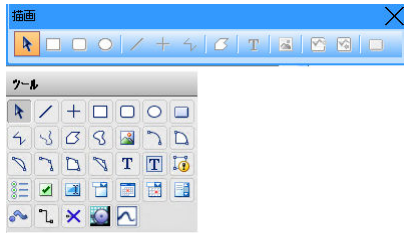
描画領域の色の設定

産業用グラフィック エディタで描画領域の色は設定できません。描画領域の色は透明で、シンボルが埋め込まれた環境の色を継承します。

産業用グラフィックを InTouch のウィンドウに埋め込んだ場合、要素間の領域には、InTouch のウィンドウの色が使用されます。

基本的なオブジェクトの使用

長方形、楕円、折れ線などの InTouch の基本的なオブジェクトは産業用グラフィック エディタと同じように描画できます。産業用グラフィック エディタでは、基本的なオブジェクトは要素と呼ばれます。



複雑なオブジェクトの使用

ActiveX コントロール、ウィザード、セル、シンボルのような InTouch オブジェクトは産業用グラフィック エディタに存在しません。

しかし、スマート シンボルを産業用グラフィックにインポートできます。スマート シンボルをインポートする場合、スマート シンボルの要素とアニメーションが変換されます。

産業用グラフィック エディタでは、要素のグループを作成できます。グループ内では、含まれている個別の要素のプロパティは維持されます。グループの **TreatAsIcon** プロパティを設定すれば、グループの動作を変更できます。

ウィザードの使用

InTouch ウィザードは産業用グラフィックおよびグラフィック ツールボックスにインポートできません。その代り、以下を使用します。

- グラフィック ツールボックスにインポートできる産業用グラフィック ライブラリー
- ツールボックスの一部である **Windows** コントロール以下を使用できます。
 - ラジオ ボタン グループ
 - チェックボックス
 - 編集ボックス
 - コンボ ボックス
 - カレンダー コントロール
 - 日付時間ピッカー
 - リスト ボックス

アニメーションの使用

産業用グラフィック エディタでアニメーションを使用して、InTouch WindowMaker と同様に、ランタイム中のシンボルの動作を設定できます。要素またはシンボルに 1 つまたは複数のアニメーションを設定できます。データはさまざまなソースに由来します。

データソースの設定

InTouch WindowMaker では、タグ名ディクショナリを使用して値を保持する変数を定義します。産業用グラフィック エディタのデータ ソースを以下に示します。

- オートメーション オブジェクトの属性
- シンボルのカスタム プロパティおよび継承されたプロパティ

- InTouch タグ(産業用グラフィック エディタでは、InTouch タグに直得接続できる特殊な InTouch 参照が使用されます)

データ型の使用

産業用グラフィックは InTouch のデータ型とは別のデータ型を使用します。

以下の表には、両方のデータ型と相互の対応関係が示されています。

InTouch	Application Server	説明
論理型	ブール型	ブール型。例：0 または 1。
整数型	整数型	整数値。例：-4、7、22 など。
実数型	浮動小数点または倍精度	<p>精度の異なる浮動小数点値または倍精度値。 例：3.141、-5.332、1.343e+17 など。</p> <p>浮動小数点: 32 ビット。IEEE 単精度浮動小数点。6 ～ 7 桁の有効桁が必要な場合に使用します。デフォルトは NAN です。</p> <p>倍精度: 64 ビット。IEEE 倍精度。15 ～ 16 桁の有効桁が必要な場合に使用します。デフォルトは NAN です。</p>
メッセージ型	文字列型	文字列値。例："Hello World"。
該当なし	日付時刻	日付時刻値。例："04/13/2006 04:03:22.222 AM"
該当なし	経過時間	<p>経過時間を秒単位で表す浮動小数点値。以下の形式で表示されることがありますが、浮動小数点値で保存されます。</p> <p>[-][DDDDDD] [HH:MM:]SS[.ffffff]</p> <p>値を以下に示します。</p> <ul style="list-style-type: none"> • DDDDDD: 0 ～ 999999 • HH: 0 ～ 23 • MM: 0 ～ 59 • SS: 0 ～ 59 • fffffff は小数点の右側にある小数部分を示します。 <p>経過時間は正の値にも負の値にもなります。</p>
該当なし	国際化された文字列	特殊文字を保存できる特殊な文字列データ型。

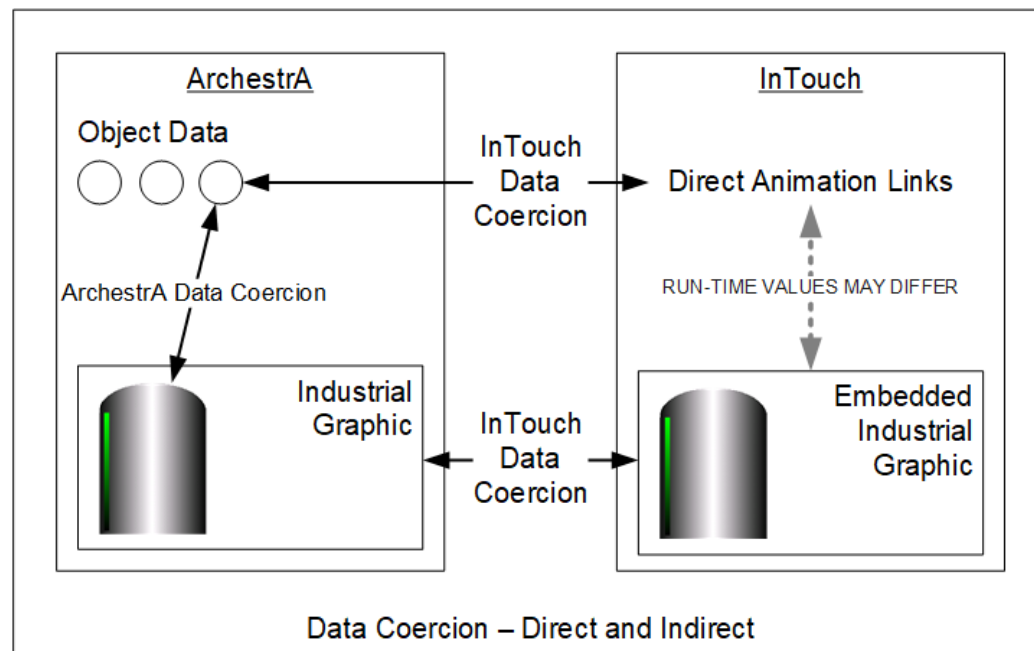
産業用グラフィックは、Galaxy からデータを取得するように設定できます。

使用するデータ型とソース データ型が異なる場合、データは IDE の強制変換のルールに基づいて強制変換され、アニメーション中では "-10" の文字列値は "True" に強制変換されます。

この産業用グラフィックを InTouch のウィンドウに埋め込んだ場合、アニメーションリンクのデータ型は InTouch の強制変換の規則に基づいて強制変換されます。埋め込まれた産業用グラフィックは InTouch HMI で "True" と表示されます。

しかし、InTouch HMI でオリジナルデータソースをポイントする論理型アニメーション表示リンクを直接作成する場合、結果の値が異なる場合があります。

この例では、文字列値 "-10" は InTouch HMI で "False" と表示されます。



アニメーションの使用

InTouch のアニメーションを設定するには、[アニメーションリンク] ダイアログボックスを使用します。このダイアログボックスを開くには、InTouch オブジェクトをダブルクリックします。

産業用グラフィックエディタのアニメーションを設定するには、通常は要素をダブルクリックすることで表示される [アニメーションの編集] ダイアログボックスを使用します。

いくつかのアニメーションタイプは異なっており、他のものは設定を単純化するためにグループ化されています。以下の表を使用して、産業用グラフィックエディタの同等のアニメーションタイプを見つけてください。

InTouch アニメーション	産業用グラフィックエディタアニメーション
ユーザー入力 - 論理型	ユーザー入力 - ブール型
ユーザー入力 - アナログ型	ユーザー入力 - アナログ型
ユーザー入力 - 文字列	ユーザー入力 - 文字列
スライダー - 垂直	垂直スライダー
スライダー - 水平	水平スライダー

InTouch アニメーション	産業用グラフィック エディタ アニメーション
タッチ押しボタン - 論理値	押しボタン - ブール値
アクション	アクション スクリプト
ウィンドウの表示	(サポートされていません)
ウィンドウの非表示	(サポートされていません)
線の色 - 論理型	線スタイル - ブール型
線の色 - アナログ型	線スタイル - 真理値表
線の色 - 論理型アラーム	線スタイルに変換
線の色 - アナログ アラーム	線スタイルに変換
塗りつぶしの色 - 論理型	塗りつぶしスタイル - ブール値
塗りつぶしの色 - アナログ型	塗りつぶしスタイル - 真理値表
塗りつぶしの色 - 論理型アラーム	塗りつぶしスタイルに変換
塗りつぶしの色 - アナログ アラーム	塗りつぶしスタイルに変換
テキストの色 - 論理型	テキスト スタイル - ブール値
テキストの色 - アナログ型	テキスト スタイル - 真理値表
テキスト色 - 論理型アラーム	テキスト スタイルに変換
テキスト色 - アナログ アラーム	テキスト スタイルに変換
オブジェクト サイズ - 高さ	Height
オブジェクト サイズ - 幅	幅
位置 - 垂直	水平位置
位置 - 水平	垂直位置
塗りつぶしパーセント - 垂直	垂直塗りつぶしパーセント
塗りつぶしパーセント - 水平	水平塗りつぶしパーセント
その他 - 表示オン/オフ	表示オン/オフ
その他 - 点滅	点滅
その他 - 向き	向き
その他 - 無効	無効化
その他 - ツールヒント	ツールヒント

InTouch アニメーション	産業用グラフィック エディタ アニメーション
値の表示 - 論理型	値の表示 - ブール型
値の表示 - アナログ型	値の表示 - アナログ型
値の表示 - 文字列型	値の表示 - 文字列型

スクリプトの使用

産業用グラフィック エディタでは InTouch WindowMaker と同じようにスクリプトを設定できます。ただし、小さな違いがあります。

InTouch スクリプト	産業用グラフィック エディタ スクリプト
アプリケーション スクリプト	(使用できません)
ウィンドウ スクリプト	シンボルの定義済みスクリプト
キー スクリプト	キー実行条件付きのアクション スクリプト アニメーション
アクション スクリプト	OnTrue、OnFalse、WhileTrue、WhileFalse 実行条件付きのシンボル名前付きスクリプト
データ変化スクリプト	DataChange 実行条件付きのシンボル名前付きスクリプト
QuickFunction	(使用できません)
ActiveX イベント スクリプト	(使用できません)
アクション スクリプト	アクション スクリプト アニメーション

アプリケーション スクリプトの使用

InTouch HMI で、アプリケーション スクリプトのトリガは以下の通りです。

- WindowViewer でアプリケーションが起動したときに 1 回実行。
- WindowViewer でアプリケーションが実行されている間、定期的に実行。
- WindowViewer でアプリケーションがシャットダウンしたときに 1 回実行。

産業用グラフィックは InTouch のアプリケーションに対応し、シンボルに直接関連付けられた定義済みスクリプトを設定できます。例:

- 表示時
- 表示中
- 非表示時

キー スクリプトの使用

産業用グラフィック エディタでキー スクリプトは使用できませんが、キーの組み合わせによって有効化されるアクション スクリプトに対し、要素を関連付けることができます。

条件スクリプトの使用

シンボル スクリプト機能を使用することで、条件が満たされたときに実行するスクリプトを設定できます。これにより、値または式が以下のときに実行する実行条件を定義できます。

- 条件を満たしている間(True の間)
- 条件を満たしたとき(True の時)
- 条件を満たしていない間(False の間)
- 条件を満たさなくなったとき(False の時)

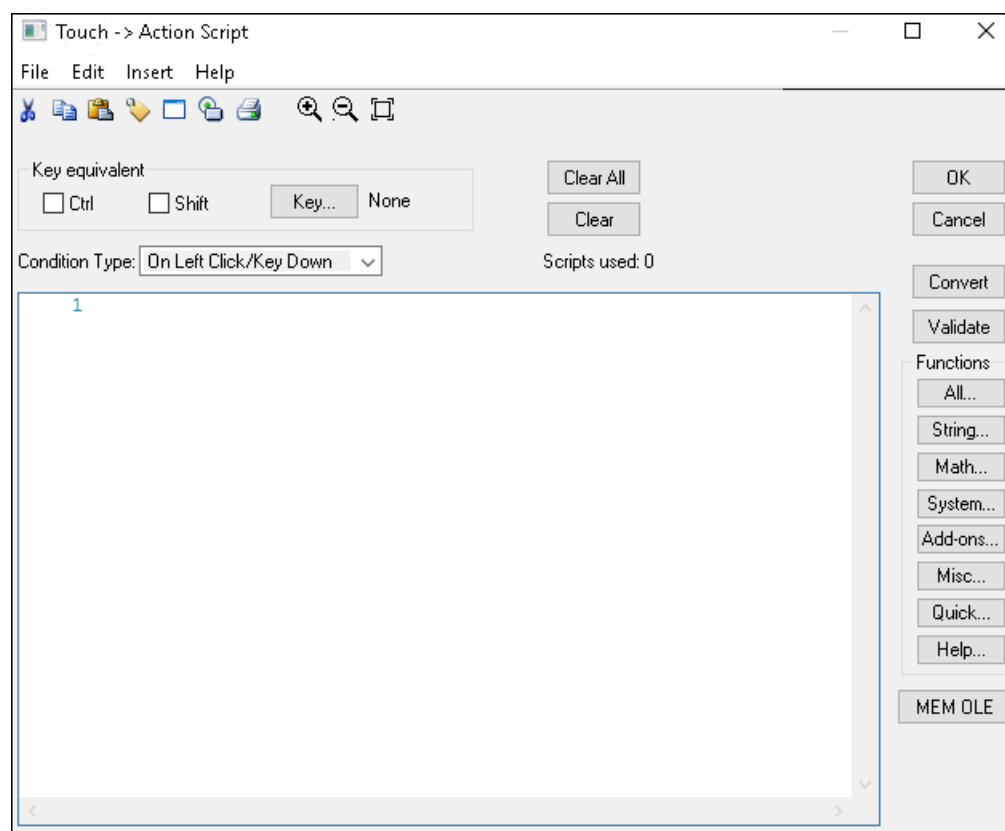
データ変化スクリプトの使用

シンボル スクリプト機能を使用することで、値または式が変更されたときに実行するスクリプトを設定できます。これにより、値または式が変化したときに実行する実行条件を定義できます。

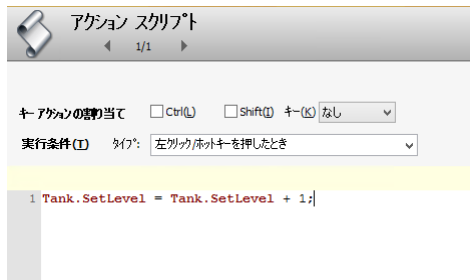
アクション スクリプトの使用

産業用グラフィック エディタでは、InTouch WindowMaker と同じようにアクション スクリプトを設定できます。ランタイムのユーザーがマウスを使用するかキーを押して要素を操作すると、アクション スクリプトを実行できます。

InTouch アクション スクリプト ウィンドウを使用してアクション スクリプトを作成します。



産業用グラフィック エディタのアクション スクリプト ウィンドウを使用してアクション スクリプトを作成します。



個別の要素、またはシンボル全体に対してアクション スクリプトを設定できます。

InTouch WindowMaker の定義済み関数の多くは産業用グラフィック エディタで使用できます。産業用グラフィックに使用できる InTouch の定義済み関数の詳細なリストについては、「[アクション スクリプトのインポート](#)」を参照してください。

アプリケーション スクリプト、キー スクリプトなど、他の InTouch スクリプト タイプは Application Server オートメーション オブジェクトで設定できます。

InTouch タグへのアニメーションの接続

要素のアニメーションと外観を InTouch タグに接続できます。InTouch タグは、要素のアニメーションと外観を制御する値をランタイム時に提供します。

以下の方法で要素のアニメーションを InTouch タグに接続できます。

- **[intouch:tag]** 構文で参照を設定します。この構文は、産業用グラフィックを使用する InTouch HMI スタンドアロン アプリケーション内の参照タグでは必要ありません。
- カスタム プロパティを使用して、InTouch 内の埋め込まれた産業用グラフィックのカスタム プロパティを、InTouch タグを参照するように設定します。属性として InTouch タグを含む管理対象の InTouchViewApp オブジェクトへの Application Server の属性参照を設定します。InTouchViewApp オブジェクトは InTouchProxy オブジェクトの機能を使用します。
- アイテムとして InTouch タグを含む InTouchProxy オブジェクトへの Application Server の属性参照を設定します。これは Application Server 属性参照の設定の特別な例です。

重要: 履歴サマリ データ型は、AVEVA OMI ViewApps または InTouch HMI マネージド アプリケーションを対象とした Application Server オブジェクト属性とのみ機能します。InTouch HMI モダン アプリケーションでは、履歴サマリ データ型が割り当てられているカスタム プロパティを使用しないでください。

InTouch:Tagname 構文の使用

intouch:tagname 構文を使用すると、アニメーションはグラフィックが使用されるノードの InTouch タグに接続します。この構文の使用には制限があります。

- Application Server とは異なり、ブール値として True と False は使用できません。代わりに 1 と 0 を使用します。
- InTouch SuperTag を参照する場合、代わりに以下の構文を使用します。

`attribute("InTouch:SuperTag\Member")`

- "InTouch:" 接頭辞は、産業用グラフィックを含む InTouch スタンドアロン アプリケーションでは不要です。タグの参照およびランタイム タグのバインディングは "InTouch:" 接頭辞なしで機能します。
- ハイフンを含む名前の InTouch タグを参照している Application Server カスタム プロパティは、ランタイム時に動作しません。たとえば、"InTouch:TAG-1" はランタイム時に動作しません。

入力モードの設定

いくつかのボックスで、静的、および/または属性と要素のプロパティに対する参照を使用する値または式を入力できます。両方の入力メソッドをサポートするボックスには **[入力モード]** 選択アイコンがあります。

以下を選択します。

- **[静的モード入力]** アイコンを選択すると、3.141 や「Test」など、静的な文字の値または式を指定できます。
- **[参照モード入力]** アイコンを選択すると、属性または要素プロパティへの参照を指定できます (Tank_001.PV など)。

注記: 静的な文字列値を参照モードで (参照と共に、または単独で) 使用するには、二重引用符で囲むことができます (例: "Description: "+Tank_001.Desc)

設定フィールドまたはスクリプトの固有のプロパティの 1 つを参照する要素を設定するとき、そのプロパティ名のみを使用することができます。産業用グラフィックの場合、オートメーションオブジェクトで使用されている「me.」のような自己を参照するキーワードはありません。

しかし、「me.」キーワードを使用して現在、設定している産業用グラフィックをホストしているオートメーションオブジェクトの属性を参照することができます。

InTouchViewApp 属性へのアニメーションの接続

InTouch タグを参照するには、最初に以下を実行する必要があります。

- WindowMaker で InTouchViewApp テンプレートを派生させて設定することによって、管理対象の InTouch アプリケーションを作成します。
- InTouchViewApp から派生したテンプレートのインスタンスを派生させます。

InTouch タグは InTouchViewApp オブジェクトのインスタンスの属性で表されます。

Galaxy ブラウザの [InTouch タグ ブラウザ] タブの使用

産業用グラフィック アニメーションまたはクライアント スクリプトに InTouch タグを必要とする参照を設定するときに、Galaxy ブラウザから直接 InTouch タグを選択できます。

アニメーションエディタまたはスクリプトエディタから呼び出したとき、Galaxy ブラウザには **[InTouch タグ ブラウザ]** タブが **[属性ブラウザ]** タブおよび **[要素ブラウザ]** タブと共に表示されます。

[InTouch タグ ブラウザ] タブでは、左ペインに現在の Galaxy の InTouchViewApp インスタンスとテンプレートがすべて一覧表示されます。右ペインには選択した InTouchViewApp に対する InTouch タグが表示されます。**[ドットフィールド]** リストボックスには、選択したタグに関連付けられたドットフィールドが表示されます。

右ペインの下にある **[ドットフィールド]** リストボックスでは、選択したタグのドットフィールドを指定できます。

[InTouch タグ ブラウザ] タブは以下のように動作します。

- InTouch タグ ブラウザの機能は、アニメーションエディタまたはスクリプトエディタのみから利用できます。
- Galaxy ブラウザはタグ名ディクショナリから InTouch タグを読み込みます。
- InTouch HMI がインストールされているかどうかに関係なく、タグ名ディクショナリのコンポーネントはインストールされ、Galaxy ブラウザで利用可能です。

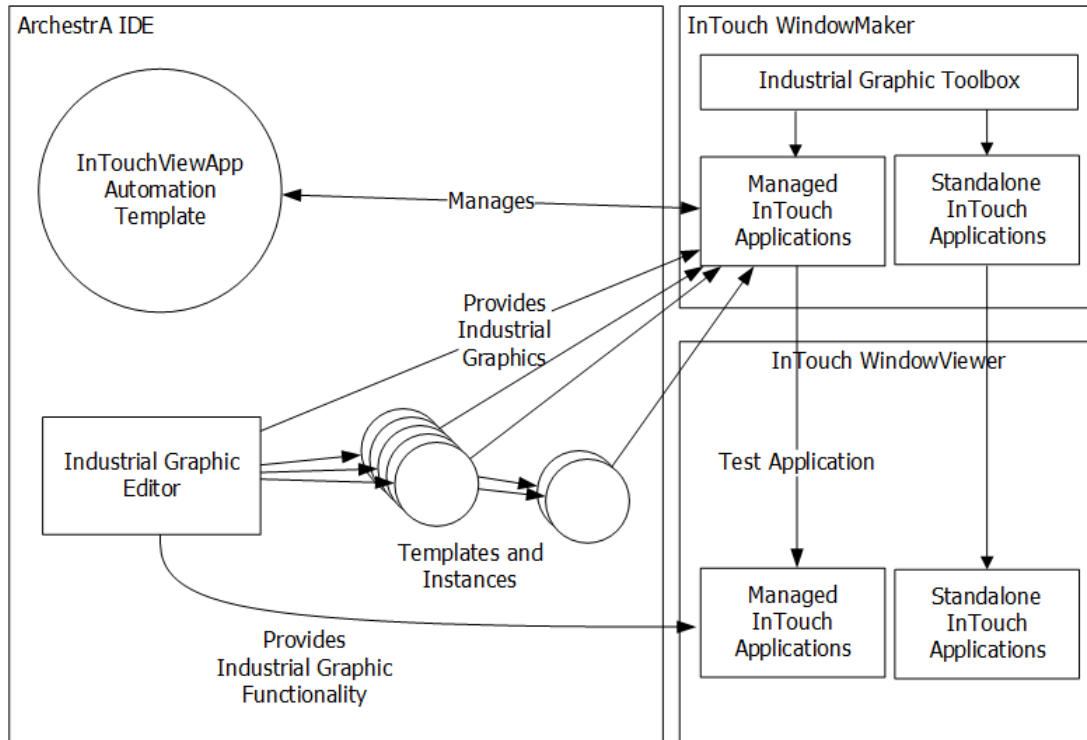
- タグは Galaxy ブラウザを閉じて再び開いたときにだけ更新されます。
- すべてのタグは Galaxy ブラウザを閉じて再び開くまでメモリに保持されます。
- 現在のユーザーが InTouchViewApp をチェックアウトした場合、Galaxy ブラウザが InTouchViewApp の最新のタグ名ディクショナリの内容を読み込みます。
- ユーザーが InTouchViewApp にチェックインしてアクセスした場合、Galaxy ブラウザは常にタグ名ディクショナリのチェックインバージョンを読み込みます。
- 現在のユーザー以外のユーザーが InTouchViewApp をチェックアウトした場合、Galaxy ブラウザはもっとも最近チェックインされたタグ名ディクショナリを読み込みます。
- InTouchViewApp テンプレートを選択する場合、出力参照文字列構文は InTouch:selectedTag です。InTouchViewApp インスタンスを選択する場合、出力参照文字列構文は SelectedInTouchViewAppInstance.selectedTag です。

アニメーションを InTouch タグに接続するには

1. 要素をダブルクリックします。
[アニメーションの編集] ダイアログボックスが表示されます。
2. アニメーションリストからアニメーションを選択します。
3. パラメータを選択します。
4. [参照] ボタンをクリックします。
[Galaxy ブラウザ] が表示されます。
5. [InTouch タグ ブラウザ] タブをクリックすると [InTouch タグ ブラウザ] ページが表示されます。
6. 管理対象の InTouch アプリケーションに対応する InTouchViewApp オブジェクトを選択します。右ペインに InTouch タグが表示されます。
7. タグを選択して、[OK] をクリックします。
InTouch タグに対して選択した参照が設定ボックスに表示されます。

WindowMaker での産業用グラフィックの使用

産業用グラフィックエディタで作成した産業用グラフィックは、マネージドまたはスタンドアロン InTouch アプリケーションで使用できます。WindowMaker の産業用グラフィックツールボックスから産業用グラフィックを直接追加することもできます。産業用グラフィックと Situational Awareness Library シンボルの操作に関する詳細については、『産業用グラフィックエディタ ユーザーガイド』または WindowMaker ヘルプを参照してください。

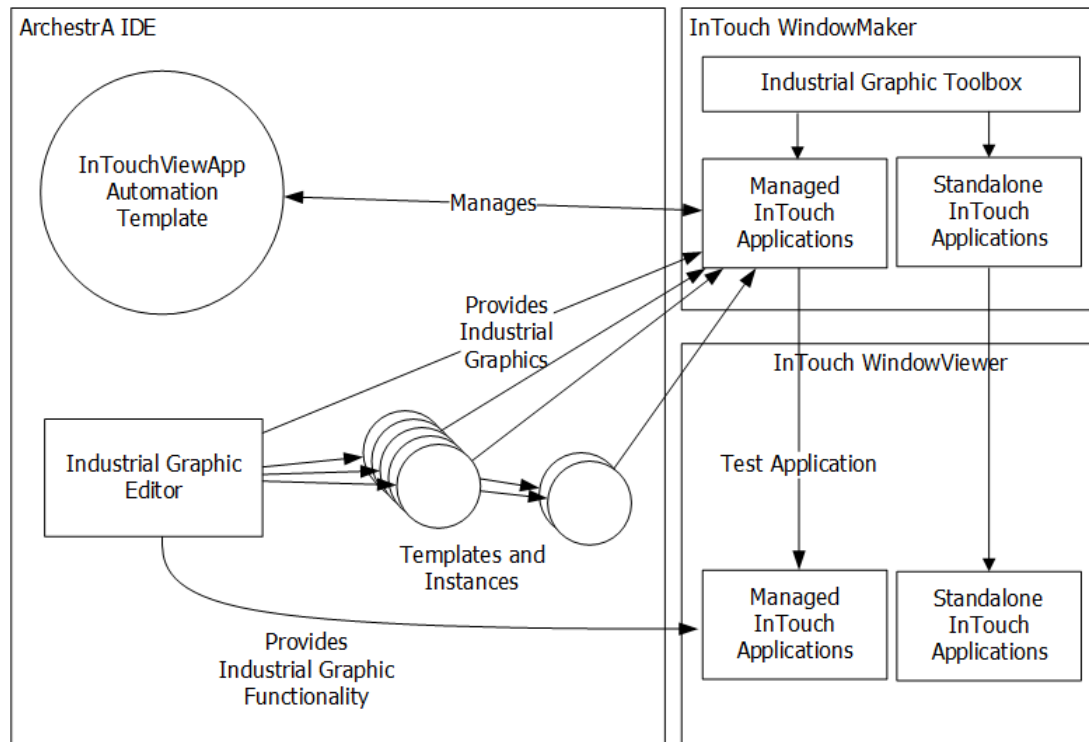


以下の操作を実行できます。

- InTouch ウィンドウへの産業用グラフィックの埋め込み。
- 産業用グラフィック ツールボックスから InTouch ウィンドウへの産業用グラフィックの直接のドラッグアンドドロップ。
- 埋め込まれた産業用グラフィックのサイズ変更。
- 産業用グラフィックへの制限された数の InTouch アニメーションの追加。
- 埋め込まれた産業用グラフィックのカスタム プロパティの設定。
- 産業用グラフィック エディタの起動。
- WindowViewer での産業用グラフィックのテスト。
- 埋め込まれた産業用グラフィックをホストするオートメーション オブジェクトの新規インスタンスの作成。

WindowMaker での産業用グラフィックの使用について

産業用グラフィック エディタで作成した産業用グラフィックは、マネージドまたはスタンドアロン InTouch アプリケーションで使用できます。WindowMaker の産業用グラフィック ツールボックスから産業用グラフィックを直接追加することもできます。産業用グラフィックと **Situational Awareness Library** シンボルの操作に関する詳細については、『産業用グラフィック エディタ ユーザー ガイド』または WindowMaker ヘルプを参照してください。



以下の操作を実行できます。

- InTouch ウィンドウへの産業用グラフィックの埋め込み。
- 産業用グラフィック ツールボックスから InTouch ウィンドウへの産業用グラフィックの直接のドラッグアンドドロップ。
- 埋め込まれた産業用グラフィックのサイズ変更。
- 産業用グラフィックへの制限された数の InTouch アニメーションの追加。
- 埋め込まれた産業用グラフィックのカスタム プロパティの設定。
- 産業用グラフィック エディタの起動。
- WindowViewer での産業用グラフィックのテスト。
- 埋め込まれた産業用グラフィックをホストする自動オブジェクトの新規インスタンスの作成。

InTouch ウィンドウへの産業用グラフィックの埋め込み

産業用グラフィックをマネージドおよびスタンドアロン InTouch アプリケーションの InTouch ウィンドウに埋め込むことができます。

産業用グラフィックは、以下の一部にすることができます。

- グラフィック ツールボックス
- オブジェクトテンプレート。
- オブジェクトインスタンス。

埋め込むことのできる産業用グラフィックは、要素に要素スタイルが適用されたシンボルおよびシンボルウィザードエディタを使用して作成されたシンボルウィザードです。

パブリッシュ済みアプリケーションに産業用グラフィックを含むマネージド InTouch アプリケーションをパブリッシュできます。ただし、新しい産業用グラフィックを追加することや、パブリッシュ済みアプリケーション内の既存のシンボルを編集することはできません。

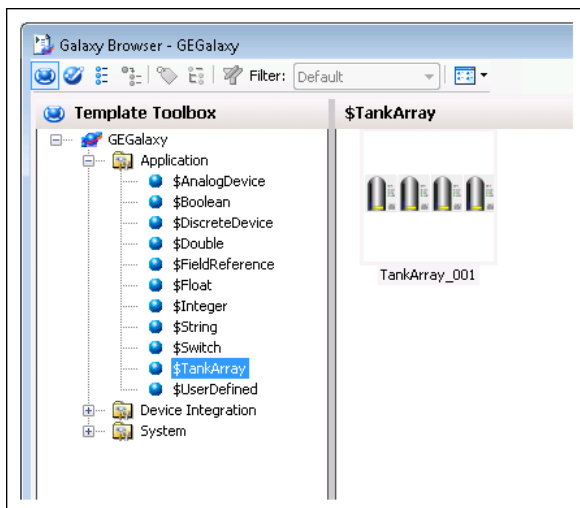
オートメーションテンプレートからの産業用グラフィックの埋め込み

産業用グラフィックをホストするオートメーションテンプレートから産業用グラフィックを埋め込むことができます。同時に、選択したテンプレートから新しく派生したインスタンスが作成されます。

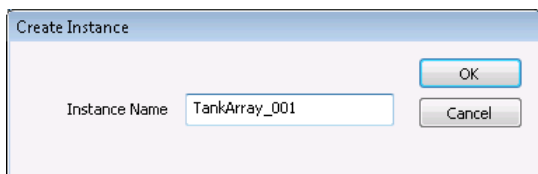
InTouch ウィンドウ上に既に存在する産業用グラフィックに基づく新規インスタンスの作成の詳細については、「[WindowViewer での産業用グラフィックのテスト](#)」を参照してください。

オートメーションテンプレートから産業用グラフィックを埋め込むには

1. WindowMaker を開きます。
2. オブジェクトを右クリックして **産業用グラフィックの埋め込み** をクリックします。
[Galaxy ブラウザ] ダイアログ ボックスが表示されます。
3. **[テンプレート ツールボックス]** アイコンをクリックします。左側に **[テンプレート ツールボックス]** リストが表示されます。

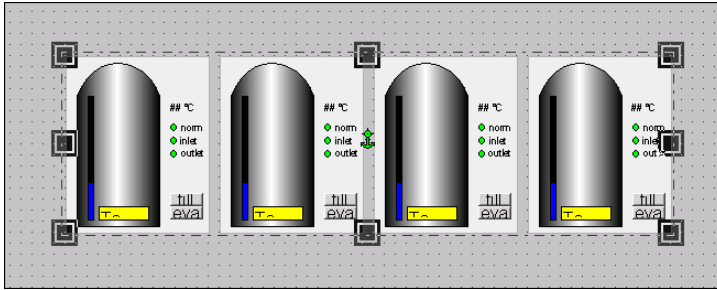


4. 埋め込む産業用グラフィックが含まれているテンプレートを選択します。選択したテンプレートに含まれている産業用グラフィックが右側に表示されます。
5. 埋め込む産業用グラフィックを選択し、**[OK]** をクリックします。InTouch ウィンドウ上にポインタを置くと、Galaxy ブラウザが閉じて挿入アイコンが表示されます。
6. InTouch ウィンドウ内で産業用グラフィックを埋め込む場所をクリックします。**[インスタンスを作成]** ダイアログ ボックスが表示されます。



7. **[インスタンス名]** ボックスで、インスタンスの名前を入力します。

8. **[OK]** をクリックします。指定した名前を持つテンプレートから、インスタンスが自動的に派生します。InTouch ウィンドウにシンボルが埋め込まれます。



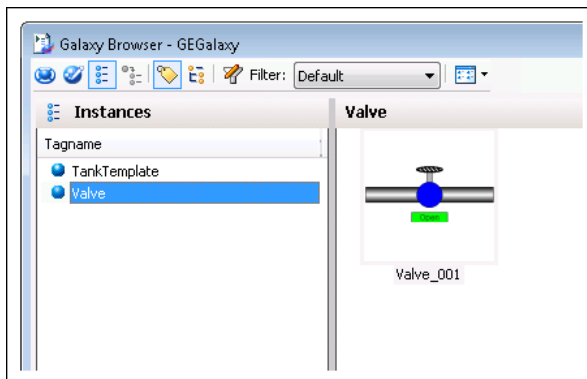
インスタンスからの産業用グラフィックの埋め込み

産業用グラフィックが関連付けられているインスタンスから産業用グラフィックを埋め込むことができます。

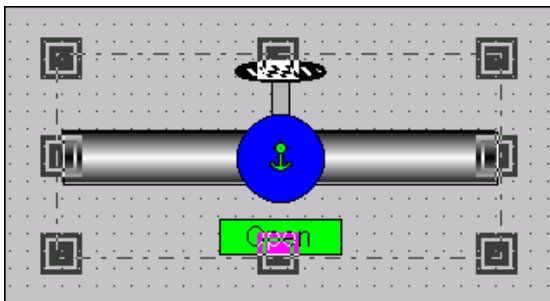
インスタンスから産業用グラフィックを埋め込むと、シンボルがそのインスタンスに関連付けられます。

インスタンスから産業用グラフィックを埋め込むには

1. WindowMaker を開きます。
2. オブジェクトを右クリックして **産業用グラフィックの埋め込み** をクリックします。
[Galaxy ブラウザ] ダイアログ ボックスが表示されます。
3. **[インスタンス]** アイコンをクリックします。左側に **[インスタンス]** リストが表示されます。



4. 埋め込む産業用グラフィックが含まれているインスタンスをクリックします。選択したインスタンスに関連付けられている産業用グラフィックが右側に表示されます。
5. 埋め込む産業用グラフィックをクリックし、**[OK]** をクリックします。InTouch ウィンドウ上にポインタを置くと、Galaxy ブラウザが閉じて挿入アイコンが表示されます。
6. InTouch ウィンドウ内で産業用グラフィックを埋め込む場所をクリックします。InTouch ウィンドウにシンボルが埋め込まれます。



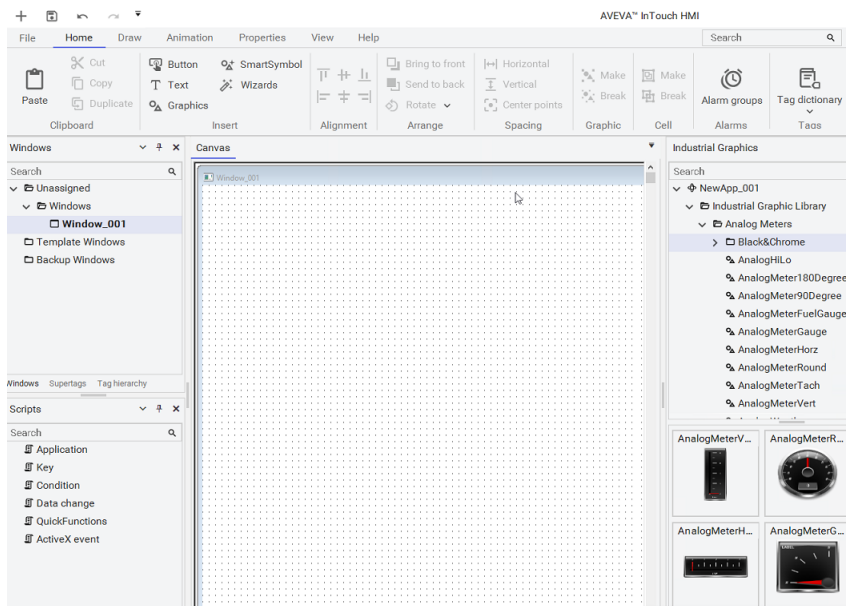
グラフィック ツールボックスからの産業用グラフィックの埋め込み

グラフィック ツールボックスから産業用グラフィックを埋め込むことができます。

グラフィック ツールボックスから産業用グラフィックを埋め込むには

オプション 1

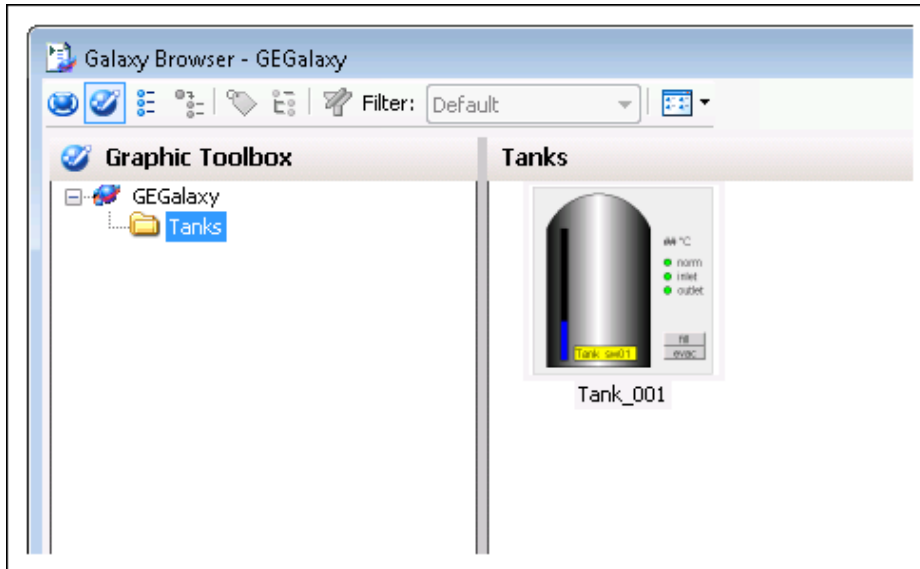
1. WindowMaker を開きます。
2. [産業用グラフィック] ペインから、目的の産業用グラフィック フォルダを選択します。
そのフォルダの下に産業用グラフィックがサムネイルとして表示されます。
または
検索ボックスで目的の産業用グラフィックを検索します。
3. 目的のグラフィックを選択して、キャンバスにドラッグアンドドロップします。
枠ウィンドウが作成されます。



オプション 2

1. WindowMaker を開きます。
2. オブジェクトを右クリックして [産業用グラフィックの埋め込み] をクリックします。
3. [Galaxy ブラウザ] ダイアログ ボックスが表示されます。

4. [グラフィック ツールボックス] アイコン  をクリックします。左側に [グラフィック ツールボックス] リストが表示されます。



5. 埋め込む産業用グラフィックを選択し、[OK] をクリックします。ポインタを InTouch ウィンドウ上に置くと、挿入アイコンが表示されます。
6. InTouch ウィンドウ内で産業用グラフィックを埋め込む場所をクリックします。InTouch ウィンドウにシンボルが埋め込まれます。

要素スタイルが適用された産業用グラフィックの埋め込み

要素スタイルが適用された産業用グラフィックを埋め込むことができます。要素スタイルは、グラフィックの塗りつぶし、線、テキスト、点滅、および外郭プロパティを定義します。要素スタイルで定義された表示プロパティがグラフィックに適用されます。要素スタイルを使用すると、要素に一貫したスタイルを適用できます。要素スタイルを使用すると、画面ビルダやシンボルを作成するその他のユーザーの表示標準を確立することもできます。

要素スタイルが適用された産業用グラフィックは、その他の産業用グラフィックと同様に InTouch ウィンドウに埋め込まれます。

産業用グラフィックでの要素スタイルの使用の詳細については、『産業用グラフィック エディタ ユーザーガイド』の「要素スタイルの操作」を参照してください。

マネージド InTouch アプリケーションの要素スタイル

WindowViewer で一度に開くことのできるアプリケーションは 1 つだけです。プラットフォームがローカル ノードに配置されている場合、他のスタンドアロン アプリケーションで設定されているスタイルよりも Galaxy で設定したスタイルが優先されます。

シンボル ウィザードの埋め込み


System Platform IDE からシンボル ウィザードをマネージド InTouch アプリケーションのウィンドウに埋め込むことができます。

シンボル ウィザードを埋め込むには

1. WindowMaker を開き、埋め込まれたシンボルを含むウィンドウを表示します。

2. オブジェクトを右クリックして **「産業用グラフィックの埋め込み」** をクリックします。

「Galaxy ブラウザ」 ダイアログ ボックスが表示されます。

3. **「グラフィック ツールボックス」** アイコン  をクリックします。左側に **「グラフィック ツールボックス」** リストが表示されます。

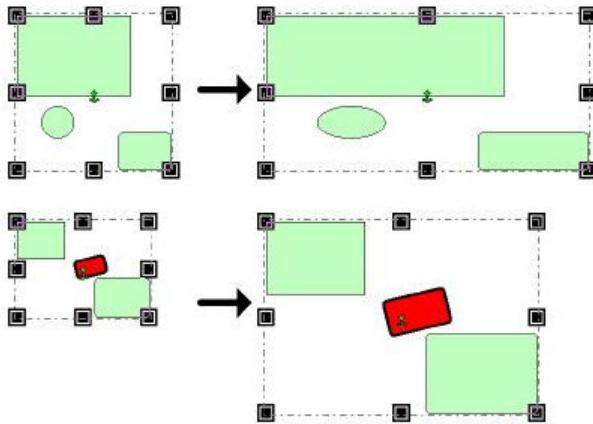
4. アプリケーションに埋め込むシンボル ウィザードを選択して、**「OK」** をクリックします。

5. InTouch ウィンドウ内でシンボル ウィザードを埋め込む場所をクリックします。デフォルトのシンボル ウィザード設定を使用して、シンボルが InTouch ウィンドウに埋め込まれます。

埋め込まれた産業用グラフィックのサイズ変更

InTouch ウィンドウに産業用グラフィックを埋め込んだ後、他の InTouch オブジェクトで行う場合と同様に、ハンドルまたは幅および高さの値エントリを使用してサイズを変更できます。

ただし、シンボルに少なくとも 1 つの回転要素が含まれている場合、産業用グラフィックのサイズ変更は縦横の比率を維持した状態でのみ行うことができます。



埋め込まれた産業用グラフィックのサイズは、その最小サイズよりも小さいサイズに変更することはできません。最小サイズは、含まれている要素のペン幅によって決まります。

埋め込まれた産業用グラフィックは、産業用グラフィック エディタで作成されたときの元のサイズにリセットできます。

埋め込まれた産業用グラフィックのサイズを変更するには

1. 産業用グラフィックを選択すると、ハンドルが表示されます。

2. 以下のいずれかを実行します。

- ハンドルの 1 つをドラッグして、産業用グラフィックのサイズを変更します。
- ステータス バーの **「W」** ボックスおよび **「H」** ボックスに、幅および高さを入力します。

埋め込まれた産業用グラフィックのサイズを元のサイズに戻すには

- 元のサイズに戻す産業用グラフィックを右クリックし、**「産業用グラフィック」** をポイントし、次に **「シンボル - オリジナル サイズ」** をクリックします。埋め込まれた産業用グラフィックが元のサイズに戻ります。

WindowMaker での産業用グラフィックの設定

WindowMaker に埋め込まれた産業用グラフィックは、以下の方法で設定できます。

- コピー、切り取り、貼り付け、複製、サイズ変更、移動、および削除などの標準の編集を行う。
- WindowMaker アニメーション リンクを設定する。
- InTouch タグ変数と産業用グラフィックを接続する。
- 同じ親の代替インスタンスを選択する。
- 同じインスタンスの代替シンボルを選択する。
- ダイナミック サイズの反映を有効化または無効化する。

産業用グラフィックの WindowMaker アニメーション リンクの設定

埋め込まれた産業用グラフィックの WindowMaker アニメーション リンクは、他の InTouch オブジェクトと同じ方法で設定します。設定できるのは、埋め込まれた産業用グラフィックの外部にある次のようなアニメーション リンクだけです。

- オブジェクト サイズ
- オブジェクトの位置
- 表示オン/オフ
- 使用可能性

WindowMaker で設定されたアニメーション リンクは、産業用グラフィック エディタで設定されたアニメーション リンクから独立しています。これらには産業用グラフィックの設定は継承されません。また、WindowMaker で実行されるときに優先されます。

埋め込まれた産業用グラフィックの WindowMaker アニメーション リンクを設定するには

1. 埋め込まれた産業用グラフィックを選択します。
2. [描画] メニューの [モード] グループで [アニメーション化] をクリックします。
または、オブジェクトを右クリックして、[アニメーション リンク] を選択します。
[アニメーション リンク] ウィンドウが表示されます。

The screenshot shows a configuration window for an 'Industrial Graphic' object named 'ClockAnalogWall1'. The window has a title bar and a header section with 'Object type: Industrial Graphic', 'Name: ClockAnalogWall1', and buttons for 'Prev Link', 'Next Link', 'OK', and 'Cancel'. The main area is divided into several sections:

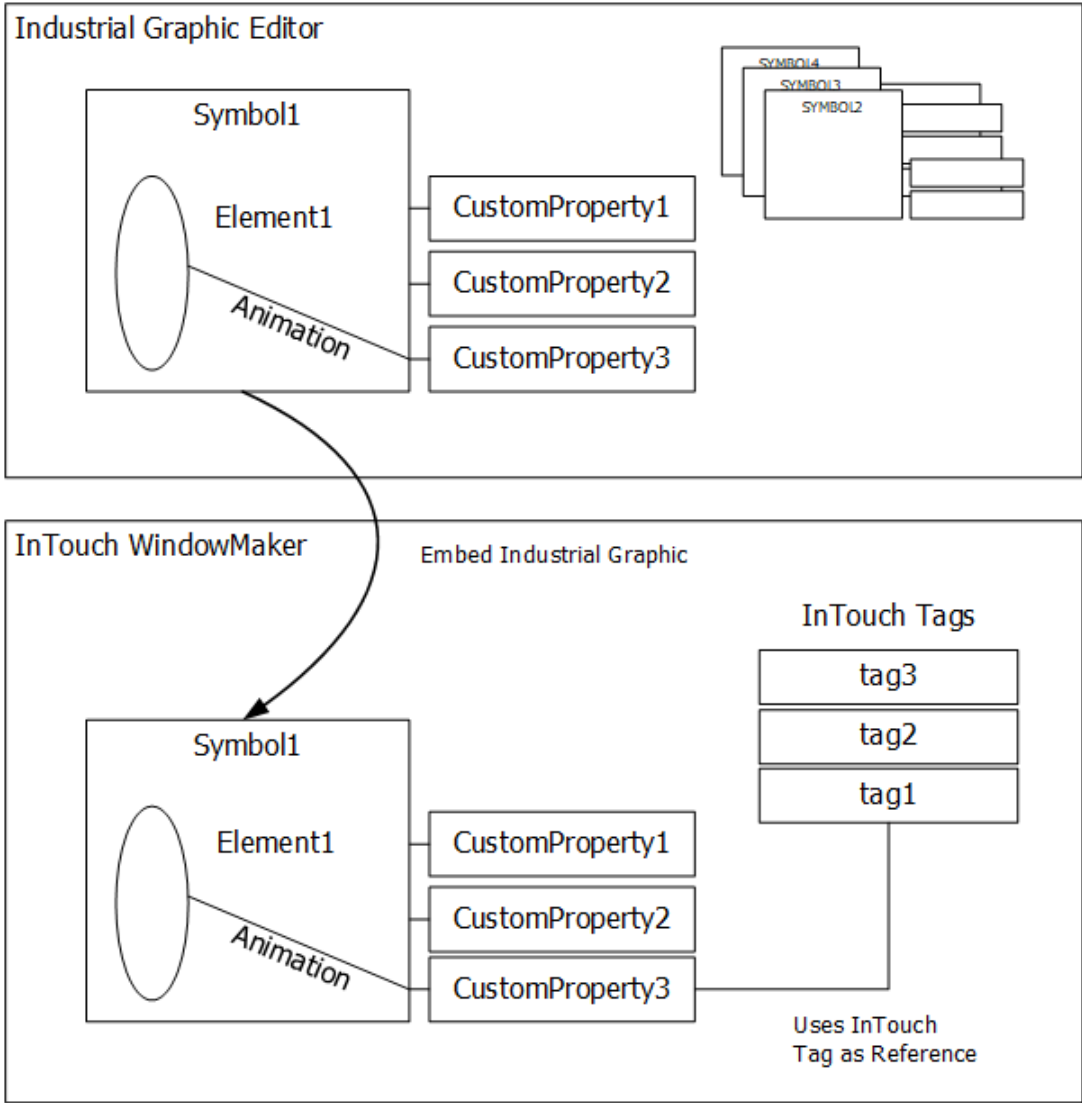
- Touch Links**: Includes 'User Inputs' (Discrete, Analog, String), 'Sliders' (Vertical, Horizontal), and 'Touch Pushbuttons' (Discrete Value, Action, Show Window, Hide Window).
- Line Color**: Includes 'Discrete', 'Analog', 'Discrete Alarm', and 'Analog Alarm'.
- Fill Color**: Includes 'Discrete', 'Analog', 'Discrete Alarm', and 'Analog Alarm'.
- Text Color**: Includes 'Discrete', 'Analog', 'Discrete Alarm', and 'Analog Alarm'.
- Object Size**: Includes 'Height' and 'Width'.
- Location**: Includes 'Vertical' and 'Horizontal'.
- Percent Fill**: Includes 'Vertical' and 'Horizontal'.
- Miscellaneous**: Includes 'Visibility', 'Blink', 'Orientation', 'Disable', and 'Tooltip'.
- Value Display**: Includes 'Discrete', 'Analog', and 'String'.

3. 他の InTouch オブジェクトと同じ方法で変更します。
4. [OK] をクリックします。

InTouch タグと産業用グラフィックの接続

埋め込まれた産業用グラフィックのカスタムプロパティを上書きすることにより、産業用グラフィックと InTouch タグを接続できます。産業用グラフィックのプロパティがカスタムプロパティによって InTouch に公開されます。カスタムプロパティは、産業用グラフィックのアニメーションによって内部的に使用できる場合と、できない場合があります。

Connecting Industrial Graphics with InTouch Tags



産業用グラフィックを InTouch ウィンドウに埋め込むと、以下の表に示すようにアニメーション リンク内の参照が変換されます。

産業用グラフィック	埋め込まれた産業用グラフィック
Object.Extension	galaxy:Object.Extension
intouch:Tagname	タグ名

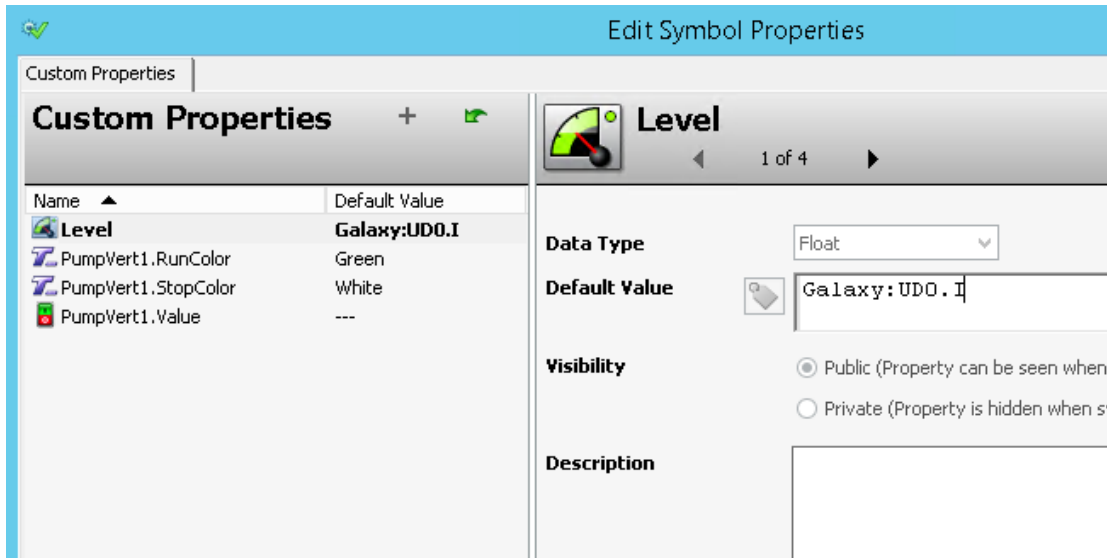
カスタム プロパティの詳細については、『産業用グラフィック エディタ ユーザー ガイド』を参照してください。

アニメーション エディタまたはスクリプト エディタ内で作業している場合、Galaxy ブラウザから InTouch タグを直接選択できます。それらのエディタのどちらかから起動された場合、Galaxy ブラウザは現在の Galaxy の InTouchViewApp インスタンスをすべてリストし、選択されている InTouchViewApp インスタンスの InTouch タグとドット フィールドをすべてリストします。

Galaxy ブラウザでの InTouch タグの選択の詳細については、『[InTouch からの Application Server のオブジェクト属性の参照](#)』を参照してください。

InTouch タグと産業用グラフィックを接続するには

1. 埋め込まれた産業用グラフィックを InTouch ウィンドウ内で右クリックし、[産業用グラフィック] をポイントし、次に [シンボルプロパティの編集] をクリックします。[シンボルプロパティの編集] ダイアログ ボックスが表示されます。



2. InTouch タグに接続するカスタム プロパティを選択します。選択したカスタム プロパティの設定が右ペインに表示されます。
3. [デフォルト値] ボックスで、以下のいずれかを実行します。
 - InTouch タグの名前を入力します。
 - 参照ボタンをクリックし、[タグの選択] ダイアログ ボックスからタグを選択します。



4. カスタム プロパティを元の値に戻すには、[復元] をクリックします。
5. [OK] をクリックします。選択したカスタム プロパティを使用して設定した産業用グラフィック内のアニメーションで、処理中に InTouch タグの値が使用されるようになります。

InTouch タグと産業用グラフィックの接続の例

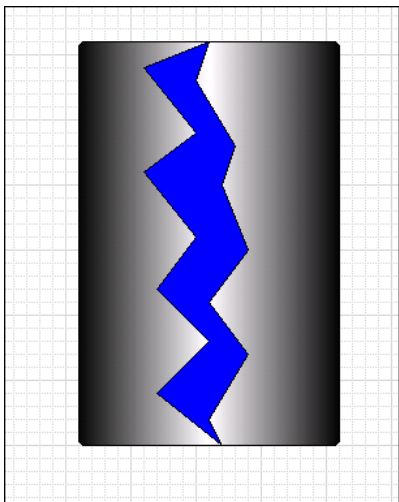
この例では、産業用グラフィック エディタで作成された垂直塗りつぶしパーセントアニメーションを持つタンク シンボルを InTouch タグに接続する方法を示します。

これを行うには、以下の 3 つの主な手順を実行します。

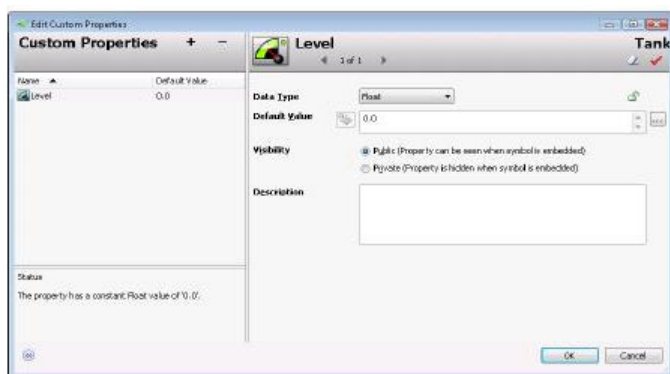
- 産業用グラフィックとしてサンプル タンクを作成します。
- InTouch アプリケーションを作成します。
- WindowViewer でサンプル タンクを派生させて表示します。

産業用グラフィックとしてサンプル タンクを作成するには

1. IDE で、"Tank" という名前の新しいシンボルを作成し、産業用グラフィック エディタで開きます。
2. キャンバスに長方形を貼り付けます。必要に応じて、外観を変更します。
3. タンク レベルを示すためのタンクの切り抜きを表す色付きの多角形要素を作成します。

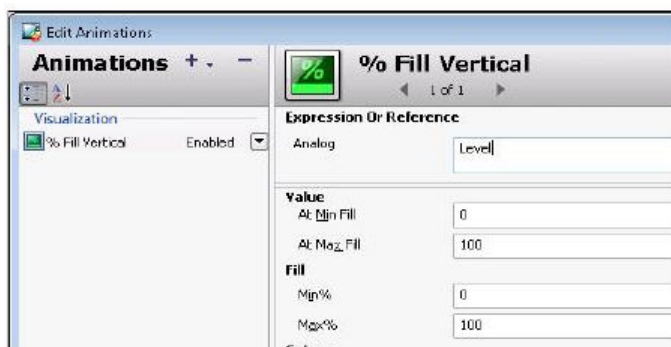


4. キャンバスをクリックします。
5. [プロパティ] メニューの [グラフィック] グループで [編集] を選択します。
または、グラフィック オブジェクトをダブルクリックします。
[カスタム プロパティの編集] ダイアログ ボックスが表示されます。
6. **Level** という名前のカスタム プロパティを追加します。
7. プロパティの詳細を設定します。以下の手順を実行します。
 - [データ タイプ] リストで、[浮動小数点] をクリックします。
 - [デフォルト値] ボックスに 0 と入力します。



8. [OK] をクリックします。
9. タンク レベルを表す多角形要素をダブルクリックします。
[アニメーションの編集] ダイアログ ボックスが表示されます。

10. [垂直塗りつぶしパーセント] アニメーションを追加します。
11. 右ペインにある [アナログ] ボックスに、カスタムプロパティの名前を入力します。この例では、これは Level です。



12. [OK] をクリックし、[アニメーションの編集] ダイアログ ボックスを閉じます。
13. [保存して閉じる] をクリックし、産業用グラフィック エディタを閉じます。

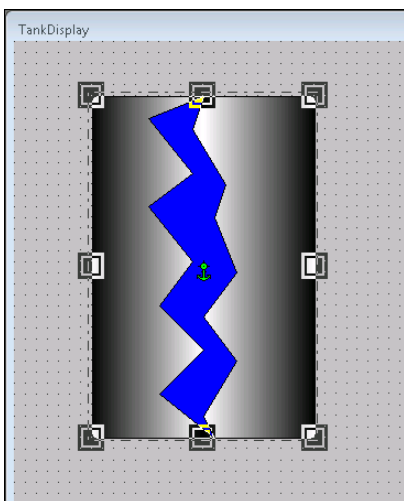
InTouch アプリケーションを作成するには

1. System Platform IDE で、新しいマネージド InTouch アプリケーションを作成します。詳細については、[「マネージド InTouch アプリケーションの作成」](#) を参照してください。
2. マネージド InTouch アプリケーションを WindowMaker で開きます。
3. TankDisplay という名前の新しいウィンドウを作成します。
4. タグ名ディクショナリを開き、TankLevel という名前の新しい実数型の InTouch タグを作成します。
5. [産業用グラフィックの埋め込み] アイコンをクリックします。

[Galaxy ブラウザ] ダイアログ ボックスが表示されます。

6. タンク シンボルを選択し、[OK] をクリックします。
7. ウィンドウ内のシンボルを埋め込む新しい位置でクリックします。

ウィンドウにタンク シンボルが埋め込まれます。



8. 埋め込まれた産業用グラフィックを右クリックし、[産業用グラフィック "Tank"] をポイントし、次に [シンボルプロパティの編集] をクリックします。
[シンボルプロパティの編集] ダイアログ ボックスが表示されます。
9. カスタム プロパティ レベルを選択します。
10. [デフォルト値] ボックスに TankLevel と入力します。また、参照ボタンをクリックし、[タグの選択] ダイアログ ボックスを使用して TankLevel を参照することもできます。
11. [OK] をクリックします。
12. ウィンドウにスライダを貼り付け、ローカル InTouch タグである TankLevel を使用して設定します。
13. 変更を保存し、WindowMaker を終了します。

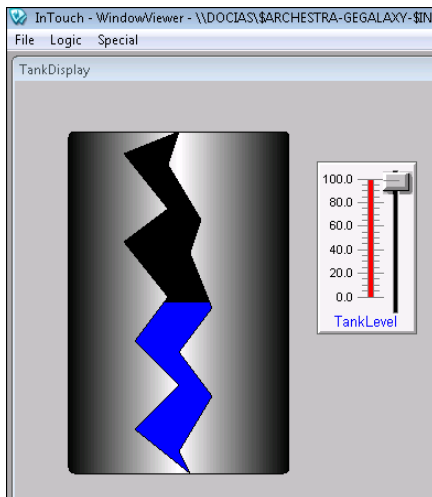
マネージド InTouch アプリケーションは自動的にチェック インされます。

サンプル タンクを派生させ、テストするには

1. System Platform IDE で、マネージド InTouch アプリケーションのインスタンスを派生させ、WinPlatform インスタンスおよび ViewEngine インスタンスと共に配置します。
2. InTouch アプリケーション マネージャを開き、WindowViewer のリストに表示されているアプリケーションを起動します。

タンクおよびスライダが WindowViewer のウィンドウに表示されます。

3. スライダを移動するとタンク レベルを変更できます。



同じ親からの代替インスタンスの選択

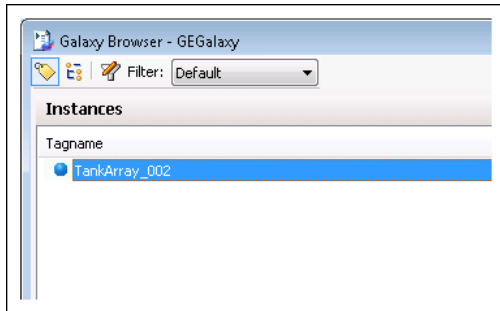
産業用グラフィック内のすべての参照は、代替インスタンスにリダイレクトできます。継承された産業用グラフィックは編集できないので、サイズを除き、産業用グラフィックの外観は変更されません。

グラフィック ツールボックスから生成される産業用グラフィックはオブジェクトには関連付けられていないため、この機能は使用できません。

同じ親から代替インスタンスを選択するには

1. 埋め込まれた産業用グラフィックを右クリックし、[産業用グラフィック] をポイントし、次に [代替インスタンスの選択] をクリックします。

[Galaxy ブラウザ] ダイアログ ボックスが表示されます。同じ親を持つその他のすべてのインスタンスが表示されます。



2. リストから代替インスタンスを選択し、[OK] をクリックします。

新しい代替インスタンスをポイントするように、産業用グラフィックの参照が更新されます。

同じインスタンスの代替シンボルの選択

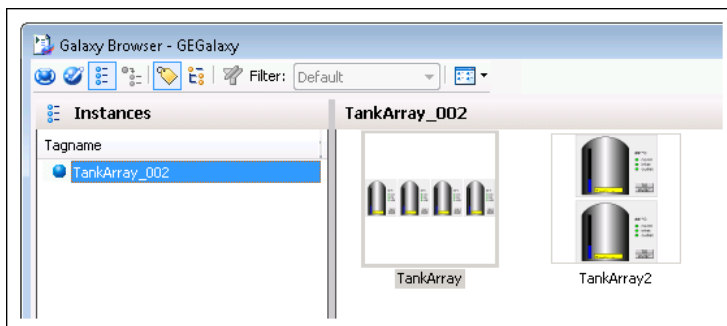
埋め込まれた産業用グラフィックは、同じインスタンスに属する他の産業用グラフィックで置き換えることができます。

グラフィック ツールボックスから生成される産業用グラフィックはオブジェクトには関連付けられていないため、この機能は使用できません。

代替の産業用グラフィックを同じインスタンスから選択するには

1. 埋め込まれた産業用グラフィックを右クリックし、[産業用グラフィック] をポイントし、次に [代替シンボルの選択] をクリックします。

[Galaxy ブラウザ] ダイアログ ボックスが表示されます。



2. 右ペインで代替シンボルを選択し、[OK] をクリックします。
3. 代替シンボルのサイズが元のシンボルのサイズと異なる場合、現在埋め込まれている産業用グラフィックのサイズを保持するかどうかを確認するメッセージが表示されます。以下のいずれかを実行します。
 - 選択されている産業用グラフィックの現在のサイズを保持するには、[はい] をクリックします。
 - 選択されている産業用グラフィックのサイズを新しい産業用グラフィックのサイズに更新するには、[いいえ] をクリックします。

どちらの場合でも、埋め込まれた産業用グラフィックは代替の新しい産業用グラフィックで更新されます。

産業用グラフィック内の文字列の置換

埋め込まれた産業用グラフィック内のすべての文字列を代替文字列で置き換えることができます。

埋め込まれた産業用グラフィック内のすべての文字列を置き換えるには

1. 埋め込まれた産業用グラフィックを選択します。
2. [アニメーション] メニューの [置換] グループで [文字列] をクリックします。
[文字列の変更] ダイアログ ボックスが表示されます。
3. 対応するボックスに新しい文字列を入力し、[OK] をクリックします。

埋め込まれた産業用グラフィック内の文字列が新しい代替文字列で置き換えられます。

産業用グラフィック内の参照の置換

埋め込まれた産業用グラフィック内のすべての参照を代替の参照で置き換えることができます。

埋め込まれた産業用グラフィック内のすべての参照を置き換えるには

1. 埋め込まれた産業用グラフィックを選択します。
2. [アニメーション] メニューの [置換] グループで [タグ] をクリックします。
[タグの置換] ダイアログ ボックスが表示されます。
3. 対応するボックスに新しい参照を入力し、[OK] をクリックします。

埋め込まれた産業用グラフィック内の文字列が新しい代替参照で置き換えられます。

埋め込まれた産業用グラフィックのダイナミック サイズ変更の反映の有効化または無効化

埋め込まれた産業用グラフィックのダイナミック サイズ変更の反映を有効化または無効化できます。

ダイナミック サイズ変更の反映が有効になっている、ソース シンボルの絶対アンカー ポイントの位置は以下のように変更されます。

- 埋め込まれたシンボルのアンカー ポイントは変更されません。
- 埋め込まれたシンボルの位置は適宜移動します。

ダイナミック サイズ変更の反映が無効になっていると、ソース シンボルの絶対アンカー ポイントの位置は以下のように変更されます。

- 埋め込まれたシンボルのアンカー ポイントが適宜移動します。
- 埋め込まれたシンボルの位置は変更されません。

ダイナミック サイズの反映の詳細については、『産業用グラフィック エディタ ユーザー ガイド』を参照してください。

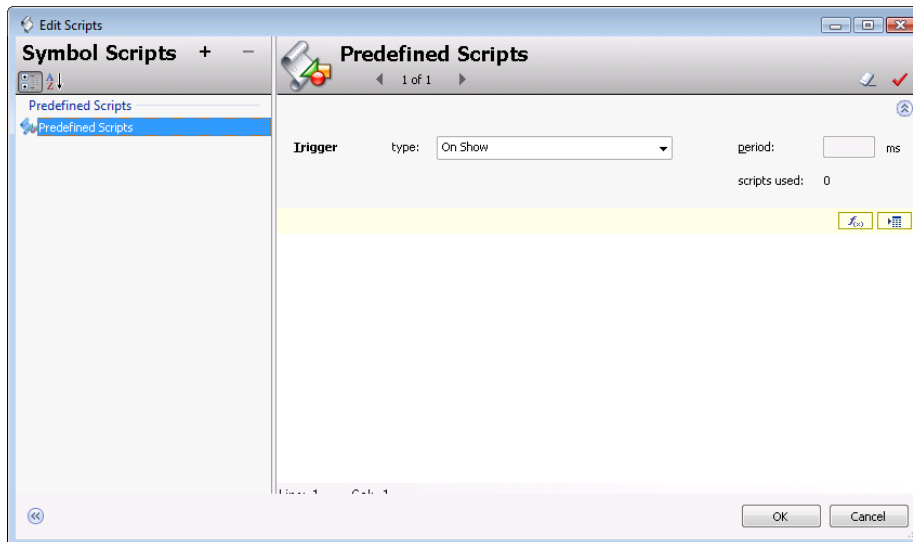
埋め込まれたシンボルのダイナミック サイズ変更の反映を有効化または無効化するには

- 埋め込まれた産業用グラフィックを右クリックし、[産業用グラフィック] をポイントして [ダイナミック サイズ変更] をオンまたはオフにします。

スクリプトと産業用グラフィックの関連付け

スクリプトと InTouch アプリケーションに配置された産業用グラフィックを関連付けることができます。スクリプトを使用して、InTouch アプリケーションの実行中にグラフィックをアニメーション化することや要素を変更することができます。

InTouch ウィンドウに埋め込まれた産業用グラフィックを選択した後、スクリプトの実行をトリガする値を持つ式またはリファレンスを選択します。産業用グラフィック エディタを使用して、スクリプトを実行するトリガを選択します。

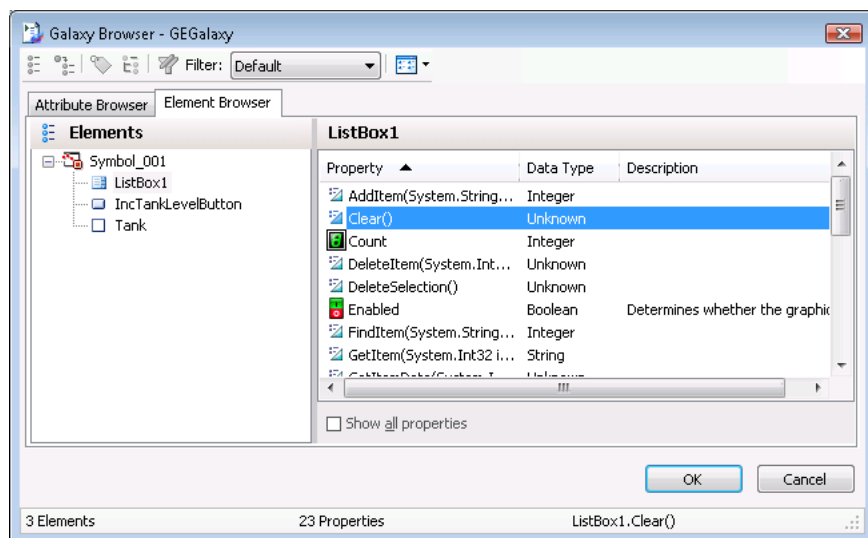


- グラフィック スクリプトは、事前定義することや名前を付けることができます。事前定義されたスクリプトは、実行中のアプリケーションのグラフィックのステータスに基づいて実行されます。名前付きスクリプトは、スクリプトに関連付けられた式またはリファレンスで状態が変更された場合に実行されます。
- 事前定義されたスクリプトは、HMI InTouch ウィンドウ スクリプトに似ています。スクリプト トリガの設定方法に応じて、事前定義されたシンボル スクリプトが以下のように実行されます。
 - グラフィックを開いた後、または表示した後に 1 回。
 - 実行中のアプリケーションにグラフィックが表示されている間、定期的に。
 - グラフィックを閉じた後、または非表示にした後に 1 回。
- トリガ値または式が **True** または **False** であるか、**True** と **False** のステータスの間を移行している場合に、スクリプトの設定方法に応じて名前付きグラフィック スクリプトを実行できます。また、トリガ式に関連付けられたデータの値が変更されたとき、または品質状態の値が変更されたときに、名前付きグラフィック スクリプトを実行できます。

産業用グラフィック スクリプトでのメソッドの使用

いくつかの要素では、スクリプト メソッドがサポートされています。これらのメソッドは、ランタイム中に要素上でさまざまな関数を実行できます。通常、メソッドにアクセスするアクション スクリプトを設定します。

Galaxy ブラウザを開いて要素を選択すると、要素のサポートされているプロパティおよびメソッドが表示されます。



- **Edit Box** コントロールのメソッドが含まれているスクリプトを実行して、ランタイム中にファイルからコントロールにテキストをロードすることができます。また、スクリプトを実行して、ランタイム中に **Edit Box** コントロールの現在のコンテンツをファイルに保存することもできます。

- **Edit Box** コントロールのメソッドは、スクリプト内で以下の形式で宣言されています。

```
ControlName.SaveText(FileName);
```

ここで、**ControlName** は **Edit Box** コントロールの名前であり、**FileName** はロードまたは保存されるコントロールのコンテンツが含まれているファイルの名前です。上記の例では、**SaveText** が、**Edit Box** コントロールのコンテンツをファイルに保存するメソッドの名前です。

- **Combo Box** コントロールおよび **List Box** コントロールのメソッドが含まれているスクリプトを使用して、ランタイム中にリストのコンテンツを変更できます。リストアイテムは、追加、削除または変更できます。
- **Combo Box** コントロールおよび **List Box** コントロールのメソッドは、**Edit Box** コントロールに似たスクリプトで宣言されています。

産業用グラフィック エディタでの産業用グラフィックの編集

埋め込まれた産業用グラフィックは、**System Platform IDE** に統合されている産業用グラフィック エディタを使用して編集できます。これを行うには、以下の手順を実行します。

1. 埋め込まれた産業用グラフィックを産業用グラフィック エディタで開き、シンボルを変更して保存します。テンプレート、インスタンス、またはグラフィック ツールボックスで産業用グラフィックが更新されます。

詳細については、「[埋め込まれた産業用グラフィックの編集](#)」を参照してください。

2. ステータス バーの右下隅にある「シンボルの変更」アイコンをクリックし、**WindowMaker** の変更を受け入れます。**WindowMaker** に変更が反映されます。

詳細については、「[WindowMaker でのシンボルの変更の受け入れ](#)」を参照してください。

埋め込まれた産業用グラフィックの編集

埋め込まれた産業用グラフィックは、**InTouch WindowMaker** 内で簡単に編集できます。

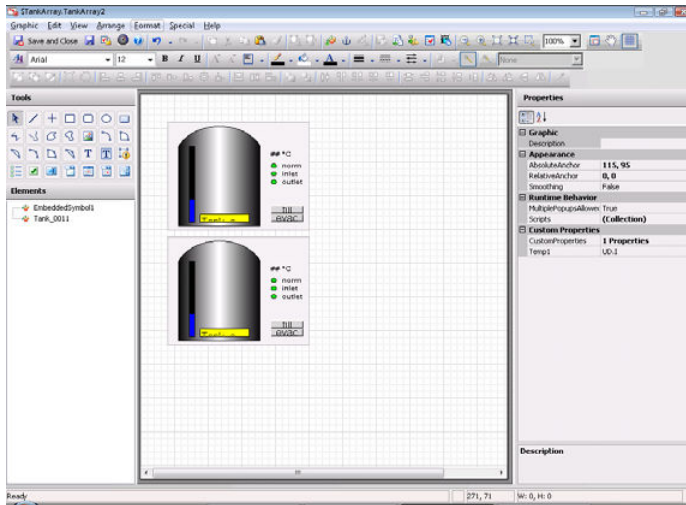
ソース シンボルまたはその埋め込まれたシンボルが他のマネージド InTouch アプリケーションで使用されている場合、埋め込まれたシンボルおよび InTouch アプリケーションに変更が反映されます。

産業用グラフィックに加えたすべての変更は、埋め込まれたシンボルに自動的に反映されるわけではありません。詳細については、「[WindowMaker でのシンボルの変更の受け入れ](#)」を参照してください。

埋め込まれた産業用グラフィックを産業用グラフィック エディタで編集するには

1. 埋め込まれた産業用グラフィックを右クリックし、[産業用グラフィック] をポイントして [シンボルの編集] をクリックします。

産業用グラフィック エディタに産業用グラフィックが表示されます。



2. 産業用グラフィックを編集します。詳細については、『産業用グラフィックの作成および管理ユーザー ガイド』を参照してください。

産業用グラフィックの編集では、シンボルに要素スタイルを適用することもできます。要素スタイルの使用の詳細については、『*Application Server* ユーザー ガイド』を参照してください。

3. [保存して閉じる] をクリックします。変更が保存され、産業用グラフィック エディタが終了します。
4. Application Server オブジェクトがインスタンスまたはテンプレートによってホストされている場合、保存して、IDE のオブジェクト エディタを終了します。

WindowMaker でのシンボルの変更の受け入れ

産業用グラフィックが変更され、それを WindowMaker の InTouch ウィンドウで現在使用している場合、WindowMaker で変更をすぐに受け入れることができます。

変更をすぐに受け入れない場合、ウィンドウを閉じてから再度開いたときに WindowMaker でシンボルが更新されます。

また、アプリケーションをテストするために WindowViewer に切り替えた場合、または対象のノードで WindowViewer のアプリケーションを開いた場合、シンボルも更新されます。

WindowMaker でシンボルの変更をすぐに受け入れるには

- 埋め込まれた産業用グラフィックが含まれた開いている InTouch ウィンドウでは、以下のいずれかを実行します。
 - ステータス バーの右下隅にある [シンボルの変更] アイコンをダブルクリックします。

- 埋め込まれた産業用グラフィックが含まれている InTouch ウィンドウを閉じ、再度開きます。

どちらの場合でも、産業用グラフィックに加えられた変更は、InTouch ウィンドウの埋め込まれた産業用グラフィックに反映されます。

WindowViewer でのシンボルの変更の受け入れ

産業用グラフィックが変更され、それを WindowViewer で現在テストしている場合、WindowViewer で変更を受け入れることができます。

埋め込まれた産業用グラフィックのテストの詳細については、「[WindowViewer での産業用グラフィックのテスト](#)」を参照してください。

テスト時に WindowViewer のシンボルの変更を受け入れるには

以下のいずれかを実行します。

- WindowMaker に高速切り替えで戻り、WindowViewer に戻します。
- InTouch ウィンドウを閉じ、再度開きます。これは、[WindowViewer のプロパティ] ダイアログ ボックスで [アプリケーションを常にディスクからロード] オプションがオンになっている場合のみ機能します。

どちらの場合でも、産業用グラフィックに加えられた変更は、InTouch ウィンドウの埋め込まれた産業用グラフィックに反映されます。

InTouch タグを使用したグラフィック要素と産業用グラフィックの作成

産業用グラフィック エディタの [プロパティ] 設定ペインにある [タグ] タブに InTouch アプリケーションで使用するすべてのタグが表示されます。タグをキャンバスにドラッグアンドドロップするだけでグラフィック要素または産業用グラフィックを作成できます。複数のシンボルを一度に作成する場合、ドットフィールドプロパティが自動的に制限されます。この方法を使用すると、グラフィック開発ワークフローを簡素化して、アプリケーション開発時間を大幅に削減できます。

注記: この [タグ] タブは AVEVA Connect とマネージド InTouch の産業用グラフィック エディタでは使用できません。

InTouch タグを使用してグラフィック要素を作成するには

1. 産業用グラフィック エディタの [タグ] ペインからタグをキャンバスにドラッグアンドドロップします。

[グラフィック要素] オプションと [産業用グラフィック] オプションが表示されます。

注記: 複数のタグを一緒にドラッグアンドドロップすると、同じ要素タイプとアニメーションの複数のグラフィック要素を作成できます。

2. デフォルト値を設定するには

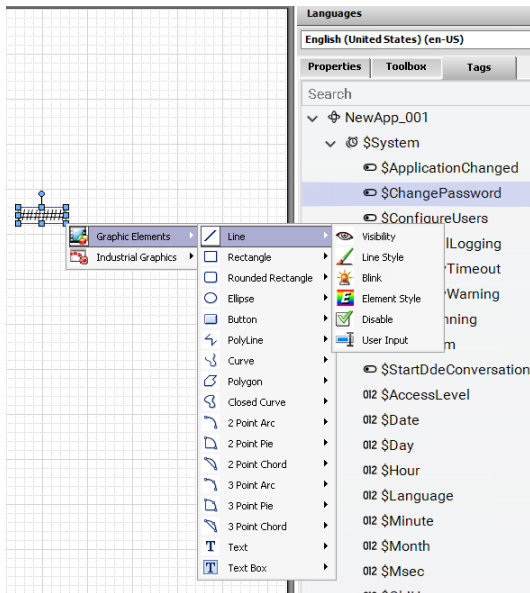
- 画面の任意の場所をクリックします。

または

- [グラフィック要素] オプションの上にカーソルを置き、目的のグラフィック要素とアニメーションを選択します。

グラフィック要素と使用可能なグラフィック アニメーションのデータ型のリストが表示されます。

3. 目的のグラフィック要素の上にカーソルを置きます。
使用可能なアニメーションが表示されます。



4. 目的のアニメーションを選択します。
タグに割り当てられた新しいグラフィック要素が作成されます。

InTouch タグを使用して産業用グラフィックを作成するには

1. 産業用グラフィック エディタの [タグ] ペインからタグをキャンバスにドラッグアンドドロップします。

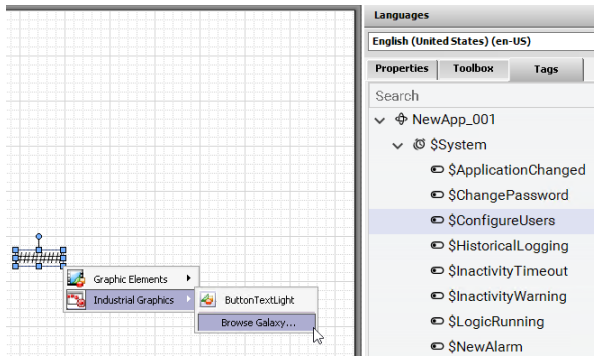
[グラフィック要素] オプションと [産業用グラフィック] オプションが表示されます。

注記: 複数のタグを一緒にドラッグアンドドロップすると、同じタイプの複数の産業用グラフィックを作成できます。

2. デフォルト値を設定するには
 - 画面の任意の場所をクリックします。
 - または
 - [産業用グラフィック] オプションの上にカーソルを置き、目的の産業用グラフィックを選択します。

[Galaxy を参照] オプションが表示されます。

注記: 同じタイプのタグを複数回ドラッグして [産業用グラフィック] オプションの上にカーソルを置くと、以前に選択された 5 つのグラフィック名が [Galaxy を参照] オプションと共に表示されます。



3. 以前に選択したグラフィックのいずれかを選択するか、[Galaxy を参照] をクリックして新しいグラフィックを選択します。
4. [Galaxy を参照] オプションを選択した場合、目的の産業用グラフィックを参照して選択し、[OK] をクリックします。

タグに割り当てられた新しい産業用グラフィックが作成されます。

注記: 選択した産業用グラフィックの [カスタム プロパティ] で、[表示オン/オフ] が [パブリック] に設定されている場合:

- [値] プロパティの [デフォルト値] が空または --- の場合、ドラッグアンドドロップしたタグの名前で置き換えられます。

- ドット プロパティの [デフォルト値] が空または --- の場合、タグ名<該当するドット プロパティ名>で置き換えられます。

上記の両方のケースにおいて、選択した産業用グラフィックのカスタム プロパティの [表示オン/オフ] が [プライベート] の場合、そのプロパティは、作成した産業用グラフィックに表示されません。[表示オン/オフ] が [プライベート] で、他のグラフィックに対して表示しない場合、[表示オン/オフ] が [プライベート] のカスタム プロパティはその他の産業用グラフィックで表示されません。

産業用グラフィック エディタの [ツールボックス] タブからのグラフィックの埋め込み

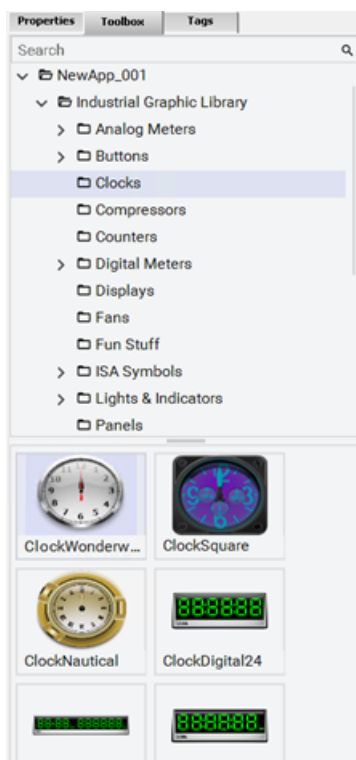
産業用グラフィック エディタの [プロパティ] 設定ペインにある [ツールボックス] タブを使用して、既存の産業用グラフィックを別のグラフィックに埋め込むことができます。[ツールボックス] タブには、InTouch ライブラリで使用できるすべての産業用グラフィックが表示されます。別のグラフィックに埋め込んだグラフィックは、グラフィックのその他のコンポーネントと同様に編集できます。産業用グラフィック エディタの [産業用グラフィックの埋め込み] アイコンを使用してグラフィックを埋め込むこともできます。詳細については、産業用グラフィック エディタ ヘルプの「グラフィックの埋め込み」トピックを参照してください。

産業用グラフィック エディタの [ツールボックス] タブから既存のグラフィックを別のグラフィックに埋め込むには

1. [プロパティ] 設定ペインで [ツールボックス] タブを選択します。
2. フォルダをナビゲートして目的の産業用グラフィックを選択するか、[検索] ボックスでグラフィックを検索します。

選択したフォルダまたはオブジェクト内のグラフィックがナビゲーション領域の下に表示されます。

3. 追加するグラフィックを選択して、キャンバスの目的の位置にドラッグします。

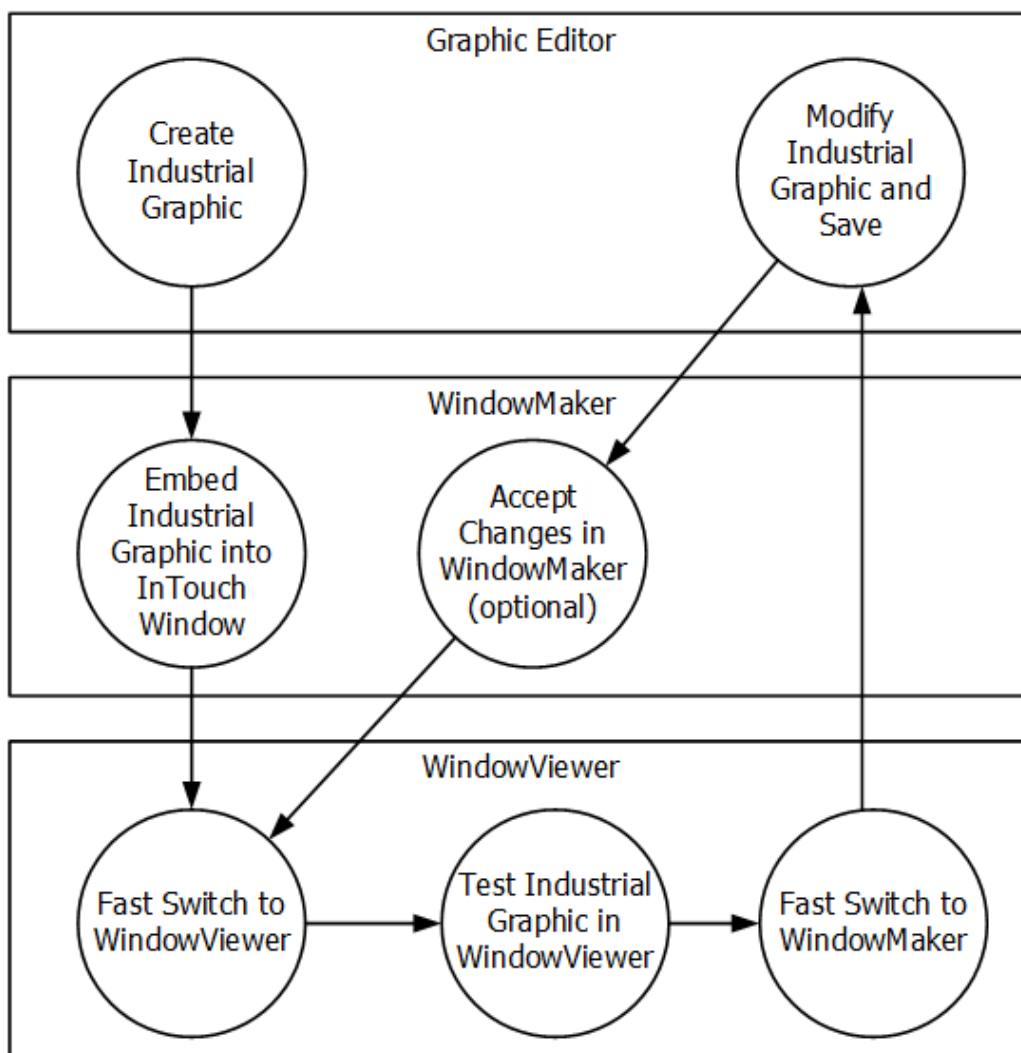


WindowViewer での産業用グラフィックのテスト

埋め込まれた産業用グラフィックを InTouch ウィンドウでテストできます。InTouchViewApp インスタンスを派生させる必要はありません。事前に以下の操作を行っている場合、埋め込まれた産業用グラフィックをテストできます。

- グラフィック ツールボックス、オートメーション テンプレートまたはオートメーション インスタンスでの産業用グラフィックの作成。
- マネージド InTouch アプリケーションの作成。
- マネージド InTouch アプリケーションへの産業用グラフィックの埋め込み。

Developing and Testing Industrial Graphics



埋め込まれた産業用グラフィックを **WindowViewer** でテストするには

1. WindowMaker で **【実行】** をクリックし、WindowViewer に切り替えます。
2. 通常のランタイム環境で行う場合と同様に、埋め込まれたシンボルのアニメーション、動作、および表示方式をテストします。
3. WindowMaker への高速切り替えを行って、産業用グラフィックを埋め込む方法を変更できます。

埋め込まれた産業用グラフィックを **WindowViewer** で変更してテストするには

1. 産業用グラフィック エディタで産業用グラフィックを変更します。
2. 変更を保存します。

WindowViewer が開いてしばらくすると、WindowViewer に変更を受け入れるようメッセージが表示されます。**【はい】** をクリックします。

WindowViewer が終了したら、WindowMaker から WindowViewer に高速切り替えし、変更を確認することができます。開いている WindowViewer セッションに変更が反映されるのを待機するよりも、WindowViewer を終了してから WindowViewer を再度開くほうがより高速です。

グラフィック パフォーマンスの予測

グラフィック パフォーマンス インデックス (GPI) を使用して、ランタイム時に産業用グラフィックのパフォーマンスを測定できます。

このツールは、産業用グラフィック エディタで作成するシンボルがランタイム時に起動するときの推定呼び出し時間を計算します。呼び出し時間は、ユーザーまたはシステムが関連グラフィックの表示を要求してからグラフィックがライブ データで画面に表示されるまでの間隔です。計算は、ランタイム時に InTouch WindowViewer で起動されたシンボルの内容、および外部参照サブスクリプションが完了した場合のベスト ケース時間予測に基づきます。

GPI の使用の詳細については、『産業用グラフィック エディタ ユーザー ガイド』の「グラフィック パフォーマンスの予測」を参照してください。

新しいオートメーション インスタンスの作成

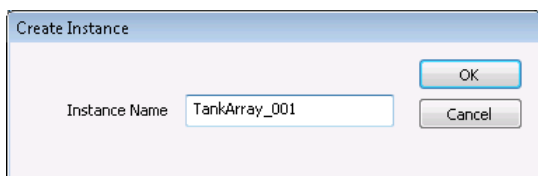
埋め込まれた産業用グラフィックをホストするオートメーション オブジェクトの新しいインスタンスをすばやく作成できます。この方法を使用すると、System Platform IDE に切り替えてインスタンスを派生させる必要がありません。

新しいインスタンスの割り当てが解除されるので、インスタンスを使用する前に System Platform IDE で割り当ておよび配置を行う必要があります。

新しいオートメーション インスタンスを作成できるのは、テンプレートまたはインスタンスによってホストされる産業用グラフィックに対してのみです。グラフィック ツールボックス内の産業用グラフィックには、この機能はありません。

新しいオートメーション インスタンスを作成するには

1. 埋め込まれた産業用グラフィックを右クリックし、[産業用グラフィック] をポイントして [新しいインスタンス] をクリックします。[インスタンスの作成] ダイアログ ボックスが表示されます。



2. [インスタンス名] ボックスで、インスタンスの名前を入力します。
3. [OK] をクリックします。指定した名前を持つテンプレートから、インスタンスが自動的に派生します。

新しい Application Server オブジェクト インスタンスの自動作成

テンプレートから産業用グラフィックを埋め込む場合、InTouch HMI で、そのオブジェクトのインスタンスおよび新しいインスタンスを参照するシンボルのインスタンスを作成できます。

以下の例は、新しい Application Server オブジェクト インスタンスを自動的に作成する方法を示しています。

新しい **Application Server** オブジェクト インスタンスを自動的に作成するには

1. オブジェクト テンプレート **\$Valve1** を作成し、IDE オブジェクト エディタで開きます。
2. [グラフィック] タブで、**ValveSymbol** という名前の産業用グラフィックを追加します。
3. InTouchViewApp オブジェクトの派生テンプレートを作成し、**WindowMaker** で開きます。
4. 新しい InTouch ウィンドウを作成し、オブジェクト テンプレート **\$Valve1** の産業用グラフィック **ValveSymbol** を埋め込みます。インスタンス名を入力するよう求めるメッセージが **WindowMaker** に表示されます。
5. たとえば、**Valve1_E122** などの名前を入力し、[OK] をクリックします。InTouch ウィンドウに産業用グラフィックが貼り付けられ、**Galaxy** でオートメーションオブジェクト インスタンス **Valve1_E122** が作成されます。

InTouch ウィンドウから産業用グラフィックへの変換

マネージド InTouch アプリケーションのウィンドウを産業用グラフィックに変換できます。変換された産業用グラフィックは **WindowMaker** グラフィック ツールボックスおよび IDE グラフィック ツールボックスに表示されます。ウィンドウに表示されるグラフィックに加えて、InTouch スクリプトも **Application Server** スクリプトに変換されます。

ウィンドウの変換の準備

InTouch ウィンドウを変換する前に以下の点に注意してください。

- 産業用グラフィックに変換できるのは InTouch スタンドアロンおよびマネージド アプリケーションのウィンドウだけです。
- 変換するウィンドウは、**WindowMaker** で閉じる必要があります。

ウィンドウの変換

ウィンドウのシンボルおよびスクリプトだけが変換されます。ウィンドウの色、タイプ、フレーム、タイトルバー、サイズコントロール、および [閉じる] ボタンは変換されたシンボルから除外されます。

InTouch のグラフィック タイプに基づいて、ウィンドウグラフィックは次のように変換されます。

- すべての InTouch グラフィック プリミティブは、対応する産業用グラフィック プリミティブに変換されます。
- 1 つの InTouch スマート シンボルは 1 つの産業用の埋め込まれたグラフィックに変換されます。
- ウィンドウ内の 1 つの産業用グラフィックは 1 つの埋め込まれたシンボルに変換されます。埋め込まれたシンボルには、新しいシンボルは作成されません。
- InTouch シンボルは、プロパティ **TreatAsIcon=True** のグループに変換されます。
- InTouch セルは、プロパティ **TreatAsIcon=False** のグループに変換されます。
- InTouch ウィンドウは、**Archestra** ウィンドウ コントロールに変換されます。
- 一部の InTouch グラフィック コンポーネントは産業用グラフィックに変換できません。
 - InTouch のリアルタイム トレンドおよび履歴トレンドは変換できません。
 - **ActiveX** コントロールおよび分散アラーム表示は変換できません。

アニメーション スクリプトの変換

ウィンドウに埋め込まれたすべての InTouch アニメーション リンクは、Application Server の対応するアニメーションに変換されます。

変換中、検証に関する警告やエラー メッセージはログに記録されません。変換されたスクリプトを産業用グラフィック スクリプト検証で検証して、サポートされていないスクリプト構文を見つける必要があります。

InTouch アニメーション スクリプトを変換するとき、以下の例外が発生します。

- 線の色リンクと塗りつぶし色の理論値アラームとアナログ アラームのアニメーション リンクはライン スタイルアニメーションと塗りつぶしスタイルアニメーションのブール型および真理値表に変換されます。
- ShowWindow アニメーション リンクは、ShowGraphic スクリプト関数を含むアクション スクリプトに変換されます。HideWindow アニメーション リンクは、HideGraphic スクリプト関数を含むアクション スクリプトに変換されます。
- アニメーション リンク式で設定されているすべての InTouch タグのタグ名には、"InTouch" というプリフィックスが追加されます。たとえば、Tag1 は Intouch:Tag1 に変換されます。
- アニメーション リンクで設定されている Application Server 属性参照のプリフィックス "galaxy:" は削除されます。たとえば、galaxy:UD001.Value は UD001.Value に変換されます。
- アクション スクリプト内で設定されているすべての InTouch スクリプト関数は変換されません。InTouch タグおよび Application Server 属性参照処理を除くすべてのスクリプトがコピーされます。

ウィンドウ変換の既知の制限事項

InTouch ウィンドウを産業用グラフィックに変換する場合、必ずしも完全な忠実性が維持されるわけではありません。ウィンドウ コンポーネントをシンボルに変換できないことがあります。このセクションでは、InTouch ウィンドウを産業用グラフィックに変換する際の既知の制限事項およびその対処方法について説明します。

- 垂直マウス リリース スライダ SmartSymbol を含むウィンドウの変換

垂直マウス リリース スライダ SmartSymbol には、2 種類のアニメーションが含まれます。シンボルは、塗りつぶしアニメーションを使用して、スケールに対する現在の測定値および値を設定する移動可能なスライダ ノブを示します。埋め込まれた垂直マウス リリース スライダ SmartSymbol を含むウィンドウを産業用グラフィックに変換すると、移動可能なスライダ アニメーションは維持されません。

- Symbol Factory シンボルを含むウィンドウの変換

Symbol Factory シンボルに組み込まれているいくつかのタイプのアニメーションは産業用グラフィックに変換できません。以下のタイプの Symbol Factory シンボル アニメーションは変換できません。

- 塗りつぶしパーセント
- 線/輪郭色
- 水平移動
- 垂直移動

- InTouch ActiveX コントロール、InTouch OCX DLL、または InTouch DLL オブジェクトを含むウィンドウの変換

ウィンドウ変換では、InTouch ActiveX コントロール、InTouch OCX DLL、および InTouch DLL をサポートしていません。これらのコンポーネントは、新しいシンボルには含まれません。

- InTouch 翻訳文字列から産業用グラフィックへの移行

InTouch 翻訳文字列は、アプリケーションのフォルダ内の XML ファイルに格納されます。各言語の翻訳文字列は、個別の XML ファイルに配置されます。すべてのウィンドウおよび SmartSymbol 文字列の翻訳は、ローカライズされた文字列をインポートした後、アプリケーションディレクトリ内の XML ファイルから使用できます。

以下の手順は、言語アシスタント ツールを使用して、InTouch ウィンドウ ローカライゼーション文字列を産業用グラフィックに移行する方法を示します。

- a. InTouch ウィンドウから産業用グラフィックへの変換
 - b. InTouch 言語 XML ファイルの内容を言語アシスタントにインポートして、語句のグローバル ディクショナリを作成します。
 - c. 産業用グラフィックを XML ファイルにエクスポートします。
 - d. シンボル XML ファイルを言語アシスタントにインポートします。グローバル ディクショナリの翻訳が産業用グラフィックの語句に自動的に適用されます。
 - e. 言語アシスタントから産業用グラフィック XML ファイルをパブリッシュします。
 - f. 翻訳されたテキスト文字列を含む産業用グラフィック ツールボックスに XML ファイルをインポートします。
- InTouch スクリプトから産業用グラフィックへの移行

InTouch または QuickScript 関数を含むスクリプトは産業用グラフィックに変換できません。

- InTouch 履歴オブジェクトから産業用グラフィックへの移行

InTouch 履歴トレンドを含む InTouch ウィンドウは、産業用グラフィックには完全に変換されません。変換されたシンボルに履歴トレンドの一部だけが表示されることがあります。スクータバーなどのトレンドコンポーネントは、変換された産業用グラフィックに表示されません。

ウィンドウを変換した後

変換されたシンボルは、System Platform IDE ツールボックスおよび WindowMaker の産業用グラフィック ツールボックスに追加されます。変換された元の InTouch ウィンドウのバックアップが作成されます。新しい InTouch ウィンドウが InTouch アプリケーションに作成され、新しく作成された産業用グラフィックが埋め込まれます。

ウィンドウのサブセットしか変換されない場合、変換されたウィンドウが未変換のウィンドウと共に機能することを手動で確認する必要があります。

変換されたウィンドウの名前が新しい産業用グラフィックの名前としてデフォルトで割り当てられます。InTouch ウィンドウの名前にサポートされない文字が含まれている場合、サポートされない各文字はアンダースコア (_) で置き換えられます。シンボルが既に存在する場合、変換されたシンボルの名前に数字のサフィックスが追加されます (Main_001 など)。

例外は、ドル記号 (\$)、シャープ記号 (#)、およびアンダースコア (_) だけです。

アプリケーションを移行した場合、ウィンドウの編集、またはウィンドウプロパティの変更を行うと、ウィンドウ名に含まれる特殊文字はアンダースコア（_）で置き換えられます。この処理は、アプリケーション、枠、およびテンプレートを始めとするすべてのタイプのウィンドウに適用されます。

新しいツールセットが作成され、変換されたすべてのシンボルに InTouch アプリケーション名が割り当てられます。ウィンドウ変換の後、InTouch のフォルダ階層は、ツールセットによって維持されます。

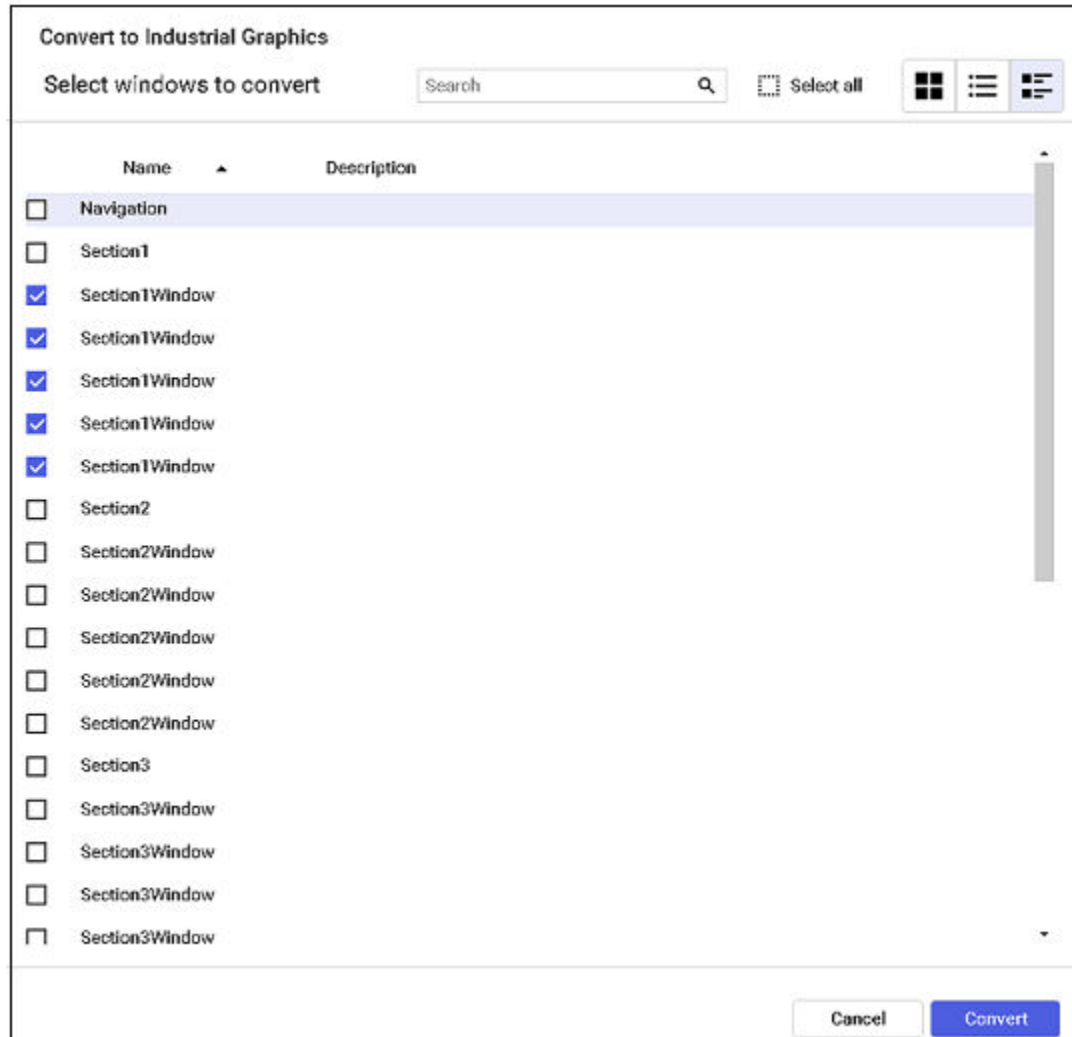
- 割り当てられていない InTouch ウィンドウから変換されたシンボルは、InTouch アプリケーション名が割り当てられたツールセットに追加されます。
- ウィンドウが割り当て済み領域に属する場合、元のフォルダ構造が両方のツールボックス内に作成され、InTouch アプリケーションの名前が最上位のルート フォルダ名として割り当てられます。

ウィンドウ変換手順の完了

ウィンドウを産業用グラフィックに変換するには

1. 変換するすべての InTouch ウィンドウを閉じます。
2. 変換するウィンドウを [ウィンドウ] ペインで選択します。
3. 右クリックしてショートカットメニューを表示し、[産業用グラフィックに変換...] を選択します。
[変換するウィンドウを選択] 画面が表示されます。
4. 変換するウィンドウを選択します。

デフォルトでは、以前の画面で選択したウィンドウが選択されます。追加のウィンドウを選択できます。



5. [変換] をクリックします。

選択した複数のウィンドウが連続して変換されていることを示すメッセージが表示されます。ウィンドウが変換された後、複数の [チェック イン] ダイアログ ボックスが連続して表示され、変換された各ウィンドウのコメント（オプション）を入力できます。

6. WindowMaker の産業用グラフィック ツールボックスおよび System Platform IDE のグラフィック ツールボックスを確認します。

変換されたウィンドウが両方のツールボックスに産業用グラフィックとして表示されていることを確認します。

ウィンドウ変換エラーの診断

変換中、進行状況バーにウィンドウ変換中に発生した警告メッセージやエラー メッセージが表示されます。ウィンドウの変換中、各エレメント、アニメーション、またはスクリプトに関する有益な診断情報を含むカスタム ログ フラグが作成されます。

変換後、進行状況バーの [Window conversion Report]（ウィンドウ変換レポート）をクリックして、すべてのメッセージを HTML 変換レポートにエクスポートします。このレポートは、メッセージ ウィンドウで表示できます。

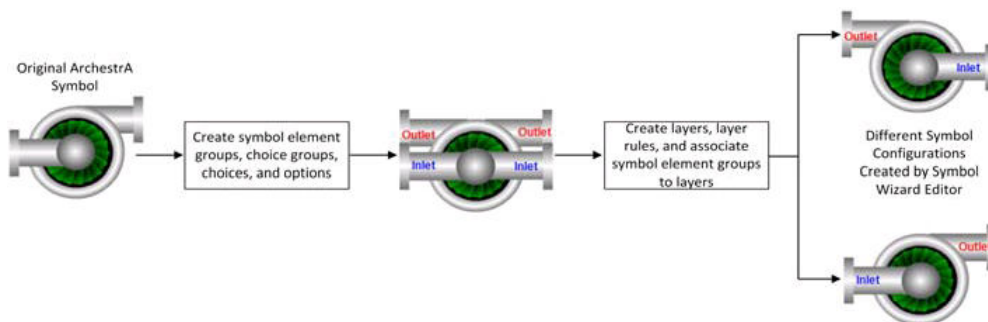


変換の進行状況ダイアログボックスと変換レポートの両方には、ウィンドウ変換に関する以下の情報が含まれます。

- ウィンドウ名。
- ウィンドウが正常に変換されたかどうか、またはエラーが発生したどうかを示す変換ステータス。
- 変換中に発生した警告メッセージ。
- ウィンドウ変換中に発生したエラーメッセージ。

シンボル ウィザード エディタでのシンボル ウィザードの作成

産業用グラフィック エディタには、シンボルのさまざまな表示設定および機能設定を作成できるシンボル ウィザード エディタが含まれています。複数の設定を含むシンボルは、シンボル ウィザードと呼ばれます。シンボル ウィザードの作成の例は、中央のブレードハウジングの左または右の排水ポンプシンボルです。シンボル ウィザード エディタを使用して、両方の設定を1つのシンボル ウィザードに含めることができます。



シンボル ウィザードは、特定の Application Server オブジェクト テンプレートや Application Server オブジェクト インスタンスに関連付けられていません。特定のシンボル設定を選択できることを除き、シンボル ウィザードは標準の産業用グラフィックと同様に動作します。

シンボル ウィザードには複数の異なる設定を含めることができるので、アプリケーションに対して作成する必要のあるシンボルの数が削減されます。その結果、保守労力が削減され、InTouch HMI アプリケーションを構築する際の設計時の問題も少なくなります。

シンボル ウィザードの作成

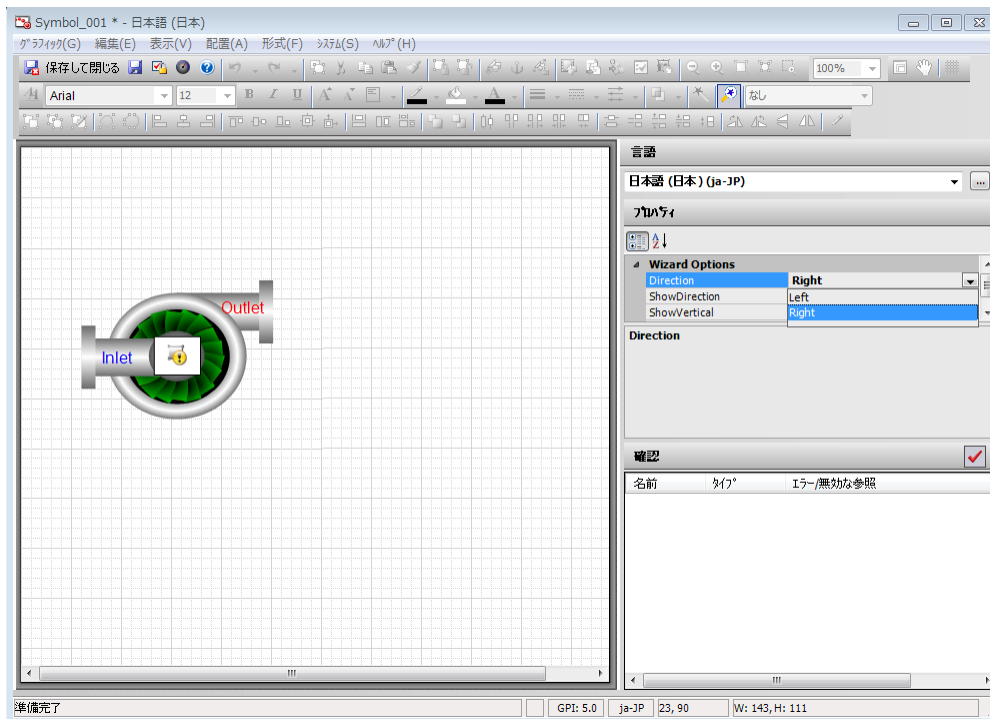
シンボル ウィザードの作成および実装のプロセスには、デザイナー ワークフローとコンシューマ ワークフローという 2 つのワークフローがあります。

- デザイナは、シンボル ウィザード エディタを使用して、1 つのシンボルの複数の設定を作成します。
- コンシューマは、シンボル ウィザードの適切な設定を選択して、シンボルをマネージド InTouch アプリケーションに埋め込みます。

シンボル ウィザードのデザイナー ワークフロー

デザイナーはシンボル ウィザード エディタを使用して、さまざまなシンボル設定を定義し、グラフィック要素、カスタムプロパティ、およびスクリプトをレイヤーの形式でシンボル設定に関連付けます。規則は、レイヤーをシンボル設定に含めるときを決定する選択グループ、選択、およびオプションに関連付けられます。デザイナーは、シンボルのデフォルトとなる設定を選択します。このデフォルト設定は、シンボルがマネージド InTouch アプリケーションに埋め込まれたときに表示されます。

すべてのシンボル設定を作成した後、デザイナーは、シンボル ウィザードプレビューを使用して、シンボルの各設定をコンシューマに示す方法を検証します。デザイナーは、「ウィザードオプション」を使用して、シンボルに対して設定されているレイヤー規則に基づいて設計されたとおりに各設定が表示されることを確認します。



デザイナーは、マネージド InTouch アプリケーションに埋め込むことができるように、終了したシンボル ウィザードを Galaxy ライブラリに保存します。

シンボルウィザードの作成に関する詳細については、『産業用グラフィック エディタ ユーザー ガイド』を参照してください。

シンボルウィザードのコンシューマ ワークフロー

コンシューマがシンボルウィザードのインスタンスをマネージド InTouch アプリケーションに埋め込むと、シンボルのデフォルト設定が選択されます。コンシューマは、シンボルウィザードの[ウィザード オプション] ビューからオプションを選択して設定を変更できます。選択した設定によっては、コンシューマが選択できる追加の設定関連プロパティがあります。

InTouch アプリケーションが実行しているとき、シンボルウィザードは、コンシューマによって選択された設定として表示されます。シンボルの設定は、ランタイムで変更することはできません。

シンボルウィザードをマネージド InTouch アプリケーションに埋め込む方法の詳細については、「[シンボルウィザードの埋め込み](#)」を参照してください。

トレンド ペンの履歴データ検索について

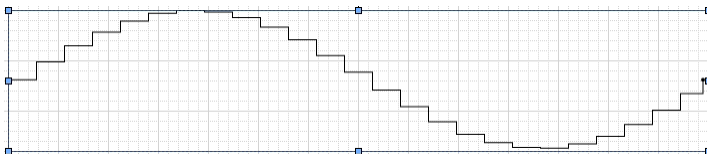
トレンド ペンを含むアプリケーションが WindowViewer で実行開始したとき、Historian クエリーが全体のトレンド ペン期間のデータを取得します。リアルタイム データは履歴データが取得されている期間中にトレンドライン上にプロットされます。履歴データは検索後、リアルタイム データに追加され、全体期間についてトレンドラインをバックフィルします。

以下はトレンド ペンを設定するための手順です。通常、トレンド ペンの設定には、トレンド ペンプロットがグラフィックのプロセス値の経時の変化を表示することを視覚的に示すためのメーター グラフィックの隣にトレンド ペンを配置する複数のステップが含まれます。

トレンド ペンを設定するには

1. [視覚化] フォルダから [トレンド ペン] をクリックします。
カーソルを産業用グラフィック エディタのキャンバス領域に置くと、十字型に変化します。
2. トレンド ペンプロットの水平の境界として設定するキャンバス上の位置にカーソルを置きます。
3. キャンバスでドラッグして、トレンド ペン コントロールの長方形を選択します。

トレンド ペンの長方形の水平方向および垂直方向の境界は、ランタイム時におけるトレンド ペンプロットの描画領域を表します。長方形の水平軸はトレンドの期間を表します。垂直軸はトレンドの可能な値の範囲を表します。



[トレンド ペン] ダイアログ ボックスが表示されます。トレンド ペン グラフィックをダブルクリックするか、または [システム] メニューの [アニメーションの編集] を選択し、[トレンド ペン] ダイアログ ボックスを表示できます。

4. [参照] フィールドに参照を入力します。

参照はトレンドによって値として表示されるデータ ソースで、オブジェクトの属性、アナログ タグ 値、またはカスタム プロパティのような外部参照にできます。定数と式は許可されません。

5. **[Historian]** または **[InTouch Log History/LGH]** を選択して、履歴ソースを表示します。**[Historian]** を選択した場合、手順 7 に進みます。
6. **[InTouch Log History/LGH]** を選択した場合、**UNC パス** に対してアイコンを使用して、フィールドへの入力を式または静的テキストとして切り替えることができます。
 - 式モードを選択した場合、参照ボタンをクリックして HMI の属性/タグブラウザを起動し、カスタムプロパティ、属性、またはタグを選択できます。
 - 静的テキストモードを選択する場合、省略ボタンをクリックしてファイルブラウザを起動し、LGH ファイル名を指定できます。
7. Historian の位置を識別するメソッドに、**[自動検出]** または **[式]** を選択します。

- **自動検出:** Historian Server は参照属性が実行されている AppEngine から自動検出されます。たとえば、**[参照]** フィールドが **UDO.UDA1** に設定されている場合、**[自動検出]** は、UDO が実行されている AppEngine に設定された Historian Server 名に設定されています。自動検出は Application Server リファレンスにのみ有効です。
- **式:** 式または参照を **[サーバー名]** フィールドに入力すると、指定した Historian Server にトレンドペンが関連付けられます。

[サーバー名] フィールドの左にあるアイコンは、フィールドへの入力を式モードまたは静的テキストモードの間で切り替えます。

トレンドペンは、**[式]** モードで **[サーバー名]** フィールドが空白の場合にライブデータのみ表示します。

8. トレンド期間のタイプとして、**[移動]** または **[固定]** を選択します。
 - **移動:** トレンド期間の開始時間は現在の時刻で、終了時間は開始時間からの期間です。次の期間の開始時間は、過去のトレンド期間の終了時間に設定されます。
 - **固定:** 固定トレンド期間において、**StartTime** および **EndTime** プロパティは自動で変化しません。初期値では、トレンド期間の開始時間は現在の時間です。**StartTime** プロパティはスクリプトによって変更できます。

トレンド期間の **EndTime** プロパティ（移動と固定の両方）は読み取り専用です。トレンド期間の終了時間は、指定した開始時間と期間から計算されます。

9. **[経過時間 (分)]** フィールドにトレンドラインの X 軸期間を設定します。

トレンド期間は定数、外部参照、式、またはカスタムプロパティとして指定できます。浮動小数点数を入力した場合、期間は最近似の分に丸められます。

最小トレンド期間は 1 分で、最大期間は 10080 分（1 週間）です。

10. プロセス値をトレンドラインに配置するために、**[自動範囲]** または **[範囲外の値を除外]** をスケールメソッドとして選択します。

[自動範囲] を選択する場合、**[最小範囲]** と **[最大範囲]** フィールドが無効になります。トレンドラインの Y 軸は、自動で調整され、トレンドペングラフィックの上限と下限の間にあるトレンド値の全範囲を表示します。

[範囲外の値を除外] を選択する場合、**[最小範囲]** と **[最大範囲]** フィールドが有効になります。**[最小範囲]** と **[最大範囲]** はトレンドの Y 軸値範囲の下限と上限を設定します。どちらのフィールドも、定数、外部参照、カスタムプロパティに設定できます。

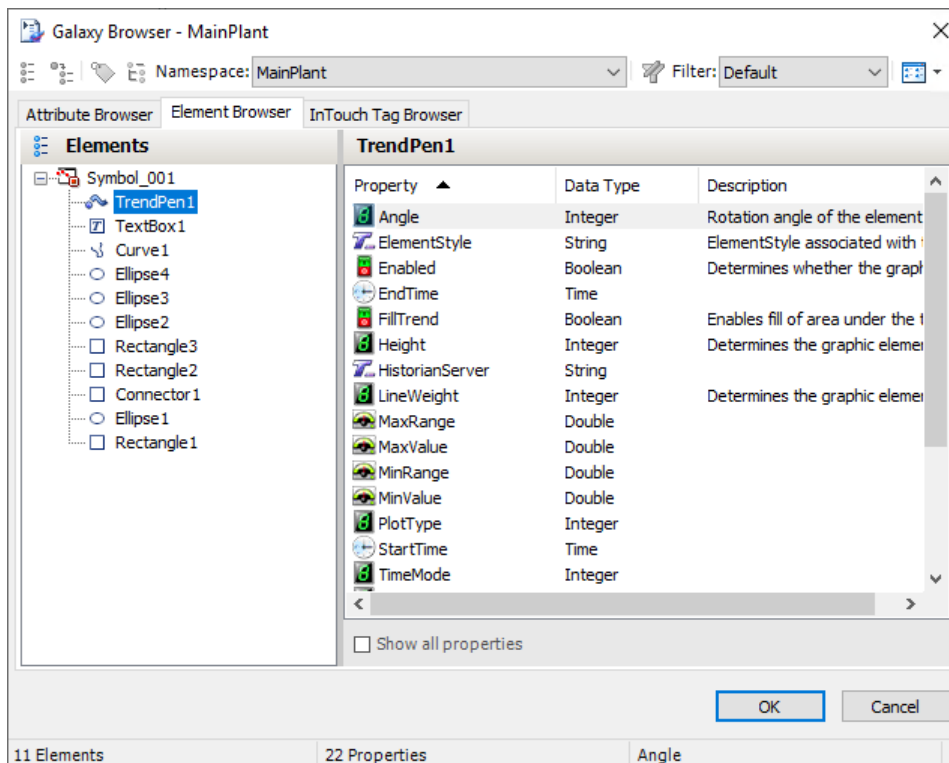
【範囲外の値を除外】を使用したトレンドの下限と上限を値が超える場合、トレンドラインは値の範囲の制限で切り捨てられ、プロセス値がトレンド値の範囲の外にあるときに水平線として表示されます。

11. [プロットタイプ] で、[ステップライン] または [線] をトレンドプロットのタイプとして選択します。

- [ステップライン] プロットはトレンドデータポイントから次のデータポイントの時間まで、トレンドのX軸に水平の線を描画し、データポイントまで垂直の線を描画します。
- [線] プロットはトレンド期間中の各連続ポイントを線で直接結びます。

ランタイム時のトレンドペンプロパティの変更

トレンドペンに含まれるプロパティの値は、ランタイム時に修正して、トレンドの視覚的および機能的な特性を変更できます。次の図は、ランタイム時に変更できる属性、タグ、または値を検索するための一般的な HMI ソフトウェアブラウザまたはエクスプローラの一例です。



以下のサブトピックでは、一般的にランタイム時に修正して、トレンドペンの視覚的、機能的な特徴を変更するプロパティについて説明します。

MinValue プロパティ

ランタイム時に、MinValue プロパティの値を修正して、トレンドに表示される最小の測定値を変更できます。デザイン時にトレンドペンの [Y 軸範囲] オプションに割り当てられた値に基づいて、ランタイム時に、MinValue は読み書き可能、または読み取り専用プロパティにできます。

- デザイン時に [Y 軸範囲] を [自動範囲] に設定する場合

トレンドによって表示される最小測定値は、履歴またはトレンド期間の現在のデータから受け取ったデータの最小値に設定されます。**MinValue** は読み取り専用です。

- デザイン時に **〔Y 軸範囲〕** を **〔範囲外の値を除外〕** に設定する場合

トレンドによって表示される最小測定値は、デザイン時における **〔最小範囲〕** オプションの下限に設定されます。

MinValue は読み書き可能で、ランタイム時に変更できます。ランタイム時に **MinValue** を変更する場合、トレンドラインは **MinValue** および **MaxValue** プロパティに割り当てられた値に基づいて再描画されます。

MinValue および **MaxValue** プロパティに割り当てられた値が等しい場合、トレンドの Y 軸範囲は自動的に自動範囲に変更されます。

MaxValue プロパティ

ランタイム時に、**MaxValue** プロパティの値を修正して、トレンドに表示される最大の測定値を変更できます。デザイン時にトレンドペンの **〔Y 軸範囲〕** オプションに割り当てられた値に基づいて、ランタイム時に、**MaxValue** は読み書き可能、または読み取り専用プロパティにできます。

- デザイン時に **〔Y 軸範囲〕** を **〔自動範囲〕** に設定する場合

トレンドによって表示される最大測定値は、履歴またはトレンド期間の現在のデータから受け取ったデータの最大値に設定されます。**MaxValue** は読み取り専用です。

- デザイン時に **〔Y 軸範囲〕** を **〔範囲外の値を除外〕** に設定する場合

トレンドによって表示される最大測定値は、デザイン時における **〔最大範囲〕** オプションの上限に設定されます。

MaxValue は読み書き可能で、ランタイム時に変更できます。ランタイム時に **MaxValue** を変更する場合、トレンドラインは **MinValue** および **MaxValue** プロパティに割り当てられた値に基づいて再描画されます。

MinValue および **MaxValue** プロパティに割り当てられた値が等しい場合、トレンドの Y 軸範囲は自動的に自動範囲に変更されます。

StartTime プロパティ

ランタイム時に、**StartTime** プロパティの値を修正して、デザイン時にトレンドペン **〔実行周期〕** に設定された値に基づいて、トレンド期間の開始時間を変更します。

- デザイン時に **〔実行周期〕** を **〔固定〕** に設定する場合

StartTime プロパティに割り当てられているデフォルト値は、トレンドペンが **WindowViewer** に最初に表示されるときで、開始時間は時間の経過と共に変化します。**StartTime** は読み取り/書き込みで、ランタイム時に変更できます。**StartTime** の値が変化すると、トレンドペンは新しい **StartTime** 値を使用してトレンドを再プロットします。

- デザイン時に **〔実行周期〕** を **〔移動〕** に設定する場合

StartTime プロパティに設定された値は現在のシステム日付時間です。**StartTime** は読み取り専用です。

EndTime プロパティ

ランタイム時に、EndTime プロパティの値を修正して、デザイン時にトレンドペン [実行周期] に設定された値に基づいて、トレンド期間の終了時間を変更します。

- デザイン時に [実行周期] を [固定] に設定する場合

EndTime プロパティに割り当てられたデフォルト値は、デザイン時に設定された終了時間です。

EndTime は読み書き可能で、ランタイム時に変更できます。EndTime の値が変化すると、トレンドペンは新しい EndTime 値を使用してトレンドを再プロットします。

- デザイン時に [実行周期] を [移動] に設定する場合

EndTime プロパティに設定された値は現在のシステム日付時間です。EndTime は読み取り専用です。

PlotType プロパティ

ランタイム時に、PlotType プロパティの値を修正して、トレンドプロットのタイプを変更できます。

- PlotType が 0 のとき、トレンドペンプロットタイプはステップラインになります。(デフォルト)
- PlotType が 1 のとき、トレンドペンプロットタイプは線になります。

PlotType の値は値が 0 でも 1 でもない場合は無視されます。

ランタイム時に PlotType の値が変化すると、トレンドを描画する前に、トレンドデータは再度読み出されます。

TimeMode プロパティ

ランタイム時に、TimeMode プロパティの値を修正して、トレンド期間のタイプを変更できます。

- TimeMode が 0 のとき、トレンド期間のモードは移動です。トレンドの終了時間は現在の時間です。(デフォルト)
- TimeMode が 1 のとき、トレンド期間のモードは固定です。トレンドの開始時間は、トレンドペンが WindowViewer に最初に表示されるときで、開始時間は時間の経過と共に変化します。

TimeMode の値は、値が 0 でも 1 でもない場合は無視されます。

ランタイム中、期間が移動から固定に変化した場合、トレンドの開始時間と終了時間は以前と同じになり、データも維持されます。ランタイム中、期間が固定から移動に変化した場合、トレンドを描画する前にデータが再度読み出されます。トレンドの開始時間と終了時間は移動モードによって自動で調整されます。

FillTrend プロパティ

ランタイム時に FillTrend プロパティを使用して、トレンドペン曲線の下の領域の外観を変更できます。

FillTrend プロパティは、スクリプトを使用して変更することもできます。たとえば、`TrendPen6.FillTrend = not TrendPen6.FillTrend;` と指定します。

一般的な要素スタイルの手順ルールに従って、塗りつぶしスタイルアニメーションおよび要素スタイルアニメーションの Fill プロパティ、または要素スタイルが FillTrend プロパティに適用されるようになりました。

塗りつぶしスタイルアニメーションの詳細については、「塗りつぶしスタイルの設定」を参照してください。

要素スタイルアニメーションの詳細については、「アニメーションスタイルを使用したアニメーションの設定」を参照してください。

注記: FillTrend プロパティはデザイン時またはランタイム時に有効化することはできます。デザイン時に色が選択されていない場合、ランタイム時、色はデフォルトで白に設定されます。

マルチ ペン トレンドの設定

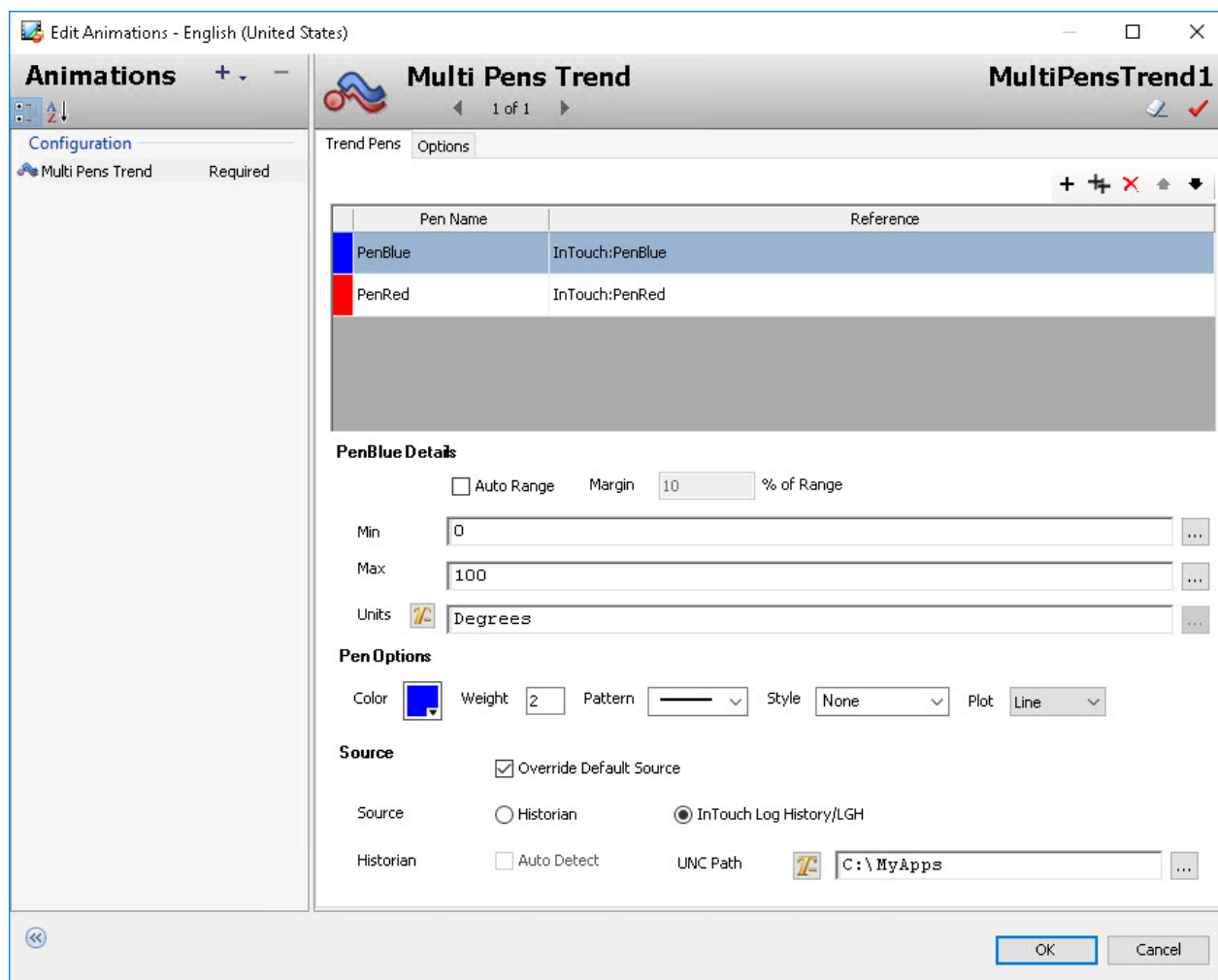
マルチ ペン トレンドはトレンドペンと同様に一連のプロセス値を表示しますが、複数のトレンド線が表示されます。トレンドペンのタイプの詳細については、「トレンドペンの設定」内のセクションを参照してください。

注記:

- マルチ ペン トレンドは英語以外のオペレーティング システムではテストされていません。
 - マルチ ペン トレンドのパフォーマンス インデックス 計算は GPI に含まれていません。
 - AVEVA OMI ViewApps はマルチ ペン トレンドをサポートしません。
 - Web Client では、マルチ ペン トレンドがサポートされません。
-

マルチ ペン トレンドを設定するには

1. [マルチ ペン トレンド] 要素を選択します。
2. グラフィック エディタ キャンバス上をクリックします。
3. [アニメーションの編集] ダイアログ ボックスが表示されます。各ペンに対して、[トレンドペン] タブの以下のセクションを設定します。
 - ペン名と参照
 - ペンの詳細
 - ペンのオプション
 - ソース



ペン名と参照の設定

[トレンド ペン] タブで各ペンに対して以下の設定を行います。

1. [ペン名] フィールドにペンの名前を入力します。

ペン名フィールドには以下のルールが適用されます。

- 入力できるのは静的テキストだけです。
- 1つの文字を含む必要があります。
- 最大文字数は32です。
- 有効な文字は、英数字および特殊文字（\$、#、_）です。
- 2つのトレンド ペンに同じ名前を付けることはできません。
- ペン名では大文字と小文字が区別されません。

設定したペン名はツールヒントに表示されます。

2. [参照] フィールドに参照を入力します。フィールドの横にある省略記号ボタンを使用して参照を選択します。

参照はトレンドによって値として表示されるデータソースで、オブジェクトの属性、InTouch アナログ型タグ値、またはカスタムプロパティのような外部参照にできます。定数と式は許可されません。詳細については、「[入力モードの設定 \(\)](#)」を参照してください。

3. デフォルトでは、2つの空の行がグリッドに作成され、最大8つのペンを作成できます。
- 4.トレンドペンを追加するには、**+** アイコンを使用します。トレンドペンを削除するには、**✗** アイコンを使用します。ペンの表示順序を変更するには、**◆** および **▼** の矢印を使用します。既存のペン設定を複製するには、**✚** アイコンを使用します。

ペンの詳細の設定

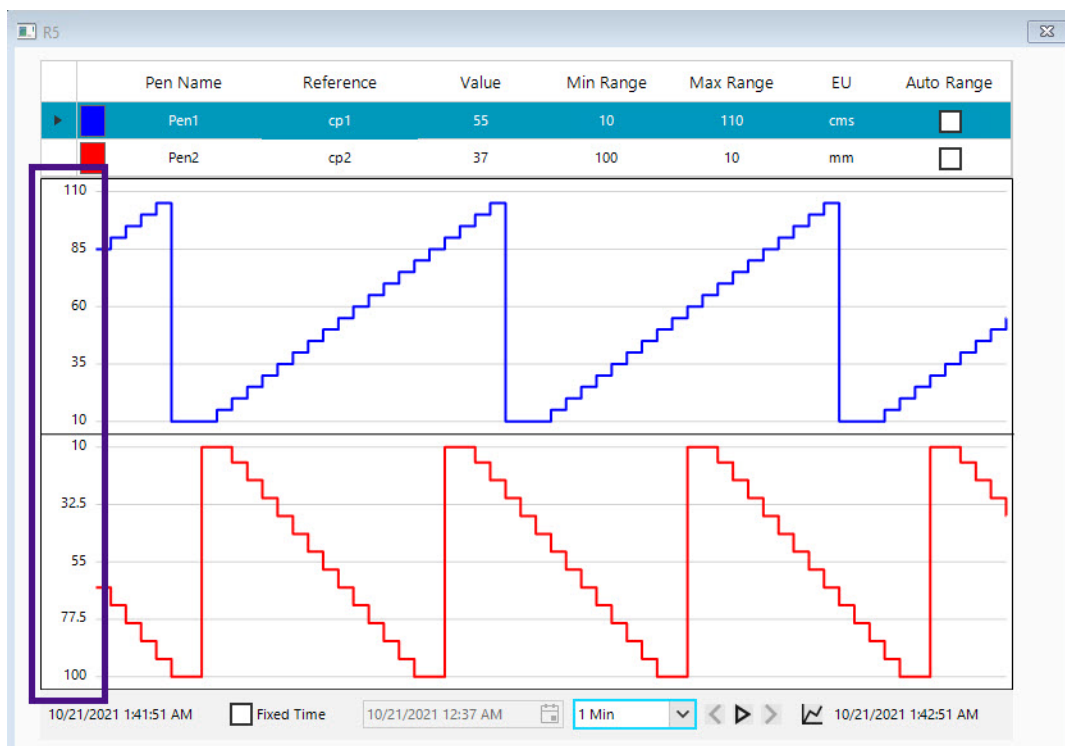
ペンの詳細セクションで個々のペンの範囲と単位を設定できます。

1. スケール方法に **【自動範囲】** を選択すると、トレンド線にプロセス値が配置されます。
 【自動範囲】 を選択すると **【最小】** と **【最大】** フィールドが無効になります。トレンド線のY軸は、自動で調整され、トレンドの上限と下限の間にあるトレンド値の全範囲を表示します。
 - a. **【マージン】** のパーセンテージ値を指定します。ランタイム時、トレンド領域のマージンはトレンド値と指定したマージンのパーセントに基づいて動的に表示されます。これは表示のみを目的としていて、軸の値には影響しません。
次に例を示します。最小値が10、および最大値が20に設定されている場合を考えてみます。範囲は10 (20-10) です。
20% のマージンを指定すると、マージンは「範囲 x マージン %」として計算されるので、この例では $10 \times 20\% = 2$ になります。
Y軸の最大値は「最大値 + マージン」です。この例では $20 + 2 = 22$ になります。Y軸の最小値は「最大値 - マージン」です。この例では $10 - 2 = 8$ になります。

2. **【自動範囲】** を選択しない場合、選択したペンのY軸の **【最小】** と **【最大】** の値を指定します。

値を直接指定するか、**【...】** .Galaxy ブラウザが表示されます。

最小値が最大値よりも大きい場合、トレンドペンはランタイム時に違う形で描画され、スケールは最大値が最小値よりも大きい場合と逆になります。



3. 属性またはタグを参照して選択します。
4. [単位]を入力します。トグル ボタンを使用して静的テキストを入力するか、Galaxy ブラウザから参照値を選択します。次に例を示します。角度を選択します。
5. [OK] をクリックします。

ペン オプションの設定

[ペン オプション] セクションを使用して個々のペンのスタイルを設定できます。

1. 色: ペンの表示色を選択します。
 - a. カラー ピッカーを使用して画面の部分から色を選択できます。
 - b. [パレットのロード...] をクリックしてカスタム パレットを読み込むこともできます。
 - c. + を押して色をカスタム パレットに追加することや、Delete ボタンを押して色を削除することもできます。
 - d. 設定したカスタム パレットは保存して後で使用できます。[パレットの保存] をクリックします。
2. 太さ: ペンの太さを入力します。トレンド ペン線の太さが指定されます。
3. パターン: 事前定義済みのオプションのリストからトレンド ペン線の表示パターンを選択します。
4. スタイル: 事前定義済みの [スタイル] オプションのリストから選択できます。スタイルは IDE の [要素スタイル] で設定されます。詳細については、「要素スタイルの視覚的プロパティの変更」を参照してください。
5. プロット: これは表示されるトレンド線のタイプです。[ステップ ライン] または [線] を選択します。

6. [OK] をクリックします。

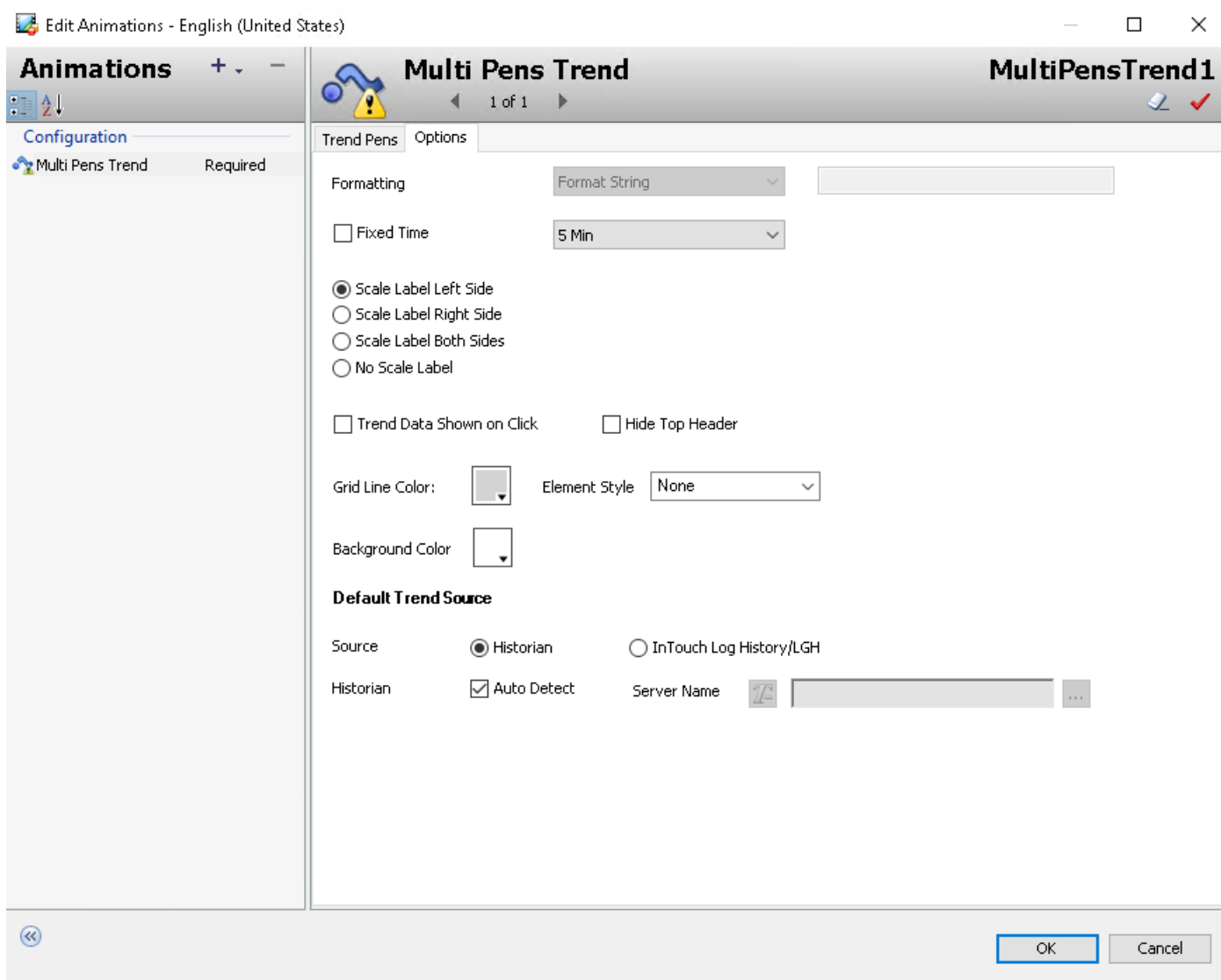
ソースの設定

[オプション] タブで設定したデータの履歴ソースを上書きできます。詳細については、「[マルチ ペン トレンドのカスタマイズ \(\)](#)」を参照してください。

- 履歴データのソースを変更するには、[デフォルト ソースの上書き] をクリックします。

マルチ ペン トレンドのカスタマイズ

[オプション] タブを使用してランタイム時のマルチ ペン トレンドの表示をカスタマイズできます。



- [書式設定] オプションは無効化されます。
- [固定時間] 設定は、固定時間モード（選択時）および移動モード（非選択時）を切り替えることができます。[固定時間] チェック ボックスをオンにすると、指定した期間のトレンドデータがトレンド領域に表示され、移動しません。

3. ドロップダウン リストの事前定義済みオプションのリストから [時間間隔] を選択します。ここで選択した時間間隔がランタイム時の初期値として設定されます。
4. スケール ラベルを表示するトレンド領域内の場所を選択します (左、右、両方、または非表示)。
5. [クリック時にトレンドデータを表示] をクリックすると、ランタイム時のカーソル読み出しダイアログの動作をカスタマイズできます。

デフォルトでは、カーソルをトレンド領域の上に置くとカーソル読み出しダイアログが表示されます。このオプションを選択すると、トレンド領域をクリックしたときにカーソル読み出しダイアログを表示することや非表示にすることができます。
6. 上部ヘッダーを非表示にするには、[上部ヘッダーの非表示] を選択します。トレンド領域とツールバーだけが表示されます。
7. [グリッド線の色] オプションを使用してグリッド線の色を選択します。
8. [要素スタイル] ドロップダウンリストから定義済みスタイルを選択します。
9. [背景色] オプションを使用して背景の色を選択することもできます。

10. [デフォルトのトレンド ソース] の [ソース] フィールドで [Historian] または [InTouch ログ履歴/LGH] を選択します。

デフォルトでは、[Historian] ソースおよび [自動検出] オプションが選択されます。[自動検出] を選択すると、[サーバー名] が無効になります。

式または参照を [サーバー名] フィールドに入力すると、指定した **Historian Server** にマルチ ペントレンドが接続されます。[サーバー名] フィールドの左にあるボタンを使用して、フィールドへの入力モード (式または静的テキスト) を切り替えます。

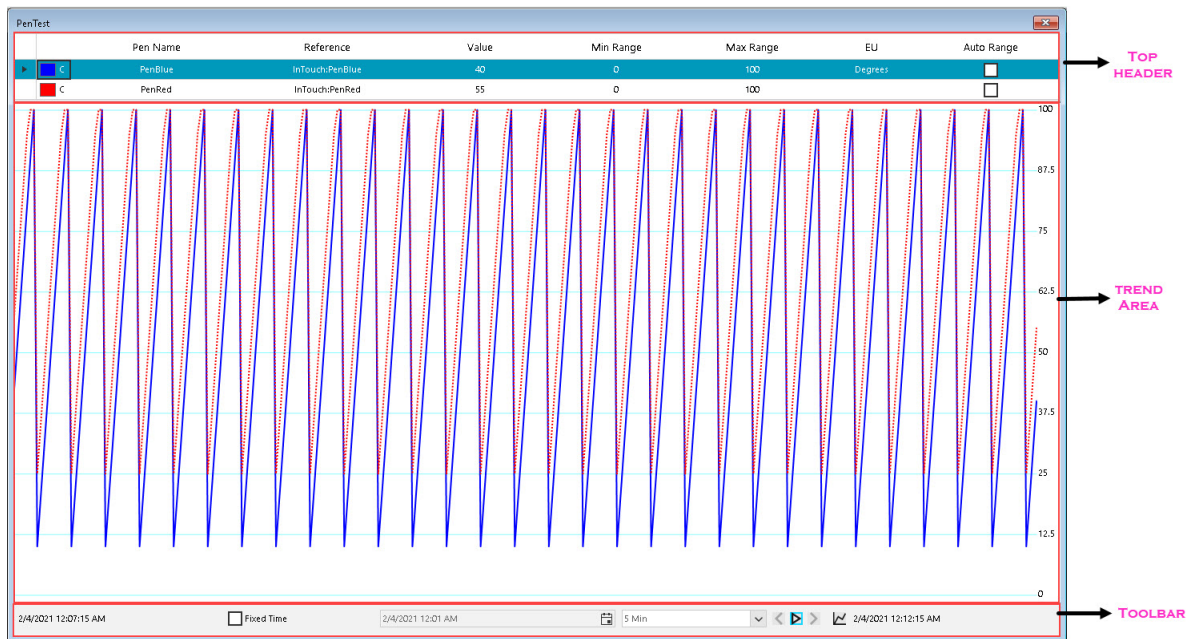
Historian Server は参照属性が実行している **AppEngine** から自動的に検出されます。たとえば、[参照] フィールドが **UDO.UDA1** に設定されている場合、[自動検出] は、**UDO** が実行している **AppEngine** に設定された **Historian Server** の名前に設定されます。[自動検出] は **Application Server** 参照に対してのみ有効です。トレンド ペンは、[式] モードで [サーバー名] フィールドが空白の場合にライブ データのみ表示します。

11. [InTouch ログ履歴/LGH] を選択した場合、[UNC パス] フィールドの横にあるボタンを使用して、フィールドへの入力モード (式または静的テキスト) を切り替えます。
 - a. 式モードを選択した場合、省略記号ボタンをクリックして **HMI** の属性/タグ ブラウザを起動し、カスタム プロパティ、属性、またはタグを選択できます。
 - b. 静的テキスト モードを選択する場合、省略記号ボタンをクリックしてファイル ブラウザを起動し、LGH ファイルの場所を指定できます。

ランタイム時のマルチ ペントレンドの使用

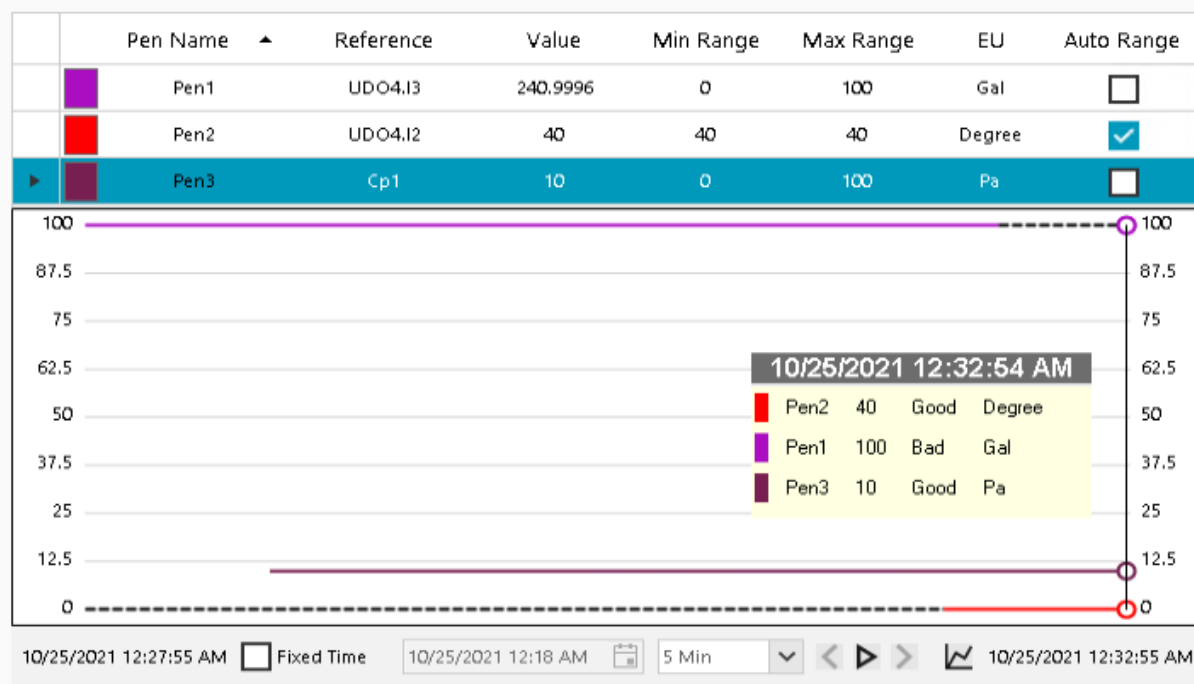
WindowViewer でのランタイム時のマルチ ペントレンドの動作を変更することができます。たとえば、ペンの追加、ペンの削除、または履歴トレンドデータの操作を行うことができます。マルチ ペントレンドは 3 つの領域で整理できます。

- 上部ヘッダー: テーブル形式でペンの詳細が一覧表示されます。
- トレンド領域: この領域ではペンが視覚的に表現されます。
- ツールバー: 再生およびスタック オプションが含まれます。



カーソル読み出しダイアログ

カーソルをペンの特定のポイント（指定した時間など）に置くと、カーソル読み出しダイアログに [ペン名]、[値]、[品質]、および [工学単位] フィールドが表示されます。カーソル読み出しダイアログを表示するには、トレンド領域の上にカーソルを置きます。



データ品質の表示

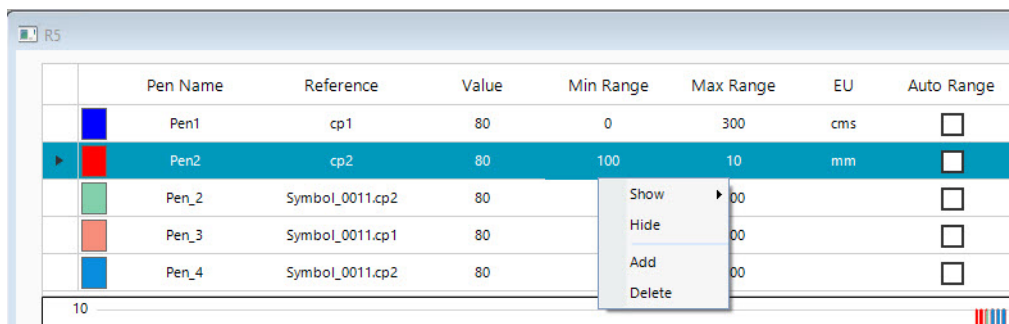
参照に高品質のデータがない場合、トレンドペンのスタイルは低品質スタイルになります。トレンドは最後の既知の有効な値を使用して引き続き表示されます。スタイルは、System Platform IDE の Galaxy スタイルを使用して変更できます。

注記: グラフィックの表示ウィンドウでは、ズームインまたはズームアウトするのはトレンド領域だけです。Ctrl キーとマウス ホイール オプションを使用してズームしたとき、ステータス コントロール バーは移動しません。

上部ヘッダー オプション

上部ヘッダー領域を使用してランタイム時にペンの詳細を編集できます。上部ヘッダーを使用して、以下の操作を行うこともできます。

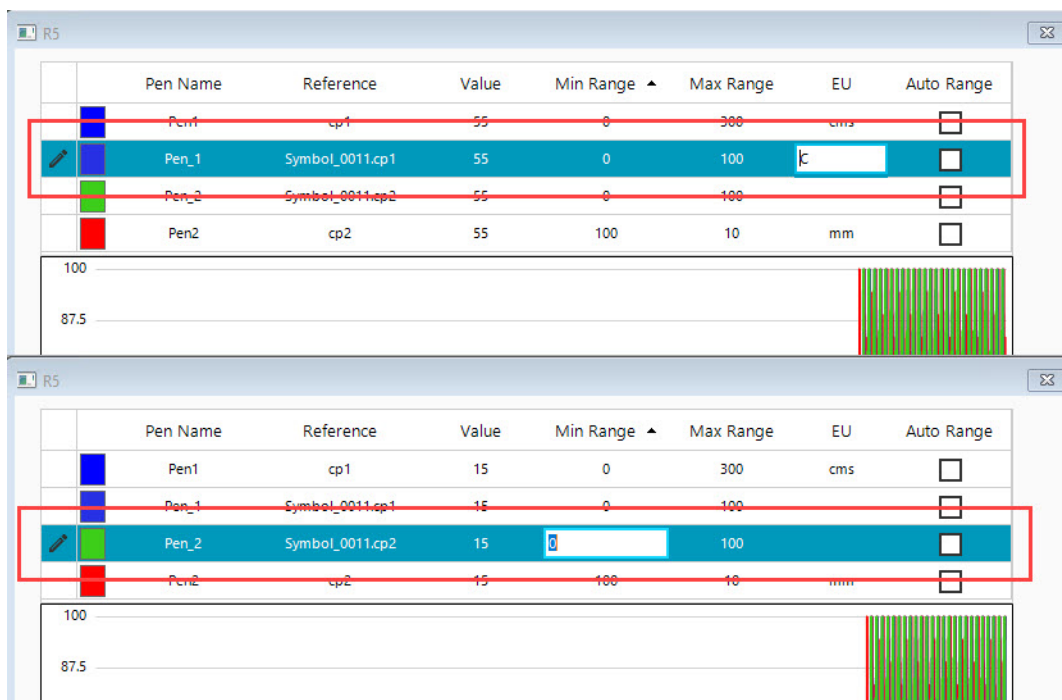
- 個々のペンの表示と非表示の切り替え
- ペンの追加
- ペンの削除



	Pen Name	Reference	Value	Min Range	Max Range	EU	Auto Range
	Pen1	cp1	80	0	300	cms	<input type="checkbox"/>
	Pen2	cp2	80	100	10	mm	<input type="checkbox"/>
	Pen_2	Symbol_0011.cp2	80		100		<input type="checkbox"/>
	Pen_3	Symbol_0011.cp1	80		100		<input type="checkbox"/>
	Pen_4	Symbol_0011.cp2	80		100		<input type="checkbox"/>

ランタイム時のペンの詳細の編集

1. フィールドをダブルクリックして編集します。
2. 値を更新します。
3. 行の外側をクリックすると新しい値が保存されます。



	Pen Name	Reference	Value	Min Range	Max Range	EU	Auto Range
	Pen1	cp1	55	0	300	cms	<input type="checkbox"/>
	Pen_1	Symbol_0011.cp1	55	0	100		<input type="checkbox"/>
	Pen_2	Symbol_0011.cp2	55	0	100		<input type="checkbox"/>
	Pen2	cp2	55	100	10	mm	<input type="checkbox"/>

	Pen Name	Reference	Value	Min Range	Max Range	EU	Auto Range
	Pen1	cp1	15	0	300	cms	<input type="checkbox"/>
	Pen_1	Symbol_0011.cp1	15	0	100		<input type="checkbox"/>
	Pen_2	Symbol_0011.cp2	15	0	100		<input type="checkbox"/>
	Pen2	cp2	15	100	10	mm	<input type="checkbox"/>

個々のペンの表示と非表示の切り替え

1. 行を右クリックします。
2. コンテキストメニューから **〔非表示〕** を選択します。
上部ヘッダー領域でペンが非表示になり、トレンド領域に描画されなくなります。
3. ペンを表示するには、上部ヘッダー領域を右クリックします。
4. コンテキストメニューから **〔表示〕** を選択して、ペンの名前を選択します。
上部ヘッダーとトレンド領域にペンが表示されます。

ペンの追加

1. 上部ヘッダー領域を右クリックし、**〔追加〕** を選択します。
新しい行が上部ヘッダー領域に追加されます。
2. 名前や参照などのペンの詳細を入力します。
3. Enter キーを押します。
データが使用可能な場合、トレンド領域でペンの描画が開始されます。

ペンの削除

- 上部ヘッダー領域を右クリックし、**〔削除〕** を選択します。Delete キーを押すこともできます。
マルチ ペントレンドコントロールからペンを削除されます。

上部ヘッダー領域のソート

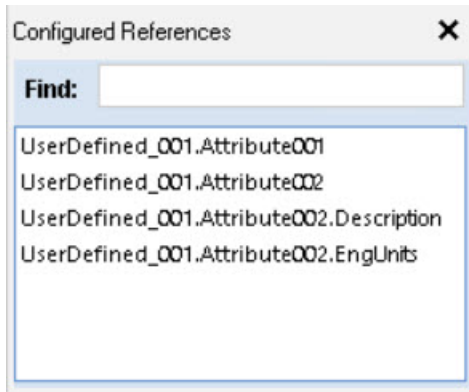
1. ペンの詳細をソートするヘッダー行の見出し名をクリックします。
ペン名、参照、EU の見出し名の場合、上部ヘッダーが昇順または降順にソートされます。
その他のすべての見出し名は、小さいものから大きいものの順にソートされます。
色以外のすべての見出しを上部ヘッダー領域のソートに使用できます。
2. 見出し名を再度クリックすると、ソートの方向が変更されます。
3. 見出し名をもう一度クリックすると、ソート前の状態に戻ります。

〔設定済み参照〕 ダイアログ ボックス

〔設定済み参照〕 ダイアログ ボックスでは、グラフィックまたはウィンドウで設定されている属性を表示できます。このダイアログ ボックスは、サイズを変更することや、ウィンドウ上の位置を変更することができます。ダイアログ ボックスの位置と寸法は **WindowViewer** を再起動しても維持されます。

〔設定済み参照〕 ダイアログ ボックスを表示するには

- ウィンドウを右クリックします。〔設定済み参照〕 ダイアログ ボックスが表示されます。



属性を見つけるには

- [検索] フィールドに属性名の最初の数文字を入力して **Enter** キーを押します。入力したテキストに一致する属性のリストが表示されます。

属性名をコピーするには

- [設定済み参照] ダイアログ ボックスで属性を右クリックして **[コピー]** をクリックします。
属性の名前がオペレーティング システムのクリップボードにコピーされます。

ドラッグ アンド ドロップによるペンの追加

参照を選択してドラッグ アンド ドロップすることでランタイム時にトレンドペンを追加できます。

1. グラフィックまたは要素を右クリックします。[設定済み参照] ダイアログ ボックスが表示され、そのグラフィックで使用できるすべての属性が一覧表示されます。
2. 参照を選択して、マルチ ペントレンドの上部ヘッダーにドラッグ アンド ドロップします。新しいペンが作成され、選択したペンのデータがトレンド領域に描画されます。

トレンド領域のスケール

トレンド領域のスケールは、選択したトレンドペンの値に応じてランタイム時に計算および表示されます。表示されるグリッドラインの数は、トレンド領域の高さに応じて異なります。マルチ ペントレンドの高さを変更すると、その変更にしたがってスケールも変更されます。高さに応じて、2、4、または8の間隔を使用できます。

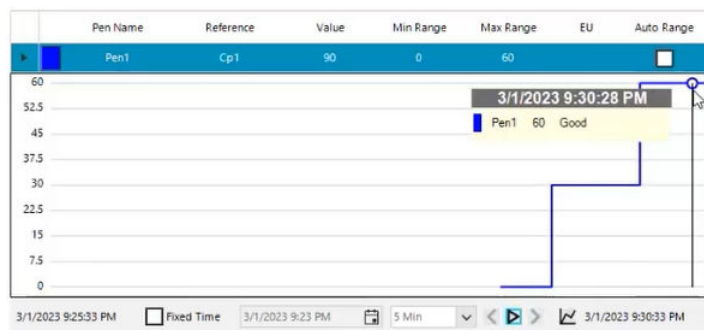
最小および最大範囲の変更

[自動範囲] チェック ボックスがオンにされていない場合、最小と最大の範囲を変更できます。トレンド領域でスケールが更新され、新しいスケールでトレンドがプロットされます。

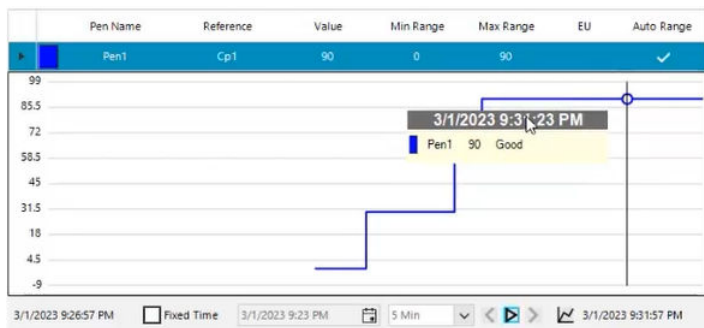
- 選択した行に対して、上部ヘッダーの最小範囲および最大範囲フィールドをクリックして新しい値を入力します。

トレンド領域のペンのスケールが更新されます。

タグ値が最小範囲と最大範囲を超えている場合、トレンド線は最小範囲と最大範囲の値に切り詰められます。カーソル読み出しには、トレンド線に基づく値が常に表示されます。下の画像では、最大範囲が60なので、タグ値が90であってもトレンドは60に切り詰められ、カーソル読み出しも60を示します。



切り詰めなしでトレンド線を表示し、カーソル読み出しで実際の値を表示するには、[自動範囲] チェック ボックスをオンにします。ペンに対して指定されている最大範囲と最小範囲の値が等しい場合、自動範囲の条件と見なされ、ペンに対して指定されている値は無視されます。下の画像では、[自動範囲] チェック ボックスがオンになっているので、トレンド線は切り詰められず、カーソル読み出しでは実際の値が表示されています。



自動範囲の詳細については、「[自動範囲の有効化](#)」トピックを参照してください。

自動範囲の有効化

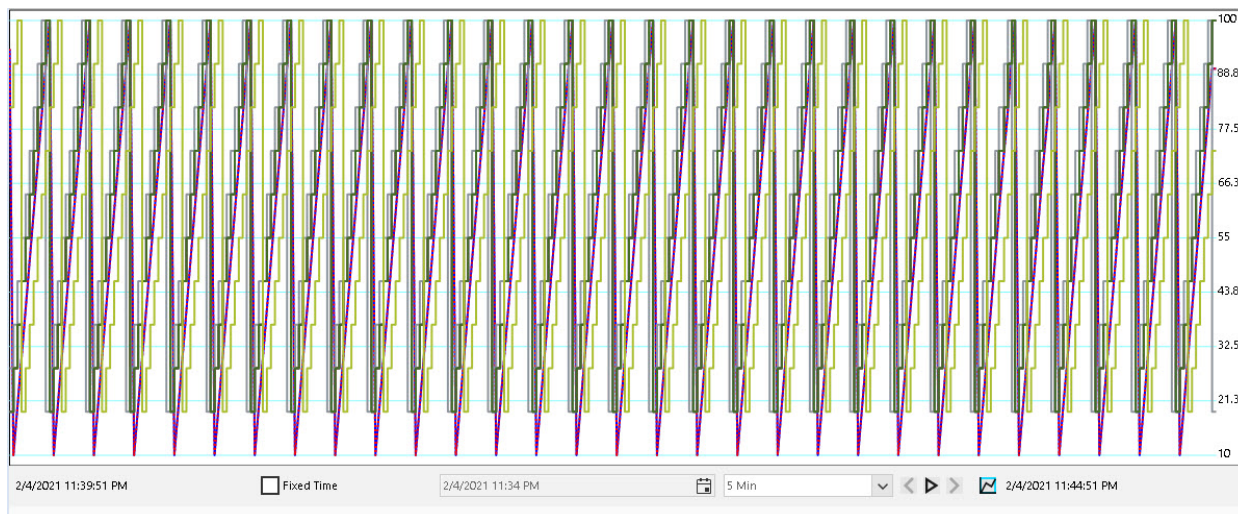
[自動範囲] を選択すると、トレンド ペン値に基づいてトレンド領域のスケールが動的に更新されます。ペンに対して指定された最小および最大の範囲は無視されます。



- 目的のペンの [自動範囲] チェック ボックスをオンにします。

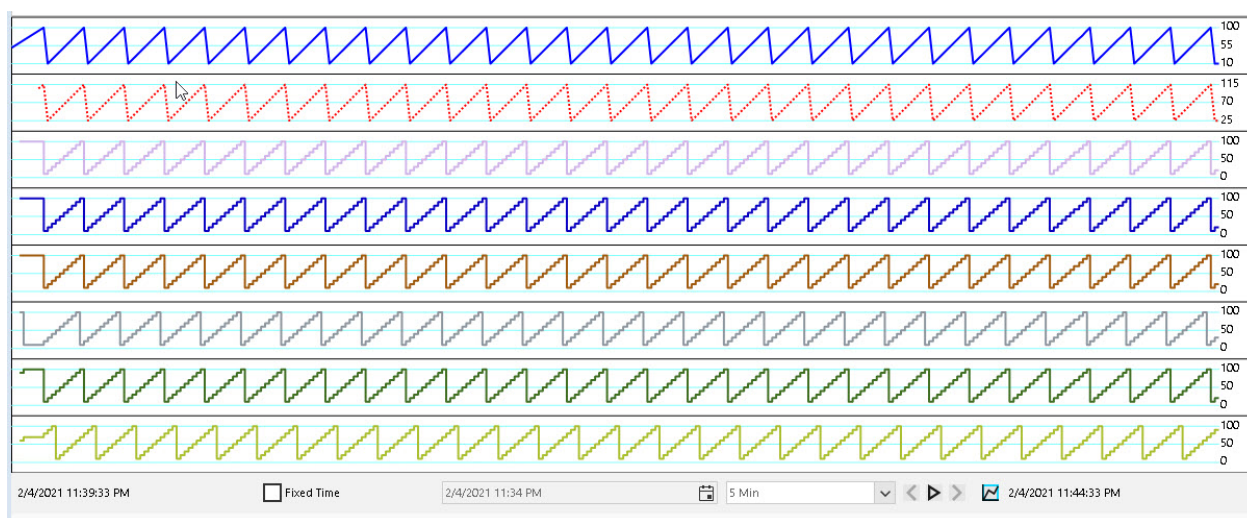
ペンの行を選択すると、トレンド領域のペンのスケールが更新されます。

トレンドのスタックおよびスタック解除

ランタイム時にスタック/スタック解除オプションを使用して個々のペンを表示できます。デフォルトでは、複数のペンが一緒に表示されます。



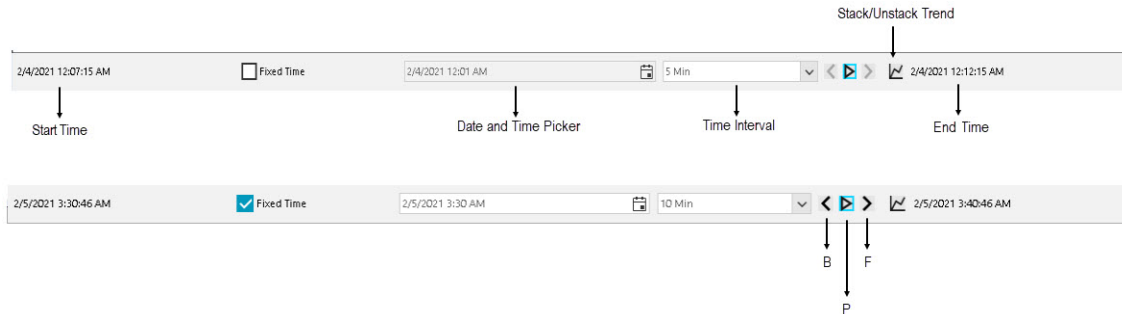
個々のペンを表示するには、ツールバーの  [トレンドのスタック] ボタンをクリックします。ペンが個々に重なった状態で表示されます。  を再度クリックすると、スタックが解除されたビューに戻ります。



注: トренд領域が小さい場合、ランタイム時にペンを追加すると個々のスケールが重なって表示されます。いくつかのペンを非表示にして結果を明瞭に表示することができます。

トレンドデータの再生

ツールバーの再生オプションを使用すると、指定した開始時間と終了時間の間のトレンドデータを再生できます。



移動モード

デフォルトでは、トレンド領域には、[時間間隔] フィールドで指定した値の現在のトレンドデータが表示されます。開始時間と終了時間の関係は、「終了時間 = 開始時間 + 時間間隔」で表されます。移動モードでは、終了時間は常に現在の時間で、入力に応じて変更されます。固定モードでは、開始時間と終了時間は、再生コントロールを使用するまで固定されます。

時間間隔は事前定義済みの値のリストから選択できます。トレンド領域のトレンドペンデータが更新され、その期間のデータだけが表示されます。

固定モード

- 過去または将来の特定の期間のトレンドペンを表示するには、[固定時間] チェックボックスをオンにします。
- [時間間隔] を選択します。

トレンド領域で、開始時間と終了時間の間のペンデータに選択した時間間隔が反映されます。

- [トレンドを後方向に移動] (B)、[トレンドの一時停止/トレンドの再生] (P)、および [トレンドを前方向に移動] (F) を使用します。

一時停止ボタンをクリックすると、選択した時間間隔のトレンド領域が固定されます。

◀ および ▶ をクリックすると、トレンド領域が開始時間と終了時間の間の時間ブロック内で時間間隔の半分だけ移動します。

次に例を示します。時間間隔が 10 分の場合の例を以下に示します。

Mode	開始時刻	終了時刻
ライブ	2/8/2021 3:15:16 AM	2/8/2021 3:25:16 AM
固定: 後方向に移動	2/8/2021 3:10:16 AM	2/8/2021 3:20:16 AM
固定: 前方向に移動	2/8/2021 3:20:16 AM	2/8/2021 3:30:16 AM

日付時刻ピッカーの使用

日付時刻ピッカーを使用して特定の開始日時を選択できます。この機能は、[固定時間] チェックボックスがオンにされている場合にのみ使用できます。

選択した日時が開始時間に表示されます。終了時間には、選択した日時に時間間隔を加えた値が表示されます。

選択した日時に時間間隔を加えた期間のデータがトレンド領域に表示されます。

SmartSymbol について

SmartSymbol は、再使用できるテンプレートに変換された InTouch グラフィックです。SmartSymbol テンプレートに対して行った変更内容は、アプリケーション全体の SmartSymbol のすべてのインスタンスに反映されます。このため、アプリケーションで繰り返し使用されるグラフィックの作成、変更、検証、および再検証を何度も行う手間が省けます。

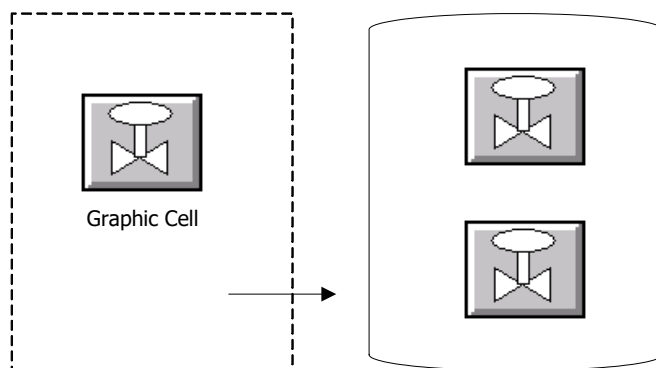
SmartSymbol は、InTouch タグおよび Application Server 属性を参照します。タグを参照しない SmartSymbol、または InTouch タグのみを参照する SmartSymbol は InTouch SmartSymbol と呼ばれます。

1 つまたは複数のオートメーション オブジェクトまたはオートメーション オブジェクト インスタンスの属性を参照する SmartSymbol は産業用グラフィック SmartSymbol と呼ばれます。産業用グラフィック SmartSymbol は InTouch タグを参照することもできます。

InTouch SmartSymbol および産業用グラフィック SmartSymbol は、産業用グラフィックではありません。産業用グラフィックの設計および管理には System Platform 統合開発環境 (IDE) を使用します。

InTouch バージョン 10 以降を使用して新しいグラフィックを作成している場合、アプリケーションには SmartSymbol より産業用グラフィックの方が適している場合があります。

SmartSymbol Template SmartSymbol Instances



SmartSymbol マネージャおよびライブラリ

SmartSymbol ライブラリには、InTouch アプリケーション用の SmartSymbol が含まれています。SmartSymbol マネージャは、SmartSymbol ライブラリの内容をインポート、エクスポート、および整理するために使用します。

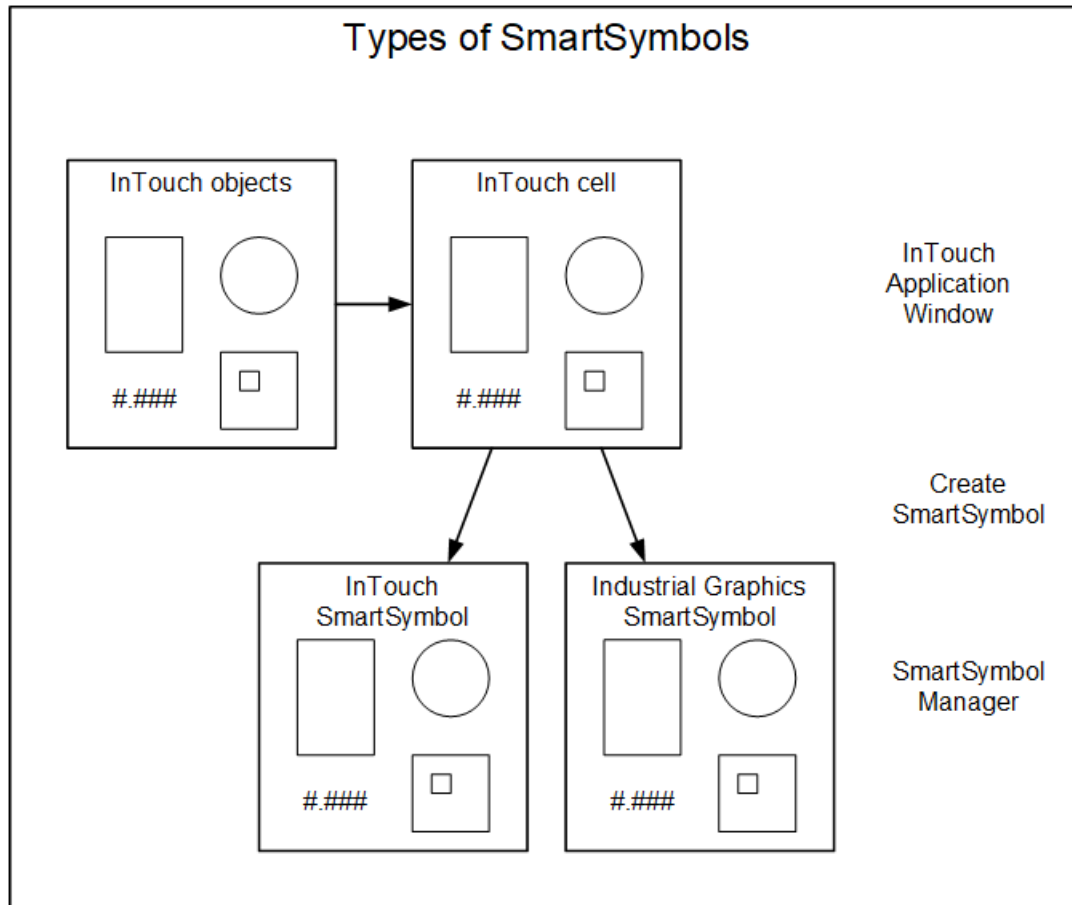
SmartSymbol は、ユーザーの InTouch アプリケーションのアプリケーション フォルダにある \Symbols フォルダに保存されています。

ライブラリの SmartSymbol についての参照情報は、関連する XML ファイルに保存されます。XML ファイルは編集しないでください。

InTouch SmartSymbol および産業用グラフィック SmartSymbol

InTouch SmartSymbol は、産業用グラフィック SmartSymbol と同じものではありません。産業用グラフィックは産業用グラフィック SmartSymbol に取って代わるものであり、System Platform IDE を使用して開発されています。InTouch SmartSymbol は、タグ変数のデータを参照します。産業用グラフィック SmartSymbol は、Galaxy オブジェクトまたはテンプレートの属性を参照します。

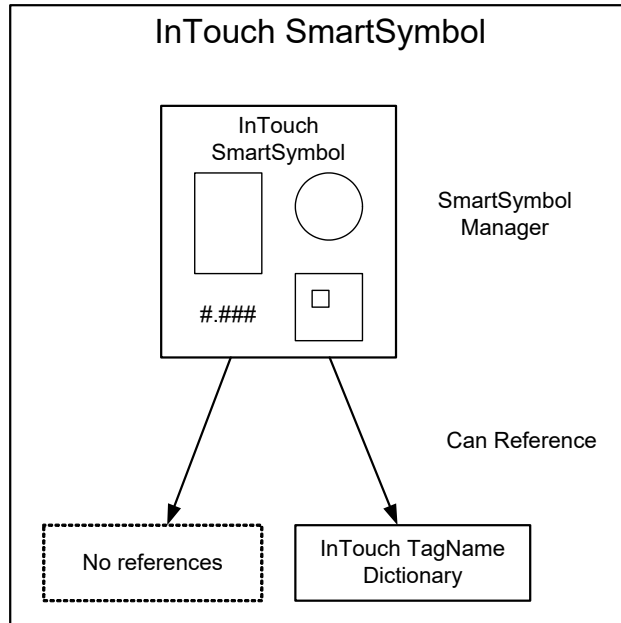
次の図は、SmartSymbol の種類を示しています。



InTouch SmartSymbol

InTouch SmartSymbol は、SmartSymbol マネージャの InTouch シンボル フォルダに保存されています。

ローカルおよびリモートの InTouch タグ変数への参照を使用して、InTouch SmartSymbol でグラフィカルな要素に対するアニメーションを設定できます。詳細については、「[SmartSymbol テンプレートとインスタンスの作成](#)」を参照してください。

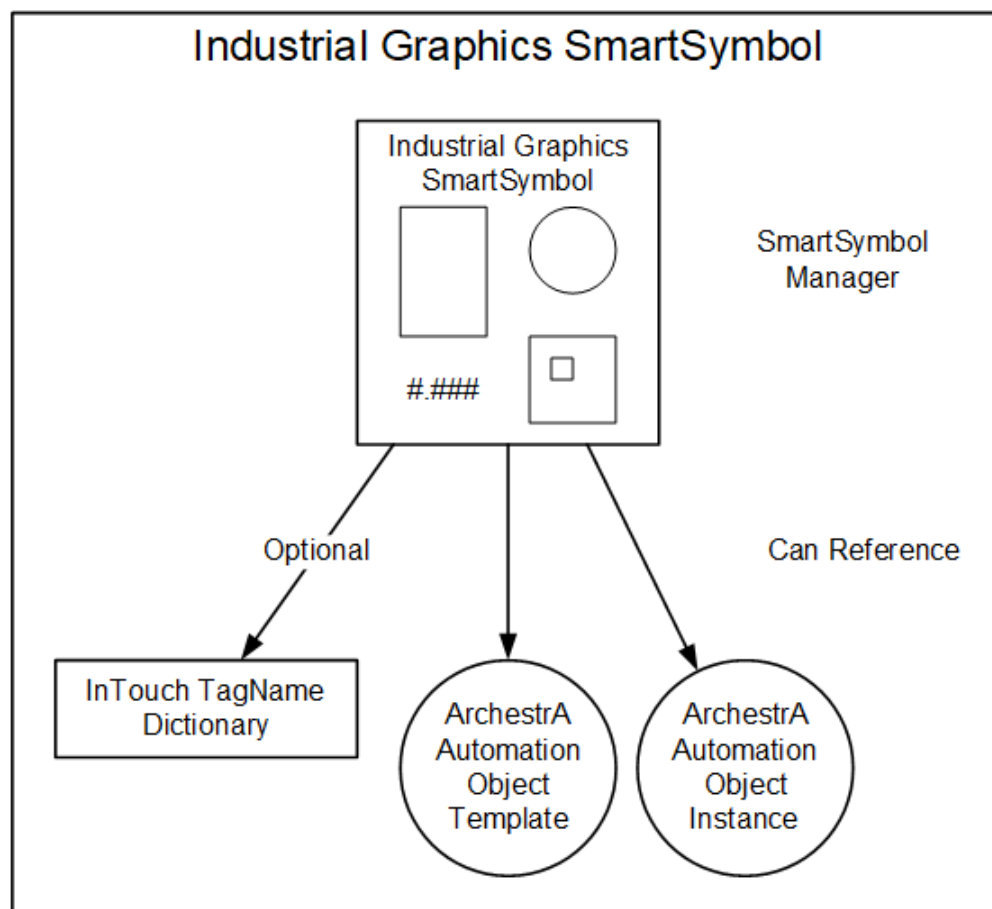


産業用グラフィック SmartSymbol

産業用グラフィック SmartSymbol は、SmartSymbol マネージャの産業用グラフィック フォルダに保存されています。

注: フォルダの名前は産業用グラフィック (Industrial Graphics) ですが、フォルダには産業用グラフィックではなく、産業用グラフィック SmartSymbol が含まれています。

産業用グラフィック SmartSymbol を作成するには、1 つまたは複数の Galaxy オブジェクト テンプレートを選択して、さまざまなグラフィカル要素に対するアニメーション参照を定義します。詳細については、「[SmartSymbol テンプレートとインスタンスの作成](#)」を参照してください。



SmartSymbol の制限

次に、SmartSymbol の既知の制限について説明します。

- SmartSymbol にはトレンドオブジェクトを含めることはできません。トレンドオブジェクト（履歴またはリアルタイム）が含まれている SmartSymbol を作成しようとすると、エラーメッセージが表示されます。
- SmartSymbol には、分散アラーム表示コントロール、Windows コントロール、AlarmViewer などの InTouch ActiveX コントロール、または InTouch アプリケーションで設定されるサードパーティの ActiveX コントロールを含めることはできません。
- Application Server バージョン 1.5 で作成された Galaxy でインスタンスを参照することはできません。インスタンスを参照するには、Application Server バージョン 2.0 以降をインストールしてください。
- SPC チャート ウィザードを使用した SmartSymbol の生成はサポートされていません。
- SmartSymbol は、ローカル スクリプト変数を参照できません。
- 属性ブラウザには、生成されたオブジェクト インスタンスは表示されません。この問題に対処するには、生成されたテンプレートを SmartSymbol マネージャで作成するか、またはブラウザでカスタム フィルタを作成します。

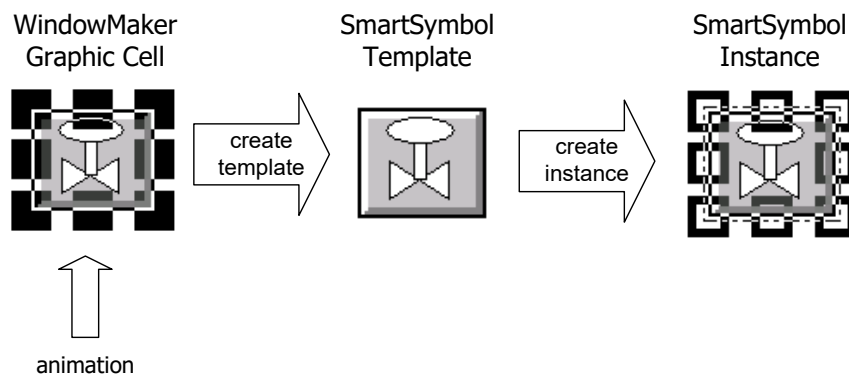
SmartSymbol テンプレートとインスタンスの作成

WindowMaker を使用すると、SmartSymbol テンプレートおよびインスタンスを作成できます。

SmartSymbol テンプレートは、WindowMaker で 1 つまたは複数のグラフィックを描画し、それらを 1 つのセルに結合してから、そのセルを SmartSymbol に変換することによって作成します。テンプレートを InTouch タグまたは Application Server オブジェクトに接続またはリンクする必要はありません。

SmartSymbol テンプレートを作成した後で、SmartSymbol のインスタンスをアプリケーション ウィンドウで作成できます。

また、既存の産業用グラフィック SmartSymbol インスタンスから Application Server オブジェクト インスタンスを作成して、InTouch WindowMaker と System Platform IDE の切り替えを回避することもできます。



InTouch データで使用する SmartSymbol テンプレートの作成

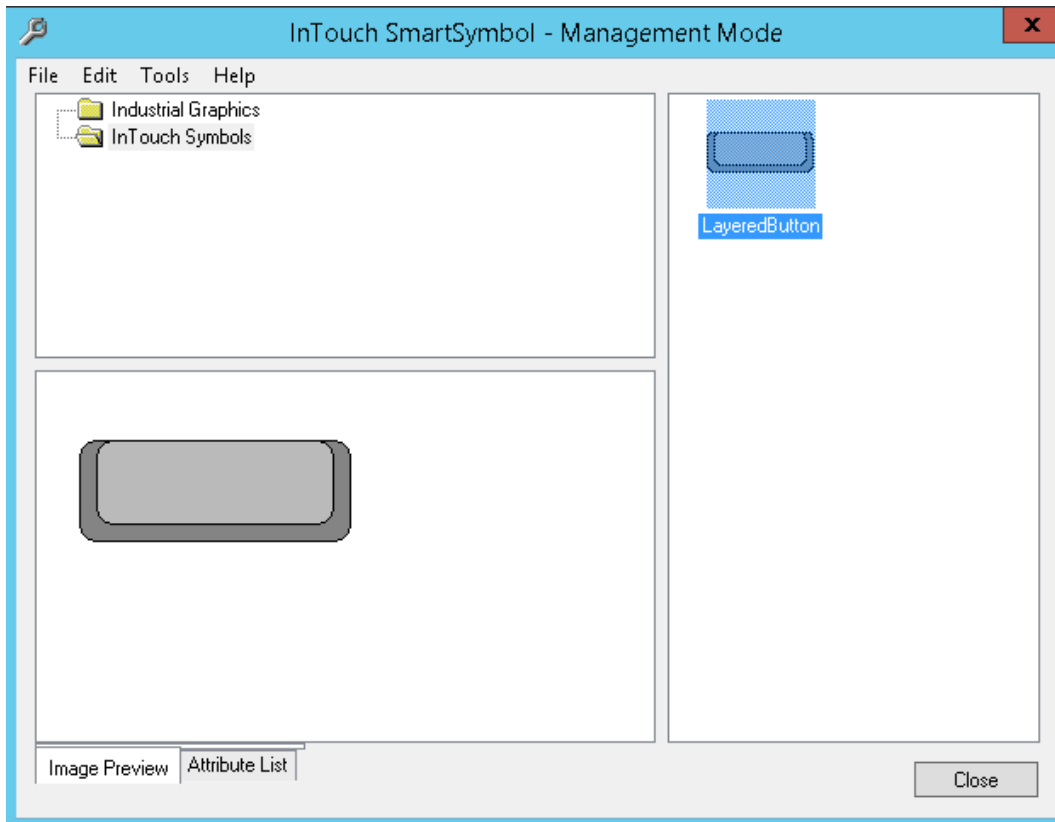
InTouch SmartSymbol は、グラフィカル要素、アニメーション、および InTouch タグへの参照を含めることのできるセルから作成されます。

以下の手順では、新しいウィンドウ、グラフィック、およびセルの作成が指定されていますが、既存のウィンドウ、グラフィック、またはセルを使用して SmartSymbol を作成することもできます。

新しい InTouch SmartSymbol テンプレートを作成するには

1. WindowMaker で新しいウィンドウを作成します。
2. グラフィック描画ツールまたはウィザードを使用して、SmartSymbol に含める 1 つまたは一組のグラフィックを作成します。
3. グラフィックのアニメーション リンクを設定します。
詳細については、「[オブジェクトのアニメーション化](#)」を参照してください。
4. SmartSymbol テンプレートに含めるオブジェクトをすべて選択します。
5. [アニメーション] メニューの [セル] グループで [セルの作成] をクリックします。
6. 先ほど作成したセルを選択します。
7. [描画] メニューの [SmartSymbol] グループで [生成] をクリックします。

[InTouch SmartSymbol - 管理モード] ダイアログ ボックスが表示され、新しい SmartSymbol が強調表示されます。



デフォルトでは、新しい SmartSymbol は InTouch シンボルの最上位レベルのフォルダに配置されます。デフォルトの名前が、シンボルに自動的に割り当てられます（例: 新規シンボル 1）。

8. 新しい名前を入力するか、デフォルトの名前を受け入れます。SmartSymbol の名前はいつでも変更できます。SmartSymbol の名前の変更の詳細については、「[SmartSymbol テンプレートの名前の変更](#)」を参照してください。
9. [閉じる] をクリックします。グラフィックセルを新しい SmartSymbol で置換するよう求めるメッセージが表示されます。[はい] または [いいえ] をクリックします。[はい] をクリックすると、グラフィックセルが SmartSymbol によって置換されます。[いいえ] をクリックすると、グラフィックセルは変更されません。いずれの場合でも、新しい SmartSymbol は SmartSymbol ライブラリに保存され、今後使用も使用できます。

産業用グラフィック SmartSymbol テンプレートの作成

オートメーションオブジェクトテンプレートまたはインスタンスへの参照を少なくとも 1 つ含む InTouch グラフィックセルから SmartSymbol を作成すると、その SmartSymbol は産業用グラフィック SmartSymbol になります。

オートメーションテンプレートへの参照には、"\$" 記号が含まれます。

オブジェクトテンプレートまたはインスタンスに関連付けられている SmartSymbol テンプレートを生成できます。

InTouch ウィンドウで SmartSymbol インスタンスを作成すると、それが参照するオブジェクトテンプレートをインスタンス化できます。

新しい産業用グラフィック **SmartSymbol** テンプレートを生成するには:

1. WindowMaker で新しいウィンドウを作成します。
2. 描画ツールまたはウィザードを使用して、**SmartSymbol** に作成するグラフィックを作成します。
3. グラフィックのアニメーション リンクを設定します。

Galaxy ソース名の設定の手順については、「[InTouch から Application Server データへのアクセス](#)」を参照してください。Application Server 属性へのリンクの作成および設定の手順については、「[オブジェクトのアニメーション化](#)」を参照してください。

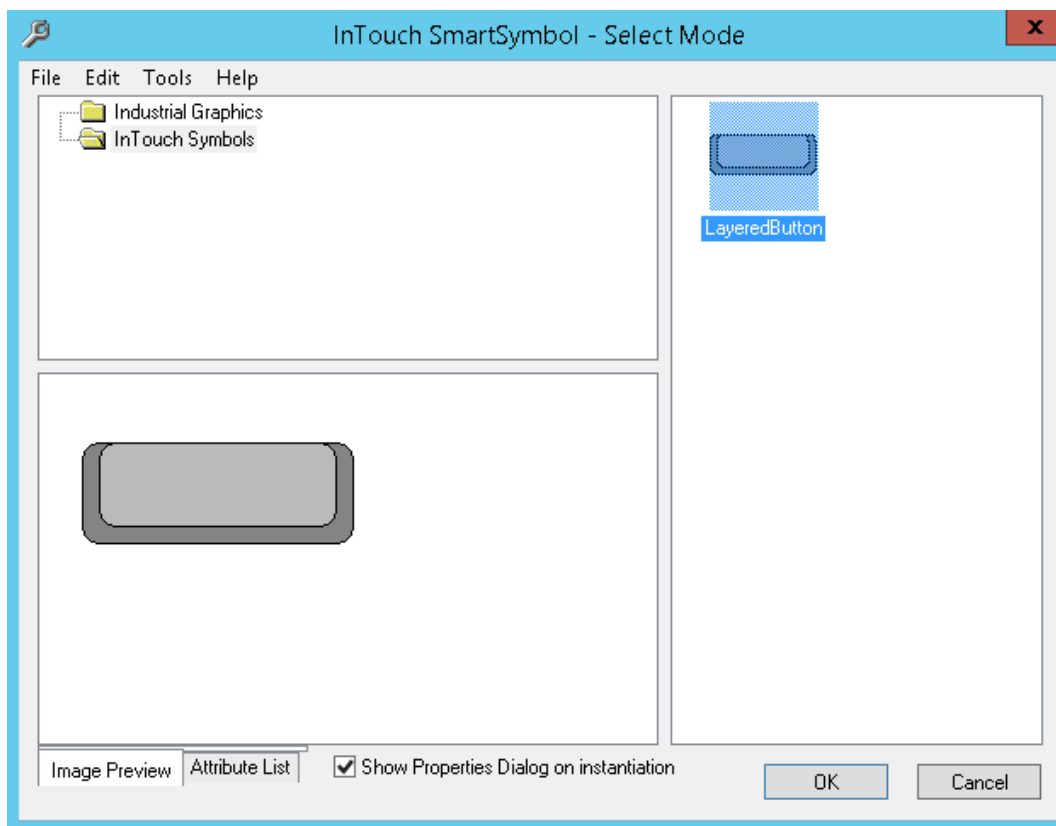
4. セルに含めるグラフィックを選択します。
5. [アニメーション] メニューの [セル] グループで [セルの作成] をクリックします。
6. [描画] メニューの [SmartSymbol] グループで [生成] をクリックします。
SmartSymbol マネージャによって Galaxy データの新しい SmartSymbol が作成されます。
7. 新しい名前を入力するか、デフォルトの名前を受け入れて後で変更します。
8. [閉じる] をクリックします。グラフィック セルを新しい SmartSymbol で置換するよう求めるメッセージが表示されます。[はい] または [いいえ] をクリックします。[はい] をクリックすると、グラフィック セルが SmartSymbol によって置換されます。[いいえ] をクリックすると、グラフィック セルは変更されません。いずれの場合でも、新しい SmartSymbol は SmartSymbol ライブラリに保存され、今後使用も使用できます。

InTouch SmartSymbol テンプレートからの SmartSymbol インスタンスの作成

1 つの SmartSymbol テンプレートから複数の SmartSymbol インスタンスを作成できます。各インスタンスには、すべての参照とテキスト ラベルが継承されます。InTouch ウィンドウにインスタンスを配置する前に、参照とテキスト ラベルを変更できます。

InTouch SmartSymbol テンプレートから SmartSymbol を作成するには

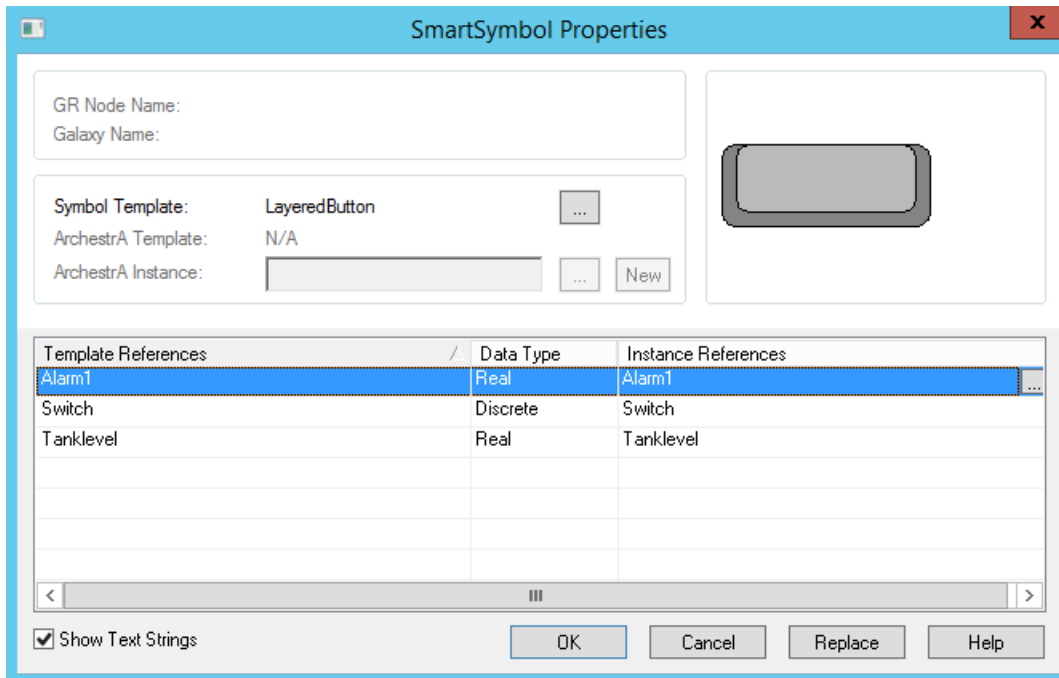
1. WindowMaker を開き、SmartSymbol を使用するウィンドウを開きます。
2. [ホーム] メニューの [挿入] グループで [SmartSymbol] アイコンをクリックします。
3. WindowMaker ウィンドウで、シンボルを配置する場所をクリックします。[InTouch SmartSymbol - 選択モード] ダイアログ ボックスが表示されます。



注記: デフォルトでは、[インスタンス化の際に [プロパティ] ダイアログを表示する] チェック ボックスはオンになっています。新しい SmartSymbol インスタンスの参照またはテキストラベルを変更しない場合は、チェック ボックスをオフにします。

4. [InTouch シンボル] フォルダで、SmartSymbol をダブルクリックします。アプリケーション ウィンドウに新しいシンボルが表示されます。

前の手順で [インスタンス化の際に [プロパティ] ダイアログを表示する] チェック ボックスをオンにした場合、[SmartSymbol プロパティ] ダイアログ ボックスが表示されます。



5. [インスタンス参照] カラムで、省略記号ボタンをクリックします。[タグの選択] または [タグ名ディクショナリ] ダイアログ ボックスが表示されます。
6. タグを選択して、SmartSymbol にリンクします。ウィンドウを閉じると、[SmartSymbol プロパティ] ダイアログ ボックスが表示されます。

注記: まだ定義されていない新しいタグ名を入力すると、[未登録のタグ名] ダイアログ ボックスが表示されます。[OK] をクリックして、タグ名ディクショナリから新しいタグを定義します。

7. [OK] をクリックします。アプリケーション ウィンドウに新しいシンボルが表示されます。

ArchestrA SmartSymbol テンプレートからの SmartSymbol インスタンスの作成

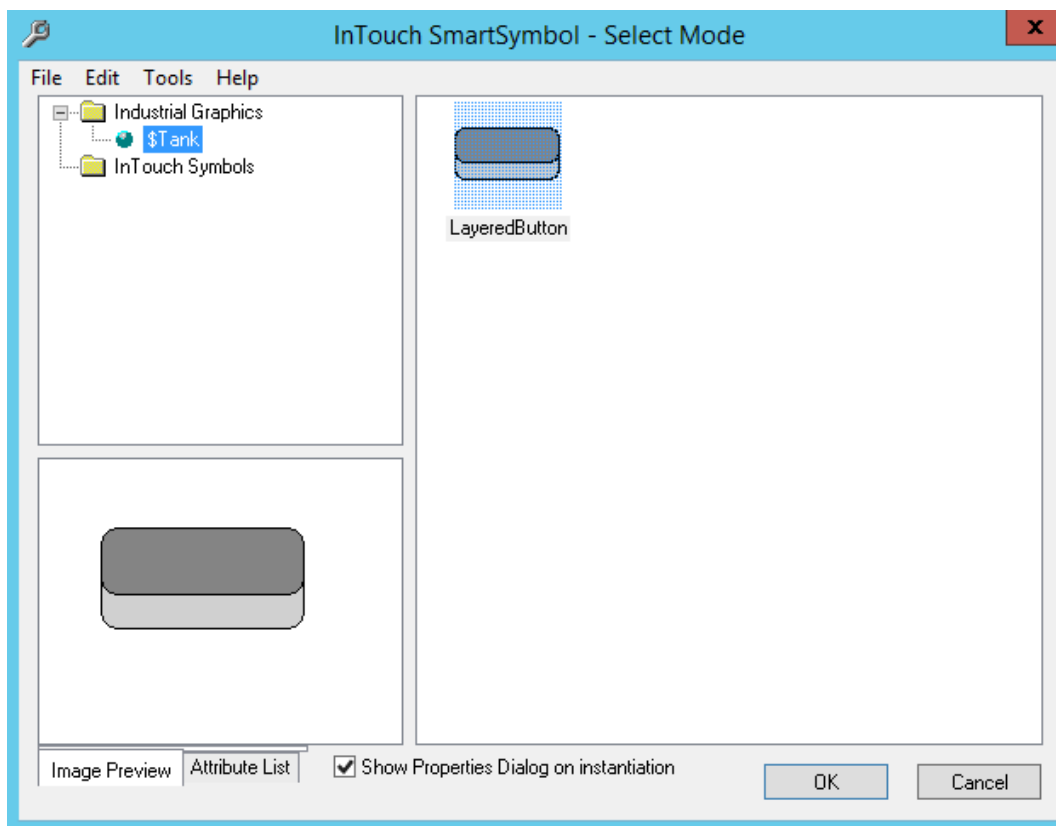
複数の ArchestrA SmartSymbol インスタンスは、1 つの ArchestrA SmartSymbol テンプレートから作成できます。

ArchestrA SmartSymbol テンプレートから SmartSymbol を作成するには

1. [SmartSymbol] アイコンをクリックし、WindowMaker ウィンドウでシンボルを配置する場所をクリックします。[InTouch SmartSymbol - 選択モード] ダイアログ ボックスが表示されます。

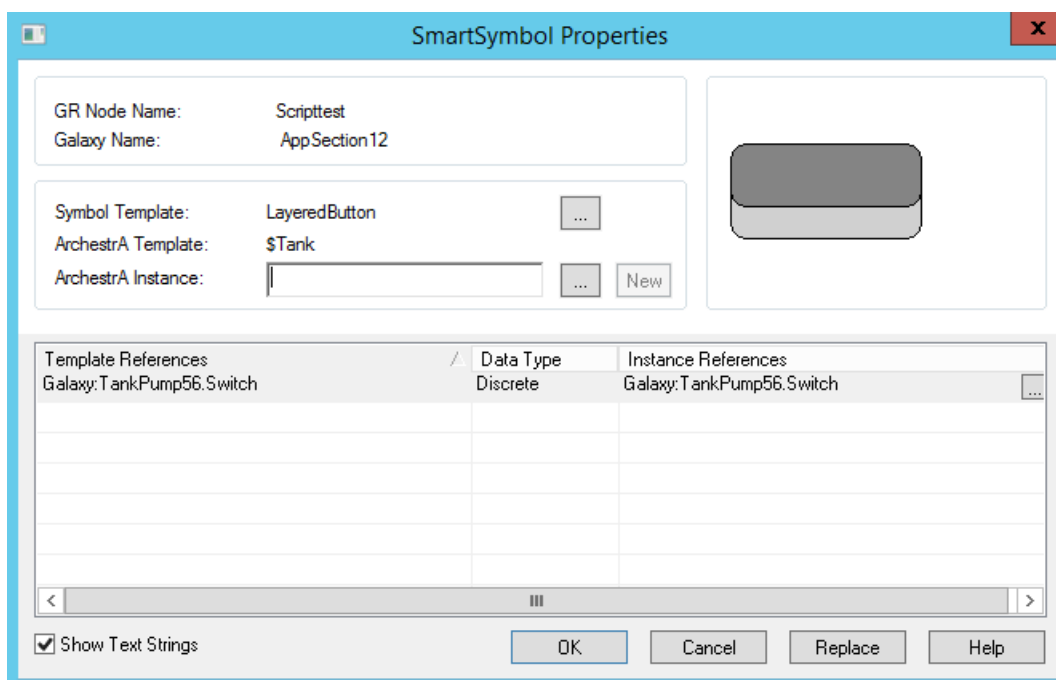
注: デフォルトでは、[インスタンス化の際に [プロパティ] ダイアログを表示する] チェック ボックスはオンになっています。

1. 「Industrial Graphics」フォルダをクリックします。右ペインに産業用グラフィックが表示されます。



2. SmartSymbol を選択して、[OK] をクリックします。アプリケーション ウィンドウに新しいシンボルが表示されます。

前の手順で [インスタンス化の際に [プロパティ] ダイアログを表示する] チェック ボックスをオンにした場合、[SmartSymbol プロパティ] ダイアログ ボックスが表示されます。



3. **[ArchestrA インスタンス]** テキスト ボックスでは、以下のいずれかを実行できます。

- ArchestrA オブジェクトを参照し、選択します。
- 関連付けられたオブジェクトテンプレートから派生した新しい ArchestrA オブジェクトインスタンスを作成します。インスタンスの名前を入力して、**[新規]** をクリックします。

[インスタンス参照] カラムにインスタンス属性参照が表示されます。

注: Galaxy にまだ接続されていなかった場合、ノード名と Galaxy 名を入力するよう求めるダイアログ ボックスが表示されます。

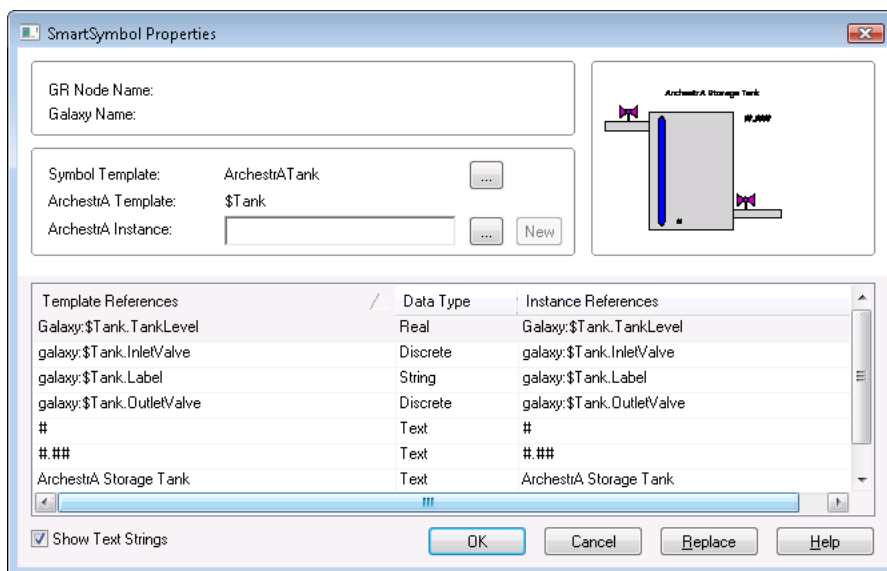
1. 必要に応じて、**[インスタンス参照]** カラムで参照を変更します。正しい構文で手動で参照を入力するか、**省略記号** ボタンをクリックして、**属性ブラウザ** を使用します。
2. **[OK]** をクリックします。ウィンドウに新しいシンボルが表示されます。

ArchestrA SmartSymbol インスタンスからの ArchestrA オブジェクトインスタンスの作成

既存の ArchestrA SmartSymbol インスタンスから新しい ArchestrA オブジェクトインスタンスを作成できます。この操作を行うことによって、WindowMaker と IDE を切り替える必要はありません。

新しい ArchestrA オブジェクトインスタンスを作成するには

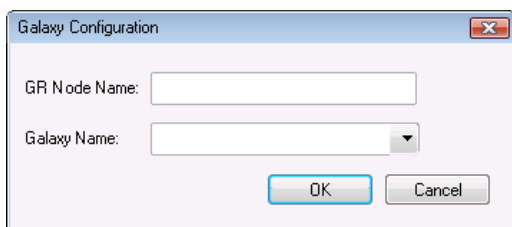
1. WindowMaker で、SmartSymbol インスタンスが配置されているウィンドウを開きます。
2. アプリケーション ウィンドウで、SmartSymbol インスタンスをダブルクリックします。**[SmartSymbol プロパティ]** ダイアログ ボックスが表示されます。



3. **[ArchestrA インスタンス]** ボックスで、新しいオートメーション オブジェクトの有効な名前を入力します。

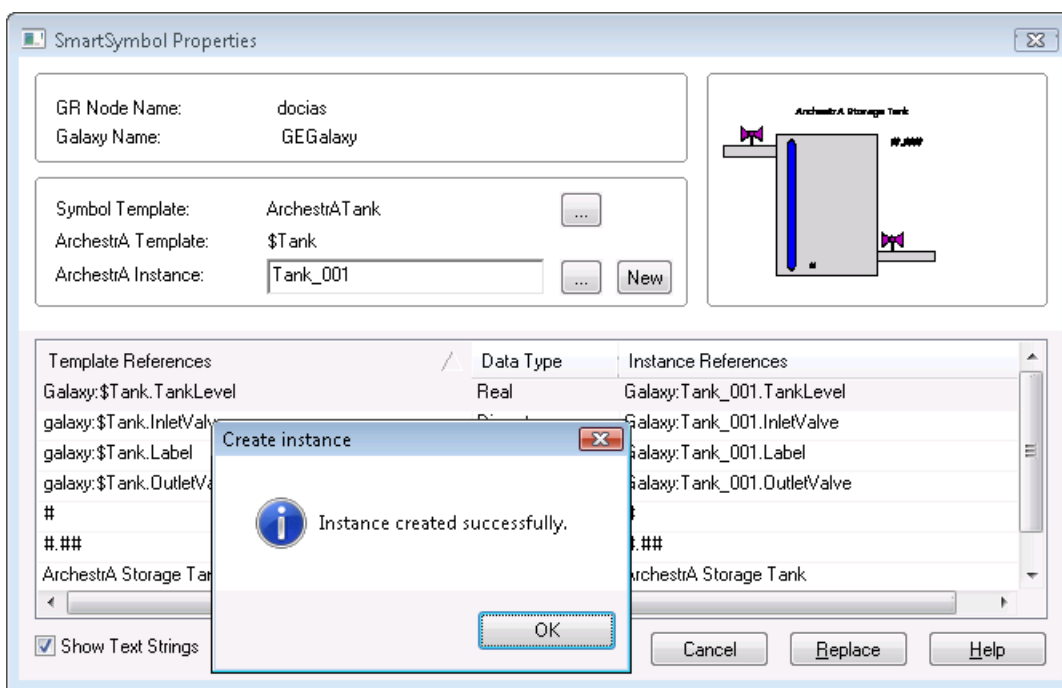
注: オブジェクトを指定するのが初めてである場合は、ログインするように求められます。有効なユーザー名、パスワード、およびドメイン名を入力します。Application Server のセキュリティが **[なし]** 以外のモードに設定されている場合、OS ユーザーまたは OS グループ ベースのセキュリティに対してのみドメイン名が必要です。

1. **[新規]** をクリックします。新しいオブジェクトを作成する有効な Galaxy を選択するようメッセージが表示されたら、**[OK]** をクリックします。**[Galaxy の設定]** ダイアログ ボックスが表示されます。



2. Galaxy を指定します。以下の手順を実行します。

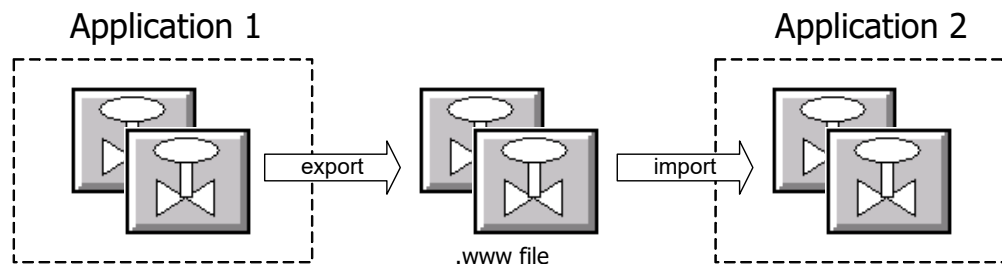
- a. [GR ノード名] ボックスで、Galaxy が稼動しているコンピュータの名前を入力します。
- b. [Galaxy 名] リストで、Galaxy をクリックします。
- c. [OK] をクリックします。ArchestrA オブジェクトインスタンスが作成され、インスタンス参照は新しいインスタンスをポイントします。



- a. 再び [OK] をクリックして、[インスタンスの作成] ダイアログ ボックスを閉じます。
3. [OK] をクリックし、[SmartSymbol プロパティ] ダイアログ ボックスを閉じます。アプリケーション ウィンドウに新しい SmartSymbol インスタンスが表示されます。

SmartSymbol の管理

SmartSymbol マネージャを使用すると、複数の InTouch アプリケーション間および異なる物理システム間で SmartSymbol をインポートおよびエクスポートできます。SmartSymbol のエクスポートおよびインポートは、InTouch アプリケーション間で SmartSymbol を移動する最適な手段です。



SmartSymbol と共にウィンドウもインポートすることができ、グラフィックもインポートされますが、テンプレート情報はインポートされません。このため、SmartSymbol のインスタンスは孤立します。詳細については、「[SmartSymbol の回復](#)」を参照してください。

SmartSymbol マネージャを使用すると、SmartSymbol テンプレートの名前の変更、複製、削除、および保存を行うことができます。

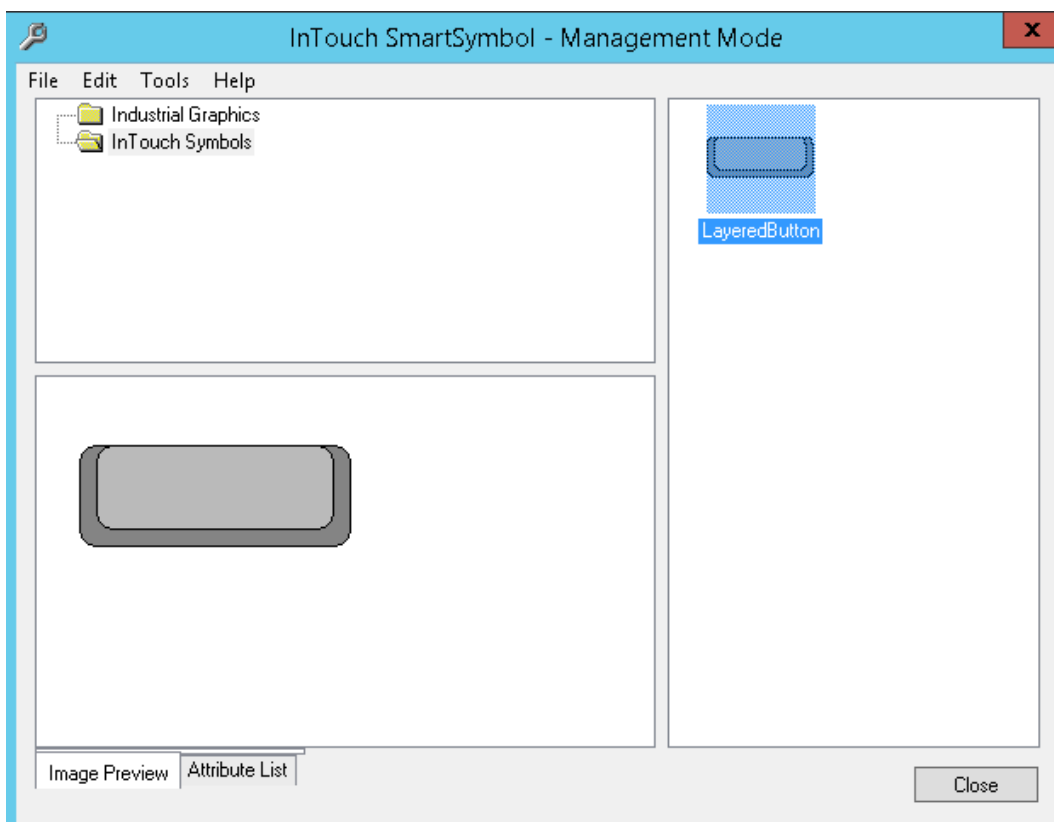
SmartSymbol のインポート

SmartSymbol は、他の InTouch アプリケーションからアプリケーションの SmartSymbol ライブラリにインポートできます。他のアプリケーションからシンボルをインポートすると、そのシンボルを再使用する代わりに、シンボルを再使用できます。

シンボル ライブラリに **SmartSymbol** をインポートするには:

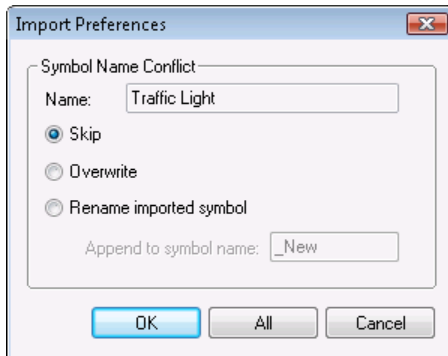
1. すべてのアプリケーション ウィンドウを閉じます。
2. [描画] メニューの [SmartSymbol] グループで [管理] をクリックします。

[InTouch SmartSymbol - 管理モード] ウィンドウが表示されます。



3. [ファイル] メニューで、[インポート] をクリックします。[シンボルのインポート] ダイアログ ボックスが表示されます。
4. インポートする SmartSymbol を含むファイルを参照します。シンボルのエクスポート ファイルはファイル拡張子 .www を持ちます。
5. ファイルを選択して、[OK] をクリックします。[SmartSymbol - 管理モード] ウィンドウに、そのファイルの SmartSymbol が表示されます。

名前が競合する場合、[インポート環境設定] ダイアログ ボックスが表示されます。



6. 以下のいずれかまたは複数の手順を実行します。
 - このシンボルのインポートを省略するには、[スキップ] をクリックします。複数のシンボルをインポートしている場合、残りのシンボルがインポートされます。
 - 既存のシンボルを新しいシンボルで上書きするには、[上書き] をクリックします。
 - 新しいシンボルを未使用の名前に変更するには、[インポートするシンボルの名前を変更] をクリックします。[シンボル名に追加] ボックスに名前を入力します。
7. 以下のいずれかを実行します。
 - [OK] をクリックして、選択したオプションを SmartSymbol に適用します。
 - [インポートするシンボルの名前を変更] をクリックした場合は、[すべて] をクリックして、[シンボル名に追加] ボックス内のテキストをパッケージファイル内で名前が競合するすべての SmartSymbol に適用します。

[InTouch SmartSymbol - 管理モード] ウィンドウに、インポートした SmartSymbol が表示されます。

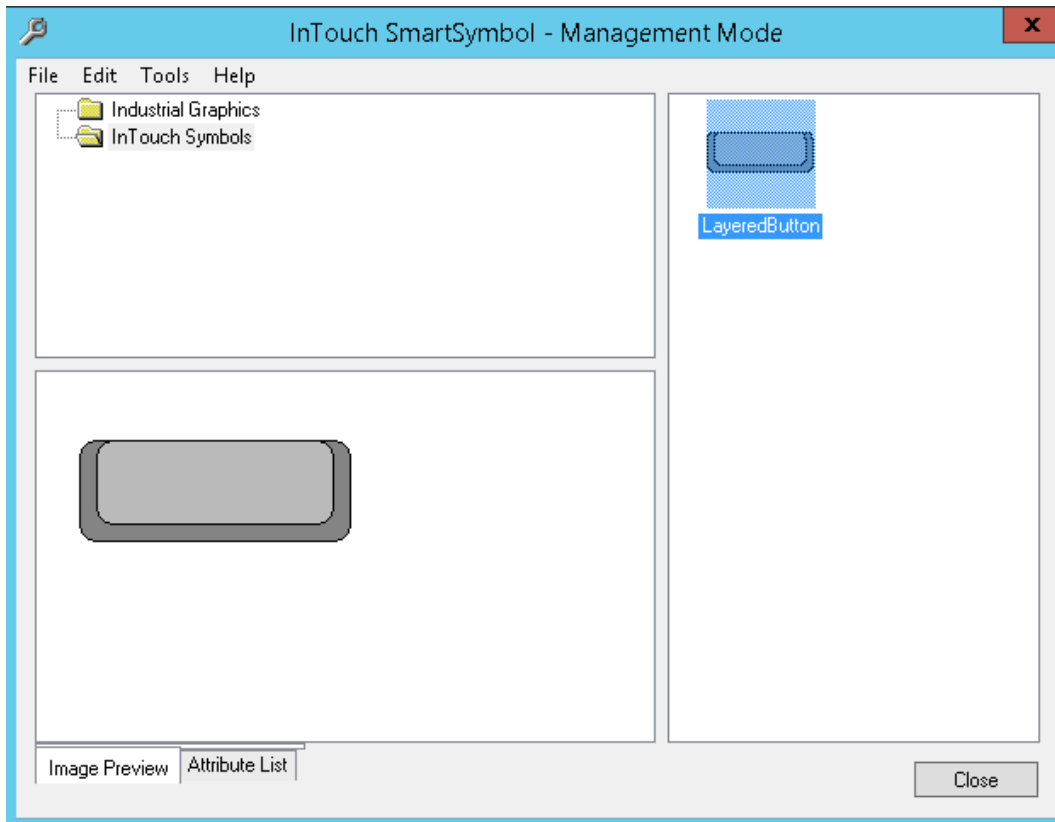
SmartSymbol のエクスポート

SmartSymbol をアプリケーションの SmartSymbol ライブラリで作成またはインポートすると、1 つまたは複数の SmartSymbol テンプレートを他の InTouch アプリケーションにエクスポートできます。SmartSymbol テンプレートのエクスポートは、SmartSymbol を InTouch アプリケーション間で移動する最適な手段です。

SmartSymbol をエクスポートするには

1. [描画] メニューの [SmartSymbol] グループで [管理] をクリックします。

[InTouch SmartSymbol - 管理モード] ウィンドウが表示されます。



2. SmartSymbol およびフォルダのリストから、エクスポートする SmartSymbol またはフォルダを選択します。
3. [ファイル] メニューで、[エクスポート] をクリックします。[シンボルのエクスポート] ダイアログ ボックスが表示されます。
4. シンボルのエクスポート先であるフォルダを参照します。
5. 拡張子 .www を付けてファイル名を入力し、[保存] をクリックします。SmartSymbol またはフォルダが、指定したフォルダにエクスポートされます。

SmartSymbol テンプレートの名前の変更

SmartSymbol マネージャを使用すると、SmartSymbol テンプレートの名前を変更できます。SmartSymbol テンプレートの名前の変更は、SmartSymbol インスタンスには影響を与えません。

SmartSymbol テンプレートの名前を変更するには：

1. SmartSymbol マネージャで、名前を変更する SmartSymbol テンプレートを選択します。
2. [編集] メニューで、[名前の変更] をクリックします。
3. シンボルの新しい名前を入力して、**Enter** キーを押します。新しい名前の SmartSymbol テンプレートが表示されます。

SmartSymbol テンプレートの複製

SmartSymbol テンプレートを作成すると、そのコピーを作成できます。たとえば、テンプレートを複製して、そのテンプレートを変更したり編集したりして、それに似た機能を持つ新しいテンプレートを作成できます。

SmartSymbol テンプレートの編集の詳細については、「[SmartSymbol の編集](#)」を参照してください。

SmartSymbol テンプレートを複製するには：

1. SmartSymbol マネージャで、複製する SmartSymbol をクリックします。
2. [編集] メニューで、[コピー] をクリックします。
3. 新しい SmartSymbol 用のフォルダをクリックします。
4. [編集] メニューで、[貼り付け] をクリックします。新しい SmartSymbol が表示されます。元の SmartSymbol と同じフォルダに配置する場合、新しい SmartSymbol の名前は「コピー～<元の名前>」となります。

SmartSymbol テンプレートの削除

SmartSymbol テンプレートを削除すると、そのテンプレートに基づく SmartSymbol インスタンスのプロパティを開いたり、編集したり、参照したりすることはできなくなります。これらの SmartSymbol インスタンスのランタイムの状態は、削除によって影響されません。

削除した SmartSymbol はそのインスタンスから回復できます。詳細については、「[SmartSymbol の回復](#)」を参照してください。

SmartSymbol テンプレート削除するには：

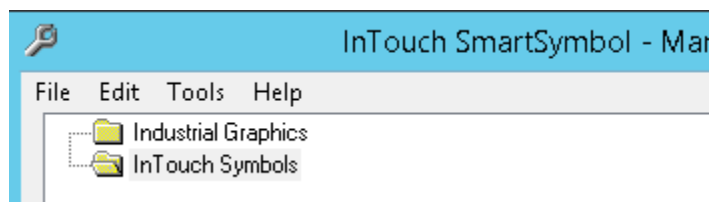
1. SmartSymbol マネージャで、削除する SmartSymbol を選択します。
2. [ファイル] メニューで、[削除] をクリックします。メッセージが表示されたら、[はい] をクリックします。

SmartSymbol テンプレートが SmartSymbol ライブラリから削除されます。この SmartSymbol のすべてのインスタンスは孤立します。

フォルダ階層への SmartSymbol の保存

SmartSymbol は、標準の階層フォルダ構造で、SmartSymbol ライブラリに保存されます。SmartSymbol ライブラリの SmartSymbol の組織を簡素化するために、次の 2 つの標準フォルダが含まれています。

- 産業用グラフィック SmartSymbol テンプレートの最上位レベルのフォルダ
- InTouch SmartSymbol テンプレートの最上位レベルのフォルダ



SmartSymbol マネージャを使用して、テンプレートのサブフォルダを作成できます。作成されるときに関連付けられる必要のあるテンプレートフォルダに産業用グラフィック SmartSymbol テンプレートを保存します。たとえば、\$Valve オブジェクトで使用するために SmartSymbol を作成した場合、シンボルテンプレートは「\$Valve」テンプレートフォルダに保存します。

産業用グラフィック SmartSymbol を InTouch シンボルフォルダにドラッグすることはできません。また、InTouch SmartSymbol を産業用グラフィック シンボルフォルダにドラッグすることもできません。

SmartSymbol を別のフォルダに移動するには

1. 移動する SmartSymbol を選択します。

2. SmartSymbol を新しいフォルダにドラッグします。

SmartSymbol と言語の切り替えのサポート

アプリケーションに SmartSymbol テンプレートが存在していれば、SmartSymbol の言語の切り替えは有効です。

SmartSymbol に翻訳可能なテキスト オブジェクトが含まれる場合、ディクショナリをエクスポートすると、SSD_<SymbolName>_<LangID>_<ID>.xml のような個別の XML が生成されます。この XML ファイルには、SmartSymbol に含まれるすべての翻訳可能な文字列が含まれています。このファイルを Excel で開き、他の InTouch アプリケーションに対して行うようにテキスト文字列を翻訳できます。

InTouch アプリケーションの翻訳をインポートすると、各 SmartSymbol の翻訳も同じようにインポートされます。

WindowViewer で言語を切り替えると、この方法で翻訳された翻訳可能な文字列を含む SmartSymbol が翻訳された状態で表示されます。

ディクショナリ ファイルを持つ SmartSymbol をエクスポートすると、ディクショナリ ファイルが .www ファイルと共にエクスポートされます。言語の切り替えの詳細については、『AVEVA™ InTouch HMI アプリケーションランタイム ガイド』の「[ランタイムでの言語の切り替え](#)」を参照してください。

SmartSymbol の回復

ライブラリから SmartSymbol テンプレートを削除すると、その SmartSymbol のすべてのインスタンスは「孤立した」インスタンスと考えられます。削除した SmartSymbol は、孤立したインスタンスから回復できます。アプリケーション ウィンドウに孤立したインスタンスが存在しない場合、SmartSymbol は回復できません。

SmartSymbol テンプレートが削除された後にインスタンスのプロパティを開こうとすると、ライブラリには SmartSymbol が存在しないことを伝える警告メッセージが表示されます。

また、SmartSymbol が含まれるウィンドウをインポートすると、孤立したインスタンスを使用することもできます。孤立したインスタンスから SmartSymbol を回復して、その SmartSymbol の名前を変更する必要があります。

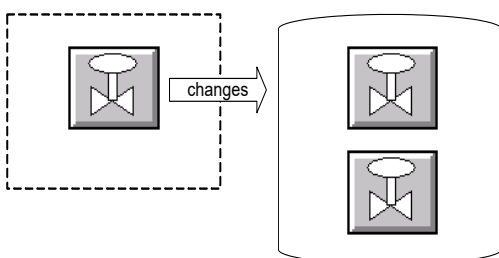
削除した SmartSymbol を回復するには:

1. InTouch HMI アプリケーション ウィンドウで、削除した SmartSymbol の孤立したインスタンスをクリックします。
2. [描画] メニューの [SmartSymbol] グループで [回復] をクリックします。
SmartSymbol が [新規シンボル] の名前と共に、[SmartSymbol - 管理モード] ウィンドウに表示されます。
3. 必要に応じて、SmartSymbol の名前を変更します。

SmartSymbol の編集

SmartSymbol を作成した後で、テンプレートまたは SmartSymbol のインスタンスを変更したり修正したりすることによって、SmartSymbol を編集できます。

SmartSymbol Template SmartSymbol Instances



SmartSymbol テンプレートの変更

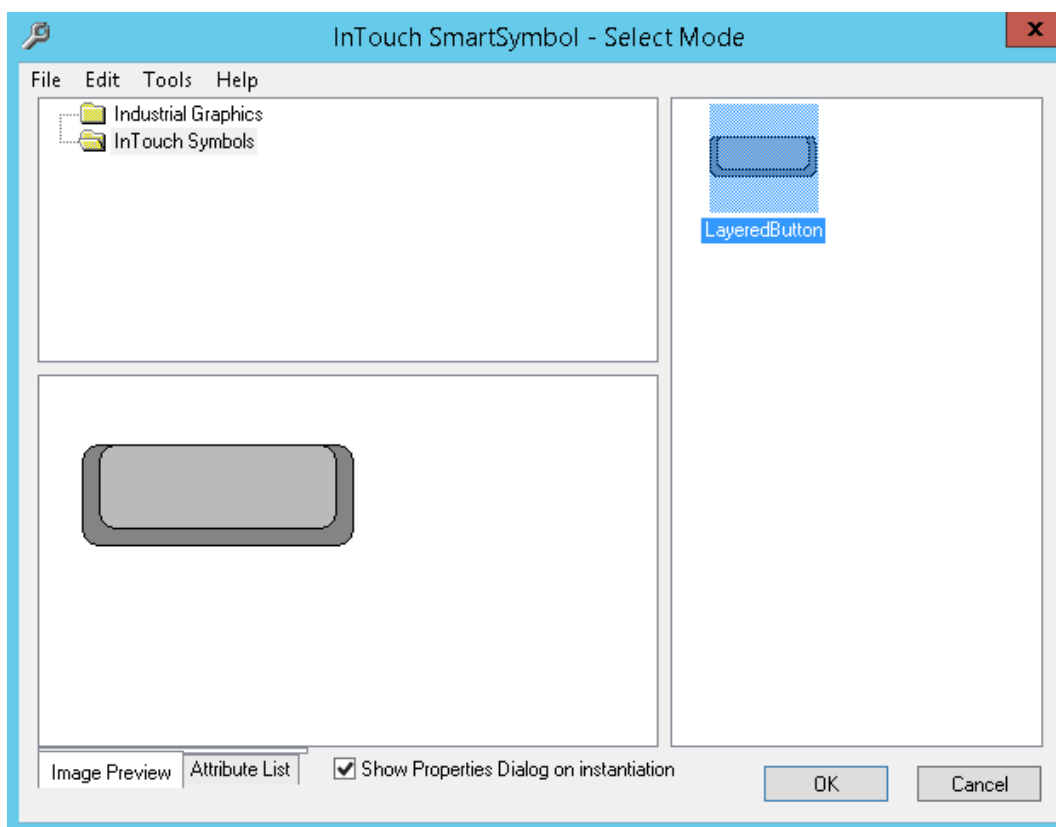
SmartSymbol を編集するには、セルを分解してから描画ツールを使用して変更を行います。SmartSymbol に関連付けられたアニメーションを変更することもできます。テンプレートを変更すると、SmartSymbol のすべてのインスタンスに影響します。

注記: SmartSymbol は、アプリケーション ウィンドウではなく一時ウィンドウで編集します。

既存の SmartSymbol テンプレートを編集するには

1. [描画] メニューの [SmartSymbol] グループで [編集の開始] をクリックします。
2. SmartSymbol を編集するウィンドウをクリックします。

[InTouch SmartSymbol - 選択モード] ウィンドウが表示されます。



3. 編集する SmartSymbol を選択して、[OK] をクリックします。

SmartSymbol のインスタンスがアプリケーション ウィンドウに配置されます。

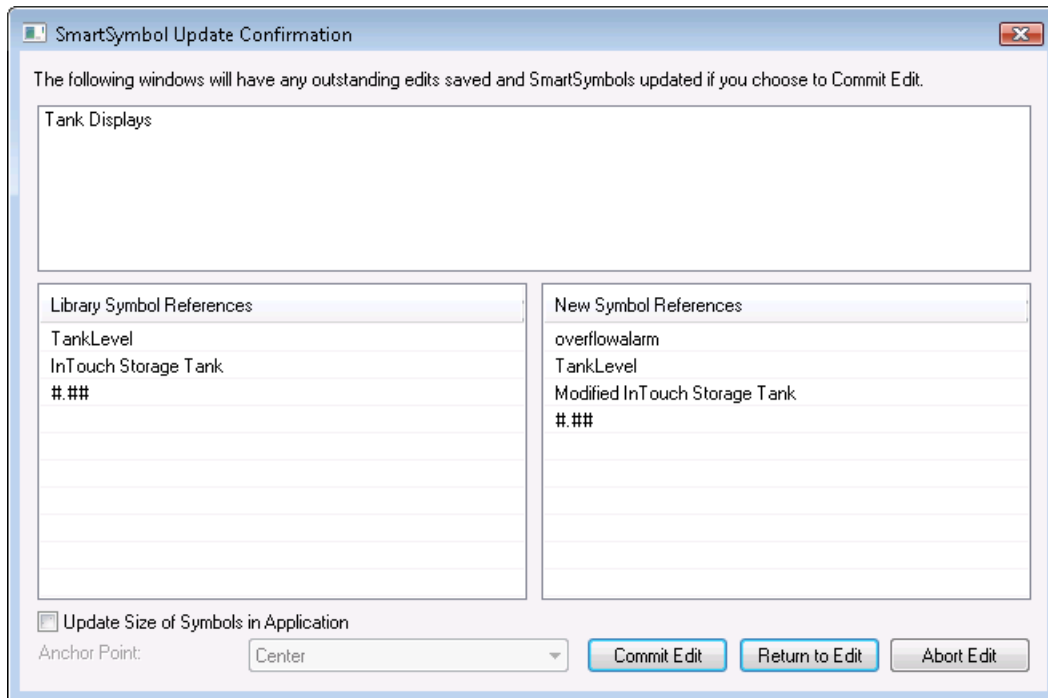
4. [アニメーション] メニューの [セル] グループで [分解] をクリックします。

シンボルがコンポーネント要素に分解されます。

5. これで、1つまたは複数の要素を編集できるようになりました。

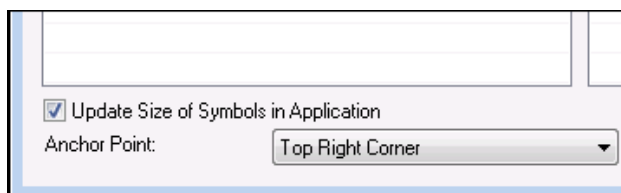
注記: SmartSymbol の一部であるセルに要素を追加すると、より大きなセルとなります。SmartSymbol インスタンスに変更を反映する場合、サイズの変更を反映させるかどうかを選択できます。

6. 編集が完了したら、シンボルのすべての要素を選択します。
7. [アニメーション] メニューの [セル] グループで [セルの作成] をクリックします。
8. [描画] メニューの [SmartSymbol] グループで [編集の終了] をクリックします。
- [SmartSymbol 更新確認] ダイアログ ボックスが表示されます。



9. 編集した SmartSymbol のサイズが変更された場合は、サイズの反映を設定できます。以下のいずれかを実行します。

- 既存の SmartSymbol インスタンスのサイズに影響を与えないようにするには、[アプリケーションでのシンボルの更新サイズ] チェック ボックスをオフにします。
- テンプレートサイズの変更を SmartSymbol インスタンスに反映させるには、[アプリケーションでのシンボルの更新サイズ] チェック ボックスをオンにして、[アンカー ポイント] リストで、サイズ変更が完了したときに画面に「アンカー」する SmartSymbol インスタンスの部分をクリックします。



10. 以下のいずれかを実行します。

- 変更した内容を適用するには、**[編集のコミット]** をクリックします。SmartSymbol マネージャにより、SmartSymbol テンプレートとすべてのインスタンスが更新されます。
- SmartSymbol の編集を続けるには、**[編集に戻る]** をクリックします。アプリケーション ウィンドウが再び表示され、編集を続けることができます。

SmartSymbol インスタンスの変更

SmartSymbol インスタンスの参照および静的テキストは変更することができます。インスタンスの静的テキストは、検索および置換することができます。

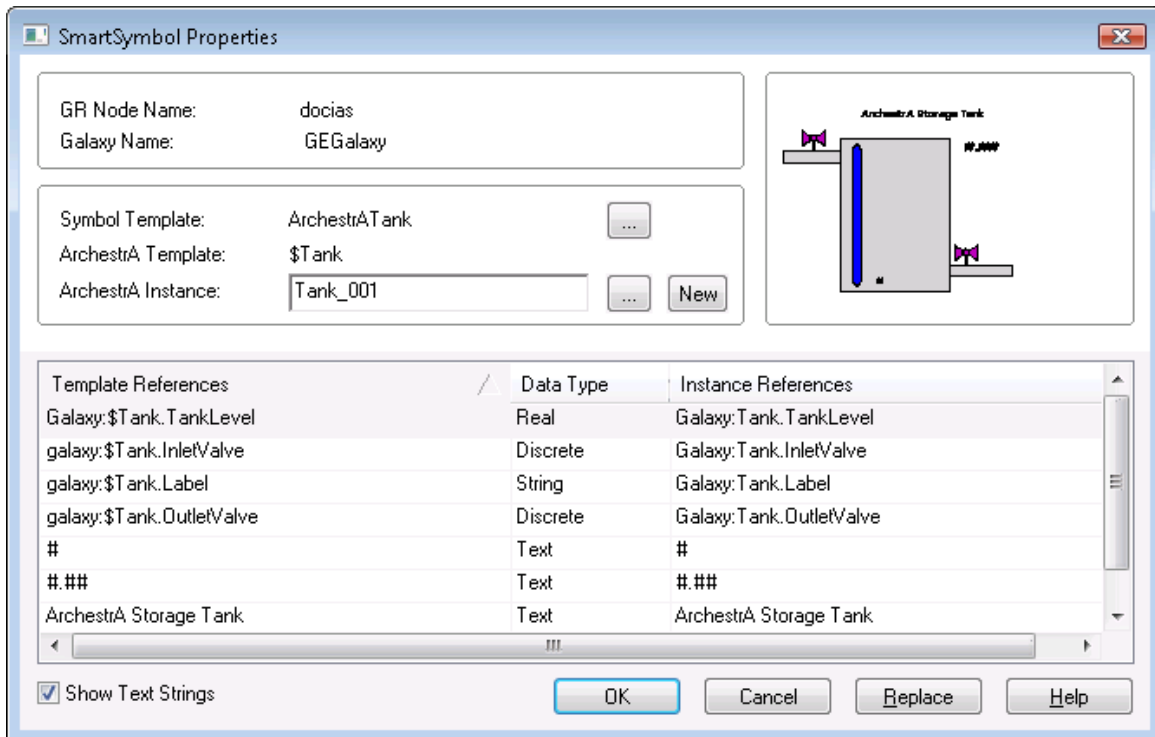
SmartSymbol インスタンスの別の参照の選択

SmartSymbol をインスタンスとしてウィンドウに配置すると、その参照先を別のオブジェクトや異なるタグ変数などに変更することができます。SmartSymbol テンプレートは影響されません。

ランタイムでは、SmartSymbol インスタンスが参照するタグ変数は、IOSetRemoteReferences() スクリプト関数を使用して変更できます。詳細については、「[ランタイム中のリモート参照のリダイレクト](#)」を参照してください。

SmartSymbol インスタンスで参照を編集するには

1. SmartSymbol インスタンスをダブルクリックします。**[SmartSymbol プロパティ]** ウィンドウが表示されます。



2. 以下のいずれかを実行します。

- [シンボル テンプレート]** の横にある**省略記号**ボタンをクリックして、新しい SmartSymbol テンプレートを選択します。

- **[ArchestrA インスタンス]** テキスト ボックスの横にある**省略記号**ボタンをクリックして、ArchestrA オブジェクトインスタンスを参照します。

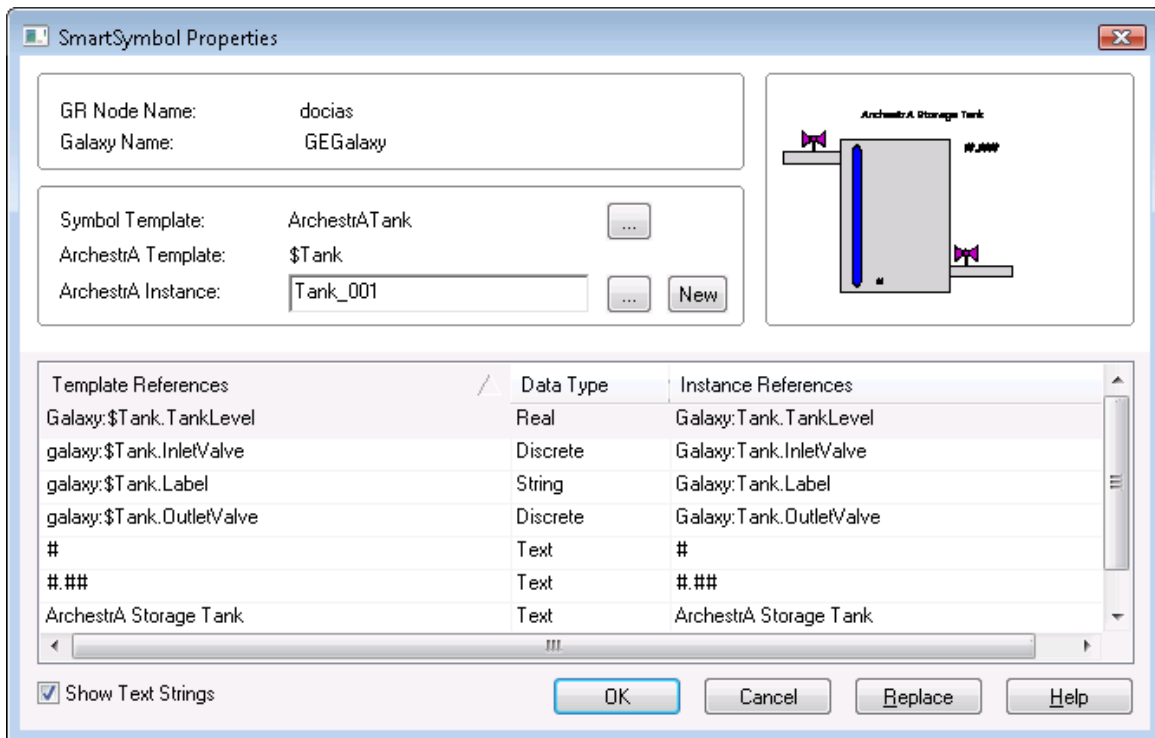
3. SmartSymbol にマップする別のオブジェクトインスタンスを選択し、**[OK]** をクリックします。

SmartSymbol インスタンスのテキストおよび参照の手動での編集

アプリケーション ウィンドウで SmartSymbol インスタンスを作成すると、インスタンスの静的テキストを変更できます。

SmartSymbol インスタンスの静的テキストを変更するには

1. SmartSymbol インスタンスをダブルクリックします。**[SmartSymbol プロパティ]** ウィンドウが表示されます。



2. **[インスタンス参照]** カラムで、テキスト ボックスをクリックし、テキストを変更します。

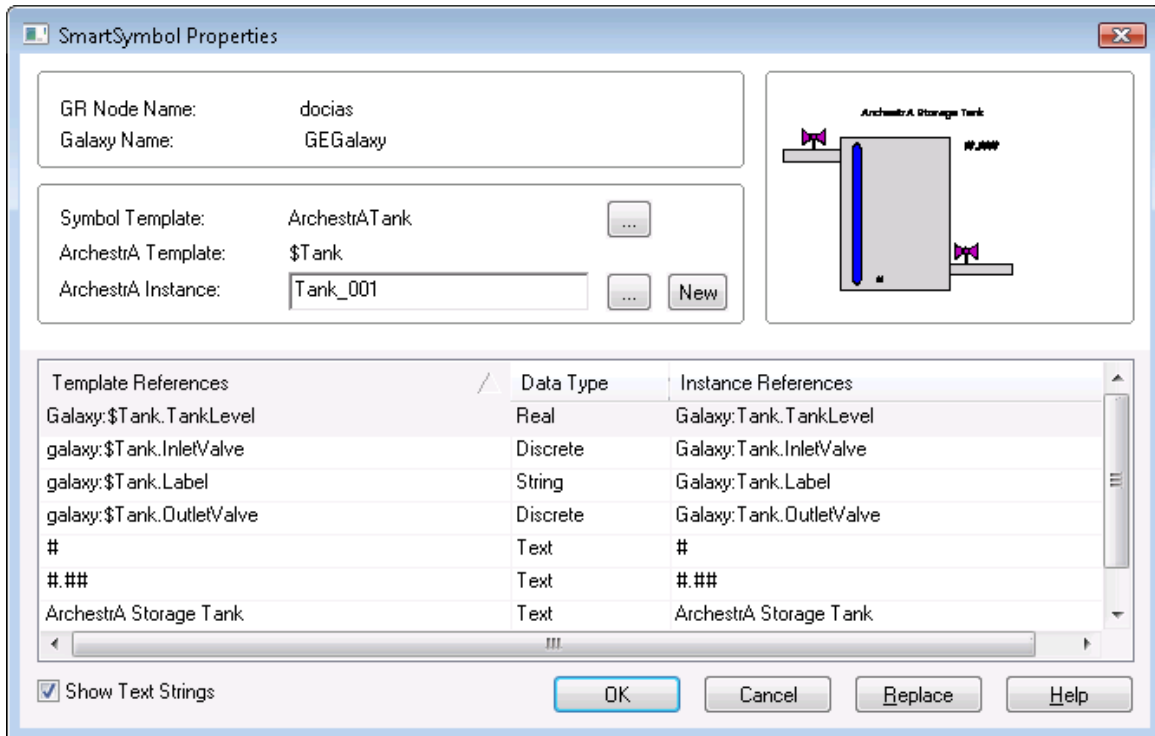
3. **[OK]** をクリックします。

SmartSymbol インスタンスのタグ変数とテキスト文字列の置換

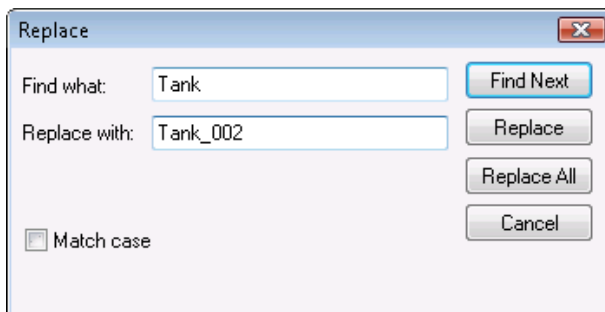
SmartSymbol インスタンスの複数の参照やテキスト文字列に同じ変更が必要な場合は、置換機能を使用できます。

SmartSymbol インスタンスの参照を置換するには

1. SmartSymbol インスタンスをダブルクリックします。**[SmartSymbol プロパティ]** ウィンドウが表示されます。



2. **[置換]** をクリックします。**[置換]** ダイアログ ボックスが表示されます。



3. 置換テキスト文字列を設定します。以下の操作を行います。
- [検索する文字列]** ボックスに、置換するテキストを入力します。**[大文字と小文字を区別する]** チェック ボックスをオンにして、検索時に大文字小文字が区別されるように設定します。
 - [置換後の文字列]** ボックスに、新しい置換テキストを入力します。置換テキストは常に入力されたとおりに使用されます。
4. 以下のいずれかを実行します。
- すべてのテキストを置換するには、**[すべて置換]** をクリックします。
 - 一度に 1 つのインスタンスのテキストを検索して置換するには、**[次を検索]** をクリックして、**[置換して次に]** をクリックし、そのインスタンスを置換します。
5. **[OK]** をクリックします。アプリケーション ウィンドウに、タグ変数とテキスト文字列が変更された状態の SmartSymbol インスタンスが表示されます。

InTouch SmartSymbol の移行

InTouch SmartSymbol は、インポート（移行）することによって産業用グラフィックで使用できます。SmartSymbol の外観および設定がインポートされ、アニメーション設定に変換されます。

インポートされた SmartSymbol では次のことが可能です。

- キャンバスの既存の要素に追加する。
- キャンバスの既存の要素を置換する。

通常、どの InTouch SmartSymbol でも産業用グラフィックにインポートできます。

注記: SmartSymbol には、インポートできないオブジェクト、またはインポートはできても機能が制限されているオブジェクトが含まれている場合があります。これらのオブジェクトの完全なリストについては、「[SmartSymbol インポートの制限](#)」を参照してください。

産業用グラフィックへの InTouch SmartSymbol のインポート

産業用グラフィックに InTouch SmartSymbol をインポートするには

1. 産業用グラフィック エディタを開きます。
2. [システム] メニューで、[InTouch SmartSymbol のインポート] をクリックします。[InTouch アプリケーション ウィザードの検索] が表示されます。
3. InTouch アプリケーションがデフォルト以外のフォルダにある場合、[参照] ボタンをクリックしてアプリケーションのパスを参照します。
4. [アプリケーションの検索] を選択します。[検索ルート] ボックスは、すべてのアプリケーションが検索されるパスを表示します。
5. 指定した検索ルート パスの下に InTouch アプリケーションがない場合、1 個を検索または参照するための新しい開始フォルダを入力して、検索ルート パスを変更します。
6. [検索] をクリックします。指定した検索ルート フォルダに含まれるすべての InTouch アプリケーションが検索され、一覧表示されます。
7. SmartSymbol をインポートするアプリケーションを選択し、[次へ] をクリックします。[InTouch SmartSymbol の選択] ダイアログ ボックスが表示されます。
8. SmartSymbol 階層にある SmartSymbol の位置を参照し、インポートする SmartSymbol を選択し、[OK] をクリックします。

キャンバス上に要素がすでにある場合、既存の要素を置換するかどうかを確認するダイアログ ボックスが表示されます。クリック:

- [はい] 既存の要素を削除し、空のキャンバスに SmartSymbol をインポートする場合。
 - [いいえ] 既存の要素をそのままにして、SmartSymbol をインポートする場合。
9. SmartSymbol に現在オペレーティング システムにインストールされていないフォントが含まれる場合、[フォント マッピングの編集] ダイアログ ボックスが表示されます。

[続行] をクリックして提案されるフォント マッピングを受け入れるか、それぞれ個別のフォントのフォント マッピングを変更できます。目的

 - a. [マップされたフォント] 列のフォント名をクリックします。[参照] ボタンが表示されます。
 - b. [参照] ボタンをクリックします。[サポートされるフォントの選択] ダイアログ ボックスが表示されます。

- c. リストからフォントを選択して、**[OK]** をクリックします。**[マップされたフォント]** 列の選択されたフォント名をクリックします。
- d. 別のフォントをマップする場合は、他のフォントでこの手順を繰り返します。

注記: 次回 SmartSymbol をインポートするためのマッピングを保存する場合、**[Save mapping]** をチェックします。

1. SmartSymbol がインポートされ、キャンバスに表示されます。

SmartSymbol インポートの制限

SmartSymbol をインポートするとき、以下の設定がインポートされます。

- InTouch グラフィック
- グラフィック アニメーション
- スクリプト
- 参照

InTouch グラフィックのインポート

以下の表には、次のような InTouch グラフィックが示されています。

- 問題なくインポート可能です。
- インポートできますが、機能に変更されたり、プロセス中にいくつかの機能が失われます。
- インポートできません。

以下の InTouch グラフィックは、問題なくインポート可能です。

InTouch グラフィック	産業用グラフィック要素	注記
長方形	長方形	
角丸長方形	角丸長方形	
楕円形	楕円形	
線	線	
水平/垂直線	線	SmartSymbols は水平/垂直線を線に変換します。従って、ArchestrA はラインのみを生成します。
折れ線	折れ線	
多角形	多角形	
テキスト	テキスト	
ビットマップ	ビットマップ	
セル	グループ	ArchestrA property "Treat as Icon" = false.

InTouch グラフィック	産業用グラフィック要素	注記
----------------	-------------	----

シンボル	グループ	ArchestrA property "Treat as Icon" = true.
------	------	--

ボタン	ボタン	
-----	-----	--

以下の InTouch グラフィックをインポートできますが、機能に変更されたり、プロセス中にいくつかの機能が失われます。

InTouch グラフィック	産業用グラフィック要素	注記
----------------	-------------	----

ウィザード	要素	SmartSymbol にグループ化するとき、要素のグループとして表示されます。
-------	----	--

SmartSymbol	要素	別の Smart Symbol にグループ化されるとき、セルに分割し、SmartSymbol プロパティ自体を失います。
-------------	----	--

SmartSymbol に追加できないため、以下の InTouch グラフィックはインポートできません。

InTouch グラフィック	産業用グラフィック要素	注記
----------------	-------------	----

リアルタイムトレンド	該当なし	SmartSymbol に追加できません。
------------	------	-----------------------

履歴トレンド	該当なし	SmartSymbol に追加できません。
--------	------	-----------------------

ActiveX コントロール	該当なし	SmartSymbol に追加できません。これは、すべての ActiveX アラーム コントロール (Alarm DB View、Alarm Viewer など) を含む可能性があります。
----------------	------	--

グラフィック アニメーションのインポート

InTouch SmartSymbol をインポートするとき、InTouch アニメーションで設定されたすべてのデータは、ArchestrA アニメーションにインポートされます。InTouch アニメーションおよび ArchestrA アニメーションは別の名前を持ちますが、同じ機能を実行します。

以下の表は、どのアニメーションがそれぞれに対応しているのかを示します。

InTouch アニメーション リンク	ArchestrA アニメーション
ユーザー入力 - 論理型	ユーザー入力

InTouch アニメーション リンク	Archestra アニメーション
ユーザー入力 - アナログ型	ユーザー入力
ユーザー入力 - 文字列	ユーザー入力
スライダー - 垂直スライダー	垂直スライダー
スライダー - 水平スライダー	水平スライダー
タッチ押しボタン - 論理値	押しボタン
タッチ押しボタン - アクション	アクション スクリプト
タッチ押しボタン - ウィンドウの表示	サポートしていません
タッチ押しボタン - ウィンドウの非表示	サポートしていません
線の色 - 論理型	線スタイル
線の色 - アナログ型	線スタイル
線の色 - 論理型アラーム	線スタイル
線の色 - アナログ アラーム	線スタイル
塗りつぶしの色 - 論理型	塗りつぶしスタイル
塗りつぶしの色 - アナログ型	塗りつぶしスタイル
塗りつぶしの色 - 論理型アラーム	塗りつぶしスタイル
塗りつぶしの色 - アナログ アラーム	塗りつぶしスタイル
テキストの色 - 論理型	テキスト スタイル
テキストの色 - アナログ型	テキスト スタイル
テキストの色 - 論理型アラーム	テキスト スタイル
テキストの色 - アナログ アラーム	テキスト スタイル
オブジェクト サイズ - 高さ	高さ
オブジェクト サイズ - 幅	幅
位置 - 垂直	水平位置
位置 - 水平	垂直位置
塗りつぶしパーセント - 垂直	垂直塗りつぶしパーセン ト
塗りつぶしパーセント - 水平	水平塗りつぶしパーセン ト
その他 - 表示オン/オフ	表示オン/オフ

InTouch アニメーション リンク	Archestra アニメーション
その他 - 点滅	点滅
その他 - 向き	回転
その他 - 無効	無効化
その他 - ツールヒント	ツールヒント
値の表示 - 論理値	値の表示
値の表示 - アナログ値	値の表示
値の表示 - 文字列の値	値の表示

アクション スクリプトのインポート

SmartSymbol をインポートすると、オブジェクトに関連付けられている SmartSymbol 内のすべてのアクション スクリプトもインポートされます。SmartSymbol 内のアクション スクリプトは、産業用グラフィックのスクリプト アニメーションになります。

定義済み InTouch 関数 (QuickScripts) のほとんどはインポートされています。

数学関数

InTouch WindowMaker に含まれる次の数学関数が産業用グラフィック エディタでサポートされています。

Abs、ArcCos、ArcSin、ArcTan、Cos、Exp、Int、Log、LogN、Pi、Round、Sgn、Sin、Sqrt、Tan、Trunc

文字列関数

InTouch WindowMaker に含まれる次の文字列関数が産業用グラフィック エディタでサポートされています。

Dtext、StringASCII、StringChar、StringCompare、StringCompareNoCase、StringFromGMTTimeToLocal、StringFromIntg、StringFromReal、StringFromTime、StringFromTimeLocal、StringInString、StringLeft、StringLen、StringLower、StringMid、StringReplace、StringRight、StringSpace、StringTest、StringToIntg、StringToReal、StringTrim、StringUpper、Text、wwStringFromTime

システム関数

InTouch WindowMaker に含まれる次のシステム関数が産業用グラフィック エディタでサポートされています。

ActivateApp

その他の関数

InTouch WindowMaker に含まれる次のその他の関数が産業用グラフィック エディタでサポートされています。

DateTimeGMT、LogMessage、SendKeys、WWControl

参照のインポート

SmartSymbol をインポートするとき、以下の変更がタグと参照に行われます。

InTouch SmartSymbol	産業用グラフィック	例
ローカル タグ	"InTouch:" キーワードの頭字語。	リアル メモリ タグ "TankLevel1" は、"InTouch:TankLevel1" に変換されます。
ドット フィールドのローカル タグ	"InTouch:" キーワードの頭字語。	論理型メモリ タグ "TankLevel1.InAlarm" は、"InTouch:TankLevel1.InAlarm" に変換されます。
スーパータグ	"InTouch:" キーワードの頭字語。以下の構文で式を手動で囲う必要があります。 属性("...");	リアル SuperTag メンバー "Reactor1\Level" は "InTouch:Reactor1\Level" に変換されます。以下のように式を手動で変更する必要があります。 attribute("InTouch:Reactor1\Level");
I/O 参照	"InTouch:" キーワードの頭字語。	整数 I/O タグ "Testprot:i00" は "InTouch:Testprot:i00" に変換されます。
Galaxy 参照	"Galaxy:" の頭字語が削除されます。	Galaxy 参照 "galaxy:Pump1.Valve1" は "Pump1.Valve1" に変換されます。

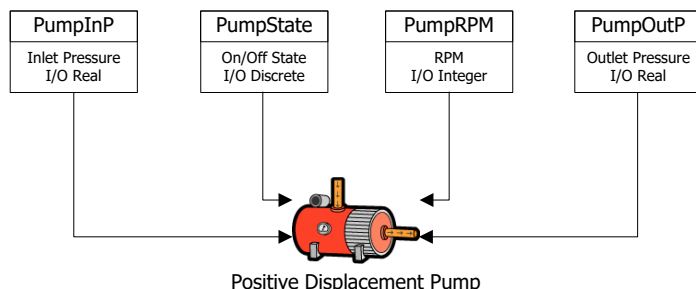
以下のアイテムは機能を変更した上でインポートされます。

InTouch SmartSymbol	産業用グラフィック	例
Galaxy:ObjectTagname. Property.#VString	"Galaxy:" の頭字語が削除されますが、#VString はサポートされていません。#VString1、#Vstring2、#VString3、#VString4 にも適用されます。	"Galaxy:Tank.PV.#VString4" は "Tank.PV" に変換されます。
Galaxy:ObjectTagname. Property.#ReadSts	"Galaxy:" の頭字語が削除されますが、#ReadSts はサポートされていません。	"Galaxy:Tank.PV.#ReadSts" は "Tank.PV" に変換されます。
Galaxy:ObjectTagname. Property.#WriteSts	"Galaxy:" の頭字語が削除されますが、#WriteSts はサポートされていません。	"Galaxy:Tank.PV.#WriteSts" は "Tank.PV" に変換されます。
Galaxy:ObjectTagname. Property.#EnumOrdinal	"Galaxy:" の頭字語が削除されますが、#EnumOrdinal はサポートされていません。	"Galaxy:Selection.Sel1.#EnumOrdinal" は "Selection.Sel1" に変換されます。

章 9 タグ

InTouch ヒューマン マシン インターフェイス (HMI) は、製造環境におけるコンポーネントをグラフィック表示するアプリケーションです。このグラフィカル インターフェイスを使用して、工場のオペレータは製造工程を監視および管理することができます。

以下の図は、製造工程のコンポーネントであるポンプの例を示しています。ポンプには、値に関連付けられているプロパティがあります。圧力、RPM、およびステータスはポンプのプロパティであり、その値は HMI から監視されます。



タグは InTouch HMI アプリケーションのデータ アイテムを表します。タグを使用すると、製造環境からの特定のデータ アイテムとして、特定のコンポーネント プロパティをアクセス可能にできます。上の図では、**PumpState** タグはポンプがオンであるかオフであることを示しています。製造環境において、InTouch アプリケーションでプロパティを監視または制御するコンポーネントに対してタグを作成します。

製造コンポーネントから収集された異なるタイプのデータに対しては異なるタイプのタグを使用できます。たとえば、**PumpState** タグは、ポンプが実行中であるか停止中であることを示すブール型のオン/オフ値を返します。アプリケーションの一部とするデータ型に対しては、適切なタイプの InTouch タグを割り当てます。

タグ名ディクショナリを使用したタグの管理

タグ名ディクショナリを使用して、InTouch アプリケーションのタグを作成できます。以下の図は、[タグ名ディクショナリ] ダイアログ ボックスと、I/O タグのプロパティを定義するすべてのオプションを示しています。

The screenshot shows the 'Tagname Dictionary' dialog box with the following sections and labels:

- Level of Tagname:** Points to the 'Main' tab.
- Existing Tag Selection:** Points to the 'Type: ...' dropdown menu.
- Tag Type Selection:** Points to the 'I/O Integer' dropdown menu.
- Tag Name Assignment:** Points to the 'Tagname: PumpRPM' field.
- Tag Data Logging:** Points to the 'Log Data' checkbox.
- Initial Data Value:** Points to the 'Initial Value: 0' field.
- Data Unit of Measure:** Points to the 'Eng Units:' field.
- Item Linked to IO Tag:** Points to the 'Item: RPM' field.
- Tag Comment:** Points to the 'Comment: AccessLevel' field.
- Data Retention:** Points to the 'Retentive Value' checkbox.
- Tag Upper and Lower Data:** Points to the 'Min EU: -32768' and 'Max EU: 32767' fields.
- Access Name Assigned to:** Points to the 'Access Name: ...' field.
- Alarm Options:** Points to the 'Alarm Value' and 'Priority' fields.

タグ変数の使用法の計画

計画段階でアプリケーションのタグ変数の主要な必要条件を事前に指定することにより、開発時間を削減できます。また、すべての計画を実行すれば、InTouch アプリケーションの作成に要する時間を短縮できます。

タグ変数を作成する前に：

- InTouch アプリケーションの中で表す必要があるプロセスのすべての物理コンポーネントを特定します。

アプリケーション内でデータ ソースとして表す必要があるコンポーネント属性のリストを作成します。

- 各コンポーネント属性に割り当てられるデータのタイプを特定します。

コンポーネント属性に割り当てられたデータに基づいて、各タグ変数にタグ変数タイプを割り当てます。タグ変数へのデータ タイプの割り当てについては、「[新しいタグの作成](#)」を参照してください。

- InTouch アプリケーションの中に組み込む必要があるデータの特性を決定します。

各タグ変数に対する以下のデータ特性を評価します。

- データ値の予測範囲
- データ値に割り当てる測定単位
- データの初期値

- タグ変数の値に変更ありと認識する際のしきい値を設定するためのデッドバンド値
- タグ変数の値によって状況が変更された場合に表示されるメッセージ

タグ変数データの特性の定義については、「[タグ変数プロパティの理解](#)」を参照してください。

- タグ変数の命名の規則と標準を作成します。

複雑なアプリケーションには通常、多くのタグ変数が必要です。命名規則を作成して標準化すれば、アプリケーションにおけるタグ変数の構成が分かるようになります。タグ変数の命名規則については、「[タグ名規則](#)」を参照してください。

- どの処理データを保存する必要があるかを決定します。

選択されたデータはログ ファイルに保存されます。ログ済みデータを使用することにより、タグ変数の値の時系列変化を示す履歴トレンドを作成できます。タグ変数のログ記録の設定については、「[タグ変数のログ記録](#)」を参照してください。

新しいタグの作成

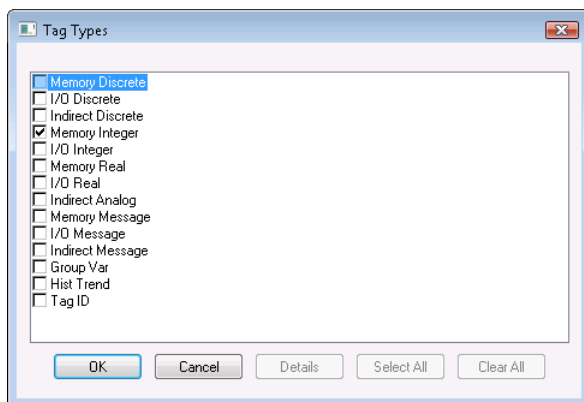
WindowMaker [タグ名ディクショナリ] を使用してタグを作成します。作成を開始する前に、工場のプロセスを分析して InTouch アプリケーション内で作成する必要があるタグを決定します。

新しいタグを作成するには

1. WindowMaker で InTouch アプリケーションを開きます。
2. [ホーム] メニューの [タグ] グループで [タグディクショナリ] をクリックします。

初めてタグ名ディクショナリを開くと、**\$AccessLevel システム タグ**の定義が [タグ名] ボックスに表示されます。新しいタグを保存した後は、[タグ名ディクショナリ] には最後に保存されたタグの定義が表示されます。

3. 以下の手順を実行します。
 - a. [新規] をクリックします。[タグ名] ボックスが空白になります。
 - b. 新しいタグの名前を入力します。タグの命名要件については、「[タグ名規則](#)」を参照してください。
 - c. オプションとして、[コメント] ボックスに新しいタグに関するコメントを入力します。
4. [タグタイプ] ボタンをクリックします。[タグタイプ] ダイアログ ボックスが表示されて、サポートされている InTouch タグタイプが表示されます。



5. リストからタグのタイプを選択して、**[OK]** をクリックします。タグ名ディクショナリが再表示されて、選択したタグのタイプが表示されます。
6. 必要に応じて**[詳細設定]** をクリックすると、選択したタグタイプに対応するタグ名ディクショナリの追加オプションが表示されます。
7. **[タグ名ディクショナリ]** ダイアログボックスでさらにタグ オプションを指定します。
タグプロパティの指定については、「[タグプロパティの設定](#)」を参照してください。
8. **[保存]** をクリックします。**[閉じる]** をクリックして、**[タグ名ディクショナリ]** ダイアログボックスを閉じます。

タグ プロパティの設定

[タグ名ディクショナリ] ダイアログボックスを使用して、すべてのタグ定義の一部である共通のタグプロパティを指定できます。各タグに名前を割り当てる必要があります。オプションとして、コメントの追加が可能です。タグ名とコメントは、すべてのタグに共通するプロパティです。

The screenshot shows the 'Tagname Dictionary' dialog box with the 'Details' tab active. The 'Tagname' field contains '\$AccessLevel', 'Type' is 'System Integer', and 'Local Tag' is unchecked. The 'Group' is '\$System', 'Read only' is selected, and 'Read Write' is unselected. The 'Comment' field contains 'AccessLevel'. At the bottom, 'Log Events' is checked and 'Priority' is set to '999'. Buttons for 'New', 'Restore', 'Delete', 'Save', 'Select...', '<<', '>>', 'Cancel', and 'Close' are visible at the top.

InTouch タグの各タイプには独自のデータ プロパティがあります。タグタイプを選択すると**[タグ名ディクショナリ]** ダイアログボックスが拡張されて、選択したタグタイプに応じたオプションのセットが表示されます。

共通のタグ変数プロパティ

各タグ変数には独自の名前を割り当てる必要があります。オプションとしてコメントをタグ変数定義の1部とすることができます。定義した各タグ変数に応じて適切なコメントを指定すると良いでしょう。

すべてのタグ変数は、別の共通のタグ変数プロパティのアラーム グループに属しています。デフォルトでは、すべてのタグ変数は **\$System** アラーム グループに属します。別のアラーム グループへのタグの割り当てについては、「[アラームの設定](#)」を参照してください。

タグ名規則

似通ったプロパティを持つ多くのタグ変数が必要な場合は、一貫した命名規則に従ってタグ変数に名前をつけます。

次に示す InTouch のタグ変数名の命名規則に従ってください。

- タグ変数名に使用できる最大文字数は **128** 文字です。
- タグ変数名の先頭の文字には英数字 (**A-Z**、**a-z**、**0-9**) を使用します。
タグ変数名には英数字のみを使用することをお勧めします。
- タグ変数名には英数字を **1** 文字以上の使用する必要があります。

オプションとして、次に示す特殊文字を使用します。

- ダッシュ	! 感嘆符	# 番号記号
\$ ドル記号	% パーセント	& アンパサンド
? 疑問符	@ アットマーク	_ 下線

タグ変数名への特殊文字の使用については、アプリケーションでどうしても必要な場合を除き、できるだけ避けるようにしてください。

- タグ変数名へのダッシュ (-) の使用は避けてください。

ダッシュは InTouch のタグ変数名に使用できる有効文字です。しかし InTouch では、ダッシュは論理演算を表す否定演算子または数値演算を表す減算演算子として評価されます。たとえば $A=B-C$ と表記すると、 A は B 引く C と解釈されるか、または $B-C$ という名前のタグ変数がタグ変数 A に割り当てられます。

タグ名が数字で始まる場合、ハイフンまたはマイナス記号 (-) は使用できません。

- タグ変数名にスペースを使用しないでください。
- タグ変数名には指数として解釈される可能性がある数字を使用しないでください。

たとえばタグ変数名を $125E4$ とすると、基数に 4 乗が付いた指数表現と解釈されるため、使用できません。

- タグ変数名には 16 進数として解釈される可能性がある数字を使用しないでください。

たとえば、タグ変数名を $0x123B$ とすると 16 進数として解釈されるため、使用できません。

タグ変数の自動的命名

タグ変数ディクショナリでタグ変数に名前を付ける場合は、使用している命名規則を InTouch HMI が追跡します。たとえばタグ変数名を Pump01、Pump02 とすると、InTouch HMI は次のタグ変数名として Pump03 を候補に挙げます。この名前は採用してもしなくても構いません。この命名支援機能はインデクシングと呼ばれています。

インデクシングは、タグ変数名の中の最後の続き番号に基づきます。たとえば、タグ変数名を PumpInP04LotB99A とした場合、InTouch HMI が次のタグ変数名として候補に挙げるのは PumpInP05LotB99A ではなく PumpInP04LotB100A です。

タグ コメント

タグ変数作成時のオプションとして、最大で 160 文字までのコメントをタグ変数ディクショナリの [コメント] ボックスに入力できます。

初めてタグ変数ディクショナリにアクセスすると、\$AccessLevel システム タグ変数のデフォルトコメントが [コメント] ボックスに表示されます。これから定義するタグ変数にこのコメントを使用しない場合は、削除してください。

タグ変数プロパティの理解

共通のタグ変数プロパティを指定したら、作成中のタグ変数のタイプに固有なその他のプロパティを定義する必要があります。タグ変数タイプごとのメモリ タグ変数の基本的なプロパティを次の表に示します。

タグ変数タ**イプ** 独自のプロパティ

論理型	初期値、オンメッセージ、オフメッセージ、コメント
整数	初期値、最小値、デッドバンド、工学単位、最大値、ログデッドバンド、コメント
実数	初期値、最小値、デッドバンド、工学単位、最大値、ログデッドバンド、コメント

メッセージ 最大文字数、初期値、コメント

I/O タグ変数には、ネットワーク通信を確立して、ネットワーク デバイスからの生データを InTouch アプリケーションで使用される正規化した値に変換するための追加のプロパティがあります。I/O タグ変数の定義については、「[I/O タグ変数のプロパティの設定](#)」を参照してください。

値の範囲、測定単位、および初期値

論理型、整数型、実数型、およびメッセージ型のタグ変数には、InTouch アプリケーションが WindowViewer で起動した際に初期値が割り当てられます。論理型タグ変数の場合の初期値は、とり得るバイナリ状態のいずれかです。整数型と実数型のタグ変数の場合、初期値はアプリケーションが起動したときにタグ変数に割り当てられた数値です。初期値はタグ変数ディクショナリで指定されます。

WindowViewer でアプリケーションの実行が停止した際に、タグ変数の最終値を初期値とするように指定できます。タグ変数ディクショナリから **【変数値保持】** オプションを選択すると、アプリケーションが再起動した際にタグ変数の初期値として最後の有効値が割り当てられます。

整数型と実数型のタグ変数には、タグ変数がとり得る数値範囲の下限と上限を設定するプロパティが含まれています。整数型と実数型のタグ変数にはどちらにも **【最小値】** および **【最大値】** プロパティがあり、範囲の下限と上限が定義されます。

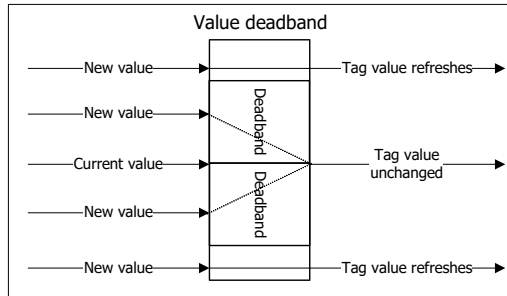
また整数型と実数型のタグ変数には、タグ変数値の測定単位を表す工学単位を割り当てる **【工学単位】** プロパティも含まれています。たとえば、ポンプ圧に関連する整数型タグ変数の **【工学単位】** プロパティには PSI を割り当てることができます。

タグ変数デッドバンド

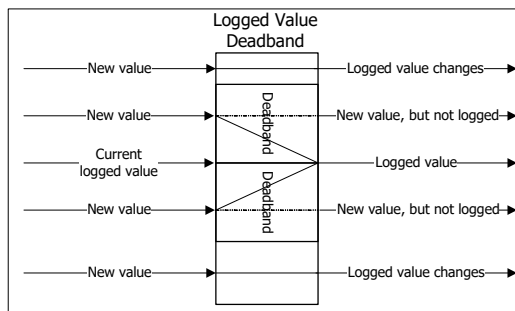
デッドバンドによりタグ変数値の感度が設定されます。デッドバンドは通常、絶えず値が変化する I/O タグ変数に割り当てられます。タグ変数の一時的な微小変化をデッドバンドによって取り除くことで、InTouch のデータ処理量が削減できます。

タグ変数ディクショナリには、整数データおよび実数データが割り当てられたタグ変数に対する 2 種類のデッドバンドプロパティが用意されています

- **デッドバンド値**：デッドバンド値プロパティにはしきい値が設定されています。WindowViewer では、このしきい値を超えた場合にのみランタイム メモリ内のタグ変数値がリフレッシュされます。以下の図は、現在のタグ変数値の前後にあるデッドバンドの領域を示しています。



- ログデッドバンド値：ログデッドバンドにはしきい値が設定され、このしきい値を超えなければタグ変数の値はログファイルに書き込まれません。以下の図は、ログ済みタグ変数の現在の値の前後にあるデッドバンドを示しています。



新しいタグ変数の値は、デッドバンドの外側の値のみがログファイルに書き込まれます。デッドバンド領域内の微小な変化は無視され、ログされません。

タグ変数デッドバンドを設定するには

1. [タグ変数ディクショナリ] ダイアログ ボックスを開きます。
2. [選択] をクリックします。[タグ変数を選択してください] ダイアログ ボックスが表示されます。現在定義されているアプリケーションのタグ変数のリストが表示されます。
3. リストから整数型か実数型のタグ変数タイプを選択します。
4. [OK] をクリックします。整数型か実数型のタグ変数を選択した場合には、[タグ変数ディクショナリ] ダイアログ ボックスの詳細設定に追加のオプションが表示されます。

Initial Value:	<input type="text" value="0"/>	Min Value:	<input type="text" value="0"/>	Deadband:	<input type="text" value="15"/>
Eng Units:	<input type="text" value="RPM"/>	Max Value:	<input type="text" value="2500"/>	Log Deadband:	<input type="text" value="25"/>

5. 選択したタグ変数タイプに従い整数値か実数値を [デッドバンド] ボックスに入力して、デッドバンド値を設定します。
デッドバンド値には、しきい値レベルの絶対値が工学単位で設定されます。
6. 選択したタグ変数タイプに従い整数値か実数値を [ログデッドバンド] ボックスに入力して、ログデッドバンドを設定します。
ログデッドバンドには、デッドバンド値と同様にしきい値の絶対値が工学単位で設定されます。
7. [保存] をクリックして、デッドバンドの変更を保存します。
8. [閉じる] をクリックして、[タグ変数ディクショナリ] ダイアログ ボックスを閉じます。

タグ変数値の保持

[タグ変数ディクショナリ]の詳細設定には、タグ変数の値とユーザーが加えたアラーム制限の変更を保持するための2種類のプロパティが用意されています。

[変数値保持] プロパティはすべてのタグ変数タイプに含まれています。アプリケーションが停止したときにタグ変数の現在の値を保持するには、[変数値保持]を選択してください。WindowViewerでアプリケーションを再起動すると、保持した値がこの変数の初期値として使用されます。

WindowViewerでアプリケーションが再起動されたときに、WindowViewerはI/Oデバイスに対して保持した値を書き込みません。I/Oの値は、I/Oサーバーがデータを供給するデバイスを最初にスキャンしたあとで更新されます。

WindowViewerの実行中は、新規または既存のタグ変数に対する[変数値保持]オプションを選択/解除することはできません。このオプションを選択すると、タグ変数の初期値は常に更新されて現在の値が反映されます。WindowViewerが終了すると、初期値は最後に保持されていた値に設定されます。このオプションを後で解除すると、タグ変数の初期値は最後に保持されていた値に設定されます。

整数型と実数型のタグ変数には、[パラメータ値保持]プロパティが含まれています。アプリケーションの実行中にユーザーがタグ変数のアラーム制限に加えたすべての変更を保持するには、[パラメータ値保持]を選択してください。WindowViewerでアプリケーションが再起動されると、変更したアラーム制限が初期値として使用されます。

I/O 接続

I/O タグ変数のすべてのタイプには、外部データ ソースのアクセス名とアイテム名を指定する必要があります。I/O タグ変数のアクセス名とアイテム名の指定については、「[I/O アクセス パラメータの設定](#)」を参照してください。

タグ変数のログ記録

WindowViewerでは、ランタイム中にログ変数のデッドバンドを超える変化がタグ変数の値に生じるたびに、履歴ログファイルに内容を書き込むようにすることができます。また WindowViewerでは、現在のタグ変数の値に関わりなく一定間隔おきにログファイルへの書き込みを行います。デフォルトの間隔は1時間です。

注: ログ記録の制御をさらに容易にしてその用途を拡大するには、Historian を使用して InTouch の履歴データを保存することをお勧めします。

[タグ名ディクショナリ] ダイアログ ボックスには、データとイベントをログファイルに記録するための個別のオプションがあります。変数値ログのオプションを設定することができます。イベントログの設定については、「[アラームの設定](#)」を参照してください。

タグ変数の値を履歴ログファイルに書き込むには、履歴ログを有効にする必要があります。一般的なログプロパティの設定については、「[履歴ログの設定](#)」を参照してください。

整数型と実数型のタグ変数に対しては、対応する詳細ダイアログ ボックスで[ログデッドバンド]を設定します。[ログデッドバンド] オプションでは、タグ変数の値が工学単位でいくつ変化したらログに内容が書き込まれるかを指定します。

タグ変数のログを設定するには

1. タグ名ディクショナリを開きます。
2. ログファイルに保存するデータのタグを選択します。
3. [データ ログ] を選択します。

☐ Log Data ☐ Log Events ☐ Retentive Value ☐ Retentive Parameters

4. オペレータ、I/O、QuickScript、またはオペレーティング システムに起因するタグ値の変更をログ記録するには、[イベント ログ] を選択します。[イベント ログ] を選択すると [優先度] ボックスが表示されます。

☒ Log Data ☒ Log Events Priority: 999 ☐ Retentive Value ☐ Retentive Parameters

[優先度] の値によって、タグ変数に対するイベントの優先度が決定されます。有効な入力値は 1～999 で、1 は最高優先度、999 は最低優先度を示します。

5. [保存] をクリックして、[タグ変数ディクショナリ] を閉じます。

論理型タグの作成

InTouch アプリケーション内で実行されている内部プロセスのバイナリ状態を示す論理型タグを指定できます。論理型タグにはオン/オフの初期値を割り当てる必要があります。タグに関連するプロセスがアラーム状態になったときやアラーム状態から抜け出したときにアラーム イベント ウィンドウに表示するメッセージを指定することもできます。

メモリ論理型タグを定義する手順を以下に示します。I/O 論理型タグは、プログラマブルコントローラ、プロセス コンピュータ、およびネットワーク ノードのデータとのすべての入出力のバイナリ状態を示します。

I/O 論理型タグのプロパティの設定の詳細については、「[論理型 I/O タグ変数の指定](#)」を参照してください。

メモリ論理型タグの初期値とメッセージを定義するには

1. [タグタイプ] ダイアログ ボックスで、タグのタイプとして [メモリ論理型] を選択します。
2. [タグ名ディクショナリ] ダイアログ ボックス上部の [詳細] を選択して詳細オプションを表示します。[タグ名ディクショナリ] ダイアログ ボックスの詳細部分が表示されます。

Initial Value
☐ On ☒ Off
 On Msg: Off Msg:

3. タグに関連付けられた初期値として [オン] または [オフ] を選択します。アプリケーションの起動時、タグがこの初期値に設定されます。
4. タグがアラーム状態になったときやアラーム状態から抜け出したときに表示するメッセージを [オンメッセージ] ボックスおよび [オフメッセージ] ボックスに入力します。

これらのメッセージは、タグにアラームの設定があるかどうかに関わらず、あらゆるアニメーション リンクやスクリプトで使用できます。

- タグ値が 1 (On, True) のときにアクティブな論理型アラームを定義した場合には、[オンメッセージ] ボックスに入力したメッセージが ActiveX アラーム表示の [値] 列および [しきい値] 列に表示されます。

タグのアラーム状態が通常状態に戻ると、[オフメッセージ] ボックスに入力したメッセージが [値] 列に表示され、[オンメッセージ] は [しきい値] 列に残ります。

- タグ値が 0 (Off, False) のときにアクティブな論理型アラームを定義した場合には、[オフメッセージ] ボックスに入力したメッセージが ActiveX アラーム表示の [値] 列および [しきい値] 列に表示されます。

タグのアラーム状態が通常状態に戻ると、[オン メッセージ] ボックスに入力したメッセージが [値] 列に表示され、[オフ メッセージ] は [しきい値] 列に残ります。

5. タグの変更を保存します。

整数型と実数型タグ変数の作成

整数型と実数型タグ変数を指定して InTouch アプリケーション内で実行されている処理の数値を表すことができます。

メモリと I/O の整数型と実数型のタグ変数を定義する方法を以下に示します。メモリ整数型とメモリ実数型のタグ変数には初期値を割り当てる必要があります。また、データの最大値と最小値を設定する必要があります。

I/O 整数型と I/O 実数型のタグ変数の I/O プロパティの設定については、[「整数型と実数型の I/O タグ変数の指定」](#)を参照してください。

重要: InTouch の実数値は、8 桁の精度に制限されています。切り捨て、切り上げによる発生を避けるために、実数型タグ変数のプロパティを定義する際は 8 桁の精度を超えないようにしてください。詳細については、[「IEEE 小数ユニット」](#)を参照してください。

メモリ整数型とメモリ実数型のタグ変数の値を定義するには

1. [タグ タイプ] ダイアログ ボックスで、タグのタイプにメモリ整数型かメモリ実数型を指定します。
[タグ変数ディクショナリ] ダイアログ ボックスの詳細設定が表示されます。

Initial Value:	<input type="text" value="0"/>	Min Value:	<input type="text" value="0"/>	Deadband:	<input type="text" value="15"/>
Eng Units:	<input type="text" value="PSI"/>	Max Value:	<input type="text" value="1500"/>	Log Deadband:	<input type="text" value="25"/>

2. 整数型と実数型タグ変数のプロパティを設定するには、以下の手順を実行します。

- アプリケーション起動時にタグ変数に関連付けられる整数値または実数値を、[初期値] ボックスに入力します。
- [最小値] ボックスに、タグ変数に対する最小整数値または最小実数値を入力します。
[最小値] フィールドに設定されるのは、メモリ整数型やメモリ実数型に関連付けられた数値がとり得る最小値です。
- [最大値] ボックスに、タグ変数に対する最大整数値または最大実数値を入力します。
[最大値] フィールドに設定されるのは、メモリ整数型やメモリ実数型に関連付けられた数値がとり得る最大値です。
- [工学単位] ボックスに、タグ変数名の工学単位を入力します。

3. タグ変数の変更を保存します。

タグ変数のデッドバンドとログ デッドバンドのプロパティの設定については、[「タグ変数デッドバンド」](#)を参照してください。

メッセージ型タグ変数の作成

内部処理と外部処理の両方に対するメッセージ型タグ変数を指定できます。これらのタグ変数には、Window Viewer でアプリケーションが起動されときに表示される初期メッセージと、Alarm Viewer から読み込むことができるコメントを指定するプロパティがあります。

I/O メッセージ型タグ変数の定義については、[「メッセージ型 I/O タグ変数の指定」](#)を参照してください。

メモリ メッセージ型タグ変数を定義するには

1. **[タグ変数タイプ]** ダイアログ ボックスで、タグ変数のタイプに **[メモリ メッセージ型]** を指定します。
2. 必要に応じて **[詳細設定]** を選択して、**[タグ変数ディクショナリ]** ダイアログ ボックスの詳細設定を表示させます。

Maximum Length: 131 Initial Value:
Alarm Comment:

3. メモリ メッセージ型タグ変数のプロパティを設定するには、以下の操作を行います。
 - **[最大文字数]** ボックスに、タグ変数のメッセージに表示できる最大文字数を整数値で入力します。あるいは、メッセージの最大文字数でデフォルト値の **131** 文字をそのまま使用します。
 - **[初期値]** ボックスには、**WindowView** でのアプリケーションの起動時にタグ変数に指定されるメッセージのテキストを入力します。
 - **[アラーム コメント]** ボックスには、メッセージ型タグ変数の **[イベント ログ]** オプションが選択されている場合に **AlarmViewer** コントロールに読み込まれるメッセージを入力します。
4. タグ変数の変更を保存します。

I/O タグ変数の作成

I/O タグ変数は、InTouch アプリケーションと外部処理間のネットワークの接続性を指定する共通プロパティのセットです。**[タグ変数ディクショナリ]** ダイアログ ボックスの詳細設定には、I/O タグ変数の外部プロパティを設定するためのオプションが含まれています。

I/O プロパティの設定については、[「I/O を使用したデータ アクセス」](#)を参照してください。

タグ変数の変更

タグ変数の変更は、タグ変数の作成に似ています。タグ変数ディクショナリから変更するタグ変数を選択します。次に、タグ変数の作成手順と同じように以下の手順に従ってタグ変数のプロパティを変更します。

重要: InTouch アプリケーションで使用された後のタグ変数のタイプを変更するのは容易ではありません。また、タグ変数のタイプの選択は、類似したデータタイプに制限されることがあります。タグ変数を定義する際は、慎重に検討して正しいデータタイプを選択してください。

タグ変数を変更するには

1. **[タグ変数ディクショナリ]** ダイアログ ボックスを開きます。
2. **[選択]** をクリックします。**[タグ変数を選択してください]** ダイアログ ボックスが表示されます。アプリケーションに対して現在定義されているタグ変数のリストが表示されます。
3. 変更するタグ変数をリストから選択します。
4. **[OK]** をクリックします。**[タグ変数ディクショナリ]** ダイアログ ボックスに、選択したタグ変数に対して指定されている設定値が表示されます。
5. タグ変数のプロパティを変更します。
6. **[保存]** をクリックして、変更したタグ変数を更新します。
7. **[閉じる]** をクリックして、タグ変数ディクショナリを閉じます。

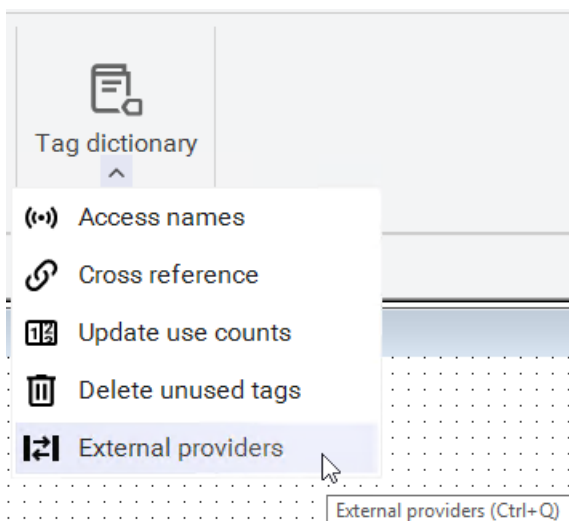
OPC UA サーバーからの InTouch タグの作成

[タグディクショナリ] の [外部プロバイダ] オプションには、同じマシン上にある Gateway Communication Driver で設定されたすべての OPC UA サーバー接続が一覧表示されます。このオプションを使用すると、いくつかの手動ステップを回避して、OPC UA アイテム参照用のアクセス名と InTouch タグを簡単に作成できます。

Gateway Communication Driver で設定された OPC UA サーバーから InTouch を作成するには
前提条件:

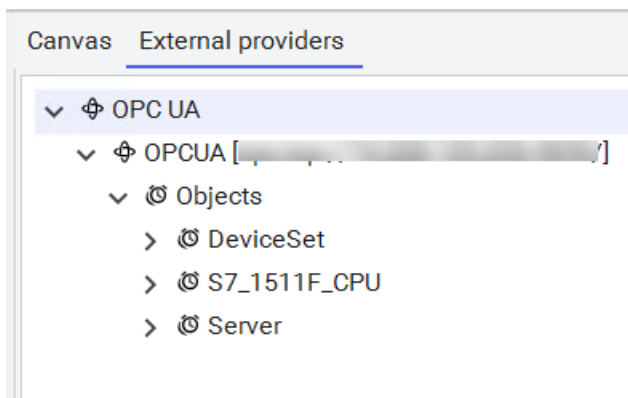
- Gateway Communication Driver で OPC UA サーバー接続を設定します。詳細については、Gateway Communication Driver ヘルプを参照してください。
- Gateway Communication Driver と InTouch WindowMaker は同じマシン上にある必要があります。

1. WindowMaker で InTouch アプリケーションを開きます。
2. [ホーム] メニューの [タグ] グループで [タグディクショナリ] の下にあるドロップダウン矢印をクリックして [外部プロバイダ] を選択します。または、キーボードショートカット **Ctrl + Q** を使用します。



Gateway Communication Driver で設定されたすべての OPC UA サーバー接続が表示されます。

3. OPC UA ツリーを展開してアイテム参照を表示および参照できます。



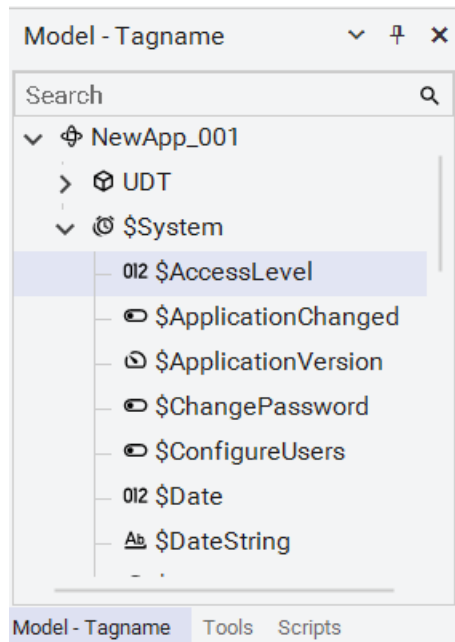
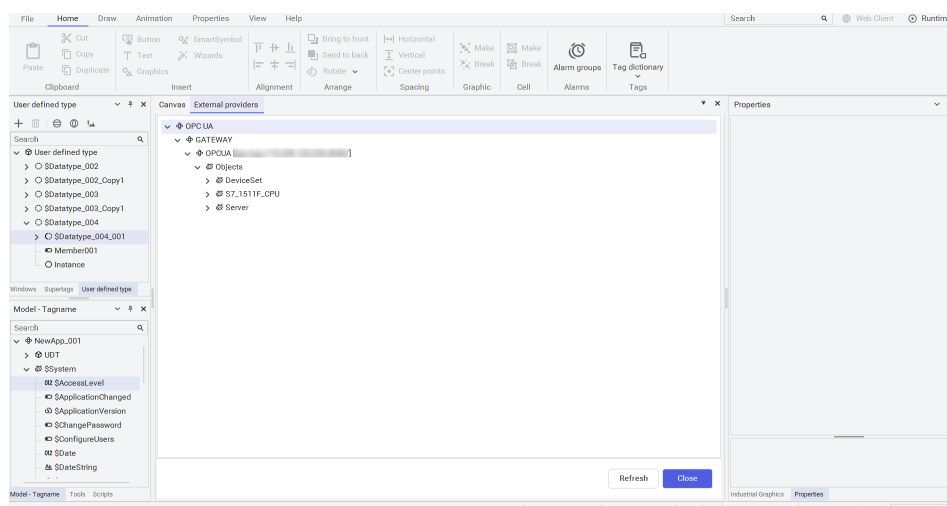
注記:

リモート Gateway Communication Driver からの OPC UA サーバーは表示されません。

OPC UA アイテムが表示された後に別の OPC UA サーバーを設定する場合、**[外部プロバイダ]** ウィンドウの下にある **[更新]** ボタンをクリックして、新しく追加した OPC UA サーバー アイテムを表示します。

4. [モデル - タグ名] ペインを開きます。

[モデル - タグ名] ペインに InTouch アプリケーションのすべてのアラーム グループとタグが表示されます。**[モデル - タグ名]** ペインの詳細については、「[モデル - タグ名](#)」トピックを参照してください。

**5. [外部プロバイダ] ウィンドウからアイテムを [モデル - タグ名] にドラッグアンドドロップします。Ctrl キーを押したままアイテムを選択することで複数のアイテムを選択できます。**

- 目的のアラーム グループ ノードにドラッグ アンド ドロップして OPC UA タグを特定のアラーム グループに追加できます。

- ランダムにドラッグアンドドロップすると、タグは **\$System** ノードの下に作成されます。

アイテムをドラッグアンドドロップすると OPC UA タグが作成されます。新しく作成された OPC UA タグは **「モデル - タグ名」** ペインに表示されます。

1. **「外部プロバイダ」** ウィンドウを閉じるには、**「閉じる」** をクリックします。

追加情報:

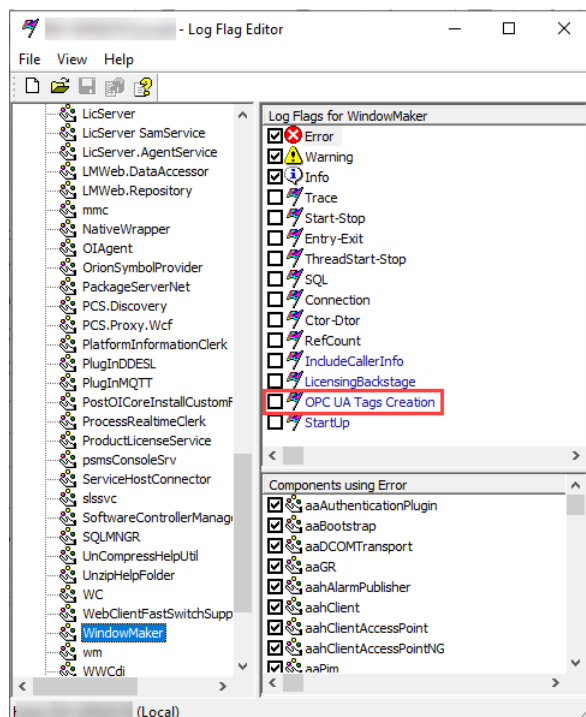
- 個々の OPC UA 接続には異なるアクセス名が追加されます。タグのアクセス名は、**Gateway Communication Driver** の OPC UA 接続のノード名です。

注記: タグを追加した後に **Gateway Communication Driver** の OPC UA ノードの名前を変更した場合、アクセス名を手動で変更する必要があります。アクセス名は自動的に変更されません。

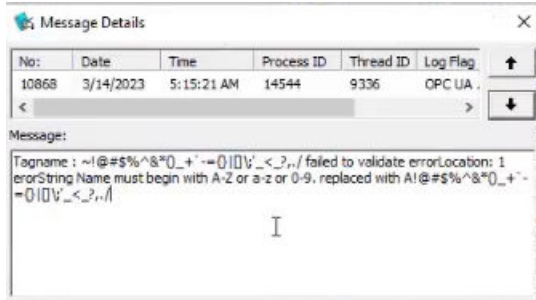
- 重複するタグ名にはインデックス番号が追加されます。たとえば、'Input' という名前の 2 つのタグがある場合、最初のタグの名前は 'Input' となり、2 番目のタグの名前は 'Input_1' になります。

注記: タグ名が最大タグ長の 128 文字を超える場合、128 文字になるまでタグ名の末尾の文字が削除されます。タグ名が 128 文字を超え、重複するタグ名がある場合、「タグ名 +_{インデックス}」が 128 文字になるようタグ名の末尾の文字が削除されます。

- OPC UA タグのデータ型が不明な場合、対応する InTouch タグは I/O メッセージ型として作成されます。
- アプリケーションマネージャで OPC UA サーバーとして設定することによって、InTouch を OPC UA サーバーとして使用することもできます。詳細については、「[InTouch OPC UA サーバーの設定および使用](#)」トピックを参照してください。
- WindowMaker のログフラグエディタで **「OPC UA タグ作成」** チェックボックスをオンにした場合、OPC UA タグに関連するログメッセージを Log Viewer で表示できます。

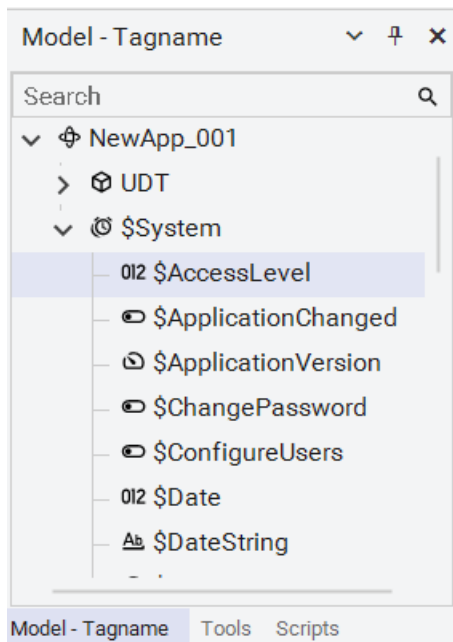


- OPC UA アイテム名にサポートされていない特殊文字が含まれる場合、タグ名のサポートされない特殊文字は、対応する InTouch タグを作成する際に 'A' または '_' で置き換えられます。サポートされない文字がタグ名の最初にある場合、その文字は 'A' で置き換えられます。その他の場所にあるサポートされない文字は '_' で置き換えられます。置換される文字の詳細については、Log Viewer の [メッセージの詳細] ウィンドウを参照してください。



モデル - タグ名

[モデル - タグ名] タブは、[ウィンドウ] タブと [スーパータグ] タブとともに InTouch WindowMaker の左側にあります。[モデル - タグ名] タブをクリックして [モデル - タグ名] ペインを表示します。InTouch アプリケーションのすべてのアラーム グループとタグが表示されます。[検索] ボックスを使用して、目的のタグを検索できます。



タグを開くには

1. [モデル - タグ名] ペインで、開くタグを右クリックします。
2. [開く] をクリックします。

タグが [タグ名ディクショナリ] ウィンドウで開きます。

タグを 1 つのアラーム グループから別のアラーム グループに移動するには

1. [モデル - タグ名] ペインで、移動するタグを選択します。

2. 目的のアラーム グループにドラッグ アンド ドロップします。

選択したタグが目的のアラーム グループに移動します。

タグを削除するには

1. [モデル - タグ名] ペインで、削除するタグを右クリックします。
複数のタグを選択できます。
2. [削除] をクリックします。
3. タグの削除を確認するメッセージが表示されたら [はい] をクリックします。
選択したタグが削除されます。

タグ変数の削除

InTouch アプリケーションに対して定義されたすべてのタグ変数のカウントが管理されています。このタグ変数のカウントは、アニメーション リンクやスクリプトが含まれるウィンドウが削除されても自動的に減算されません。削除されたウィンドウに関連するタグ変数はそのまま使用可能とみなされ、削除することはできません。

今後使用しないタグ変数の削除を可能にするには、**WindowViewer** を閉じてローカル タグとリモート タグの使用カウントを更新する必要があります。タグ変数がどこで使用されているかは、**InTouch** クロスリファレンス ユーティリティで確認できます。クロスリファレンス ユーティリティの使用とタグ変数のカウントの更新に関して詳しくは、「[タグの使用数の削減](#)」を参照してください。

使用カウントが更新されればタグ変数を削除することができます。詳細については、「[未使用タグの削除](#)」を参照してください。

タグ変数リストと使用情報の印刷

WindowMaker の印刷ユーティリティを使用して、**InTouch** アプリケーションのデータベース、ウィンドウ、およびスクリプトの内容を **WindowMaker** で印刷することができます。

詳細については、「[タグのクロスリファレンスリストの保存と印刷](#)」を参照してください。

タグ ドットフィールドを使用したタグ プロパティの表示または変更

InTouch タグの各タイプには、タグに関連付けられているデータまたは可能な条件を記述する独自のプロパティセットがあります。ドットフィールドはタグプロパティを識別します。タグ名ディクショナリに表示されるほぼすべてのタグプロパティにドットフィールドがあります。一部のドットフィールドは、すべての InTouch タグのタイプに共通しています。たとえば **.Name** ドットフィールドは、タグの名前に必ず関連付けられています。その他のドットフィールドは、特定のタイプのタグの一意的プロパティにのみ適用されています。

ドットフィールドをスクリプト、式、またはユーザー入力で使用する、アプリケーションの実行中にタグのプロパティを監視および変更できます。以下の例は、スクリプトまたは式でタグプロパティにアクセスするドットフィールドの構文を示します。

tag_name.property_dotfield

たとえば、アプリケーションの実行中にオペレータが **HiHi** アラームしきい値を変更できるように、アナログ - データ入力タッチリンクを作成できます。その後、このリンクを **Analog_Tag.HiHiLimit** ドットフィールドの式で定義されたボタンに適用します。ランタイム時、オペレータはボタンを簡単にクリックするだけで、タグの **HiHi** アラームしきい値に新しい値を入力できます。

ドットフィールドはタグに関連付けられたデータの入出力に使用できます。履歴ドットフィールドは実行中のアプリケーションで現在表示されている履歴トレンドへの変更可以使用できます。たとえば、ドットフィールドをスクリプトで使用すると、オペレータが履歴トレンドスクロールの変更、トレンドでのスクータのロックまたは再配置、または新しいタグへのペンの再割り当てを行うことができます。

利用可能なタグ タイプのドットフィールド

InTouch タグの各タイプには、タイプ独自のプロパティに関連付けられた一組のドットフィールドがあります。以下の表には、すべてのタイプのタグに対するドットフィールドがアルファベット順に一覧表示されています。

ドットフィールド	タグ タイプ													
	メモリ型				I/O 型				間接型			その他		
	履歴値	履歴値	履歴値	履歴値	履歴値	履歴値	履歴値	履歴値	履歴値	履歴値	履歴値	履歴値	履歴値	履歴値
.Ack	•	•	•		•	•	•		•	•		•		
.AckDev		•	•			•	•			•		•		
.AckDsc	•				•				•			•		
.AckROC	•	•	•			•	•			•		•		
.AckValue		•	•			•	•			•		•		
.Alarm	•	•	•		•	•	•		•	•		•		
.AlarmAccess													•	
.AlarmAckModel	•	•	•		•	•	•		•	•				
.AlarmClass													•	
.AlarmComment	•	•	•	•	•	•	•	•	•	•	•	•	•	•
.AlarmDate													•	
.AlarmDev		•	•			•	•			•		•		
.AlarmDevCount		•	•			•	•			•		•		
.AlarmDevDeadband		•	•			•				•				
.AlarmDevUnAckCount		•	•			•	•			•		•		

ドットフィールド	タグ タイプ														
	メモリ型				I/O 型				間接型			その他			
	論理型	整数型	実数型	メッシュ型	論理型	整数型	実数型	メッシュ型	論理型	アナログ型	メッシュ型	アラームグループ	履歴	分散アラーム	プロセスアラーム
.AlarmDisabled	•	•	•		•	•	•		•	•		•			
.AlarmDsc	•				•				•			•			
.AlarmDscCount	•				•				•			•			
.AlarmDscDisabled	•				•				•			•			
.AlarmDscEnabled	•				•				•			•			
.AlarmDscInhibitor	•				•				•			•			
.AlarmDscUnAckCount	•				•				•			•			
.AlarmEnabled	•	•	•		•	•	•		•	•		•			
.AlarmGroup														•	
.AlarmGroupSel														•	
.AlarmHiDisabled		•	•			•	•			•					
.AlarmHiEnabled		•	•			•	•			•					
.AlarmHiHiDisabled		•	•			•	•			•					
.AlarmHiHiEnabled		•	•			•	•			•					
.AlarmHiHiInhibitor		•	•			•	•			•					
.AlarmHiInhibitor		•	•			•	•			•					
.AlarmLimit														•	
.AlarmLoDisabled		•	•			•	•			•					
.AlarmLoEnabled		•	•			•	•			•					
.AlarmLoInhibitor		•	•			•	•								
.AlarmLoLoDisabled		•	•			•	•			•					
.AlarmLoLoEnabled		•	•			•	•			•					

ドットフィールド	タグ タイプ														
	メモリ型				I/O 型				間接型			その他			
	論理型	整数型	実数型	メッシュ型	論理型	整数型	実数型	メッシュ型	論理型	アナログ型	メッシュ型	アラームブザー	履歴	分散アラーム	アラームベロ
.AlarmLoLoInhibitor		•	•			•	•			•					
.AlarmMajDevDisabled		•	•			•	•			•					
.AlarmMajDevEnabled		•	•			•	•			•					
.AlarmMajDevInhibitor		•	•			•	•			•					
.AlarmMinDevDisabled		•	•			•	•			•					
.AlarmMinDevEnabled		•	•			•	•			•					
.AlarmMinDevInhibitor		•	•			•	•			•					
.AlarmName														•	
.AlarmOprName														•	
.AlarmOprNode														•	
.AlarmPri														•	
.AlarmProv														•	
.AlarmROC		•	•			•	•			•					
.AlarmROCCount		•	•			•	•			•					
.AlarmROCDisabled		•	•			•	•			•					
.AlarmROCEnabled		•	•			•	•			•					
.AlarmROCIInhibitor		•	•			•	•			•					
.AlarmROCUnAckCount		•	•			•	•			•					

[illegible]

ドットフィールド	タグ タイプ												
	メモリ型				I/O 型				間接型			その他	
	論理型	整数型	実数型	メッシュ型	論理型	整数型	実数型	メッシュ型	論理型	アナログ型	メッシュ型	アナログ型	履歴
.DevTarget		•	•			•	•			•			
.DisplayMode													•
.Enabled													•
.EngUnits		•	•			•	•			•			
.Freeze													•
.HiHiLimit		•	•			•	•			•			
.HiHiSet		•	•			•	•			•			
.HiHiStatus		•	•			•	•			•			
.HiLimit		•	•			•	•			•			
.HiSet		•	•			•	•			•			
.HiStatus		•	•			•	•			•			
.ListChanged													•
.ListCount													•
.ListIndex													•
.LoLimit		•	•			•	•			•			
.LoLoLimit		•	•			•	•			•			
.LoLoSet		•	•			•	•			•			
.LoLoStatus		•	•			•	•			•			
.LoSet		•	•			•	•			•			
.LoStatus		•	•			•	•			•			
.MajorDevPct		•	•			•	•			•			
.MajorDevSet		•	•			•	•			•			

[illegible]

[illegible]

[illegible]

ドットフィールド	タグ タイプ												
	メモリ型				I/O 型				間接型			その他	
	論理型	整数型	実数型	メジャーセグメント型	論理型	整数型	実数型	メジャーセグメント型	論理型	アナログ型	メジャーセグメント型	アラームグループ型	履歴トレンド型
.Visible													•

ドットフィールドは、目的とされる機能によって分類できます。ドットフィールドの機能分類に関する詳細については、後続のセクションを参照してください。

カテゴリ	参照先
値としきい値	タグ変数のしきい値の変更。
アラーム パラメータ	『AVEVA™ InTouch HMI アプリケーション ランタイム ガイド』の「 ランタイム時のタグとグループのアラーム プロパティの制御 」を参照してください。
I/O	「 I/O を使用したデータ アクセス 」
分散アラーム オブジェクト	分散アラーム表示オブジェクトの操作。
トレンド表示	タグデータのトレンド。
ウィンドウ コントロール	ウィザード

タグ変数のしきい値の変更

I/O サーバーからの生入力データは、InTouch アプリケーションに応じた値の範囲に変換されます。タグ変数の値は、タグ変数ディクショナリのプロパティの **生データ最小値**と**生データ最大値**で指定される最小値と最大値の範囲にクランプされます。

その後、これらの生データ値は、**[工学値最小値]** オプションと **[工学値最大値]** オプションで設定された工学値範囲に変換されます。一連のドットフィールドを使用することにより、タグ変数の生データ値と工学単位範囲を監視して、変更できます。

以下の表は、アプリケーションの実行中にタグ変数の値を監視したり、変更したりするドットフィールドを一覧表示しています。

ドットフィールド	読み取り/書き込み	内容
.MinRaw	書き込み禁止	I/O サーバーから受け取った生データの下限クランプの設定
.MaxRaw	書き込み禁止	I/O サーバーから受け取った生データの上限クランプの設定
.MinEU	書き込み禁止	タグ変数に割り当てられた工学単位での最小値
.MaxEU	書き込み禁止	タグ変数に割り当てられた工学単位での最大値
.EngUnits	読み取り/書き込み	タグ変数ディクショナリの 工学単位 オプションでアナログ型タグ変数に割り当てられたテキスト値
.RawValue	書き込み禁止	スケーリングが適用される前に I/O サーバーからタグ変数が受け取った実際の論理値かアナログ値
.Value	読み取り/書き込み	現在のタグ変数の値
.OnMsg	読み取り/書き込み	論理型タグ変数の値が True 、 On 、または 1 を評価するときに割り当てられるメッセージ
.OffMsg	読み取り/書き込み	論理型タグ変数の値が False 、 Off 、または 0 を評価するときに割り当てられるメッセージ
.Comment	読み取り/書き込み	タグ変数ディクショナリで指定されたタグ変数コメント

生データのしきい値の表示

I/O サーバーからの入力タグ変数の生データ値は、InTouch アプリケーションで使用する前にクランプする必要があります。クランプによって、定義済みの下限値と上限値の範囲に生データの値が制限されます。**.MinRaw** ドットフィールドおよび**.MaxRaw** ドットフィールドによって、生データの入力範囲の上限値および下限値が示されます。

.MinRaw ドットフィールド

.MinRaw ドットフィールドは、タグ変数に割り当てられた生データ最小値クランプ設定を示しています。**.MinRaw** ドットフィールドの値は、タグ変数ディクショナリで I/O タグ変数に対して割り当てられた**生データ最小値**です。この設定より小さい生データ値は、この最小値にクランプされます。

カテゴリ

タグ

使用法

```
tag_name.MinRaw;
```

パラメータ

Tag_name

任意の I/O 整数型、I/O 実数型、および間接アナログ型のタグ変数の名前。

備考

読み取り専用のドットフィールドで、生データ最小値の下限クランプ設定に割り当てられた値を示しています。

データタイプ

実数型または整数型（読み取り専用）

有効値

任意のアナログ値

例

以下のスクリプトは、ポンプの入力圧に関連付けられた生データ値が、このタグ変数の**生データ最小値**プロパティと**生データ最大値**プロパティで設定された上下限値の範囲外の場合にエラー ウィンドウを表示します。

```
IF ((PumpInP.RawValue > PumpInP.MaxRaw) OR  
    (PumpInP.RawValue < PumpInP.MinRaw)) THEN  
    Show "Instrument Failure Window";  
ENDIF;
```

参照項目

.EngUnits、**.MinEU**、**.MaxEU**、**.MaxRaw**、**.RawValue**

.MaxRaw ドットフィールド

.MaxRaw ドットフィールドは、タグ変数ディクショナリの**生データ最大値**プロパティによって I/O タグ変数に割り当てられた、生データ最大値の上限クランプを示します。この設定を超える生データ値は、この生データ最大値にクランプされます。

カテゴリ

タグ

使用法

```
Tag_name.MaxRaw
```

パラメータ

Tag_name

任意の I/O 整数型、I/O 実数型、および間接アナログ型のタグ変数の名前。

備考

読み取り専用のドットフィールドで、生データ最大値の上限クランプ設定に割り当てられた値を示しています。

データタイプ

実数型または整数型（読み取り専用）

有効値

任意のアナログ値

例

このスクリプトは、タグ変数の値が通常の動作範囲外であるかを判定し、範囲外と判定された場合にウィンドウを表示します。

```
IF ((Temp01.RawValue > Temp01.MaxRaw) OR (Temp01.RawValue <
    Temp01.MinRaw)) THEN
    Show "Instrument Failure Window";
ENDIF;
```

参照項目

.EngUnits、**.MinEU**、**.MaxEU**、**.MinRaw**、**.RawValue**

タグ変数の生データ値の表示

.RawValue ドットフィールドは、I/O サーバーから受け取った監視されたプロパティの実際の論理値またはアナログ値を示します。この生データ値は、クランピングやスケーリングが適用されてタグ変数の工学単位に値が正規化される前の実際の入力値です。

.RawValue ドットフィールド

.RawValue ドットフィールドは、WindowViewer が I/O サーバーから受け取った実際の値を示しています。**.RawValue** ドットフィールドによって、InTouch がスケーリングを適用する前の I/O タグ変数の値にアクセスできます。

カテゴリ

タグ

使用法

`Tag_name.RawValue`

パラメータ

Tag_name

任意の I/O 整数型、I/O 実数型、I/O 論理型、間接論理型、および間接アナログ型のタグ変数の名前。

備考

この読み取り専用ドットフィールドは、InTouch がスケーリングを適用する前の実際の論理値またはアナログ I/O 値を表示するために使用されます。

データタイプ

.RawValue ドットフィールドに関連付けられているタグ変数のタイプに適したすべてのデータ。たとえば、実数型タグ変数には実数、論理型タグ変数には論理値など（読み取り専用）。

例

以下のスクリプトは、ポンプの入力圧の生データ値がタグ変数の最小値と最大値によるクランプ制限外になった場合に警告メッセージを発行します。

```
IF ((PumpInP.RawValue > PumpInP.MaxRaw) OR (PumpInP.RawValue < PumpInP.MinRaw)) THEN
    AlarmMessage = "Pump sensor is out of calibration or requires replacement.";
ENDIF;
```

参照項目

.EngUnits、**.MinEU**、**.MaxEU**、**.MinRaw**、**.MaxRaw**

工学値のしきい値の表示

I/O サーバーから WindowViewer で受け取った値は最初は生データと見なされます。生データ値にはスケールが必要な場合があります。InTouch HMI は、クランプされた生の入力データ値に対して演算による変換を行い、タグ変数の工学値範囲にスケールします。I/O 整数型と実数型のタグ変数には、工学値範囲の上下限値を示す**工学値最小値**プロパティと**工学値最大値**プロパティが用意されています。

.MaxEU ドットフィールド

.MaxEU ドットフィールドは、タグ変数ディクショナリで特定のタグ変数に対して割り当てられた工学単位の最大値を示しています。

カテゴリ

タグ

使用法

`Tag_name.MaxEU`

パラメータ

Tag_name

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

.MaxEU ドットフィールドは、タグ変数に対して定義された工学単位の範囲に生データ値をスケールするために使用されます。このドットフィールドにより、工学値範囲の上限値が定義されます。

データタイプ

実数型タグ変数には実数型、整数型タグ変数には整数型（読み取り専用）

有効値

指定したタグ変数のタイプにより異なります。

例

フィールドのプログラマブル ロジック コントローラ (PLC) によって、レベル ゲージが読み取られます。レベル送信機は、4～20 mA の範囲の信号を送信します。PLC はこの信号を 0 ～ 4095 までの整数値に変換します。この値が **TankTwoLevel** タグ変数に割り当てられます。

生データ値 (0 ～ 4095) を表示しても、オペレータに対して役立つデータは提供されません。したがって、この値を対応する工学単位の範囲にスケールする必要があります。

これを達成するには、最小工学単位値フィールドと最大工学単位値フィールドを正確に設定する必要があります。この例では、生データ値 0（フィールドからは 4 mA）を「0 ガロン」に、値 4095（フィールドからは 20 mA）を「100 ガロン」に変換する場合、画面に正確な値を表示するには以下の設定が必要です。

```
TankTwoLevel.MinRaw = 0;  
TankTwoLevel.MaxRaw = 4095;  
TankTwoLevel.MinEU = 0;  
TankTwoLevel.MaxEU = 100;
```

この設定では、フィールドの生データが 4095 の場合に、オペレータに対して表示される値は 100 となります。

参照項目

.EngUnits、**.MinEU**、**.MinRaw**、**.MaxRaw**、**.RawValue**

.MinEU ドットフィールド

.MinEU ドットフィールドは、タグ変数ディクショナリから特定のタグ変数に対して割り当てられた工学単位の最小値を示しています。

カテゴリ

タグ

使用法

`Tag_name.MinEU`

パラメータ

Tag_name

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

.MinEU ドットフィールドは、タグ変数に対して定義された工学単位の範囲に生データ値をスケールするために使用されます。このドットフィールドにより、工学値範囲の下限值が定義されます。

データタイプ

実数型タグ変数には実数型、整数型タグ変数には整数型（読み取り専用）

有効値

指定したタグ変数のタイプにより異なります。

例

この例では、**Tag1** タグ変数に対して定義された工学単位の範囲が、**AbsoluteTagRange** タグ変数に割り当てられます。

```
AbsoluteTagRange = (Tag1.MaxEU - Tag1.MinEU);
```

参照項目

.EngUnits、**.MaxEU**、**.MinRaw**、**.MaxRaw**、**.RawValue**

タグ変数の工学単位の変更

タグ変数にドットフィールドを関連付けて、タグ変数に割り当てられた工学単位のテキスト値を決定できます。

.EngUnits ドットフィールド

.EngUnits ドットフィールドは、**工学単位** プロパティでアナログ型タグ変数に割り当てられたテキスト値を示しています。**.EngUnits** ドットフィールドは、工学単位をテキスト値として示しています。

注: このフィールドに書き込まれた値は保持されません。

カテゴリ

タグ

使用法

`Tag_name.EngUnits`

パラメータ

Tag_name

任意の整数型、実数型、または間接アナログ型のタグ変数。

データ タイプ

メッセージ型（読み取り/書き込み）。

備考

.EngUnits ドットフィールドは、タグ変数に関連付けられた実際のデータのスケール、変換、または形式には影響を与えません。

有効値

0 ～ 31 文字を含む任意の文字列

例

以下のスクリプトは、タグ変数の工学単位が摂氏の場合に、華氏温度への変換関数を呼び出します。

```
IF Temperature.EngUnits == "Celsius" THEN  
    CALL TempFConvert(Temperature);  
ENDIF;
```

参照項目

.MinEU、**.MaxEU**、**.MinRaw**、**.MaxRaw**、**.RawValue**

工学単位でのタグ変数値の表示

アプリケーション内で明示的にドットフィールドが指定されていない場合、InTouch タグ変数にはデフォルトのドットフィールドが割り当てられます。

.Value ドットフィールド

.Value ドットフィールドは、指定したタグ変数の現在の値を工学単位で示しています。**.Value** ドットフィールドは、すべてのタグ変数に自動的に適用されるデフォルトの InTouch ドットフィールドです。タグ変数にドットフィールドの指定がない場合は、**.Value** ドットフィールドが使用されます。

カテゴリ

タグ

使用法

`tag_name.Value`

パラメータ

tag_name

履歴トレンド以外の任意のタイプのタグ変数

備考

.Value ドットフィールドの使用が必要になることはほとんどありません。ただし、場合によっては、このドットフィールドによって計算やパラメータの使用法が明確になることがあります。

データ タイプ

指定したタグ変数のタイプと同じタイプ（読み取り／書き込み）

有効値

指定したタグ変数のタイプにより異なります。

例

次のステートメントにより、メモリ整数型タグ変数 **PumpRPM** の値が 100 に設定されます。

```
PumpRPM.Value=100;
```

以下のように指定しても機能は同じです。

```
PumpRPM=100;
```

論理型タグ変数のメッセージの表示または変更

.OnMsg ドットフィールドと **.OffMsg** ドットフィールドは、論理型タグ変数のオンとオフの状態に対してタグ変数ディクショナリで指定したメッセージを示します。論理型タグ変数のオンとオフのメッセージは、15 文字以内の短い文字列です。

.OnMsg ドットフィールド

.OnMsg ドットフィールドを使用すると、タグ変数ディクショナリで論理型タグ変数に割り当てられたオンメッセージにアクセスできます。

カテゴリ

タグ

使用法

```
Tag_name.OnMsg
```

パラメータ

Tag_name

任意の論理型タグ変数

データタイプ

メッセージ型（読み取り／書き込み）。このドットフィールドに書き込まれた値は保持されません。

有効値

0 ～ 15 文字を含む任意の文字列

例

以下のステートメントは、間接型タグ変数 **IndPumpState** のオンメッセージに「Pump1 running」という文字列が指定された場合にメッセージを発行します。

```
IF IndPumpState.OnMsg == "Pump1 running" THEN  
    TypeOfTag = "The IndPumpState tag is assigned to Pump1.";  
ENDIF;
```

参照項目

.OffMsg

.OffMsg ドットフィールド

.OffMsg ドットフィールドを使用すると、タグ変数ディクショナリで論理型タグ変数に割り当てられたオフメッセージにアクセスできます。

カテゴリ

タグ

使用法

```
Tag_name.OffMsg
```

パラメータ

Tag_name

任意の論理型タグ変数

データタイプ

メッセージ型（読み取り／書き込み）。このドットフィールドに書き込まれた値は保持されません。

有効値

0 ～ 15 文字を含む任意の文字列

例

以下のステートメントでは、**MyDiscrete** タグ変数の状態に基づいて、**StateMessage** タグ変数に対して適切な文字列が割り当てられます。

```
StateMessage=Dtext (MyDiscrete, MyDiscrete.OnMsg, MyDiscrete.OffMsg);
```

参照項目

.OnMsg

タグ変数のコメントの表示または変更

タグ変数コメントは、タグ変数ディクショナリでのみ永久的に変更できます。アプリケーションの実行中には、**.Comment** ドットフィールドを使用して別のコメントを割り当てることができます。このコメント変更が有効なのは、ランタイムセッション中だけです。タグ変数ディクショナリのタグ変数のコメントを永久的に変更することはできません。**WindowViewer** がシャットダウンされて再起動されると、元のコメントがタグ変数に割り当てられます。

.Comment ドットフィールド

.Comment ドットフィールドは、タグ変数ディクショナリでタグ変数に割り当てられたコメントを示しています。タグ コメントは、160 文字以内の文字列です。

カテゴリ

タグ

使用法

```
Tag_name.Comment
```

パラメータ

Tag_name

任意のタグ変数

備考

タグ変数に関連付けられたコメントは、InTouch アプリケーションの実行中に .Comment ドットフィールドを使用して変更できます。アプリケーションが停止されると、タグ変数ディクショナリで指定された元のコメントがタグ変数に割り当てられたまま残ります。

データ タイプ

メッセージ型

有効値

1 ～ 160 の文字を含む任意の文字列

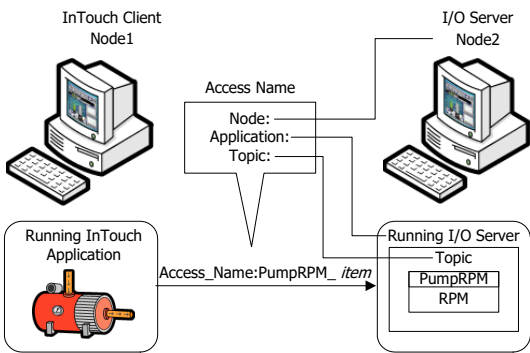
例

以下のステートメントでは、タグ変数に割り当てられたコメントをタグ変数の名前に組み合わせることによって、オペレータ メッセージが作成されます。

```
OperatorMessage=PumpRPM.Name + " has a comment of: " + PumpRPM.Comment;
```

I/O を使用したデータ アクセス

分散アプリケーションを開発して、InTouch システムの複数の機能コンポーネントを別のノード上に配置することができます。次の図に、別のノードに保存されているデータへの I/O リクエストの設定方法を示します。



3 部構成のアドレス指定規則を使用して、別のノードに保存されているデータの要素を識別するように InTouch アプリケーションを設定できます。この形式には、ノード名、アプリケーション名、トピック名が含まれています。リモート ノードからデータを取得するには、これらの 3 アイテムを指定した InTouch アプリケーションに対してアクセス名を設定する必要があります。

たとえば、別のノードで稼動しているリモート I/O サーバーのデータにアクセスする場合、アクセス名は以下の要素で構成されます。

アクセス名オプション	説明
ノード名	I/O サーバー プログラムを実行しているコンピュータのノード名です。

アクセス名オプション	説明
アプリケーション名	<p>ノードで実行されている I/O サーバー プログラム名です。</p> <p>たとえば、"UA_SampleServer" は OPC UA サーバーの名前である可能性があります。</p> <p>Operations Integration (OI) サーバーに関連付けられたアプリケーション名の詳細については、OI サーバーのマニュアルを参照してください。</p>
トピック名	I/O サーバー デバイス グループに割り当てられるラベルです。

InTouch データ ソースに Excel のスプレッドシートを使用する場合は、アクセス名を以下のように定義できます。

アクセス名オプション	説明
ノード名	Excel プログラムを実行しているコンピュータのノード名です。
アプリケーション名	アプリケーション名に Excel を指定します。
トピック名	要求データが格納されている Excel のブック名とスプレッドシート名です。たとえば、[Book1]Sheet1 と指定します。

ノード、アプリケーション、トピック、アイテムに加えて、リモート ノードにあるデータのタイプを指定する必要があります。この情報に基づいて、タグ名ディクショナリで定義されるタグの I/O タイプが決定されます。

Gateway Communications Driver の操作

Gateway Communications Driver は、ポータルおよびプロトコル コンバータとして機能し、2 つのコンピュータ システムまたはプログラムが相互に通信できるようにするアプリケーションです。OI Gateway を使用すれば、OPC などの異なるプロトコルを使用して通信するクライアントとデータ ソースをリンクさせることができます。InTouch から Gateway Communication Driver にアクセスしてフィールド デバイスを接続します。

InTouch は Gateway Communication Driver を活用して、オートメーションと制御アプリケーションの間、フィールド システムとデバイスの間、およびビジネスとオフィス アプリケーションの間の通信に OPC、OPC UA、およびその他のプロトコルを使用します。

Gateway Communication Driver は WindowViewer にバンドルされており、InTouch をインストールする際に自動的にインストールされます。スタンドアロン アプリケーションとしてインストールすることもできます。

Gateway Communication Driver について

Gateway Communication Driver は通信プロトコル コンバータとして機能します。Gateway Communication Driver を使用して、複数の異なるプロトコルを使用して通信するクライアントとデータ ソースをリンクさせることができます。

このユーザー支援マニュアルでは、Gateway Communication Driver コンポーネントの設定と実行に必要な情報のみが記載されています。その操作については、関連コンポーネントに付属するマニュアルを参照してください。

Operations Control Management Console (OCMC) へのスナップインである Log Viewer を使用して、Gateway Communication Driver の問題のトラブルシューティングを行うことができます。以下の詳細については、Log Viewer ヘルプを参照してください。

- エラー メッセージの表示
- 表示するメッセージの決定
- エラー メッセージのブックマーク設定

InTouch HMI ソフトウェアなどのクライアント アプリケーションを使用して問題のトラブルシューティングを行うこともできます。クライアント アプリケーションは、システム デバイス アイテムを使用して、ノードのステータスおよび一部のパラメータの値を特定できます。

Gateway Communication Driver の基本ルールを以下に示します。

- ノードあたり Gateway Communication Driver の 1 つのインスタンスを実行できます。
- Gateway は、OI サーバー マネージャ スナップインを使用してアクティブおよび非アクティブにすることができます。
- Gateway は、標準の COM アクティブ化メカニズムを使用して、COM サーバー（OPC サーバー）としてアクティブ化することができます。
- Gateway は OPC クライアント内で out-of-proc としてのみ実行できます。
- Gateway が通信できるのは、Application Server v2.0 以降で提供される Archestra データ ソース コンポーネントだけです。以前のバージョンの Application Server はサポートされません。

開始するためのワークフロー

以下の手順は、Gateway Communication Driver の使用を開始するためのワークフローを説明します。

1. InTouch をインストールします。
 - Gateway Communication Driver は InTouch インストールの一部として含まれています。
 - OPC サーバーと OPC UA サーバーは別にインストールする必要があります。
2. OPC サーバーまたは OPU UA サーバーをインストールし、フィールドデバイスとの通信を設定します。
3. 必要に応じて、その他のプロトコルを設定します。
4. InTouch でアプリケーションを作成します。
5. Gateway Communication Driver を設定して、インストール済みの OPC サーバー、OPC UA サーバー、またはその他のデータ ソースをポイントします。
6. Gateway Communications Driver をアクティブ化します。

Gateway Communication Driver のセットアップの詳細については、Gateway Communications Driver ヘルプの「初めての Gateway Communication Driver の設定」を参照してください。

サポートされている InTouch 通信プロトコル

InTouch HMI を設定することにより、DDE か SuiteLink を使用することができます。InTouch HMI では、メッセージ交換通信プロトコルもサポートされています。

InTouch アプリケーション内での Message Exchange の使用の詳細については、「[InTouch から Application Server データへのアクセス](#)」を参照してください。

動的データ交換

動的データ交換（DDE）通信プロトコルによって、Windows アプリケーション間での通信が可能になります。DDE は、同時に実行されている 2 つのアプリケーション間に、クライアント/サーバー関係を構築します。サーバー アプリケーションはデータを提供し、このデータを必要とする他のアプリケーションからの要求を受け付けます。要求元のアプリケーションは、クライアントと呼ばれます。InTouch アプリケーションは、クライアントにもサーバーにも同時になることができます。

SuiteLink

SuiteLink は、産業用アプリケーション向けに特に設計された TCP/IP ベースのプロトコルです。SuiteLink は、データの整合性、高スループット、およびシンプルな診断手順を提供します。SuiteLink プロトコルをサポートしているのは、Microsoft Windows NT 4.0 以降です。

SuiteLink は DDE または NetDDE に取って代わるものではありません。クライアントとサーバー間の接続はそれぞれ、ご使用のネットワーク要件によって異なります。

SuiteLink では、以下のことが行えます。

- VTQ（Value Time Quality）は、VTQ 対応のクライアントに届けられるデータ値のすべてに時間の目盛りと品質表示を挿入します。
- データのスループット、サーバーのロード、コンピュータ リソースの消費、およびネットワークの転送に対する包括的な診断を、Microsoft Windows オペレーティング システムのパフォーマンス モニタから行うことができます。
- アプリケーションが 1 つのノード上にある場合や、数多くのノードに分散されている場合にかかわらず、アプリケーション間で一貫した大容量のデータを維持できます。
- ネットワーク転送プロトコルは、Microsoft の標準 Winsock インターフェイスを使った TCP/IP です。
- TLS 1.2 プロトコルを使用して、Suitelink ノードを安全に設定できます。

標準ユーザーとしてのセキュアな Suitelink 通信の設定

TLS 1.2 プロトコルによるセキュアな Suitelink 接続では、クライアントとサーバーの間の通信チャネルが暗号化されます。リモート データにアクセスするために暗号化された SuiteLink を使用する場合、コンフィグレータから System Management Server（SMS）を設定する必要があります。暗号化は、SMS によって提供される証明書を介して行われます。SuiteLink サーバーとクライアントで暗号化された通信を使用するには、SuiteLink 3.0 をインストールする際に ASB ランタイム コンポーネント機能を選択する必要があります。

セキュアな SuiteLink（V3）の導入以降、SuiteLink はフォールバックを提供しているので、SuiteLink サーバーは、セキュア（V3、暗号化）または非セキュア（V2、非暗号化）モードの受信接続を受け入れることができます。System Platform 2023 リリースから、V2 接続を受け入れるフォールバック モードはデフォルトで無効化されています。フォールバック モードを有効にして非セキュアな接続を受け入れる場合、System Management Server で設定を変更する必要があります。SuiteLink 接続のモード（セキュアまたは非セキュア）は、Log Viewer で WWSLS コンポーネントをチェックすることによって確認できます。

注記: System Platform 2023 の時点で、SuiteLink 接続はデフォルトで TLS 暗号化によって保護されます。非セキュア フォールバック モードが有効化されておらず、セキュアな接続の要件が満たされていない場合、SuiteLink 接続は失敗します。詳細については、「[コンフィグレータでの System Management Server の設定](#)」を参照してください。

レジストリ エディタを使用してフェールバック（非セキュア）SuiteLink 接続を有効化または無効化するには

1. レジストリ エディタを開きます。
2. 次のパスに移動します。
[HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\ArchestrA\WebApplications\Default\SuiteLink]
3. 次のエントリを見つけます。
"V2Server"=dword:00000001
"V3Server"=dword:00000001
4. フォールバック モードを有効にするには、V2Server の値を 1 に設定します。
5. フォールバック モードを無効にするには、V2Server の値を 0 に設定します。
6. コンピュータを再起動します。

コンフィグレータでの System Management Server の設定

System Management Server (SMS) は、System Platform 共通プラットフォーム サービスの一部で、System Platform 2023 の重要なセキュリティ対策を実装するために使用されます。以下の内容が含まれます。

- ノード間通信のポート番号の設定。
- SuiteLink セキュリティ モードの設定: 暗号化された（セキュアな）通信だけを使用するか、セキュア（TLS）通信を確立できない場合に暗号化されていない通信を許可するよう SuiteLink 接続を介した通信を設定することができます。SuiteLink は、System Platform のさまざまな多くのアプリケーションで使用されます。SuiteLink セキュリティの設定の詳細については、[詳細設定] ボタンを選択して [通信] タブを参照してください。
- 証明書の管理。
- 外部 ID プロバイダを介したシングル サインオン（SSO）が許可される OpenID Connect 標準によるユーザー認証。

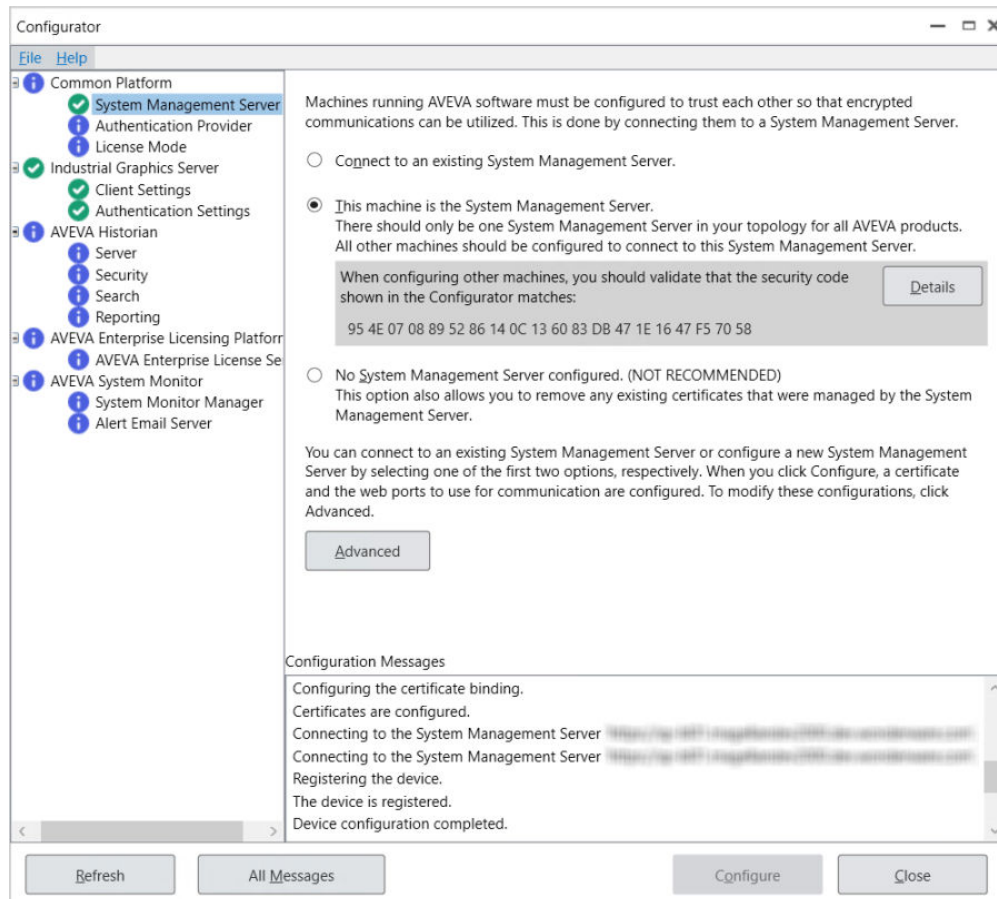
セキュリティを有効にするには、各 System Platform ノードが System Management Server と通信する必要があります。System Platform トポロジ内の System Management Server は 1 つだけである必要があります。そうでない場合、通信途絶が発生する可能性があります。System Management Server は、共有セキュリティ証明書を格納し、複数のマシン間で信頼関係を確立します。1 つの追加ノードを冗長 SSO サーバーとして設定できます。この冗長 SSO サーバーは、System Management Server にアクセスできない場合のシングルサインオンのバックアップとして機能します。

System Platform 2017 Update 3 以降にアップグレードされていないノードがある場合、それらのノードとの通信には非セキュアな通信を使用する必要があることがあります。ただし、System Management Server と通信するよう設定されている場合、System Platform 2017 Update 3 以降を実行するノード間の通信は暗号化されます。

System Management Server を設定するには

1. コンフィグレータで [共通プラットフォーム] を展開して [System Management Server] を選択します。

注記: System Management Server のユーザー資格情報が要求された場合、**DomainName\UserName** の形式でユーザー名を入力します。ドメイン管理特権があってもローカルマシンの管理者でない場合は、ユーザー資格情報が要求されることがあります。System Management Server を設定するには、**Administrators** または **aaAdministrators** OS グループのメンバーである必要があります。詳細については、AVEVA System Platform ヘルプの「System Management Server を設定するためのユーザー資格情報」を参照してください。



注記: インストールが完了するとコンフィグレータが自動的に起動します。コンフィグレータは、任意の System Platform ノードの Windows [スタート] メニューから起動することもできます。

2. 以下の 3 つのオプションが提示されます。

- **既存の System Management Server に接続:** これはデフォルト オプションです。System Platform 探索サービスによってネットワーク上の既存の System Management Server が検索されます。検出された System Management Server はドロップダウンリストに表示されます。使用するサーバーを選択するか、サーバーのマシン名を入力します。System Platform トポロジのすべてのコンピュータは同じサーバーに接続する必要があります。
- **このマシンを System Management Server にする:** このコンピュータを System Management Server にする場合、このオプションを選択します。System Platform トポロジ内の他のすべてのコンピュータで「既存の System Management Server に接続」オプションを使用して、このサーバーに接続するように設定する必要があります。

- **System Management Server** を設定しない（推奨されません）：暗号化とセキュア通信なしでコンピュータをセットアップする場合、このオプションを選択します。トポロジ内の他のコンピュータは **System Management Server** を使用するように設定できます。

3. [詳細] ボタンをクリックします。

[詳細設定] ウィンドウが開きます。

Advanced Configuration

Certificates Ports Communications

In order to enable communications via encrypted channels (e.g. HTTPS), certificates are required to be configured.

Certificates can either be provided by your IT department or automatically generated.

Configuration

Please select the appropriate options below.

Certificate Source: Automatically Generated

Certificate: ASB Details

System Management Server: Port: 443

OK Cancel

a. [証明書] タブで証明書パラメータを設定します。

- **証明書ソース**: [自動生成]（デフォルト）または [IT から提供] を選択します。IT 部門が証明書を提供している場合、[インポート] ボタンをクリックして証明書ファイルを参照します。詳細については、『System Platform インストール ガイド』の「証明書のインポート」を参照してください。
- **証明書**: 証明書名が表示されます。[詳細] をクリックすると、インポート済みの証明書が表示されます。証明書は自動更新プロセスを介してサーバー ノードとリモート ノードの両方で定期的に更新されます。
- **System Management Server**: 既存の System Management Server に接続する場合、選択したサーバーの名前とポート番号が表示されます。
- **共通プラットフォームのポート**: 共通プラットフォームのポートは、特定の AVEVA ソフトウェア（Sentinel System Monitor など）との通信に使用されます。一般的に、デフォルト設定を使用できます。リモート ノードは、以下に示すポート番号で設定する必要があります。
デフォルト HTTP ポート: 80
デフォルト HTTPS ポート: 443

詳細については、『AVEVA System Platform インストールガイド』の「詳細設定オプション」セクションを参照してください。

b. [通信] タブで通信パラメータを設定します。

- [暗号化されていない SuiteLink 接続を受け入れる (混合モード)] チェック ボックスをオンにして [OK] をクリックします。混合モードが有効になり、暗号化された接続と暗号化されていない接続の両方が受け入れられるようになります。

The screenshot shows the 'Advanced Configuration' dialog box with the 'Communications' tab selected. The dialog has three tabs: 'Certificates', 'Ports', and 'Communications'. The 'Communications' tab contains instructions for configuring AVEVA communications protocols. It includes a section for 'SuiteLink' with a description, a checked checkbox for 'Accept non-encrypted SuiteLink connections (mixed mode)', and a red warning note. Below this is a section for 'Network Message Exchange (NMX)' with a description and an unchecked checkbox for 'Grant access to NMX for all users (NOT RECOMMENDED)', followed by a grey note. At the bottom right are 'OK' and 'Cancel' buttons.

Advanced Configuration ×

Certificates Ports Communications

Use this tab to configure the behavior of AVEVA communications protocols.

Many AVEVA and 3rd Party products that integrate with System Platform use these protocols. For example: InTouch HMI, Historian, OI Servers (CDP), Batch Management, Workflow, and others. Refer to your product's documentation or contact technical support for more information.

SuiteLink

SuiteLink is a TCP/IP based communications protocol.

SuiteLink communications between processes on this node, and between processes on this node and other nodes can be encrypted. Please select the appropriate handling for non-encrypted SuiteLink connection requests.

☒ Accept non-encrypted SuiteLink connections (mixed mode).
Mixed mode is recommended for use only during online (node-by-node) upgrades and/or supporting legacy applications.

Network Message Exchange (NMX)

NMX is an AVEVA application communication protocol that uses a DCOM-based communication transport mechanism. Authorization to access NMX can be restricted to users that are members of a well-known OS User Group. Please select the appropriate handling for NMX access authorization on this node.

☐ Grant access to NMX for all users (NOT RECOMMENDED)
NOTE: Changes to this setting require a reboot in order to take effect.

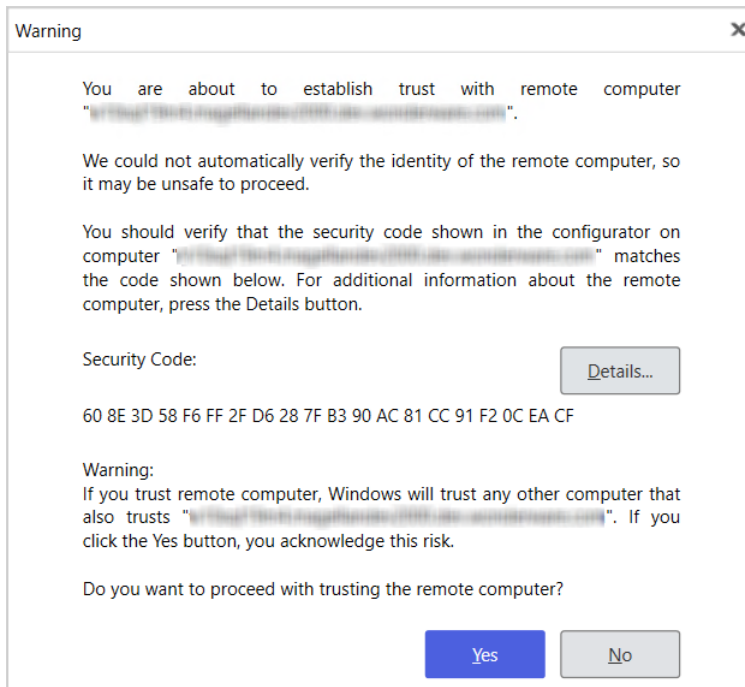
OK Cancel

注記: 暗号化された接続 (V3) のみを使用する場合は、[暗号化されていない SuiteLink 接続を受け入れる (混合モード)] チェック ボックスをオフにして [OK] をクリックします。

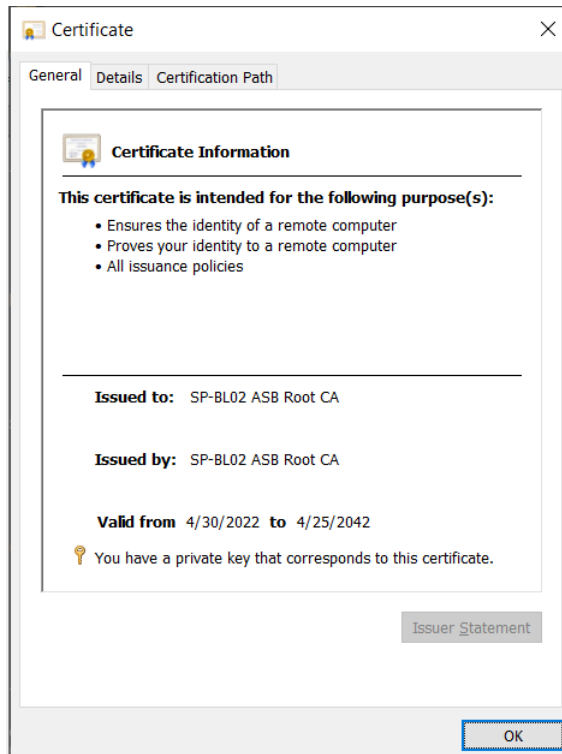
4. [OK] をクリックして変更を保存し、SMS 設定ウィンドウに戻ります。

5. [設定] をクリックします。

[セキュリティ警告] ウィンドウが表示されます。



マシン間の信頼を確立することによって通信は自由に通過できますが、リモートコンピュータの ID が不明な場合はセキュリティの懸念が生じます。接続先のコンピュータに懸念がある場合、[詳細設定] ダイアログの【詳細...】ボタンを選択して証明書を開き、セキュリティコードと証明書の詳細を確認します。



6. 設定が必要な次の項目を左ペインで選択します。必要なすべてのアイテムを設定したら **「閉じる」** ボタンをクリックしてインストールを完了します。『Application Server ユーザー ガイド』の「設定後のシステムの再起動」を参照してください。

System Management Server 設定の詳細については、『System Platform インストール ガイド』の「System Management Server の設定」セクションを参照してください。

標準ユーザーとしてのサーバーへのアクセス

標準ユーザーとしてサーバーにアクセスする場合、セキュアな SuiteLink チャンネルを確立することはできません。暗号化されたセキュアな通信ワークフローを確立するために、次のいずれかを行ってください。

- **標準ユーザーをユーザー グループに追加する:** 標準ユーザーは、サーバー側の 'ArchestraWebHosting' ユーザー グループに追加する必要があります。ユーザー グループへのユーザーの追加の詳細については、Windows 固有のマニュアルを参照してください。
- **WindowViewer をサービスとして実行する:** タグ サーバー アーキテクチャ内の InTouch タブ ベースのアプリケーションの場合、タグ サーバー ノードで InTouch WindowViewer をサービスとして実行することによって暗号化されたセキュアな Suitelink 通信を確立できます。
- **管理者特権で WindowViewer を対話型モードで実行する:** こうすると、SuiteLink サーバーとクライアント間の通信を暗号化するプライベートキーにアクセスするために必要なユーザー特権を持つ SuiteLink サーバーとして動作する非対話型ユーザーの下で WindowViewer を実行できます。この設定は、対話型ユーザーを使用した場合には使用できません。WindowViewer が実行されているユーザーは、SuiteLink サーバーがセキュアな通信を行うために必要なプライベート キーを取得するために使用されるセキュリティ コンテキストを提供するので、これは必要です。WindowViewer が対話型アプリケーションとして実行されている場合、(前に説明した通り) ユーザーは、暗号化された SuiteLink 通信をセキュリティを強化するプライベート キーにアクセスできない対話型ユーザーです。

InTouch をサービスとしてのタグ サーバーとして実行する場合、クライアント ノードの InTouch Tag Server クライアント ライセンスと組み合わせて、InTouch の通信をタグ サーバーに制限することができます。このシナリオを使用する場合、多くのユーザーは HyperV や VMWare などの PC 仮想化テクノロジーを使用して、その仮想マシン (VM) 内のサービスとして WindowViewer を実行し、ソリューションの総保有コストを削減しています。

標準ユーザーとしての WindowViewer の実行

以下のシナリオでは、WindowViewer は管理特権のない標準ユーザーとして実行していることになります。

- 管理特権のない標準ユーザーとして System Platform IDE を起動し、WindowMaker で \$InTouchViewApp を編集した後に WindowViewer への高速切り替えを行う。
- AVEVA アプリケーション マネージャから WindowViewer を起動するか、管理特権のない標準ユーザーとして WindowViewer を起動する。

SuiteLink の通信障害のトラブルシューティング

SuiteLink の通信障害が発生した場合は、以下の操作を行います。

- InTouch HMI がインストールされているコンピュータ上で、Microsoft TCP/IP が動作していることを確認します。
- コンピュータのノード名が 15 文字以内であることを確認します。

- SuiteLink が、InTouch HMI がインストールされているコンピュータのサービスとして実行していることを確認します。
- SuiteLink は、InTouch のインストール中に自動的にインストールされます。また、SuiteLink のサービスは自動的に起動されます。SuiteLink のサービスが停止した場合は、再起動する必要があります。

標準ユーザーとしてサーバーにアクセスする際の Suitelink 通信の問題

以下の表には、さまざまなバージョンの System Platform での非管理者ユーザー（標準ユーザー）の Suitelink 接続の動作、および該当するトラブルシューティング手順が示されています。

シナリオ	System Platform 2020 R2 SP1 まで	System Platform 2023	トラブルシューティング
SMS が設定されていない状態で WindowViewer を実行	ロガー メッセージなし	ロガー メッセージなし	該当なし
SMS が設定されていない状態でクライアントを WindowViewer に接続	WindowViewer への非セキュアな接続が確立されています。ロガー内に警告メッセージがあります。*	接続に失敗します。 ロガー内に警告メッセージがあります。*	<ul style="list-style-type: none"> • コンフィグレータで非セキュアな接続を許可するオプションを選択します。 <p>非セキュアな接続が確立され、ロガーに警告メッセージが記録されます。</p> <ul style="list-style-type: none"> • セキュアな Suitelink 接続を設定します。
SMS が設定された状態で WindowViewer を実行	接続に失敗します。 ロガー内にエラーメッセージがあります。*	接続に失敗します。 ロガー内にエラーメッセージがあります。*	<ul style="list-style-type: none"> • WindowViewer サービスとして実行 • 管理者特権で WindowViewer を対話型モードで実行するか、 • WindowViewer ユーザーを "ArchestrAWebHosting" グループに追加します。
SMS が設定された状態でクライアントを WindowViewer に接続	WindowViewer への非セキュアな接続が確立されています。ロガー内に警告メッセージがあります。*	接続に失敗します。 ロガー内に警告メッセージがあります。*	<ul style="list-style-type: none"> • コンフィグレータで非セキュアな接続を許可するオプションを選択します。 <p>非セキュアな接続が確立され、ロガーに警告メッセージが記録されます。</p> <ul style="list-style-type: none"> • セキュアな Suitelink 接続を設定します。

SMS とセキュアな Suitelink 接続 (V3) が設定されておらず、フォールバックとしての非セキュアな Suitelink 接続 (V2) が無効化	<ul style="list-style-type: none"> • Suitelink クライアントが WindowViewer に接続します。 • Web Client が InTouch タグにバインドします。 • ロガー内に警告メッセージがあります。* 	<ul style="list-style-type: none"> • Suitelink クライアントが WindowViewer に接続しない。 • Web Client が InTouch タグにバインドしない。 <p>WindowViewer の起動時にエラーメッセージがロガーに記録される。 **</p>	<ul style="list-style-type: none"> • コンフィグレータで SMS を設定します。 • セキュアな Suitelink 接続を設定します。 • フォールバックとして非セキュアな接続を許可するオプションを選択します。
---	---	--	---

警告メッセージ* - 警告 WWSLS 暗号化されていない通信チャネルが 127.0.0.1(VIEW) から確立されました。System Management Server を両方のノードにインストールして、クライアントとサーバー製品の両方を最新のバージョンにアップグレードして、このチャネルを保護してください。

エラー メッセージ** - エラー WWSLS [複数行メッセージ] - この設定では暗号化されたインバウンド SuiteLink 接続を確立できません。

OPC

OPC 自体は通信プロトコルではありませんが、ドライバとして機能します。これは専有されていない標準インターフェイスのセットで、Microsoft の OLE/COM テクノロジーに基づいています。この標準インターフェイスにより、オートメーション/制御アプリケーション、フィールドシステム/デバイス、ビジネス/オフィス アプリケーション間の相互運用性が可能になります。

ソフトウェアアプリケーション開発者にとって従来の要件だったフィールドデバイスとのデータ通信用のカスタム ドライバを作成する必要性を省くため、OPC は汎用的な高性能インターフェイスを定義することで、簡単に HMI、SCADA、制御、およびカスタム アプリケーションを簡単に再利用できるようにしています。

ネットワーク上では、OPC は DCOM（分散 COM）を使用してリモート通信を行います。

OPC UA

OPC Unified Architecture (OPC UA) は、相互運用性を目的としたマシン間の通信プロトコルです。OPC UA は、強化されたセキュリティに加え、高度な通信、セキュリティ、および情報モデルを備えたプロセス制御、そして情報モデル、およびクロスプラットフォームの接続性を提供します。

OPC UA は、OI Gateway のクライアントとして実装されています。

OPC UA は、OPC とは大きく異なります。ここでは、クラシック OPC と OPC UA の主な違いについて説明します。

クラシック OPC	OPC UA
通信に Microsoft の COM/DCOM 技術を使用します。設定可能なタイムアウトがありません。OPC UA はシステムで設定される DCOM タイムアウトに依存します。	データのエクスポートにサービス アーキテクチャを使用し、通信と接続を容易にします。

Windows オペレーティング システムに依存します。	プラットフォームに依存し、広範なデバイスとプラットフォームに接続します。
セキュリティが制限されています。	セキュリティが組み込まれています。
メッセージの損失などの問題を処理する機能は組み込まれていません。	メッセージの損失などの問題を処理する機能が組み込まれています。

MQTT

MQTT (Message Queuing Telemetry Transport) は、TCP/IP を介して使用する Publish/Subscribe メッセージング プロトコルです。MQTT は、必要な電力と帯域幅を最小限にしながら、デバイスが相互に通信できるようにします。MQTT は、低速または不安定なネットワークに依存するデバイスでの使用に適したシンプルなメッセージング プロトコルです。

MQTT プロトコルはアプリケーション レイヤー仕様で、標準の ISO/IEC PRF 20922 として公開されています。MQTT は、仲介ブローカーを必要とする Publish/Subscribe メカニズムです。パブリッシャーがブローカーにデータを送信し、サブスクライブ クライアントはブローカーにパブリッシュされたデータを受信します。特定のトピックをサブスクライブ (購読) するクライアントだけが、そのトピックに関するメッセージを受信します。プロトコルは、双方向通信をサポートするので、パブリッシャーであるデバイスが更新を受信することも可能です。

アクセス名の設定

InTouch I/O タグまたはリモート タグ 参照には、アクセス名を関連付ける必要があります。アクセス名によって、別の I/O データ ソースとの通信リンクが定義されます。各アクセス名には、ノード名、アプリケーション名、およびトピックで構成される I/O アドレスが指定されます。

分散アプリケーションでは、I/O 参照をネットワーク I/O サーバーのグローバル アドレスとして、またはローカル I/O サーバーのローカルアドレスとして設定できます。

InTouchView には、Application Server 環境で使用するために特に設計された HMI アプリケーションの視覚的なインターフェイスが表示されます。InTouchView アプリケーションはクライアントとして実行し、ほとんどの HMI 機能を提供するサーバーとして機能する Application Server も同時に実行します。

InTouchView アプリケーションでは、フル機能の InTouch アプリケーションで利用できる標準的な関数の一部のみが提供されています。InTouchView アプリケーションは、Application Server Galaxy 以外の I/O ソースに接続できません。InTouchView アプリケーションで利用できるのはデフォルトの Galaxy アクセス名だけで、アクセス名を作成することはできません。

アクセス名を作成するには

1. [ホーム] メニューの [タグ] グループで [アクセス名] をクリックします。
[アクセス名] ダイアログ ボックスが表示されます。
2. [追加] をクリックします。
[アクセス名の追加] ダイアログ ボックスが表示されます。

Add access name

Primary source

Access name Node name Application name Topic name

Which protocol to use? When to advise server?

☐ DDE ☒ Suitelink ☐ All items ☒ Only active items

Secondary source

☒ Enable secondary source

Node name Application name Topic name

Which protocol to use? When to advise server?

☐ DDE ☒ SuiteLink ☐ All items ☒ Only active items

Failover

☒ Enable failover

Expression (optional) Deadband: 0 sec

☒ Switchback to primary when conditions clear

Deadband: 0 sec

Cancel Add

3. [アクセス名の追加] ダイアログ ボックスのプロパティを設定します。以下の手順を実行します。

- [アクセス名] ボックスに、このアクセス名を識別する名前を入力します。
- データがネットワーク I/O サーバーに保存されている場合は、[ノード名] ボックスにリモートサーバーのノード名を入力します。
- [アプリケーション名] ボックスに、データの取得元となる I/O サーバー プログラムの実際のプログラム名を入力します。

I/O データ ソースが DAServer の場合は、DAServer プログラムの名前をプログラム ファイル名の拡張子 .exe を含めずに入力します。

- [トピック名] ボックスに、アクセスするトピック名を入力します。

トピック名はアプリケーション固有のデータ要素のサブグループです。DAServer プログラムから取得されるデータの場合、トピック名は、DAServer プログラムでそのトピックに対して設定された名前と同じです。Microsoft Excel との通信では、トピック名はファイルを保存したときに付けたブック名およびスプレッドシート名でなければなりません。たとえば、[Book1]Sheet1 と指定します。

- I/O サーバーとの通信プロトコルを選択します。
- サーバーに保存されている情報へのポーリング オプションを選択します。

オプション	定義
全てのアイテムを要求	データが表示されているウィンドウ内にあるかどうか、またはスクリプトで使用されているかどうか、あるいはアラーム、ログ、トレンドグラフが設定されているかどうかに関わらず、すべてのデータがポーリングされます。このオプションを選択するとパフォーマンスに影響が出るため、使用はお勧めできません。
有効アイテムのみ要求	表示されているウィンドウ内のデータ、およびスクリプトで使用されているデータまたはアラーム、ログ、トレンドグラフが設定されているデータだけをポーリングします。
注記: ボタンアクションスクリプトがポーリングされるのは、表示されているウィンドウにボタンが見えている場合だけです。	

- セカンダリ バック アップ サーバーを選択する場合は、[セカンダリ ソースを有効にする] を選択します。
- アクセス名を指定して [追加] をクリックします。
新しいアクセス名がリストに追加されます。
- [閉じる] をクリックします。

セカンダリ サーバーのフェイルオーバー切り替えの設定については、「[アクセス名によるフェイルオーバー機能の使用](#)」を参照してください。

アクセス名の削除

今後使用しないアクセス名を削除することができます。アクセス名の削除する前に、以下を確認してください。

- アクセス名にタグが関連付けられていないこと。
- WindowViewer が停止していること。

アクセス名を削除するには

- [ホーム] メニューの [タグ] グループで [アクセス名] をクリックします。
[アクセス名] ダイアログ ボックスが表示されます。現在のアクセス名のリストが表示されます。
- アクセス名を削除するには、削除するアクセス名をリストで選択して [削除] をクリックします。
アクセス名削除の確認を促すメッセージが表示されます。
- [はい] をクリックします。
- [閉じる] をクリックするか、他のアクセス名を削除する場合はこの手順を繰り返します。

I/O タグ変数による I/O データへのアクセス

InTouch HMI では、I/O タグ変数を使用してローカルおよびリモートの Windows アプリケーションとのデータの送受信が可能です。各 I/O 型タグ変数は、I/O サーバー プログラムの有効アイテムを参照します。タグ変数ディクショナリで、異なるタイプの I/O タグ変数を定義できます。

I/O タグ変数のプロパティの設定

タグ変数ディクショナリで、異なるタイプの I/O タグ変数を定義できます。

論理型 I/O タグ変数の指定

I/O 論理型タグ変数は、プログラマブルコントローラ、プロセスコンピュータ、およびネットワークノードのデータとの、すべての入出力のバイナリ状況を表します。

I/O 論理型タグ変数にはオン／オフの初期値を割り当てる必要があります。論理型 I/O タグ変数をバイナリソースの反対値に切り替えるように設定することもできます。また、タグ変数に割り当てられたプロセスがアラーム状況になったときにアラームイベントウィンドウに表示されるメッセージを指定することができます。

タグ変数ディクショナリでタグ変数を作成する手順の全般については、「[新しいタグの作成](#)」を参照してください。

I/O 論理型タグ変数を定義するには

1. タグ変数ディクショナリを開いて新しいタグ変数名を指定します。
2. [タグ変数タイプ] ダイアログボックスでタグ変数を I/O 論理型に指定します。[タグ変数ディクショナリ] ダイアログボックスの詳細設定が表示されます。

3. 以下の操作を行います。
 - タグ変数の初期値として [オン] または [オフ] を選択します。
アプリケーションの起動時に InTouch HMI はこの値をタグ変数に割り当てますが、I/O デバイスにはこの初期値を書き込みません。
 - リモート I/O タグ変数から受け取った値に適用される入力データの変換に、[そのまま] または [反転] を選択します。

入力データの変換	説明
そのまま	入力値は、そのままの状態で I/O サーバープログラムから直接読み取られます。
反転	I/O 論理値がサーバープログラムから読み取られるときに、値が反転されます。たとえば、I/O 入力データが 0 の場合、値には自動的に 1 がセットされます。

4. タグ変数がアラーム状況になったりアラーム状況から抜け出したときに表示するメッセージを、[オンメッセージ] ボックスおよび [オフメッセージ] ボックスに入力します。

これらのメッセージは、タグ変数にアラームの設定があるかどうかに関わらず、あらゆるアニメーションリンクやスクリプト用に利用することができます。

- タグ変数の値が 1 (On, True) のときにアクティブな論理型アラームを定義した場合には、[オンメッセージ] ボックスに入力したメッセージが ActiveX アラーム表示の [値] カラムおよび [しきい値] カラムに表示されます。

アラーム状況のタグ変数が平常に戻った場合、[オフメッセージ] ボックスに入力したメッセージが [値] カラムに表示され、[オンメッセージ] は [しきい値] カラムに残ります。

- タグ変数の値が 0 (Off, False) のときにアクティブな論理型アラームを定義した場合には、[オフメッセージ] ボックスに入力したメッセージが ActiveX アラーム表示の [値] カラムおよび [しきい値] カラムに表示されます。

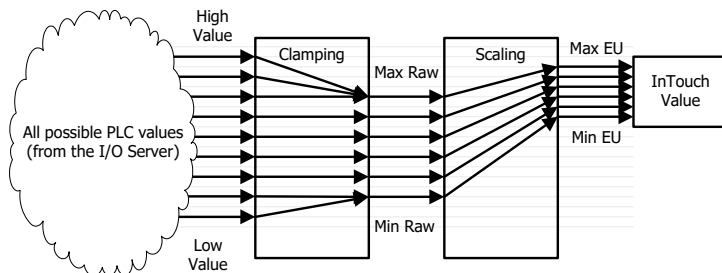
アラーム状況のタグ変数が平常に戻ると、[オンメッセージ] ボックスに入力したメッセージが [値] カラムに表示され、[オフメッセージ] は [しきい値] カラムに残ります。

5. タグ変数の変更を保存します。

整数型と実数型の I/O タグ変数の指定

I/O 整数型と I/O 実数型のタグ変数には、InTouch アプリケーションと外部プロセス間でやり取りされる数値データの特性を表す属性のセットを割り当てる必要があります。

InTouch HMI は、PLC から入力される生データを正規化します。以下の図は、生の I/O データ値がクランプされてから、InTouch アプリケーションで表示できる工学単位にスケーリングされるプロセスを示しています。



I/O 整数型と I/O 実数型のタグ変数には、PLC から送られる生の入力データの最小値と最大値を設定する属性があります。InTouch HMI は、この生データ値の範囲外の I/O データ値をクランプします。クランプによってこの範囲外の値には、生データの最小値か最大値が再割り当てされます。

I/O 整数型と I/O 実数型のタグ変数には、クランプされた生データ値を工学単位の範囲にスケーリングするための属性があります。最小工学値と最大工学値の属性には、スケーリングされる値の上限値と下限値が設定されます。

I/O 整数型と I/O 実数型のタグ変数の定義では、工学値の算出の際に生データ値をスケーリングするために使用される変換タイプを指定します。線形または二乗根から選択できます。

線形スケーリングでは、最小と最大の両終点間の線形補完法を使用して結果が計算されます。入力 of 線形スケーリングのアルゴリズムは以下のとおりです。

$$EUValue = (RawValue - MinRaw) * ((MaxEU - MinEU) / (MaxRaw - MinRaw)) + MinEU$$

出力の線形スケーリングのアルゴリズムは以下のとおりです。

$$RawValue = (EUValue - MinEU) * ((MaxRaw - MinRaw) / (MaxEU - MinEU)) + MinRaw$$

二乗根スケーリングでは、生データの最小値と最大値が補完に使用されます。これは、圧力変換器などの非線形装置からの入力のスケーリングに役立ちます。入力の二乗根スケーリングのアルゴリズムは以下のとおりです。

$$EUValue = \sqrt{RawValue - MinRaw} * ((MaxEU - MinEU) / \sqrt{MaxRaw - MinRaw}) + MinEU$$

出力の二乗根スケーリングのアルゴリズムは以下のとおりです。

$$\text{RawValue} = \text{square}((\text{EUValue} - \text{MinEU}) * (\text{sqrt}(\text{MaxRaw} - \text{MinRaw}) / (\text{MaxEU} - \text{MinEU}))) + \text{MinRaw}$$

整数型と実数型の I/O タグ変数を定義するには

1. タグ変数ディクショナリを開いて新しいタグ変数名を指定します。
2. [タグ変数タイプ] ダイアログ ボックスで、タグ変数のタイプに I/O 整数型か I/O 実数型を指定します。[タグ変数ディクショナリ] ダイアログ ボックスの詳細設定が表示されます。

3. 以下の操作を行います。
 - アプリケーション起動時にタグ変数に関連付けられる整数値または実数値を、[初期値] ボックスに入力します。
アプリケーションによる、この初期値の外部プロセスへの書き込みは行われません。
 - [工学値最小値] ボックスに、タグ変数に対する最小工学値を入力します。
 - [工学値最大値] ボックスに、タグ変数に対する最大工学値を入力します。
 - [生データ最小値] ボックスに、I/O 整数型または I/O 実数値の生データが取り得る最小クランプの値を入力します。
 - [生データ最大値] ボックスに、I/O 整数型または I/O 実数値の生データが取り得る最大クランプの値を入力します。
 - [工学単位] ボックスに、タグ変数の工学単位の名前を入力します。
 - 工学値の算出の際に生データ値をスケーリングするために使用する変換のタイプとして、[線形] または [二乗根] を選択します。
4. タグ変数の変更を保存します。

メッセージ型 I/O タグ変数の指定

I/O メッセージ型タグ変数のオプションを指定して、リモートプロセスのネットワーク アドレスを特定することができます。このメッセージのプロパティは、メモリ メッセージ型タグ変数と同じです。

メモリ型と I/O メッセージ型のタグ変数値を定義するには

1. タグ変数ディクショナリを開いて新しいタグ変数名を指定します。
2. [タグ変数タイプ] ダイアログ ボックスで、タグ変数のタイプに [I/O メッセージ型] を選択します。[タグ変数ディクショナリ] ダイアログ ボックスの詳細設定が表示されます。

3. [最大文字数] ボックスに、タグ変数のメッセージの最大文字数を入力します。
入力できるメッセージの最大文字数は 131 文字です。

4. [初期値] ボックスに、WindowViewer でアプリケーションが起動されたときに表示される文字列を入力します。
5. タグ変数の変更を保存します。

I/O アクセスパラメータの設定

タグ変数ディクショナリでタグ変数の I/O 属性を設定することができます。この属性により、タグ変数に関連付けられた外部データソースが識別されます。

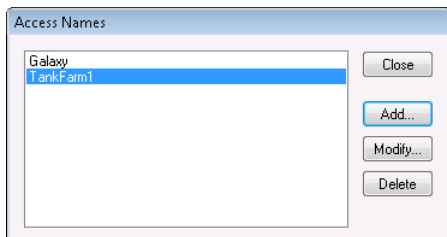
ここに示す手順では、タグ変数ディクショナリで I/O 属性を指定する方法だけを説明します。Galaxy とアクセス名の設定については、「[I/O を使用したデータアクセス](#)」を参照してください。

タグ変数の I/O 属性を設定するには

1. [タグタイプ] ダイアログボックスで、タグに I/O タグタイプを指定します。[タグ変数ディクショナリ] ダイアログボックスの詳細設定が表示されます。



2. [アクセス名] をクリックして、タグ変数に割り当てるアクセス名を定義するか選択します。[アクセス名] ダイアログボックスに、InTouch HMI が認識している現状のアクセス名のリストが表示されます。(Archestra の接続に対するデフォルトのアクセス名は Galaxy です。)



3. アクセス名を追加するか、またはデフォルト値をそのまま使用します。
4. I/O タグ変数がデータを読み書きするサーバー プログラム内のデータポイントを選択します。
 - サーバー プログラム内のプロセス データポイントでデータを読み書きできるように、[アイテム名] ボックスにアイテム名を入力します。たとえば、PLC のレジスタから値を読み取るには、レジスタの識別名をアイテム名に入力します。次に例を示します。

Allen-Bradley PLC のレジスタ 1 をアイテム名として使用するには、[アイテム名] ボックスに「R1」と入力します。

Allen-Bradley PLC のレジスタ 1 の最下位ビットをアイテム名として使用するには、[アイテム名] ボックスに「R1:0」と入力します。
 - タグ変数をアイテムとして使用するには、[タグ変数名をアイテム名として使用] を選択します。アイテム名には、最大 254 文字を入力できます。

ランタイム時の I/O タグ変数関連情報の取得

アクセス名の定義で指定したノード名、アプリケーション名、トピック名を返す関数を使用してスクリプトを記述できます。

IOGetNode() 関数

IOGetNode() 関数は、特定のアクセス名に対して定義されたノードアドレスを、スクリプトによってこの関数に関連付けられたタグ変数に返します。

カテゴリ

その他

構文

```
IOGetNode("AccessName");
```

引数

AccessName

ノード情報を取得する既存のアクセス名

備考

アクセス名はリテラル文字列で指定するか、他の InTouch タグ変数または関数が返す文字列の値で指定できます。

例

以下の例は、アクセス名 **ModbusPLC1** のノード情報をタグ変数 **NodeName** に返します。

```
NodeName = IOGetNode("ModbusPLC1");
```

IOGetApplication() 関数

IOGetApplication() スクリプト関数は、特定のアクセス名に対して定義されたアプリケーション名を、関数の引数に指定したタグ変数に返します。

カテゴリ

その他

構文

```
IOGetApplication("AccessName");
```

引数

AccessName

アプリケーションが定義されている既存のアクセス名

備考

アクセス名はリテラル文字列で指定するか、他の InTouch タグ変数または関数が返す文字列の値で指定できます。

例

以下の例は、アクセス名 **ModbusPLC1** に対して指定されたアプリケーション名を、タグ変数 **AppName** に返します。

```
AppName = IOGetApplication ("ModbusPLC1");
```

IOGetTopic() 関数

IOGetTopic() 関数は、特定のアクセス名に対して定義されたトピック名を、スクリプトによってこの関数に関連付けられたタグ変数に返します。

カテゴリ

その他

構文

```
IOGetTopic("AccessName");
```

引数

AccessName

トピック名が返されるアクセス名

備考

アクセス名はリテラル文字列で指定するか、他の InTouch メッセージ型タグ変数または関数が返す文字列の値で指定できます。

例

以下の例は、アクセス名 ModbusPLC1 のトピック情報を、タグ変数 **TopicName** に返します。

```
TopicName = IOGetTopic("ModbusPLC1");
```

ランタイム時の I/O タグ リファレンスの動的変更

InTouch HMI では、ダイナミック リファレンス アドレスを使用して、診断アプリケーションのように値を一時的に必要なデータ ポイントを表示します。ダイナミック リファレンス アドレスは、1 つのタグ変数で複数のデータ ソースのアドレス設定を可能にするものです。

1 つのタグ変数で複数のデータ ソースを動的に参照するには、いくつかの方法があります。

- I/O タグ変数の **.Reference** ドットフィールドによって、異なるアクセス名の特性を割り当てる
- **IOSetItem()** スクリプト関数を使用して、I/O タグ変数の **.Reference** ドットフィールドを設定する
- **IOSetAccessName()** スクリプト関数を使用して、ランタイム中にアクセス名の特性を変更する。

.Reference ドットフィールド

I/O タグ変数の **.Reference** ドットフィールドに対して有効なリファレンスを割り当てることにより、ダイナミック リファレンス アドレスを使用できます。**.Reference** ドットフィールドを使用して I/O タグ変数に割り当てられたアクセス名の特性を修正することにより、データ ソースを動的に変更することができます。

.Reference の構文は以下の通りです。

<code>tag.Reference="accessname.item"</code>	タグ変数に割り当てられたアクセス名とアイテムを変更します。
<code>tag.Reference="[/item"</code>	I/O タグ変数に割り当てられたアイテムを変更します。
<code>tag.Reference="accessname."</code>	I/O タグ変数のアクセス名を変更します。
<code>tag.Reference=""</code>	I/O タグ変数の使用を解除します。

各 I/O 型タグ変数には **.ReferenceComplete** ドットフィールドがあります。この論理型ドットフィールドの値は、**.Reference** ドットフィールドで要求したアイテムが、**.Value** ドットフィールドの内容に反映されたかどうかを示します。

WindowViewer でアプリケーションが起動されると、**.ReferenceComplete** フィールドに False (0) が設定されます。**.Reference** ドットフィールドに指定したソースにより **.Value** ドットフィールドが更新されたことが確認されると、**.ReferenceComplete** の値は True (1) に設定されます。**.Reference** ドットフィールドを変更すると、**.ReferenceComplete** ドットフィールドは自動的に False (0) に設定され、新しい値が更新された時点で True (1) になります。

IOSetItem() 関数

スクリプトに **IOSetItem()** 関数を使用して、ダイナミック リファレンス アドレスを使用できます。

IOSetItem() には、I/O タグ変数の **.Reference** ドットフィールドに割り当てられた値をランタイム中に変更するための引数が含まれています。

カテゴリ

その他

構文

```
IoSetItem ("Tag", "AccessName", "Item");
```

引数

Tag

引用符で囲んだ任意の InTouch I/O タグ変数

AccessName

I/O タグ変数に割り当てられたアクセス名。

Item

I/O タグ変数に割り当てられたアイテム。

タグ変数、アクセス名、およびアイテムは、リテラル文字列として指定するか、その他の InTouch タグ変数または関数が返す文字列の値を指定できます。

例

以下の例では、タグ変数 **PumpInP1** の **.Reference** ドットフィールドが、アクセス名 **excel** と アイテム **R1C1** をポイントするように変更されます。

```
IOSetItem("PumpInP1", "excel", "R1C1");
```

または、

```
Number = 1;  
TagNameString = "PumpInP" + Text(Number, "#");  
IOSetItem(TagNameString, "excel", "R1C1");
```

アクセス名とアイテムの値として空の文字列 ("") が指定された場合、タグ変数は無効になります。たとえば、以下のように指定するとタグ変数 **PumpInP2** は解除されます。

```
IOSetItem("PumpInP2", "", "");
```

アイテムのみに空の文字列を指定すると、タグ変数の現在のアイテム値は保持され、アクセス名の値のみが更新されます。たとえば、以下ではタグ変数 **PumpInP3** のアクセス名は **excel2** に変更されますが、現在のアイテム値は変更されません。

```
IOSetItem("PumpInP3", "excel2", "");
```

同様に、アクセス名の値のみに空の文字列を指定すると、タグ変数の現在のアクセス名の値が保持され、アイテム値が更新されます。たとえば、以下ではタグ変数 **PumpInP4** のアイテムは **R1C2** に変更されますが、現在のアクセス名の値は変更されません。

```
IOSetItem("PumpInP4", "", "R1C2");
```

IOSetAccessName() 関数

スクリプトに **IOSetAccessName()** 関数を使用して、ダイナミック リファレンス アドレスを使用できます。**IOSetAccessName()** は、アプリケーションまたは I/O タグ変数のアクセス名のトピック名の特性をランタイム中に修正します。

注記: **IOSetAccessName()** 関数が処理されると、既存の通信を終了してから新しい通信を開始するまでの時間遅れが生じます。この間に新しいトピックに対して実行された **POKE** や書き込みは無効になります。

カテゴリ

その他

構文

```
IOSetAccessName("AccessName", "NodeName", "AppName", "TopicName");
```

引数

AccessName

新しい AppName と Topic Name の値を割り当てる既存のアクセス名。実際の文字列またはメッセージ型タグ変数です。

NodeName

割り当てる新しいノード名。実際の文字列またはメッセージ型タグ変数です。

AppName

割り当てる新しいアプリケーション名。実際の文字列またはメッセージ型タグ変数です。

TopicName

割り当てる新しいトピック名。実際の文字列またはメッセージ型タグ変数です。

備考

AccessName、AppName、および TopicName の引数に指定する値は、リテラル文字列として指定するか、その他の InTouch タグ変数または関数が返す文字列の値を指定できます。

[アクセス名の追加] ダイアログ ボックスを使用してアクセス名のフェイルオーバーのセカンダリ ソースを設定した場合、**IOSetAccessName()** 関数を使用してランタイム時のセカンダリ ソースの設定を変更することはできません。

注記: WindowMaker でアクセス名を作成するとき、このアクセス名が SuiteLink タイプを使用している場合は、InTouch HMI は作成するアクセス名による同じノード、アプリケーションとトピックへのアクセスができないようにします。**IOSetAccessName** 関数がランタイム中にアクセス名をリダイレクトしてアクセス名を複製することがないようにしてください。ランタイム中に **IOSetAccessName()** 関数を使用すると、SuiteLink タイプのアクセス名がリダイレクトされてトピックが複製されます。ただし、リダイレクトしたアクセス名は作動しません。

例

以下のスクリプト関数の使用により、アクセス名 **MyAccess1** はアプリケーション **Excel** とトピック **[Book1]Sheet1** をポイントするように変更されますが、現在のノード名は変更されません。

```
IOSetAccessName("MyAccess1", "", "EXCEL", "[Book1]Sheet1");
```

トピックに空の文字列が指定されると、アクセス名の現在のアプリケーションの値は更新され、トピックの値は保持されます。

たとえば、以下のスクリプトではアクセス名 **MyAccess2** のアプリケーション名を **EXCEL** に変更しますが、現在のトピックの値は変更されません。

```
IOSetAccessName("MyAccess2", "", "EXCEL", "");
```

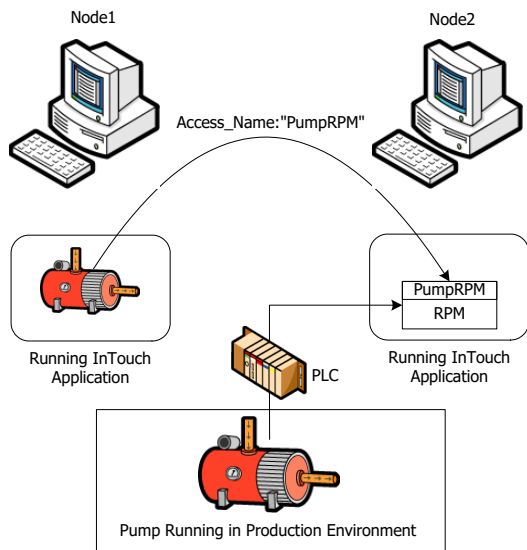
アプリケーション名のみに空の文字列が指定されると、タグ変数の現在のトピックの値は更新され、アプリケーションの値は保持されます。たとえば、以下のスクリプトではタグ変数 **MyAccess3** のトピックを **[Book2]Sheet1** に変更しますが、現在のアプリケーション名の値は変更されません。

```
IOSetAccessName("MyAccess3", "", "", "[Book2]Sheet1");
```

この例は、PLC の冗長性が必要な場合に使用できます。

タグのリモート 参照への変換

クライアント/サーバー アーキテクチャ ベースの分散 InTouch アプリケーションを作成することができます。クライアントアプリケーションを、別のリモート ノードで定義されたタグを使用する 1 つのネットワーク ノード上で実行することができます。以下の図に、ノード 1 で稼動し、ノード 2 のタグ PumpRPM へのリモート 参照を作成する InTouch アプリケーションを示します。



この例では、タグ PumpRPM の値をノード 2 から 2 つの方法で取得できます。

- I/O 型タグに関連付けられているアクセス名のノード名としてノード 2 を使用する I/O 型タグをノード 1 のタグ名ディクショナリに作成します。
- タグ PumpRPM への直接リモート 参照を使用します。たとえば、**PLC1:"TempTag"** を使用します。

次のような形式でリモート タグ名の前にアクセス名を追加することにより、ウィンドウや QuickScript でリモート タグを参照できます。

```
access_name:"tag_name"
```

ウィンドウまたは QuickScript をインポートすると、プレースホルダ タグをリモート タグ 参照に変換できます。たとえば、ウィンドウをインポートしたアプリケーションをポイントするようにプレースホルダ タグを変換できます。このタグは、ローカルタグ名ディクショナリで定義する必要はありません。

ローカル タグをリモート タグ 参照に変換するには、いくつかの方法があります。

- リモート タグ 参照を追加します。
- インポートしたウィンドウに関連付けられたプレースホルダ タグを変換します
- タグ ブラウザを起動してタグ ソースのタグ名ディクショナリを開き、リモート タグ 参照を選択します。

タグをリモート タグ 参照に手動で変換するには

1. WindowMaker でアプリケーション ウィンドウを開きます。
2. リモート タグ 参照に変換するローカル タグに関連付けられたオブジェクトを選択します。
3. [アニメーション] タブの [置換] グループで [タグ] をクリックします。

[タグ名の置換] ダイアログ ボックスが開き、オブジェクトに関連付けられたタグのリストが表示されます。

Current Name:	Required Type	New Name:
?d:AlarmHistoryNextPage	Discrete	?d:AlarmHistoryNextPage
?d:AlarmHistoryPreviousPage	Discrete	?d:AlarmHistoryPreviousPage
?m:AlarmHistoryFilter	String	?m:AlarmHistoryFilter
?v:AlarmHistoryGroupDisplayed	Group	?v:AlarmHistoryGroupDisplayed

Buttons: OK, Cancel, Index, Convert, Replace

4. 各タグ名にインデックスを追加するには、[索引] をクリックします。
5. [変換] をクリックします。タグをローカルまたはリモート 参照 タグに変換するオプションが付いた、[変換] ダイアログ ボックスが表示されます。
6. [リモート] をクリックします。[アクセス名] ダイアログ ボックスが表示されます。ローカルの InTouch アプリケーションで定義されているすべてのアクセス名が表示されます。
7. リストからアクセス名を選択します。
8. [閉じる] をクリックします。[タグの置換] ダイアログ ボックスにリスト表示されているすべてのタグは、タグ名にアクセス名が付けられたリモート タグ 参照に自動的に変換されます。
9. [OK] をクリックします。

インポートしたウィンドウのタグをリモート 参照に変換するには

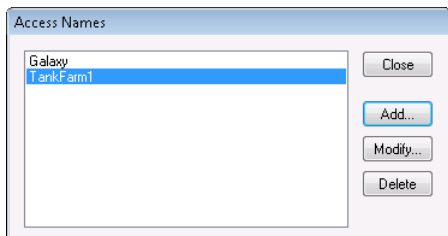
1. インポートしたウィンドウを開いてすべてのオブジェクトを選択します。
2. [アニメーション] タブの [置換] グループで [タグ] をクリックします。
[タグ名の置換] ダイアログ ボックスが表示されます。
3. [変換] をクリックします。[変換] ダイアログ ボックスが表示されます。

Convert

Local Remote

Cancel

4. **[リモート]** をクリックします。**[アクセス名]** ダイアログ ボックスが表示されます。ローカルの InTouch アプリケーションで定義されているすべてのアクセス名が表示されます。

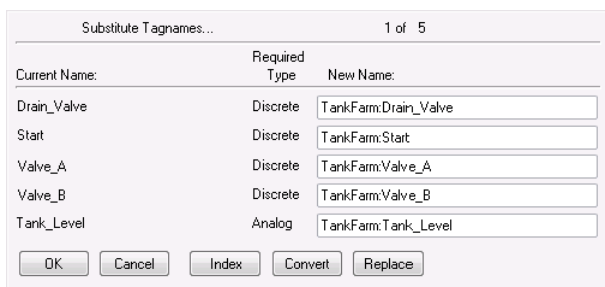


5. リストからアクセス名を選択します。

アクセス名が正しく設定されたことを確認するには、**[修正]** をクリックします。

タグ ソースにアクセスするためのアクセス名が現在定義されていない場合には、**[追加]** をクリックして定義します。アクセス名にはアプリケーションが常駐しているリモート ノードの名前を含める必要があります。

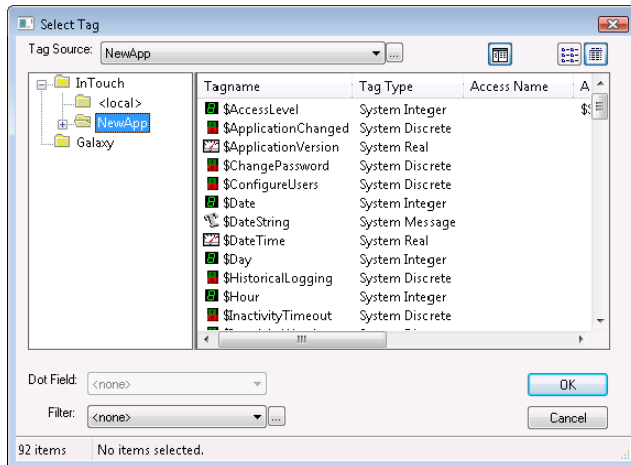
6. **[閉じる]** をクリックします。**[タグの置換]** ダイアログ ボックスにリスト表示されているすべてのタグは、タグ名にアクセス名が付けられたリモート タグ 参照に自動的に変換されます。



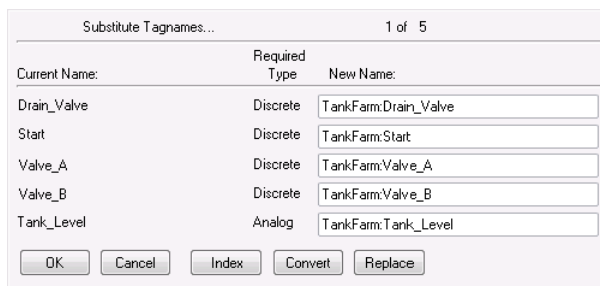
7. **[OK]** をクリックします。

リモート タグ 参照をタグ ブラウザで選択するには

1. リモート タグ 参照に変換するローカル タグに関連付けられたオブジェクトを選択します。
2. **[アニメーション]** タブの **[置換]** グループで **[タグ]** をクリックします。
[タグ名の置換] ダイアログ ボックスが開き、選択したタグが表示されます。
3. リモート タグ 参照で置き換えるタグ名を **[新しい名前]** ボックス内で削除します。
4. **[新しい名前]** ボックスをダブルクリックします。**[タグの選択]** ダイアログ ボックスが開き、アプリケーションに関連付けられたタグのリストが表示されます。
5. ツリー ビューを使用してリモート タグを選択します。
 - a. **[ツリー]** アイコンをクリックすると、左ペインにローカルとリモートの全アクセス名の階層リストが表示されます。



- b. リモート アクセス名のフォルダを選択すると、フォルダに割り当てられたタグが右ペインに表示されます。
- c. リモート 参照として使用するリモート タグを選択します。
- d. **[OK]** をクリックします。**[タグ名の置換]** ダイアログボックスが開き、選択したリモート タグの名前が **[新しい名前]** ボックスに表示されます。

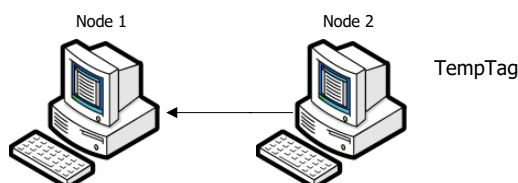


6. **[OK]** をクリックしてダイアログ ボックスを閉じ、リモートタグを選択したオブジェクトに関連付けます。
7. リモート 参照に関連付ける各タグに対して以上の手順を繰り返します。

リモート 参照による I/O データ アクセス

InTouch HMI では、工場のオートメーションアプリケーションのクライアント/サーバー アーキテクチャをサポートします。InTouch アプリケーションが稼動しているノード上にあるローカル タグ名ディクショナリのタグを一切使用することなく、クライアントアプリケーションの開発が可能です。また、リモート タグの参照を使用することにより、リモート ノードのタグを使用する 1 つのノード上でアプリケーションを実行できます。

以下の図に、**TempTag** がノード 2 でローカルに定義されているシンプルな例を示します。



この例では、ノード 1 で稼動している InTouch アプリケーションは、2 つの方法によってノード 2 の TempTag の値を取得できます。

- 関連付けられているアクセス名のノード名にノード 2 を使用する I/O 型タグを、ノード 1 のタグ名ディクショナリで作成します。
- リモート参照を使用して直接 TempTag を参照します。たとえば、Node2:"TempTag" のように使用します。

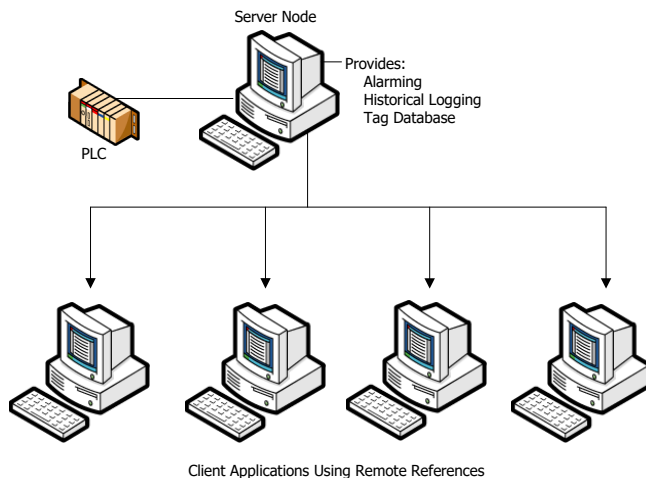
他のアプリケーションでリモートタグを直接参照する場合には、AccessName:item の記述だけが必要です。ローカルタグ名ディクショナリでリモートタグを定義する必要はありません。リモート参照により、DAServer や Microsoft Excel などのあらゆる I/O データソースのデータにアクセスすることもできます。

スーパータグへのリモート参照を作成することもできます。スーパータグへのリモートタグ参照に有効な構文を次に示します。

Access_name:Parent_Instance\ChildMember\SubMember.

スーパータグに対するリモート参照については、「[スーパータグメンバーの参照](#)」を参照してください。

ウィンドウまたは QuickScript をインポートすると、プレースホルダタグをリモートタグ参照に変換できます。ローカルのタグ名ディクショナリでタグを定義する必要はありません。リモート参照は、ネットワーク上の任意のアプリケーションからアクセスできます。



ランタイム中のリモート参照のリダイレクト

スクリプトを使用して、Application Server オブジェクト参照またはリモート参照を、ランタイム時の InTouch タグにリダイレクトできます。これにより、想定される特定の状態に基いて、またはオペレータアクションを直接介して、グラフィックシンボルのオブジェクトインスタンスを切り替えることができます。

IOSetRemoteReferences() 関数

IOSetRemoteReferences() スクリプト関数を使用して、InTouch アプリケーションの実行中に Application Server オブジェクト参照またはリモート参照をタグにリダイレクトできます。**IOSetRemoteReferences()** は、指定した文字列に一致するすべてのリモート参照を検索して、指定した引数の値に応じて参照を変更します。この関数をトリガするスクリプトを記述して、想定されるさまざまな状態に基き、またはユーザーアクションを介して参照をリダイレクトすることができます。

カテゴリ

その他

構文

```
IOSetRemoteReferences(BaseAccess, NewAccess, MatchString, SubstituteString, Mode)
```

引数

BaseAccess

この文字列引数は、参照での照合に使用する、元の設定されたアクセス名を指定します。

NewAccess

新しいアクセス名です。このアクセス名は、元のアクセス名が **BaseAccess** で指定された文字列に一致し、さらに **MatchString** の値が指定された場合には元のアイテム名がこの値に一致する、すべての参照に適用されます。

MatchString

参照の元の設定されたアイテム名での照合に使用する文字列です。**MatchString** の値が空の文字列の場合は、すべてのアイテム名が一致するとみなされます。

SubstituteString

元のアイテム名に置換される文字列です。この文字列は、**MatchString** の値と置き換わって参照に対する新しいアクティブなアイテム名を生成します。これが空の文字列の場合、置換は行われません。

Mode

元の設定されたアイテム名と **MatchString** 値との比較モードを定義します。マッチングは常にアイテム名の先頭から行われます。**Mode** の値が **0** の場合は、アイテム名全体の一致、またはピリオド (.) の位置までの一致が必要であることが指定されます。**1** の場合は、次の文字がピリオドでなくても部分的な一致が可能であることが指定されます。

備考

IOSetRemoteReferences() は、オブジェクト参照の変更前に、新しいタグまたはアクセス名の有効性のチェックは行いません。

- **IOSetRemoteReferences()** はリモート参照のみを変更します。この関数は、元の設定されたアクセス名が **BaseAccess** 引数に指定した値に一致し、さらに元のアイテム名が **MatchString** の値に一致する参照をリダイレクトします。
- **IOSetRemoteReferences()** の単一の呼び出しは、メモリ上のアクティブウィンドウ内にあり、元の設定された名前の文字列が **BaseAccess** および **MatchString** に指定した値に一致するすべてのリモート参照に影響します。
- **BaseAccess** 引数に値を指定しないと、**IOSetRemoteReferences()** はリモート参照をリダイレクトしません。
- **MatchString** 引数が空の値の場合、**IOSetRemoteReferences()** は、元のアクセス名が **BaseAccess** 引数に指定した値に一致するすべてのリモート参照をリダイレクトします。
- **Mode** 引数が **0** の場合、アイテム名での置換はオブジェクト名（またはタグ）全体、またはプロパティ名（またはドットフィールド）全体に対してのみ行われます。**MatchString** 引数の値は、元のアイテム名全体、またはピリオドの直前にある文字まで一致する必要があります。
- **Mode** 引数が **1** に設定されている場合は、アイテムの文字列と照合アイテムの文字列の開始が同じ場合に、アイテム文字列の部分的な置換が可能です。つまり、**MatchString** は元のアイテム文字列の一

部と一致し、一致部分はアイテム文字列の先頭文字から一致する必要があります。一致した文字列の最終文字の次の文字が、ピリオドである必要はありません。

- 元の設定されたリモート参照の名前は、変更されません。引き続き **IOSetRemoteReferences()** を呼び出す場合、現在アクティブな名前を指定する必要はありません。**IOSetRemoteReferences()** の呼び出しは、任意の順序で実行できます。
- 1つのリモート参照を参照するために複数のウィンドウが必要になる場合、リモート参照は I/O タグのように機能します。リモート参照をリダイレクトすると、すべてのウィンドウは同じリモート参照を表示します。2つの異なる対象を同時に参照するために、1つの名前を使用しないでください。

注記:たとえば、ウィンドウの「表示時」スクリプトなどで多くの参照を同時に変更する場合は、すべての参照が解決されるまでに時間を要することがあります。

例

以下の例は、元のアイテム名が **pumpX** に正確に一致した場合に、**Galaxy** アクセス名のアイテム名 **pump001** にすべてのリモート参照をリダイレクトします。

```
IOSetRemoteReferences("Galaxy","", "pumpX", "pump001",0);
```

以下の例は、アイテム名が **pumpX** に正確に一致した場合に、アクセス名 **Galaxy** を **TagServer1** に変更します。さらに、アイテム名を **p2** に変更します。

```
IOSetRemoteReferences("Galaxy","TagServer1", "pumpX", "p2",0);
```

以下の例は、アイテム名が **pumpX** の場合に、アクセス名 **TagServer1** を **TagServer2** に変更します。さらに、アイテム名を **backpump3** に変更します。

```
IOSetRemoteReferences("TagServer1","TagServer2", "pumpX", "backpump3",0)
```

以下の例は、アクセス名 **TagServer1** のアイテム名 **Tank** を **Plant** に変更します。

```
IOSetRemoteReferences("TagServer1","", "Tank", "Plant",1)
```

以下の例では、**BaseAccess** 引数の値が指定されていないため、リモート参照はリダイレクトされません。

```
IOSetRemoteReferences("", "Galaxy", "pumpX", "pump001",0);
```

リファレンスの回復

NewAccess 引数に値が指定されず空白の場合、**IOSetRemoteReferences** は、アクティブなアクセス名を元のベースアクセス名に戻します。

MatchString 引数に値が指定されず空白の場合、**IOSetRemoteReferences()** は、アクティブなアイテム名を元のアイテム名に戻します。

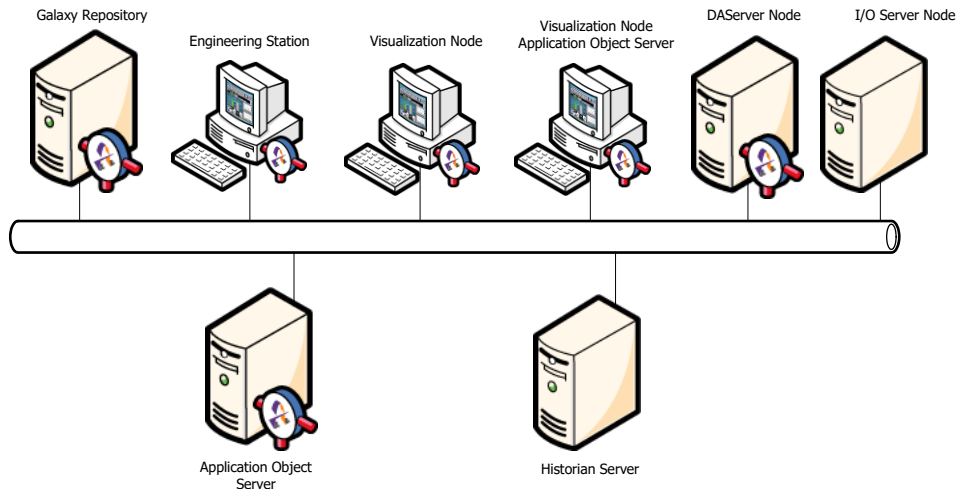
注: **SubstituteString** が空白でない場合でも、**MatchString** が空白である場合は、アイテム名は元のアイテム名に戻されます。名前の最初にテキストを挿入することはできません。たとえば、**IOSetRemoteReferences("Access1","", "", "Valve",0);** というスクリプトを実行しても、すべての元のアイテム名の最初または最後に、「Valve」という文字列は付加されません。

SubstituteString に値が指定されず空白の場合、**IOSetRemoteReferences()** はアクティブなアイテム名を元のアイテム名に戻します。空白でない **MatchString** と空白の **SubstituteString** を使用すると、指定した元のベースアクセス上のリモートリファレンスからサブセットを選択し、元のアイテム名に戻します。

InTouch から Application Server データへのアクセス

ArchestrA には、一連の製品のための共通サービスのセットと基本アーキテクチャが用意されています。このようなさまざまな製品群から製品を選択し、モジュール方式の ArchestrA コンポーネントを使用して工場の自動化システムや情報システムを構築できます。

Application Server によって、工場の自動化アプリケーションを構築するための一連のサービスが提供されます。Application Server のサービスは、システム内の複数のノードを介して配信されます。



通常は、InTouch HMI が Application Server と連携して、工場のプロセス管理のためにユーザーがインタラクティブに使用するアプリケーションにビジュアルインターフェースを提供します。

InTouch と Application Server オブジェクトの使用

InTouch タグ変数を Application Server オブジェクト属性のやり取りに使用して、InTouch アプリケーションと Application Server データ レポジトリ間でデータ転送を行えます。

InTouch HMI の通信プロトコルには、メッセージ交換がサポートされています。WindowViewer で InTouch アプリケーションが実行されると、メッセージ交換は WindowViewer を匿名エンジンとみなします。

匿名性があるということは、InTouch アプリケーションには他のメッセージ交換クライアントからアクセスできる属性がないことを意味します。WindowViewer は、Application Server Galaxy ではオートメーションオブジェクトとしては設定されず、また管理も表示もされません。そのため InTouch HMI は、メッセージ交換のみを使用して Application Server の利用可能なアクティブアイテムを要求します。

InTouch タグブラウザを使用してリモート タグのタグ ソースとして Galaxy を選択し、Galaxy のネームスペースを参照することができます。Application Server のオブジェクト属性や属性のプロパティは、リモート リファレンスで使用したり、InTouch I/O タグ変数のアイテムとして使用できます。

リモート タグ変数ソースとしての Application Server オブジェクトの使用の詳細については、「[リモート タグ ソースとして Galaxy を使用するように InTouch HMI を設定する](#)」を参照してください。

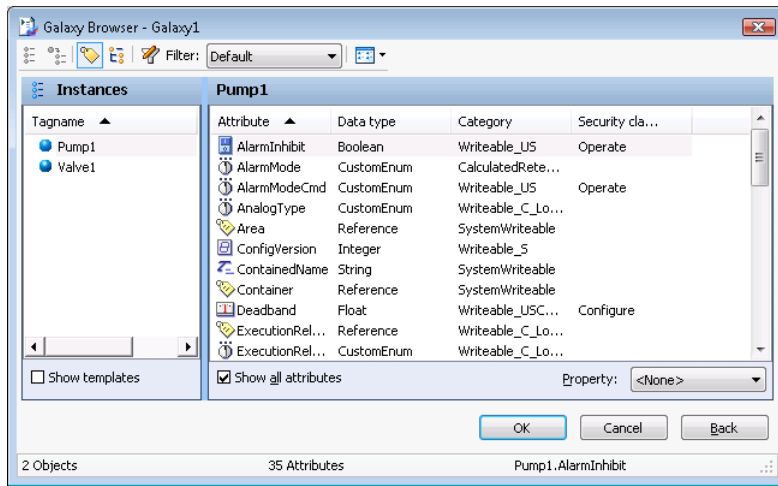
あらかじめ設定されている Galaxy というアクセス名を、WindowMaker でのメッセージ交換アクセス用に利用することができます。アクセス名 Galaxy は、ArchestrA 環境の InTouch HMI に対してのみ関連性を持ちます。アクセス名 Galaxy にはユーザーが変更できるプロパティはありません。

InTouch からの Application Server のオブジェクト属性の参照

InTouch HMI から Application Server 属性を参照して選択するには、最初に InTouch HMI で Galaxy のタグソースを設定する必要があります。詳細については、「[リモートタグソースとして Galaxy を使用するように InTouch HMI を設定する](#)」を参照してください。

InTouch の「タグ変数を選択してください」ダイアログボックスからタグソースを選択します。

InTouch の「Galaxy Browser」ダイアログボックスに、対象の Galaxy にあるすべてのオブジェクトのリストが表示されます。オブジェクトを展開すると、その配下のオブジェクトやランタイムでのアクセスが可能な属性を表示できます。「Galaxy Browser」ダイアログボックスには、「_」（隠しマーク）で始まる属性や、QualifiedStruct のタイプの属性は表示されません。



「Galaxy Browser」ダイアログボックスから、通常の InTouch の「タグ変数を選択してください」ダイアログボックスに戻るには、「Back」をクリックします。

Application Server ブラウザの制限事項

InTouch の「属性ブラウザ (Attribute Browser)」ダイアログボックスでの Application Server オブジェクトの表示には、次のような制限事項があります。

- ランタイム中に単一の Galaxy のネームスペース内で表示できる属性のみが表示されます。これには、オブジェクトの TagName と HierarchicalName 間を切り替える機能が含まれます。

以下の要件に適合する場合に、「属性ブラウザ」ダイアログボックスからオブジェクト属性を選択できます。

- ランタイム中に表示可能であること
- オートメーションオブジェクト内で現在チェック済みであること
- 属性名の「.」の後に「_」が続かないこと
- 「属性ブラウザ」ダイアログボックスには、InTouch HMI でサポートされている Application Server オブジェクト属性のデータタイプのみを表示できます。サポートされるデータタイプの詳細については、「[Application Server データタイプの InTouch データタイプへのマッピング](#)」を参照してください。
- InTouch の「属性ブラウザ」ダイアログボックスには、InTouch アイテム名が最大文字数の 95 文字以上の属性は表示されません。

- オートメーション オブジェクトの配列要素は、「TagName.AttributeName[index]」をリファレンスに使用することにより InTouch HMI で表示または取得されます。すべての配列要素の値を表示または取得するには、インデックスに -1 を使用してください。
- タグブラウザを使用して、オブジェクト属性のプロパティを選択できます。デフォルトでは、属性の選択時に Value プロパティが選択されます。

Application Server オブジェクトの特殊な拡張子

WindowMaker タグブラウザと WindowViewer のメッセージ交換クライアントは、各 Application Server オブジェクト属性に特別な拡張子を追加して認識します。この拡張子によって、InTouch HMI では利用できなかった情報にアクセスできます。

この特別な拡張子は省略可能で、InTouch アプリケーションでは使用する必要はありません。ただし、ステータス情報や品質情報を頻繁に扱うアプリケーションでは、この拡張子を使用する必要があります。

これらの項目によって属性のネームスペースが拡張され、WindowViewer がアプリケーション スクリプトとウィンドウに対して公開できる追加プロパティが包含されます。たとえば、「TIC101.PV.#ReadSts」へのリファレンスによって、TIC101.PV への要求に関する MxStatus の情報にアクセスできます。この情報は、メッセージ交換によって公開される拡張情報の表示に非常に有用です。

これらのプロパティは、指定された要素としての Application Server 内のオブジェクト属性としては存在しません。このプロパティは、クライアント抽象レイヤーで提供されたクライアント側の拡張子であり、InTouch HMI でのオブジェクト属性の表示を可能にします。次の表で、InTouch HMI に対する属性の拡張子を説明します。

属性の拡張子	データ タイプ	説明
なし	強制	デフォルトの拡張子。拡張子が提供されていないことを示します。アイテムは、強制的に与えられた InTouch に適合するデータ タイプで読み書きされます。クライアントが、次に説明する #ReadSts アイテムや #WriteSts アイテムを要求する場合、失敗した読み取り／書き込み情報を取得することができます。例："Pump1.PV".

属性の拡張子	データ タイプ	説明
.#VString	文字列型 (読み取り／書き込み)	接尾辞として「.#VString」を持つリファレンスへの要求を設定します。この拡張子は基本のリファレンスです。読み取りと書き込みがどちらも正しく作動しているときには、基本リファレンスの現在の値を文字列として返します。
float/double 属性の場合のみ:		UserGetAttribute が不良ステータスを返す場合、このアイテムは現在の値の代わりに MxStatus に基づくステータスの短い説明文字列を返します。短いステータスの説明文字列には、以下のものがあります。
.#VString1		“?Pending” – 保留
.#VString2		“?Warning” – 警告
.#VString3		“?Comms” – 通信エラー
.#VString4		“?Config” – 設定エラー
		“?Oper” – 操作エラー
		“?Security” – セキュリティ エラー
		“?Software” – ソフトウェア エラー
		“?Other” – その他のエラー

.#VString では、ステータスは良好でも品質が不良の場合、このアイテムは値の代わりに、メッセージ交換で利用可能な品質の説明文字列を返します。

UserGetAttribute に対する品質とステータスが良好、または品質が良好でステータスが不明な場合は、値が文字列として返されます。メッセージ交換で返されたデータ タイプが文字列でない場合は、強制機能が必要となることがあります。品質やステータスが不明な場合、値には接尾辞として「?」が表示されます。たとえば、「3.27?」または「True?」となります。

Application Server データ タイプの InTouch データ タイプへのマッピング

Application Server に含まれる一部の属性とデータ タイプは、InTouch タグ変数がサポートする主要な 4 つのデータ タイプに直接マップされません。

次の表で、読み取り／書き込み操作に対してクライアント抽象レイヤーがデータ タイプをマップする方法を説明します。また、Galaxy デictionary が InTouch に対して公開しているデータ タイプについても説明しています。

属性	InTouch データ タイプ	ノート
プロパティ データ タイプ	タイプ	
Float	実数型 - 32 ビット	そのまま渡します。

属性 プロパティ データ タイプ	InTouch データ タイプ	ノート
Double	実数型 - 32 ビット	ダブル型が IEEE NAN の場合、浮動型 IEEE NAN に変換されます。オーバーフローする場合は、 Quality を不良に設定して浮動型 IEEE NAN に渡します。ダブル型の値が最小浮動小数値 1.17549E-38 より小さい場合は、この値を 0.0 の浮動型として処理し、 Quality を良好に設定します。
Boolean	論理型	False = 0、True = 1
Integer	整数型 - 32 ビット	そのまま渡します。
String (常に Unicode)	メッセージ型 - MBCS (コード化された複数バイト文字列セット)	InTouch の文字列として長すぎる場合は文字列を切り詰めて、品質を不明に設定します。各 Unicode 文字の両方のバイトは保持されます。
Time	メッセージ型 - MBCS	その地域に応じた文字列形式です。 MxValue を使用して文字列を変換します。
ElapsedTime	実数	浮動型の秒数として渡します。 MxValue はこのタイプへの強制機能をサポートしています。
MxDataType	メッセージ型 - MBCS	文字列を渡します。
MxSecurityClassification	メッセージ型 - MBCS	文字列を渡します。
MxQuality	メッセージ型 - MBCS	文字列を渡します。
MxReference	メッセージ型 - MBCS	Unicode としてのみ参照文字列を渡します。
MxCategorizedStatus	メッセージ型 - MBCS	文字列を渡します。
MxQualifiedStruct	サポートしていません。	サポートされていません。
MxQualifiedEnum	メッセージ型 - MBCS	Enum 文字列を渡します。順番を示す整数値には、 .#EnumOrdinal を参照することによりアプリケーションからアクセスできます。たとえば、

属性 プロパティ データ タイプ	InTouch データ タイプ	ノート
		“Pump1.PV.#EnumOrdinal” でアクセスできます。
文字列の配列	メッセージ - MBCS (読み取り専用)	配列の各要素を、以下のようなカンマ区切り文字列に配置します。 “String1,String2,String3” 配置できるのは、InTouch 文字列値の上限までです。この文字列が切り詰められた場合、InTouch HMI に送られる品質は不明になります。文字列配列の全体に対する書き込みはできませんが、配列の個々の要素に対する書き込みは可能です。
すべての配列	整数型、実数型、メッセージ型、または論理型	配列の単一要素に対してのみの要求をサポートします。この場合、上で説明した変換が適用されます。それ以外の場合、返り値は不良品質の空の文字列です。
MxInternationalizedText	メッセージ型	ランタイム時に文字列タイプとしてアクセスされます。

Application Server 属性の読み取り/書き込み動作

システムがオートメーション オブジェクト属性を書き込む場合、初期の書き込みステータスは“Pending”に設定されます。

書き込みが完了すると、#WriteSts 文字列が書き込み結果で更新されます。書き込みが正常に終了した場合は、#WriteSts の値には空の文字列がセットされます。書き込み処理がエラーを返して保留状態になると、要求の更新が引き続き読み取り時に発生しても、#WriteSts アイテムは最後の書き込みステータスをそのまま表示します。

#VString1 ～ #VString4 のアイテムを使用して、浮動値やダブル値を文字列形式に変換することもできます。N の値は、返される値の小数点以下の桁数を示します。たとえば、「3.1234」は #VString4 を使用した文字列です。N の値を使用しない #VString アイテムを使用すると、浮動型やダブル型の値を整数型に四捨五入して、文字列型の値として返すことができます。

属性の拡張子	データ タイプ	説明
.#EnumOrdinal	整数型 (読み取り/書き込み)	Qualified Enum 型の属性を読み取っている現在の序数を示す値です。この場合、列挙型に対して文字列ではなく整数型が返されます。

属性の拡張子	データ タイプ	説明
.#ReadSts	文字列（読み取り専用）	<p>この文字列が連結されるアイテムに対して要求された現在の読み取りステータスです。「TIC101.PV.#ReadSts」のように表します。このステータスは、メッセージ交換によって文字列で供給されます。以下のいずれかの値をとります。</p> <p>“?Config” – 設定エラー</p> <p>“?Comms” – 通信エラー</p> <p>“?Oper” – 操作エラー</p> <p>“?Pending” – 保留</p> <p>“?Warning” - 警告</p> <p>“?Security” – セキュリティ エラー</p> <p>“?Software” – ソフトウェア エラー</p> <p>“?Other” – その他のエラー</p> <p>注: 関連付けられたアイテム（たとえば、TIC101.PV）が要求を受けていない場合、返される文字列は空白になります。</p>
.#WriteSts	文字列（読み取り専用）	<p>この文字列が連結されるアイテムの最後の書き込みステータスで、たとえば Pump1.Cmd.#WriteSts のように表します。このステータスは、メッセージ交換によって文字列で供給されます。この文字列が空白の場合は、このアイテムへの最後の書き込みは正常です。空白以外には、以下のいずれかの値をとります。</p> <p>“?Config” – 設定エラー</p> <p>“?Comms” – 通信エラー</p> <p>“?Oper” – 操作エラー</p> <p>“?Pending” – 保留</p> <p>“?Warning” - 警告</p> <p>“?Security” – セキュリティ エラー</p> <p>“?Software” – ソフトウェア エラー</p> <p>“?Other” – その他のエラー</p> <p>注: 関連付けられたアイテム（たとえば、TIC101.PV）が要求を受けていない場合、返される文字列は空白になります。</p>

リモートタグソースとして Galaxy を使用するように InTouch HMI を設定する

InTouch タグブラウザを使用して、タグソースとして Application Server オブジェクトを選択し、Galaxy データベースを参照することができます。Application Server オブジェクト属性または属性プロパティは、リモート参照で使用したり、InTouch I/O タグのアイテムとして使用できます。

InTouch アプリケーションが、Application Server アプリケーションのビジュアルインターフェースを提供するクライアントとして実行されている場合、InTouch アプリケーションと同じノードに Application Server Bootstrap と WinPlatform オブジェクトをインストールする必要があります。

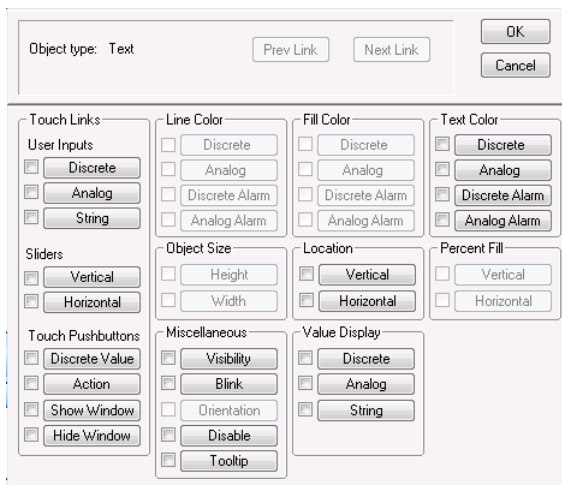
また、InTouch HMI から Galaxy ネームスペースを参照するには、Integrated Development Environment (IDE) をインストールする必要があります。

InTouch HMI では、プラットフォームのメッセージ交換機能を使用して、Galaxy ネームスペースの表示や優れた通信機能を実現しています。

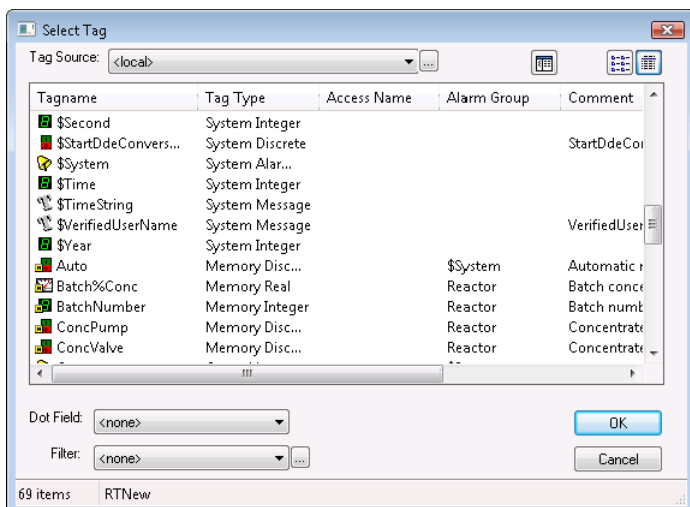
リモートタグソースとして Galaxy を使用するように InTouch を設定するには

1. WindowMaker でアプリケーション ウィンドウを開きます。
2. テキスト オブジェクトをダブルクリックします。

[アニメーションリンク] ダイアログボックスが表示されます。



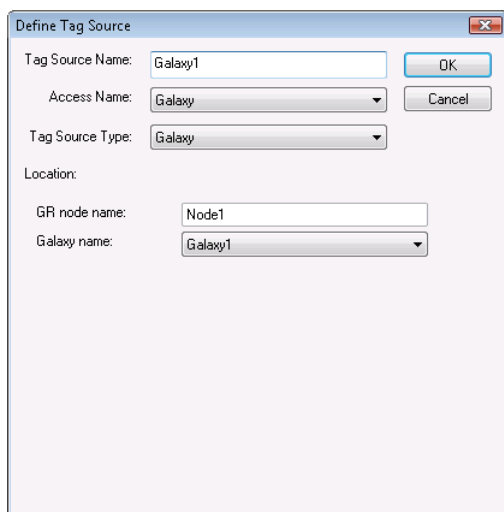
3. [値の表示] 領域で [アナログ] をクリックすると、式を挿入するダイアログボックスが表示されます。
 4. [式] ボックスに表示されている式をすべて削除します。
 5. [式] ボックス内をダブルクリックします。
- [タグの選択] ダイアログボックスが表示されます。



6. [タグ ソース] ボックスの右のボタンをクリックします。
[タグ ソースの一覧] ダイアログ ボックスが表示されます。

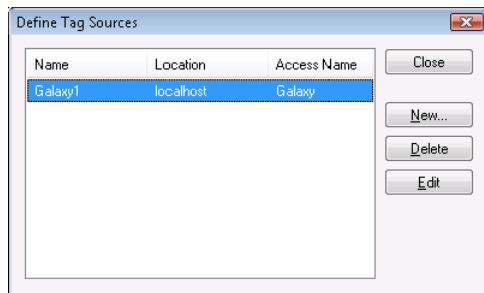


7. [新規] をクリックして [タグソースの一覧] ダイアログ ボックスを表示します。

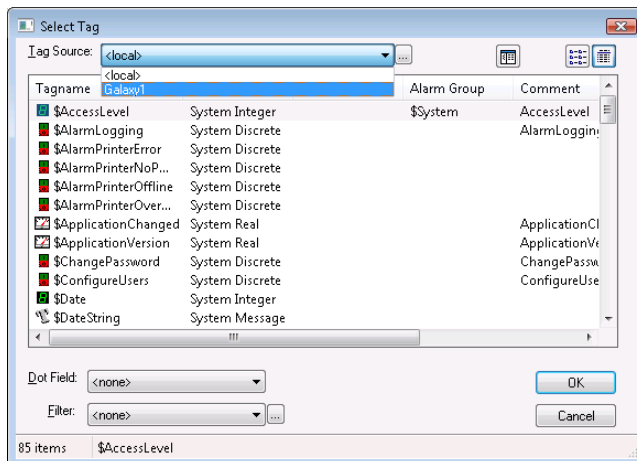


8. [タグ ソースの一覧] ダイアログ ボックスに値を入力します。以下の手順を実行します。
- [タグ ソース名] ボックスに、リモート Galaxy タグ ソースの名前を入力します。
 - [アクセス名] ボックスのリストから Galaxy を選択します。
 - [タグ ソース タイプ] ボックスのリストから Galaxy を選択します。

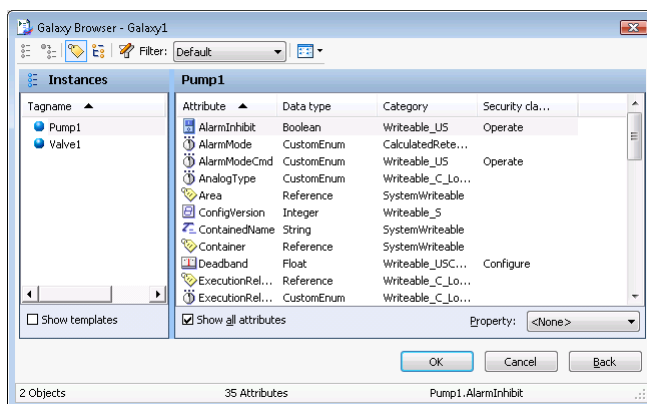
- d. [場所] 領域では Galaxy レポジトリ ノード名を入力してから、リストから Galaxy を選択します。
- e. [OK] をクリックします。[タグソースの一覧] ダイアログボックスに、リストで定義したリモートタグソースが表示されます。



9. [閉じる] をクリックして [タグソースの一覧] ダイアログボックスを閉じます。[タグの選択] ダイアログボックスの [タグソース] ボックスのリストに、新しいタグソースが表示されます。



10. [タグソース] ボックスで、作成した新しいタグソースを選択します。[Galaxy ブラウザ] ダイアログボックスが開き、左ペインにタグのリストが表示されます。



11. [Galaxy ブラウザ] ダイアログボックスの左ペインでタグを選択します。[Galaxy ブラウザ] ダイアログボックスの右ペインが、選択したタグの属性で更新されます。
12. 使用する属性をクリックしてから [OK] をクリックします。[アナログ値表示リンク] ダイアログボックスが開き、[式] ボックスに式が表示されます。

Object type: Button Prev Link Next Link OK Cancel

Output -> Analog Expression

Expression: Galaxy:Pump1.PV OK Cancel Clear

13. 文字列式が正しいことを確認します。

式には次の形式が使用されます。

Galaxy:tag_name.attribute_name

例:

Galaxy:Pump1.PV

14. [OK] をクリックして [アナログ値表示リンク] ダイアログ ボックスを閉じます。

15. 必要に応じて、残りのオブジェクト リンクを設定します。

16. アニメーション リンクのダイアログ ボックスで、[OK] をクリックします。

17. [実行] をクリックします。テキスト オブジェクトに、設定したタグの属性に対する値が表示されます。

I/O タグ変数の時間スタンプと品質情報の表示

InTouch HMI は、VTQ 対応のクライアントに配信されるすべてのデータ値に値、時間、および品質 (VTQ) のインジケータを挿入します。InTouch は、トラブルシューティングに有用なデータ品質のインジケータとしてドットフィールドのセットを使用しています。

- **.Value** ドットフィールドには、指定したタグ変数の値が格納されています。また、このドットフィールドは、すべての InTouch タグ変数のデフォルトのドットフィールドです。他のドットフィールドが指定されない場合は、.Value ドットフィールドが使用されます。
- 一連の **Time** ドットフィールドは、タグ変数が最後に更新された時間を示す時間スタンプです。
- **Quality** ドットフィールドは、I/O タグ変数に割り当てられたデータ値の信頼性を表します。

I/O タグ変数の時間スタンプ情報の表示

Time ドットフィールドのセットは、次の形式でタグ変数に割り当てられます。

Tag_name.Time_Dotfield

.TimeDate ドットフィールド

.TimeDate ドットフィールドは、1970 年 1 月 1 日から I/O サーバーのタグ変数の値が最後に更新された日付までの日数を示します。

カテゴリ

タグ

使用法

Tag_name.TimeDate

パラメータ

Tag_name

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データ タイプ

整数型（読み取り専用）

参照項目

.TimeDateString、.TimeDay、.TimeDateTime、.TimeHour、.TimeMinute、.TimeMsec、.TimeMonth、.TimeSecond、.Time.Time、.TimeTimeString、.TimeYear

.TimeDateString ドットフィールド

.TimeDateString ドットフィールドは、I/O サーバーのタグ変数の値が最後に更新された日時を示します。

カテゴリ

タグ

使用法

`Tag.TimeDateString`

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

メッセージ型（読み取り専用）

参照項目

.TimeDate、.TimeDay、.TimeDateTime、.TimeHour、.TimeMinute、.TimeMsec、.TimeMonth、.TimeSecond、.Time.Time、.TimeTimeString、.TimeYear

.TimeDateTime ドットフィールド

.TimeDateTime ドットフィールドは、1970 年 1 月 1 日から I/O サーバーのタグ変数の値が最後に更新された日付までの日数を小数点以下を含めて表します。

カテゴリ

タグ

使用法

`Tag.TimeDateTime`

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

実数型（読み取り専用）

参照項目

.TimeDate、.TimeDateString、.TimeDay、.TimeHour、.TimeMinute、.TimeMsec、.TimeMonth、.TimeSecond、.Time.Time、.TimeTimeString、.TimeYear

.TimeDay ドットフィールド

.TimeDay ドットフィールドは、I/O サーバーのタグ変数の値が最後に更新された日付からのその月の経過日数を表します。

カテゴリ

タグ

使用法

Tag.TimeDay

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

整数型（読み取り専用）

有効値

値の範囲は 1 ～ 31 です。

参照項目

.TimeDate、.TimeDateString、.TimeDateTime、.TimeHour、.TimeMinute、.TimeMsec、.TimeMonth、.TimeSecond、.Time.Time、.TimeTimeString、.TimeYear

.TimeHour ドットフィールド

.TimeHour ドットフィールドは、I/O サーバーのタグ変数の値が最後に更新された時刻からのその日の経過時間を表します。

カテゴリ

タグ

使用法

Tag.TimeHour

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

整数型（読み取り専用）

有効値

値の範囲は 0 ～ 23 です。

参照項目

.TimeDate、.TimeDateString、.TimeDay、.TimeDateTime、.TimeMinute、.TimeMsec、.TimeMonth、.TimeSecond、.Time.Time、.TimeTimeString、.TimeYear

.TimeMinute ドットフィールド

.TimeMinute ドットフィールドは、I/O サーバーのタグ変数の値が最後に更新された時刻の分の部分を示します。

カテゴリ

タグ

使用法

Tag.TimeMinute

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

整数型（読み取り専用）

有効値

値の範囲は 0 ～ 59 です。

参照項目

.TimeDate、.TimeDateString、.TimeDay、.TimeDateTime、.TimeHour、.TimeMsec、.TimeMonth、.TimeSecond、.Time.Time、.TimeTimeString、.TimeYear

.TimeMonth ドットフィールド

.TimeMonth ドットフィールドは、I/O サーバーのタグ変数の値が更新された時点の日付の月数（1 ～ 12）を表します。

カテゴリ

タグ

使用法

Tag.TimeMonth

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

整数型（読み取り専用）

有効値

値の範囲は 1 ～ 12 です。

参照項目

.TimeDate、.TimeDateString、.TimeDay、.TimeDateTime、.TimeHour、.TimeMinute、.TimeMsec、.TimeSecond、.Time.Time、.TimeTimeString、.TimeYear

.TimeMsec ドットフィールド

.TimeMsec ドットフィールドは、I/O サーバーのタグ変数の値が最後に更新された時刻のミリ秒の部分を示します。

カテゴリ

タグ

使用法

`Tag.TimeMsec`

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

整数型（読み取り専用）

有効値

値の範囲は 0 ～ 999 です。

参照項目

.TimeDate、.TimeDateString、.TimeDay、.TimeDateTime、.TimeHour、.TimeMinute、.TimeMonth、.TimeSecond、.Time.Time、.TimeTimeString、.TimeYear

.TimeSecond ドットフィールド

.TimeSecond ドットフィールドは、I/O サーバーのタグ変数の値が最後に更新された時刻の秒の部分を示します。

カテゴリ

タグ

使用法

`Tag.TimeSecond`

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

整数型（読み取り専用）

有効値

値の範囲は 0 ～ 59 です。

参照項目

.TimeDate、.TimeDateString、.TimeDay、.TimeDateTime、.TimeHour、.TimeMinute、.TimeMsec、.TimeMonth、.TimeSecond、.TimeTimeString、.TimeYear

.TimeTime ドットフィールド

.TimeTime ドットフィールドは、I/O サーバーのタグ変数の値が更新された時点のタイムスタンプを表し、午前 0 時から経過したミリ秒の数で示されます。

カテゴリ

タグ

使用法

`Tag.TimeTime`

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

整数型（読み取り専用）

有効値

値の範囲は 0 ～ 86399999 です。

参照項目

.TimeDate、.TimeDateString、.TimeDay、.TimeDateTime、.TimeHour、.TimeMinute、.TimeMsec、.TimeMonth、.TimeSecond、.TimeTimeString、.TimeYear

.TimeTimeString ドットフィールド

.TimeTimeString ドットフィールドは、I/O サーバーのタグ変数の値が最後に更新された時刻を示します。

カテゴリ

タグ

使用法

`Tag.TimeTimeString`

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

メッセージ型（読み取り専用）

参照項目

.TimeDate、.TimeDateString、.TimeDay、.TimeDateTime、.TimeHour、.TimeMinute、.TimeMsec、.TimeMonth、.TimeSecond、.TimeTimeString、.TimeYear

.TimeYear ドットフィールド

.TimeYear ドットフィールドは、I/O サーバーのタグ変数の値が更新された年を 4 桁の数字で表します。

カテゴリ

タグ

使用法

Tag.TimeYear

パラメータ

Tag

論理型、整数型、実数型、メッセージ型、間接アナログ型、間接論理型、または間接メッセージ型の任意のタグ変数

データタイプ

整数型（読み取り専用）

有効値

4 桁の数字で表される年。

参照項目

.TimeDate、.TimeDateString、.TimeDay、.TimeDateTime、.TimeHour、.TimeMinute、.TimeMsec、.TimeMonth、.TimeSecond、.Time.Time、.TimeTimeString。

I/O タグ変数の品質情報の表示

品質ドットフィールドのセットを使用して、I/O サーバーと InTouch アプリケーション間でやり取りされるデータの完全性を確保できます。品質ドットフィールドはアイテムのデータ値の品質状況を表します。この品質属性により、ネットワーク ノード間で転送される InTouch データの完全性の監視が極めて容易になります。

データ品質基準は、OLE for Process Control (OPC) で提案された標準に基づいており、同様にフィールドバス データ品質規格にも基づいています。

ランタイム時の数値の書式設定は、データ タイプとデータの品質に基づいて設定することができます。

品質データ形式

I/O サーバーは、6 種類の相互に独立したデータ品質状況を報告することができ、InTouch .Quality ドットフィールドに値を割り当てることによりクライアントに送信されます。Quality ドットフィールドの下位の 8 ビット（最下位バイト）は現在、品質 (Q)、サブステータス (S)、しきい値 (L) という 3 つのビット フィールドの形式で定義され、それらのフィールドは、QQSSSLL のように配列されています。クライアントアプリケーションがサーバーと通信できない場合は、.Quality ドットフィールドの値は 0 になります。

.Quality ドットフィールドを使用して I/O サーバーが報告するデータ品質状況を以下の表に示します。

品質状況	10 進数値	16 進数値	最上位バイト XXXXXXXX	最下位バイト QQSSSSL	品質	品質サブ ステータ ス	しきい 値
良好	192	0x00C0	00000000	11000000	Q=3	S=0	L=0
クランプされた上限（範囲外）	86	0x0056	00000000	01010110	Q=1	S=5	L=2
クランプされた下限（範囲外）	85	0x0055	00000000	01010101	Q=1	S=5	L=1
変換不可	64	0x0040	00000000	01000000	Q=1	S=0	L=0
通信に失敗	24	0x0018	00000000	00011000	Q=0	S=6	L=0
ポイントへのアクセス不可	4	0x0004	00000000	00000100	Q=0	S=1	L=0

データ Quality ドットフィールドについて

.Quality ドットフィールドは、データが最後に受信されたときのデータ値の品質を表します。SuiteLink および DDE プロトコルは、I/O サーバーによってデータ変更が行われたときのみ、更新された品質をクライアント（たとえば、WindowViewer）に送ります。そのため、新しいデータ値を受け取ったときだけ、品質が変化したことが確認されます。一部の I/O サーバーは、データに関連付けられた品質が変化したときに、現在のそのデータ値と共に更新された品質を送ることができます。

SuiteLink および DDE プロトコルを使用して、サーバー アイテムの値の品質を直接参照できないことがあります。参照するには、I/O サーバーは Item.Quality を直接サポートする必要があります。このサポートがなければ、アイテムは通知を続けることができず、.Quality ドットフィールド値はずっと 0 のままです。

TestProt I/O サーバー シミュレータは、Item.Quality を直接サポートしていません。また、このシミュレータは、品質メニュー コマンドを使用して品質を修正したときに、新しいデータ値を送り出しません。

I/O アイテムのデータ品質を観察しようとするときに、I/O サーバーが Item.Quality のアドレス指定を直接サポートしていない場合は、サーバー アイテムを観察できるように InTouch I/O タグ変数を定義してから、InTouch タグ変数の .Quality を監視します。タグ変数の制限を超える場合には、スクリプトで IOSetRemoteReferences() 関数を使用して動的に I/O ポイントを調整する方法を検討してください。

SuiteLink および DDE プロトコルは、接続状況や I/O サーバー ステータスにおけるその他の変化をクライアントへ送る品質アイテムとはみなしません。結果として、アイテムの品質が必ずしもデータ サーバーやクライアント/サーバー接続の現在のステータスを表すとは限りません。I/O サーバーのプロセスは停止できますが、.Quality ドットフィールドの値は変化しません。また、通信リンクが切断されても、.Quality フィールドの値が変わらないこともあります。

I/O サーバーの接続を監視するには、DDE または SuiteLink の内部ステータス アイテムを使用します。

.Quality ドットフィールド

.Quality ドットフィールドは、I/O サーバーが提供するデータの品質に関する数値評価を表します。

カテゴリ

タグ

使用法

Tag.Quality

パラメータ

Tag

論理型、整数型、実数型、間接アナログ型、またはメッセージ型の任意のタグ変数タイプ

データタイプ

整数型（読み取り専用）

有効値

値の範囲は 0 ～ 255 です。

参照項目

.QualityLimit、.QualityStatus、.QualitySubstatus

例

```
IF I0Tag.Quality <> 192 THEN
    LogMessage("このデータは適切ではありません。2 バイトの高バイトの品質がゼロであることが想定されています");
    LogMessage(".QualityStatus を確認し、この想定を無効にしてください");
ENDIF;
```

.QualityLimit ドットフィールド

.QualityLimit ドットフィールドは、接続されている I/O サーバーが提供するデータ値の品質しきい値を表します。

カテゴリ

タグ

使用法

Tag.QualityLimit

パラメータ

Tag

論理型、整数型、実数型、間接アナログ型、またはメッセージ型の任意のタグ変数タイプ

データタイプ

整数型（読み取り専用）

有効値

0 = 限度なし

1 = 下限

2 = 上限

3 = 一定

参照項目

.Quality

.QualityLimitString ドットフィールド

.QualityLimitString ドットフィールドは、接続されている I/O サーバーが提供するデータ値の品質しきい値文字列を表します。

カテゴリ

タグ

使用法

Tag.QualityLimitString

パラメータ

Tag

論理型、整数型、実数型、間接アナログ型、またはメッセージ型の任意のタグ変数タイプ

データタイプ

メッセージ型（読み取り専用）

参照項目

.Quality、.QualityLimit

.QualityStatus ドットフィールド

.QualityStatus ドットフィールドは、I/O サーバーが提供するデータ値の整数型の品質ステータスを表します。

カテゴリ

タグ

使用法

Tag.QualityStatus

パラメータ

Tag

論理型、整数型、実数型、間接アナログ型、またはメッセージ型の任意のタグ変数タイプ

データタイプ

整数型（読み取り専用）

有効値（SSSS）

0 = 不良

1 = 不明

3 = 良好

例

```
IF I0Tag.QualityStatus <> 3 THEN
    LogMessage("このデータは適切ではありません。");
ENDIF;
```

参照項目

.Quality、.QualitySubStatus

.QualityStatusString ドットフィールド

.QualityStatusString ドットフィールドは、I/O サーバーが提供するデータ値の品質ステータス文字列を表します。

カテゴリ

タグ

使用法

Tag.QualityStatusString

パラメータ

Tag

論理型、整数型、実数型、間接アナログ型、またはメッセージ型の任意のタグ変数タイプ

データタイプ

メッセージ型（読み取り専用）

参照項目

.QualityStatus、.QualitySubStatus、.Quality

.QualitySubstatus ドットフィールド

.QualitySubstatus ドットフィールドは、I/O サーバーが提供するデータ値の品質サブステータスを表します。

カテゴリ

タグ

使用法

Tag.QualitySubstatus

パラメータ

Tag

論理型、整数型、実数型、間接アナログ型、またはメッセージ型の任意のタグ変数タイプ

データタイプ

整数型（読み取り専用）

有効値（SSSS）と（QQ）

不良品質（QQ=0）のサブステータス（SSSS）。

0 = 非特定

1 = 設定エラー

2 = 未接続

3 = デバイスの故障

4 = センサーの故障

5 = 最後に知られている値

6 = 通信エラー

7 = 動作不能

不明品質（QQ=1）のサブステータス（SSSS）。

0 = 非特定

1 = 最後に知られている値

4 = センサーが不正確

5 = 工学値の超過

6 = サブノーマル

良好品質（QQ=3）のサブステータス（SSSS）。

0 = 非特定

6 = ローカル オーバーライド

参照項目

.QualityStatus、.QualitySubStatus、.Quality

.QualitySubstatusString ドットフィールド

.QualitySubstatusString ドットフィールドは、I/O サーバーが提供するデータ値の品質ステータス文字列を表します。

カテゴリ

タグ

使用法

Tag.QualitySubstatusString

パラメータ

Tag

論理型、整数型、実数型、間接アナログ型、またはメッセージ型の任意のタグ変数タイプ

データタイプ

メッセージ型（読み取り専用）

参照項目

.QualityStatus、.QualitySubstatus、.Quality

ランタイム時の I/O 接続の初期化と再設定

WindowViewer は、InTouch アプリケーションが起動されたときにすべての I/O 通信を初期化します。また、InTouch アプリケーションの動作中に手動で I/O 通信を再起動することもできます。ランタイム中に I/O 通信の初期化と再設定を行うには、WindowViewer のコマンドまたはスクリプトを使用します。

また、デフォルト値に基づいて I/O 接続を再初期化する必要があることを指定することもできます。このオプションを選択すると、アクセス名の再初期化時にはデフォルトの設定が使用されて現在の設定は

無視されます。アクセス名によって I/O 通信を再初期化するには、InTouch アプリケーションのアクセス名を事前に定義しておく必要があります。

コマンドによる I/O 接続の再初期化

WindowViewer の [システム] メニューには、すべての I/O 通信を再初期化したり、特定の I/O 通信を選択するためのコマンドセットが用意されています。

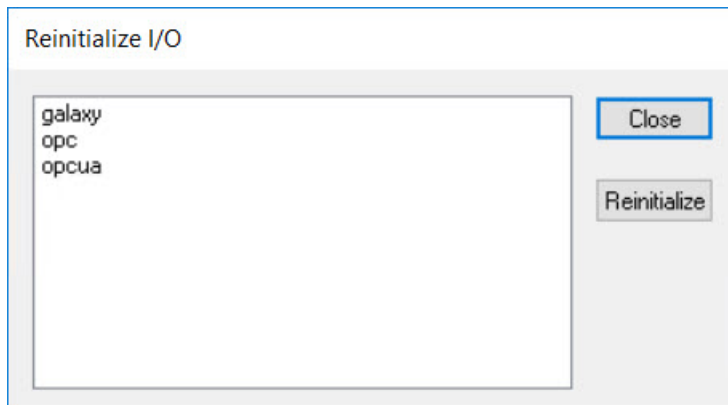
InTouch のデフォルト設定を使用して、アクセス名を再初期化することができます。デフォルトの再初期化を行うとアクセス名は、ノード名、アプリケーション名、およびトピックに現在指定されている値を無視します。アクセス名は、元のアクセス名の設定で再初期化されます。

ランタイム時にすべてのアクセス名を再初期化するには

1. [システム] メニューで、[I/O の再初期化] をクリックします。
2. [すべて再初期化] をクリックします。全アクセス名が再初期化されます。

ランタイム時にアクセス名を選択して再初期化するには

1. [システム] メニューで、[I/O の再初期化] をクリックし、[選択] をクリックします。[I/O の再初期化] ダイアログ ボックスに、アクセス名のリストが表示されます。



2. 再初期化するアクセス名を（1 つまたは複数）クリックして、[再初期化] をクリックします。選択したアクセス名が再初期化されます。

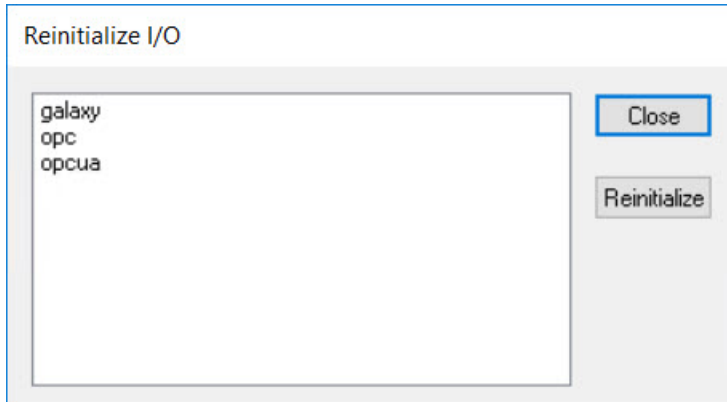
デフォルト設定を使用してアクセス名を再初期化するには

1. WindowMaker 内でアプリケーションを開きます。
1. [ファイル] メニューで、[設定] をポイントし、[WindowViewer] をクリックします。WindowViewer の設定画面が表示されます。
2. [環境設定] タブの [I/O] 領域で、[デフォルトの再初期化] チェック ボックスをオンにします。



3. [OK] をクリックします。
4. WindowViewer でアプリケーションを開きます。

5. [システム] メニューで、[連結中の I/O 通信の再初期化] をクリックして、[選択] をクリックします。[I/O の再初期化] ダイアログ ボックスが表示されます。



6. 再初期化するアクセス名を 1 つまたは複数選択して、[再初期化] をクリックします。ノード名、アプリケーション名、およびトピックの現在の設定は無視されます。アクセス名は、元のアクセス名の設定で再初期化されます。

スクリプトによる I/O 接続の再初期化

以下に示す関数を使用したスクリプトを記述することにより、1 つまたは複数のアクセス名への I/O 接続を再初期化できます。

- **IOReinitAccessName()**
- **IOReinitialize()**
- **IOStartUninitConversations()**

IOReinitAccessName() 関数

IOReinitAccessName() 関数は、特定のアクセス名への I/O 接続を再初期化します

カテゴリ

I/O 通信

構文

```
IOReinitAccessName("AccessName", Default);
```

引数

AccessName

再初期化するアクセス名

Default

Default = 1。WindowMaker から割り当てられた元のデフォルト アクセス名の値を再初期化に使用します。

Default = 0。アクセス名に割り当てられた現在のノード、アプリケーション、トピックの値を再初期化に使用します。

備考

デフォルト設定は、アクセス名設定パネルおよび WindowViewer の設定 (I/O 通信リトライ時間、ローカルのサーバーを起動、デフォルトの再初期化) の設定内容によって決まります。

例

以下の例は、ノード、アプリケーション、およびトピックに割り当てられたデフォルト値を使用して、AccessName1 への I/O 接続を再初期化します。

```
IOReinitAccessName("AccessName1", 1);
```

以下の例は、ノード、アプリケーション、およびトピックに割り当てられた現在の値を使用して、AccessName2 への I/O 接続を再初期化します。

```
IOReinitAccessName("AccessName2", 0);
```

IOReinitialize() 関数

IOReinitialize() 関数は、InTouch アプリケーションに対して定義されたすべてのアクティブな I/O 接続を終了してから再起動します。

カテゴリ

その他

構文

```
IOReinitialize();
```

引数

なし。

備考

IOReinitialize() 関数が実行する処理は、WindowViewer の [システム] メニューの [連結中の I/O 通信の再初期化] コマンドと同じです。

WindowViewer がサービスとして実行している場合、**IOReinitialize()** 関数と WindowViewer の [システム] メニューの [すべて再初期化] コマンドではすべてのアクセス名を初期化することはできません。[連結中の I/O 通信の再初期化] ダイアログボックスにリストされているアクセス名を選択して [再初期化] をクリックすると、選択したアクセス名が再初期化されます。

[連結中の I/O 通信の再初期化] ダイアログボックスへのナビゲーションの詳細については、「[コマンドによる I/O 接続の再初期化](#)」を参照してください。

例

以下の例では、InTouch アプリケーションに対して定義されたすべてのアクティブな I/O 接続を終了してから再起動します。

```
IOReinitialize();
```

IOStartUninitConversations() 関数

WindowViewer で InTouch アプリケーションが起動されると、すべての I/O 通信を開始するための初期化要求が自動的に処理されます。プログラムが I/O サーバーの初期化要求に応答しない場合は、**IOStartUninitConversations()** スクリプト関数を使用して WindowViewer に I/O 通信の起動を再試行させることができます。

カテゴリ

その他

構文

```
IOStartUninitConversations();
```


引数

なし。

備考

IOStartUninitConversations() 関数が実行する処理は、WindowViewer の [システム] メニューの [未連結の I/O 通信の再初期化] コマンドと同じです。

例

以下の例は、WindowViewer アプリケーションに対して定義されたすべての I/O 接続を起動するための初期化要求を再発行するように InTouch に指示します。

```
IOStartUninitConversations();
```

アクセス名によるフェイルオーバー機能の使用

メイン I/O サーバーに通信障害が発生した場合に、自動的にセカンダリ I/O サーバーに切り替わるように InTouch HMI を指定できます。この機能は、I/O フェイルオーバーと呼ばれています。

フェイルオーバーの設定

InTouch アプリケーションがメイン I/O サーバーと通信できなくなった場合に、フェイルオーバー用のセカンダリ I/O サーバーに切り替わるように指定できます。

フェイルオーバーの設定では、フェイルオーバー デッドバンドの指定を行います。フェイルオーバー デッドバンドは、メイン アクセス名からセカンダリ アクセス名に切り替わるまでの秒単位の遅延時間です。InTouch HMI は、式の評価が True の状態または I/O 通信障害の状態がデッドバンドの期間続いた場合にフェイルオーバーをトリガします。フェイルオーバー デッドバンドを 0 か空白に設定した場合、I/O 通信障害が検出されると直ちにフェイルオーバーがトリガされます。

アクセス名に対してフェイルオーバーを設定するには

1. 必要に応じて、WindowViewer を停止します。
2. [ホーム] メニューの [タグ] グループで [アクセス名] をクリックします。
[アクセス名] ダイアログ ボックスが開き、すべての定義済みアクセス名のリストが表示されます。
3. リストからアクセス名を選択してフェイルオーバー サーバーを追加します。
4. [編集] をクリックします。
[アクセス名の追加] が表示されます。
5. [セカンダリ ソース] セクションで [セカンダリ ソースを有効にする] チェック ボックスをクリックします。
6. 以下の手順を実行します。
 - [ノード名] ボックスに、セカンダリ I/O サーバーのノード名を入力します。
 - [アプリケーション名] ボックスに、データの取得元となるセカンダリ I/O サーバー プログラムの名前を入力します。
 - [トピック名] ボックスに、セカンダリ I/O ソースからアクセスするトピック名を入力します。
 - [プロトコル名] 領域で、DDE または SuiteLink のいずれかをセカンダリ I/O サーバー通信プロトコルとして選択します。

- [サーバーへの通知] 領域で、セカンダリ I/O ソースに対して [すべてのアイテムを要求] または [有効アイテムのみ要求] を選択します。

7. [フェイルオーバー] セクションで [ファイルオーバーを有効化] チェック ボックスをクリックします。

8. 以下の手順を実行します。

- オプションのフェイルオーバー式を入力するか、[フェイルオーバー式] ボックス内をダブルクリックしてタグを選択します。フェイルオーバー式の詳細については、[「バックアップ アクセス名へのフェイルオーバーの強制」](#)を参照してください。
- [フェイルオーバー デッドバンド] ボックスで、フェイルオーバー デッドバンドの長さを秒単位で入力します。
- フェイルオーバー状態が解除された後で、セカンダリ アクセス名からメイン アクセス名に戻す切り替え機能を有効にする場合は、[フェイルオーバー条件の解除時にメインへ切り替える] をオンにします。

デフォルトでは、メイン アクセス名に戻す機能は無効です。[フェイルオーバー条件の解除時にメインへ切り替える] をオンにすると、[フェイルオーバー設定] ダイアログ ボックスでの [フェイルバック デッドバンド] オプションの選択が可能になります。

- [フェイルバック デッドバンド] に、フェイルバック デッドバンドの長さを秒単位で入力します。

InTouch HMI は、式の評価と関連するすべての I/O 通信障害の解除がデッドバンドの期間継続するとフェイルバックをトリガします。式が空白または 0 のままの場合は、I/O 通信障害の状態が解除されると直ちにフェイルバックが発生します。

9. [更新] をクリックします。

フェイルオーバー ペアのアクセス名パラメータの編集

フェイルオーバー ペアの一部であるアクセス名パラメータを編集するには、セカンダリ I/O ソースを持つフェイルオーバーにアクセス名が設定されている必要があります。

フェイルオーバー ペアのアクセス名パラメータを編集するには

1. 必要に応じて、WindowViewer を停止します。
2. [ホーム] メニューの [タグ] グループで [アクセス名] をクリックします。

[アクセス名] ダイアログ ボックスが表示されます。

3. アクセス名ペアを選択して、[編集] をクリックします。

[アクセス名の追加] ダイアログ ボックスに、メインとセカンダリのアクセス名のパラメータが表示されます。

4. メインとセカンダリのアクセス名パラメータを変更します。

5. [更新] をクリックします。

アクセス名へのフェイルオーバーの削除

アクセス名へのフェイルオーバーを削除するには、セカンダリ I/O ソースを持つフェイルオーバーにアクセス名が設定されている必要があります。

アクセス名ペアへのフェイルオーバーを削除するには

1. [ホーム] メニューの [タグ] グループで [アクセス名] をクリックします。
2. アクセス名ペアを選択して、[編集] をクリックします。
[アクセス名の追加] ダイアログ ボックスが表示されます。
3. [セカンダリ ソースを有効にする] チェック ボックスをオフにします。
4. [OK] をクリックします。

アクセス名ペアへのフェイルオーバーが無効になります。

バックアップ アクセス名へのフェイルオーバーの強制

フェイルオーバーが発生しなくても、アクセス名のソースをメインとセカンダリ間で手動で切り替えることができます。この機能は、フェイルオーバーの強制と呼ばれています。フェイルオーバーを強制するには、セカンダリ I/O ソースを持つフェイルオーバーにアクセス名が設定されている必要があります。

フェイルオーバー式または **IOForceFailover()** スクリプト関数を使用して、フェイルオーバーを強制することができます。

フェイルオーバー式

[フェイルオーバー設定] セクションに、フェイルオーバーをトリガするタグや式を指定するための [フェイルオーバー式] オプションが表示されます。[フェイルオーバー式] の値として入力されたフェイルオーバー メモリ論理型タグを次の図に示します。

The screenshot shows a configuration window titled "Failover". It contains the following elements:

- A checked checkbox labeled "Enable failover".
- A text input field labeled "Expression" followed by "(optional)".
- A "Deadband:" label followed by a numeric input field containing "4", and two arrow buttons (down and up) to the right, with "sec" at the far right.
- An unchecked checkbox labeled "Switchback to primary when conditions clear".
- Below the second checkbox, a "Deadband:" label followed by a numeric input field containing "0", and two arrow buttons (down and up) to the right, with "sec" at the far right.

フェイルオーバー タグを **True** に設定するなどによりフェイルオーバー式が **True** に設定されると、アクセス名の I/O データ ソースはメイン (**False**) からセカンダリ (**True**) に切り替わります。

IOForceFailover() 関数

IOForceFailover() スクリプト関数は、アクセス名のデータ ソースをメインとセカンダリ間で切り替えます。また、このスクリプト関数が呼び出されるたびに、アクティブな I/O ノードはメイン ノードとセカンダリ ノード間で切り替わります。

IOForceFailover() 関数は通常、ボタンや他のウィンドウ オブジェクトに関連付けられたスクリプトの一部として記述されます。ユーザーがアプリケーション ウィンドウからこのオブジェクトを選択すると、フェイルオーバーが強制されます。再度このオブジェクトをクリックすると、**IOForceFailover** 関数によって I/O 接続は以前のアクティブ I/O ノードに戻されます。

カテゴリ

I/O 通信

構文

```
IOForceFailover("AccessName");
```

引数

AccessName

フェイルオーバーに設定されているアクセス名

例

アクセス名 **Acc1** はメインとセカンダリのデータ ソースを持ち、メインがアクティブです。以下のスクリプトを実行すると、**Acc1** はセカンダリ データ ソースにフェイルオーバーします。

```
IOForceFailOver("Acc1");
```

フェイルオーバー機能の一時的な無効化

アクセス名の I/O ノードをメインとセカンダリ間で切り替えるフェイルオーバーを、手動で無効にすることができます。通常は、InTouch システムのコンポーネントが起動されてから準備完了状態になるまでの短期間に、フェイルオーバーの一時的な無効化が利用されます。コンポーネントが安定化した後で、フェイルオーバーの切り替え機能を回復できます。

アクセス名へのフェイルオーバーを無効化するには、セカンダリ I/O ソースを持つフェイルオーバーにアクセス名が設定されている必要があります。

手動でフェイルオーバーの切り替え機能を無効にするには、2 つの方法があります。

- [フェイルオーバー] セクションで [フェイルオーバー不可] オプションを選択します。
- **IODisableFailover()** 関数を記述したスクリプトを実行します。

[フェイルオーバー不可] 設定オプション

アクセス名がプライマリとセカンダリ I/O ノードの間で切り替わらないようにするには、[フェイルオーバー設定] セクションで、[フェイルオーバーを有効化] オプションをオフにします。

Failover
☒ Enable failover

Expression (optional)

Deadband: 4 sec

☐ Switchback to primary when conditions clear

Deadband: 0 sec

[フェイルオーバー不可] オプションを有効に設定するには、アクセス名の定義を編集しなければなりません。このオプションがアクセス名に対して有効である間は、フェイルオーバーは無効です。

IODisableFailover() スクリプト関数

IODisableFailover() 関数をスクリプトに使用して、特定のアクセス名に対するフェイルオーバーを無効にできます。**IODisableFailover()** は、**IOForceFailover()** スクリプト関数による方法を除く、すべてのフェイルオーバー手段による切り替えを無効にします。

カテゴリ

I/O 通信

構文

```
IODisableFailover("AccessName",Option);
```

引数

AccessName

フェイルオーバーに設定されているアクセス名

Option

1 = フェイルオーバー不可

0 = フェイルオーバー可

備考

アクセス名はリテラル文字列で指定するか、他の InTouch タグ変数または関数が返す文字列の値で指定できます。

例

以下の例は、アクセス名 **ModbusPLC1** に対するフェイルオーバーを無効にします。

```
IODisableFailover ("ModbusPLC1",1)
```

以下の例は、アクセス名 **ModbusPLC1** に対するフェイルオーバーを有効にします。

```
IODisableFailover ("ModbusPLC1",0)
```

スクリプトによるフェイルオーバー ペア関連情報の取得

スクリプトを記述して、アクセス名のメイン、セカンダリ、およびアクティブな I/O ソースのステータスを返す関数を使用することができます。ユーザーは通常、フェイルオーバーの強制を実行する前にアクセス名のセカンダリ I/O ソースのステータスを確認するスクリプトを実行します。

アクセス名情報を返すスクリプトを記述するには、セカンダリ I/O ソースを持つフェイルオーバーにアクセス名が設定されている必要があります。

IOGetAccessNameStatus() 関数

IOGetAccessNameStatus() スクリプト関数は、アクセス名のメイン、セカンダリ、またはアクティブな I/O ソースの接続ステータスを示す整数値を返します。

IOGetAccessNameStatus() の戻り値は、通常は整数型タグ変数に関連付けられます。このタグ変数の値により、アクティブ、メイン、およびセカンダリの、アクセス名の I/O ソースのステータスをユーザーに示す論理値表示アニメーションリンクを動作させることができます。

カテゴリ

その他

構文

```
Result=IOGetAccessNameStatus("AccessName", Mode);
```

引数

AccessName

ステータスを返す既存のアクセス名。

Mode

この引数に割り当てる値に応じて、どのフェイルオーバー ペアのアクセス名に対して現在のステータスに関するクエリーを実行するかが決まります。

0 - アクティブなアクセス名 I/O ソースのステータス

1 - アクセス名のメイン I/O ソースのステータス

2 - アクセス名のセカンダリ I/O ソースのステータス

結果

返り値	説明
-1	アクセス名に設定エラーがあります。アクセス名が存在しないか、またはアクセス名にセカンダリ I/O ソースが定義されていません。
0	要求した I/O ソースとの接続に失敗しました。
1	要求した I/O ソースとの接続に成功しました。

備考

IOGetAccessNameStatus() 関数は通常、現在非アクティブなセカンダリ I/O ソースのステータスを確認するスクリプトに使用されます。ユーザーはこのスクリプトを実行して、フェイルオーバーの強制を実行する前にセカンダリ接続のステータスを確認します。

例

以下の例は、アクセス名 **ModbusPLC1** のセカンダリ I/O ソースのステータスを返します。返り値は **ANStatus** タグ変数に関連付けられます。

```
ANStatus = IOGetAccessNameStatus ("ModbusPLC1",2)
```

IOGetActiveSourceName() 関数

IOGetActiveSourceName() スクリプト関数は、アクセス名が現在プライマリ データ ソースまたはセカンダリ データ ソースを使用しているかどうかを返します。

IOGetActiveSourceName() 関数は通常、ボタンや他のウィンドウ オブジェクトに関連付けられたスクリプトに記述されます。ユーザーは、アプリケーション ウィンドウからこのオブジェクトを選択して、アプリケーションの I/O Servers のステータスを要求します。

カテゴリ

その他

構文

```
Result=IOGetActiveSourceName("AccessName");
```

引数

AccessName

ソース名を返す既存のアクセス名

備考

IOGetActiveSourceName() は、アクセス名のメインまたはセカンダリのノードがアクティブにポーリングされているかどうかを示す文字列を返します。**IOGetActiveSourceName()** 関数の返り値を以下に示します。

メイン	アクセス名のメイン ノードがアクティブにポーリングされています。
セカンダリ	アクセス名のセカンダリ ノード、つまりフェイルオーバー ノードがアクティブにポーリングされています。
Null	アクセス名のメイン ノードおよびセカンダリ ノードとも非アクティブです。

例

以下の例では、アクセス名 **ModbusPLC1** の現在のアクティブ ノードを示す返り値（メイン、セカンダリ、または Null）が、メッセージ型タグ変数 **ActiveServer** に割り当てられます。

```
ActiveServer = IOGetActiveSourceName ("ModbusPLC1");
```

I/O 接続ステータスの監視

WindowViewer には **IOStatus** というビルトインのトピックが用意されており、PLC と通信している I/O サーバーと InTouch アプリケーション間の特定の I/O 通信のステータスを監視できます。

注: InTouch の 7.0 以前のバージョンでは、トピック名は **DDEStatus** でした。

I/O 通信を監視するように **IOStatus** トピックを設定することができます。

IOStatus トピック名の使用

IOStatus トピックを準備して、WindowViewer と I/O サーバー間の I/O 通信を監視できます。この例では、WindowViewer は、I/O サーバー内でトピック名として **PLC1** に定義されている PLC に対するシミュレーション I/O サーバーと通信します。

注記: Simulation Server サーバーは、トレーニング用ツールとして使用できる汎用 **DAServer** で、**c:\program files\common files\ArchestrA** フォルダ内にあります。

I/O 通信のステータスを監視するには

1. WindowMaker でアプリケーションを開きます。
2. タグ名ディクショナリを開きます。
3. I/O 論理型タグを作成します。

IOStatus を使用して I/O 通信を監視する場合、監視対象のアクセス名に対して少なくとも 1 つの I/O 型タグを定義する必要があります。

The 'Tagname Dictionary' dialog box has tabs for Main, Details (selected), Alarms, Details & Alarms, and Members. It contains buttons for New, Restore, Delete, Save, and navigation arrows. The 'Tagname' field is set to 'PLC1', 'Type' is 'I/O Discrete', 'Group' is '\$System', and 'AccessLevel' is selected in the 'Comment' field. There are checkboxes for 'Log Data', 'Log Events', and 'Retentive Value'.

4. トピック名として **IOStatus** を定義しているアクセス名の定義にタグを割り当てるために、**[アクセス名]** をクリックします。

The 'Access Names' dialog box shows a list of access names: Galaxy, IOStatus, **PLC1** (highlighted), TankFarm, and TankFarm1. It includes buttons for Close, Add..., Modify..., and Delete.

この時点で、**PLC1** のアクセス名の定義は存在しています。

5. **PLC1** を選択して **[修正]** をクリックします。

The 'Modify Access Name' dialog box shows fields for Access (PLC1), Node Name (Demo), Application Name (demo), and Topic Name. It has buttons for OK, Cancel, and Failover. There are radio buttons for 'Which protocol to use' (DDE, SuiteLink selected, Message Exchange) and 'When to advise server' (Advise all items, Advise only active items selected). There is also a checkbox for 'Enable Secondary Source'.

この例では、タグとトピック名が同じであるため、正しいトピック名を含むアクセス名を容易に見つけ出すことができます。

6. **[キャンセル]** をクリックしてダイアログ ボックスを閉じ、**[アクセス名]** ダイアログ ボックスに戻ります。
7. **[追加]** をクリックします。

[アクセス名の追加] ダイアログ ボックスが表示されます。

8. 以下の手順を実行します。

- a. [アクセス名] ボックスに、IOStatus と入力します。
 - b. [アプリケーション名] ボックスに、**View** と入力します。これにより WindowViewer でステータスが監視されます。
 - c. [トピック名] ボックスに、InTouch の内部トピックとして **IOStatus** と入力します。
 - d. [有効アイテムのみ要求] を選択します。
9. [OK] をクリックして、ダイアログ ボックスを閉じます。[アクセス名] ダイアログ ボックスが再表示され、新しいアクセス名、IOStatus がリストに表示されます。

10. [閉じる] をクリックしてダイアログ ボックスを閉じると、新しいアクセス名が I/O 論理型タグに関連付けられます。

[アイテム名] ボックスに、監視する実際のトピック名のアクセス名を入力します。

11. タグがトピック名と同じであるため、[タグ名をアイテム名として使用] を選択すると、トピック名がアイテム名として自動的に入力されます。

注記: ビルトインのトピック IOStatus (InTouch 7.0 以前のバージョンでは DDEStatus) を使用して I/O 通信を監視する場合、[アクセス名] ボックスに入力する名前が常にアイテム名としても使用されます。

アイテム名は、最大 254 文字の文字列です。アイテム名の長さは、トピック名、アイテム名、および 1 の区切り文字の合計として計算されます。

Excel での IOStatus トピックの使用

Excel のスプレッドシート セルの式に同じ情報を入力することにより、I/O ステータスを監視できます。たとえば、前記手順の説明と同じトピックを監視するには、セルに以下の式を入力します。

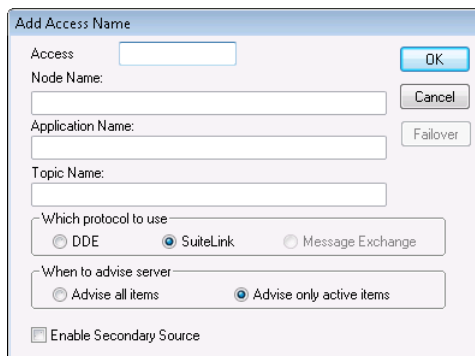
```
=view|IOStatus!'PLC1'
```

I/O 通信ステータスの監視

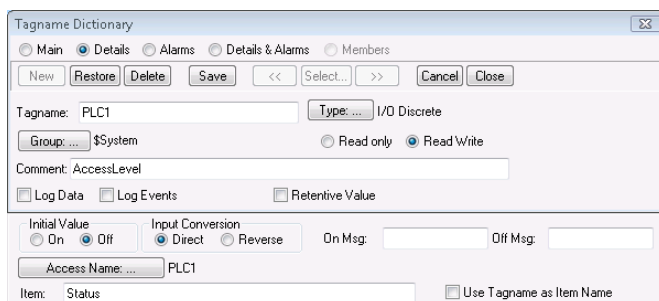
使用されている各トピック名に対してビルトインの論理型アイテム **Status** を使用して、I/O サーバー プログラムとの通信のステータスを監視できます。通信障害が発生すると、**Status** アイテムは **0** に設定されます。I/O サーバー プログラムとの通信が正常の場合は、**Status** アイテムは **1** に設定されます。

注記: IOStatus アイテムを使用してトピックのステータスを監視する場合、監視対象のトピックに対して少なくとも 1 つの I/O ポイントがアクティブであることが必要です。

タグ変数を定義して、アイテム名に **Status** という語を使用してデバイス用に設定されたトピックをそのタグ変数に関連付けることにより、InTouch HMI からサーバー通信のステータスを読み取ることができます。たとえば、WindowViewer が Modbus DAServer を使用して PLC と通信している場合、アクセス名の定義は以下の例に類似したものになります。



トピック PLC1 へのすべての通信ステータスを監視するには、タグ変数の定義を次のように作成します。



Excel 使用時には、以下の式をセルに入力して PLC 通信ステータスを読み出すことができます。

```
=SIMULATE|PLC1!'STATUS'
```

他のアプリケーションからの InTouch タグ変数データへのアクセス

他のアプリケーションが InTouch HMI のデータ値を要求する場合、3 つの I/O アドレスのアイテムを認識する必要があります。以下に InTouch の I/O アドレス 形式について説明します。

VIEW (アプリケーション名) は、データ要素を含む InTouch のランタイム プログラムを識別します。

TAGNAME (トピック名) は、タグ変数の読み取り／書き込み時に常に使用する名前です。

ActualTagname（アイテム名）は、InTouch タグ変数ディクショナリでアイテムに対して定義された実際のタグ変数です。

たとえば、Excel から InTouch HMI のデータ値にアクセスするには、データを書き込むセルに DDE リモートトリファレンスの式を指定します。

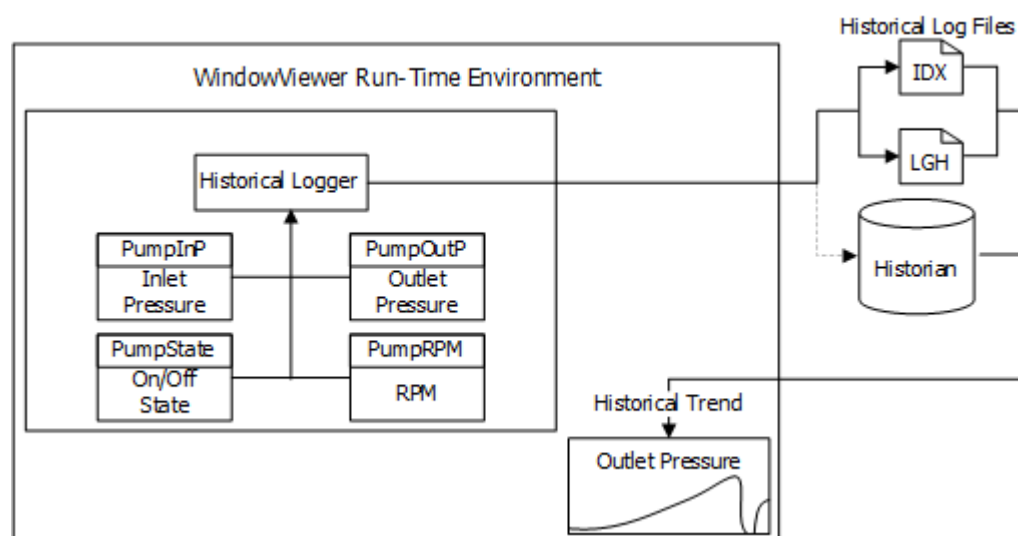
=VIEW|TAGNAME!'ActualTag_name'

タグ値の記録

InTouch アプリケーションの実行中に、タグをログ記録し、永続的に保存できます。このログデータを使用すると、一定期間に行われるプラントプロセスのいくつかの側面を表示する履歴トレンドグラフを作成できます。

注記: 大きなアプリケーションで InTouch の履歴データを保存する場合や詳細なレポートを作成する場合は、**Historian Server** を使用する必要があります。履歴ログ記録の設定の詳細については、**Historian Server** のマニュアルを参照してください。

以下の図は、履歴ログファイルおよび **Historian Server** データベースに保存されたポンプからのタグデータを示しています。InTouch 履歴ロガーは、タグの値が変化して、指定したログのデッドバンド範囲を超えるたびにログエントリを書き込みます。



履歴ログファイルにデータを保存すると、InTouch HMI によってログファイルが作成されます。1つのファイルには、専用の形式で保存されるログデータが含まれます。もう1つのファイルは、そのデータへのインデックスです。

この2つのログファイルには、以下の形式で名前が割り当てられます。

YYMMDD00.LGH および YYMMDD00.IDX

構文の要素を以下に示します。

- YY** ログファイルが作成された年の最後の2桁
- MM** ファイルが作成された月の2桁の数（01～12）
- DD** ファイルが作成された日付の2桁の数（01～31）

00 ログ ファイル名の定数 00

1 日のログ サイクルは深夜に開始および終了します。履歴ロガーは、深夜に最後のエントリをアクティブなログ ファイルに書き込み、そのファイルをアーカイブします。翌日には、2 つの新しいファイルが作成され、データはそのファイルにログ記録されます。

ログ ファイルは指定された日数の間保存されます。保存期間を過ぎたログ ファイルは削除されます。ログ ファイルを保持するための日数の設定の詳細については、「[一般的なログ プロパティの設定 - 履歴ログ ファイル](#)」を参照してください。

履歴ログの設定

InTouch アプリケーションで履歴ログを設定するには、主に 3 つの作業を行う必要があります。

- 履歴ログのタグ変数を設定する
- InTouch アプリケーションの一般的なログ プロパティを設定する
 - タグ変数値を InTouch 履歴ログ ファイルに保存するためのプロパティを設定する
 - または、タグ値を Historian Server に保存するためのプロパティを設定する
- (オプション) 履歴ログの頻度を設定する

注: LGH ファイルおよび Historian Server へのログ記録を設定できます。

履歴ログのタグの設定

履歴ログのタグは、タグ名ディクショナリから選択します。選択したタグの値が変更されると、履歴ロガーは、各タグのログ デッドバンドとその現在の値に基づいてエントリをログに書き込むかどうかを決定します。

タグをログ記録有効からログ記録無効に変更すると、そのタグに関連付けられたデータはログ ファイルに保存されなくなります。ログ記録を再び有効にすると、ログ記録は再開されます。ただし、履歴トレンドには、ログ記録が無効になっていた間のギャップが示されます。

WindowViewer がアプリケーションを実行している間は、タグのログ記録への変更は無視されます。タグに対してログ記録を変更しても、稼動しているアプリケーションを停止して再起動するまでは有効になりません。

各タグのログ記録は、タグ名ディクショナリから設定します。

タグの履歴ログを有効にするには:

1. 必要であれば、WindowViewer がアプリケーションを実行しないようにします。
2. WindowMaker でアプリケーションを開きます。
3. タグ名ディクショナリを開きます。
4. タグ名ディクショナリ リストから、データをログに記録するタグを選択します。
5. [データ ログ] チェック ボックスをオンにします。

Tagname Dictionary

☐ Main
 ☒ Details
 ☐ Alarms
 ☐ Details & Alarms
 ☐ Members

New Restore Delete Save << Select... >> Cancel Close

Tagname: Tlag6 Type: Memory Integer ☐ Local Tag

Group: \$System ☐ Read only ☒ Read Write

Comment: AccessLevel

☒ Log Data ☒ Log Events Priority: 999 ☐ Retentive Value ☐ Retentive Parameters

Initial Value: 0 Min Value: 0 Deadband: 0

Eng Units: Max Value: 100 Log Deadband: 0

【タグ名ディクショナリ】ダイアログボックスには、ログ記録に密接に関連付けられたその他のタグの属性が含まれています。

- **【ログデッドバンド】**は、超過するとタグの値がログファイルに書き込まれる工学値のしきい値を設定します。デッドバンドの範囲外の新しい値だけが、ログファイルに書き込まれます。デッドバンド範囲内でのわずかな値の変化は無視されます。
- **【工学値最小値】**プロパティおよび**【工学値最大値】**プロパティは、クランピングされた生データ値を工学値の範囲内で調整します。**【工学値最小値】**プロパティおよび**【工学値最大値】**プロパティでは、調整値の上限と下限を設定します。

工学値最小値および工学値最大値は、トレンドに表示されるログ値の範囲の境界を決定します。デフォルトでは、InTouch 履歴トレンドにはログデータが工学値範囲の 0 ～ 100% で表示されます。

6. **【保存】** をクリックします。
7. これらの手順を繰り返して、ログ記録するデータが含まれる各タグのログ記録を有効にします。
8. 完了したら、**【閉じる】** をクリックして、タグ名ディクショナリを閉じます。

一般的なログプロパティの設定 - 履歴ログファイル

選択したアプリケーションに適用する一般的なログ記録プロパティを設定できます。

履歴ログを設定するには

1. 必要であれば、WindowViewer で稼動している InTouch アプリケーションを閉じます。
2. WindowMaker を開きます。
3. **【ファイル】** メニューの **【設定】** をクリックし、**【履歴ログ】** をクリックします。
【履歴ログ】 設定画面が表示されます。

Historical logging

Historical logging Historian logging Printing control

☒ Enable historical logging

Keep log files for: 0 days

☒ Store log files in application directory

☐ Store log files in specific directory

Directory
C:\Users\Public\Wonderware\Intouch Application

Logging node

4. [履歴ログを有効化] チェック ボックスをオンにします。
5. [ログ ファイルの保存期間] ボックスに、ログ ファイルを現在の日付から保持する日数を入力します。

ログ ファイルは、現在の日付から指定した保存期間だけ保存されます。保存期間を過ぎたログ ファイルは削除されます。この値を 0 に設定すると、すべてのログ ファイルは無期限に保存されます。

例:

保存期間を 5 日間に設定し、月の最初の日からログ記録を開始しました。その月の 7 日の時点で、その前の 5 日分とその日の分 (02 ~ 07) のログ ファイルが保持されています。月の最初の日に作成されたログ ファイルは削除されます。

ログ データの保存日数を設定する際は、ディスク容量の使用率を考慮してください。ハードディスクの空き容量が不足すると、履歴ログは停止します。ログ記録を再開するには、ディスク容量を解放する必要があります。

6. ログ ファイルを保存するフォルダの場所を選択します。

注記: ログ データを保存するファイルのフォルダ パスと名前は最大 55 文字です。

[アプリケーションディレクトリにログ ファイルを保存] を選択すると、ログ データを作成している InTouch アプリケーションと同じフォルダにログ ファイルを保存できます。

[特定のディレクトリにログ ファイルを保存] を選択すると、ログ ファイルの保存先として別のフォルダを指定できます。ログ ファイルを保存するフォルダは、以下のように指定できます。

- Windows のフォルダ パス (C:\History Log Files など)
- UNC (汎用名前付け規則) パス (\\node\share\directory など)

ログ ファイルを分散ノードに保存している場合は、UNC パスでディレクトリを指定する必要があります。

マスター アプリケーション ノードのアプリケーション ディレクトリに履歴データが書き込まれるように設定されている場合、すべての NAD ノードは、履歴データをマスター アプリケーションに書き込もうとします。この問題を避けるには、各 NAD ノードで、マスター アプリケーション ノードではなくローカル ディレクトリに書き込まれるように履歴データを設定します。

7. [ロギング ノード] ボックスに、ログ データを作成している InTouch アプリケーションを実行しているコンピュータのノード名を入力します。

8. [保存] をクリックします。

ログ記録設定の変更内容は、すぐに有効になります。次回アプリケーションを起動すると、ログ記録が開始されます。

トレンドの印刷の詳細については、「[ランタイム中のトレンド印刷](#)」を参照してください。

一般的なログ プロパティの設定 - Historian への保存

選択したアプリケーションに適用する一般的なログ記録プロパティを設定できます。

Historian への保存を設定するには

1. 必要であれば、WindowViewer で稼動している InTouch アプリケーションを閉じます。
2. WindowMaker を開きます。
3. [ファイル] メニューの [設定] をクリックし、[履歴ログ] をクリックします。
[履歴ログ] 設定画面が表示されます。
4. [Historian ログ] タブを選択します。

5. [Historian へのストレージを有効にする] チェック ボックスをオンにします。
6. [Historian ノード名] ボックスに、Historian Server を実行しているコンピュータのノード名を入力します。IP アドレスを指定することもできます。サーバーが同じマシンにインストールされている場合は、localhost を使用できます。使用できる特殊文字は、ピリオド (.), アンダースコア、(_), およびハイフン (-) だけです。
7. ストア フォワードに関連するファイルを保存するローカル フォルダの場所へのパスを [履歴ストア フォワード ディレクトリ] ボックスに入力します。これらのファイルを使用すると、Historian Server への接続が失われた場合に履歴データを一時的に保存できます。接続が確立されたら、Historian Server は、このストア フォワード ディレクトリのファイルと同期し、すべての情報が維持されます。

8. [アラームとイベントをログ] を選択してアラームとイベントのログを有効にします。[アラーム クエリー] テキスト ボックスでアラーム クエリーを指定します。

注記: 異なる InTouch アプリケーションがタグ データを同じ Historian Server に保存し、同じタグ名を使用した場合、そのようなシナリオは InTouch アプリケーションで検出されず、Historian データが重複することがあります。複数のアプリケーションを区別するには、一意の接頭辞または接尾辞を使用します。詳細については、「[追加文字列の設定](#)」を参照してください。

9. [保存] をクリックします。

ログ記録設定の変更内容は、すぐに有効になります。次回アプリケーションを起動すると、ログ記録が開始されます。

トレンドの印刷の詳細については、「[ランタイム中のトレンド印刷](#)」を参照してください。

追加文字列の設定

同じアプリケーションを複数のノードで使用し、同じ Historian Server に接続するシナリオでは、一意の接尾辞または接頭辞を使用してノードの異なるタグを区別できます。文字列は、WindowMaker から設定するか、各ノードの 'dhistcfg.ini' ファイルを手動で編集することによって設定することができます。

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[履歴ログ] をクリックします。[履歴ログ] 設定画面で [Historian ログ] タブを選択します。
3. [Historian へのストレージを有効にする] チェック ボックスをオンにします。
4. [常に一意の文字列を追加する] チェック ボックスをオンにします。
5. [一意の接頭辞を追加] または [一意の接尾辞を追加] を選択します。
6. [一意の文字列] フィールドに文字列を入力します。文字列の最大長は 6 文字まで指定できます。
7. [OK] をクリックします。

手動での .ini ファイルの追加文字列の編集

追加文字列は、各ノードのアプリケーション フォルダ内にある dhistcfg.ini を変更して手動で更新できます。

1. アプリケーション フォルダに移動します。
2. dhistcfg.ini ファイルを編集します。

例 1: "aa" という一意の接頭文字列の割り当て

```
szHistorianNode=<MachineName>
bStorageLoggingEnabled=1
bAffixEnabled=1
bPrefixEnabled=1
bSuffixEnabled=0
szHistUniqueString=aa
```

例 2: "xx" という一意の接尾文字列の割り当て

```
szHistorianNode=<MachineName>
bStorageLoggingEnabled=1
bAffixEnabled=1
bPrefixEnabled=0
bSuffixEnabled=1
szHistUniqueString=xx
```

3. .ini ファイルを保存して閉じます。

変更内容は WindowViewer を再起動した後に適用されます。

詳細設定の構成

〔詳細設定〕 セクションを更新して、Historian の接続に関連する設定を指定できます。

1. 〔接続〕 では〔TCP ポート〕を指定できます。Historian Server ノードの TCP ポートには、履歴データが送信されます。TCP ポートは、Historian Server のインストール時に設定されます。デフォルトポートは 32565 です。デフォルトの Historian TCP ポートは設定可能で、データ レプリケーションに加えて、リモート IDAS バージョン 2023 R2 以降（gRPC 通信を暗示）との通信に使用されます。

注記: WCF 通信をサポートする従来の Historian TCP ポート 32568 はデータ レプリケーションと Historian バージョン 2023 以前とのリモート IDAS 通信に使用され、以前のバージョンとの互換性の目的で維持されています。InTouch HMI 2023 以前で作成されたアプリケーションのデフォルトポート番号は 32568 です。アップグレード中、移行したアプリケーションの TCP ポートは自動的に 32565 に更新されます。ただし、ポート番号を手動で変更した場合、変更したポート番号は、アップグレードしたアプリケーションで維持されます。

2. 〔帯域幅の最適化〕 セクションでは、〔圧縮を有効にする〕 チェック ボックスをオンにすることができます。このチェック ボックスをオンにすると、以下を指定できます。
 - **帯域幅制限:** Historian と通信するとき HCAL が使用するネットワーク通信の帯域幅の制限を kbps 単位で指定します。0 の値を設定すると、この機能が無効になります（デフォルト）。帯域幅のニーズの推定に関する詳細については、『System Platform インストール ガイド』に記載されている Historian Server のパフォーマンスおよびサイズ設定に関する推奨事項を参照してください。指定できる値は、0 kbps ～ 65535 kbps です。
 - **未完了のパケットを送信するまでの待機時間:** Historian Server に送信するまでに、部分的に充填された Historian クライアントアクセスレイヤー（HCAL）バッファを維持する最大時間をミリ秒で指定します。いっぱいになったバッファは、このフィールドで設定された値に関係なく直ちに送信されます。変化する頻度が高いデータの場合、この設定は関係ありません。データの変更頻度が遅く、ネットワーク帯域幅が制限されている場合、この値を増やしてネットワーク接続の頻度を下げることを検討してください。指定できる値は、1000 ミリ秒 ～ 30000 ミリ秒です。
3. 〔データ管理〕 セクションでは、以下のオプションを指定できます。
 - **前処理バッファ サイズ:** Historian クライアントアクセスレイヤー（HCAL）で使用するすべてのバッファの合計サイズ（MB）。デフォルト（最小）値は 8 です。非常に高いデータ バーストが

ある場合、この値を増やしてください。値が低すぎると、データ損失が発生し、バッファのオーバーフローに関連するエラーメッセージが **Logger** に表示されます。この値は、**10 MB** 単位で増加できます。指定できる値は **8 MB ~ 65535 MB** です。

- **ストア フォワードのしきい値:** HCAL 保存転送ディスクで維持する空き容量のサイズ (MB)。指定した容量は、保存転送中に使用されません。負の値を指定することはできません。指定できる値は **0 MB ~ 65535 MB** です。
- **ストア フォワードの最短時間:** HCAL が保存転送モードで機能する最小時間 (秒)。HCAL が保存転送モードで機能する条件が存在しなくなっても、この時間、HCAL は保存転送モードで機能します。指定できる値は、**30 秒 ~ 3600 秒** です。
- **[可能な限り早急に再接続し、切断をマークしない]** チェック ボックスをオンにします。Application Server と Historian の間の通信が切断されたときにトレンドを表示する方法を指定します。これは、通信が回復した後の履歴データのトレンドの表示方法には影響しません。TRUE の場合、切断中のクライアント側のトレンドにギャップは表示されません。切断中のギャップは切断前に取得された値で補完されます。FALSE の場合、切断中のクライアント側のトレンドにギャップが表示されます。切断時に NULL 値が挿入されてギャップが作成されます。どちらの場合でも、再接続後は保存転送データでギャップが埋められます。
- **[信頼された接続を使用する]** チェック ボックスをオンにします。Historian Server とのセキュアな通信のみが保証されます。
[信頼された接続を使用する] チェック ボックスは、すべての新規アプリケーションと移行したアプリケーションに対してデフォルトで有効になります。
 - **[信頼された接続を使用する]** チェック ボックスをオンにすると、信頼された接続のみが Historian Server と通信できます。クライアントと Historian ノードの間に共通の証明書がない場合、接続は失敗します。
 - **[信頼された接続を使用する]** チェック ボックスがオフの場合、ロガーに SSL エラーがある場合でも Historian との接続が確立されます。

注記: コンフィグレータで **SMS** を設定し、このチェック ボックスをオンにしてセキュアな接続を確立することが推奨されます。

4. [保存] をクリックします。

WCF および gRPC 通信をサポートするサービス

Historian のバージョンが **2023 R2** 以降の場合、下位互換性をサポートするために以下のサービスが実行されていることを確認してください。

- **aahClientAccessPoint** (WCF 通信) - バージョン **2023** 以前のクライアントをサポートします。
- **aahClientAccessPointNG** (gRPC 通信) - バージョン **2023 R2** 以降のクライアントをサポートします。

履歴ログの頻度の制御

オプションで、エントリをログ記録する 2 つの条件を指定できます。

- タグ変数値がログ デッドバンド値より大きい工学単位値によって変更されると、すぐにログ エントリが書き込まれます。
- ログ記録されるすべてのタグ変数の現在の値が、固定間隔で書き込まれます。また、現在の値に関わらず、記録されるすべてのタグ変数のエントリが書き込まれます。デフォルトの固定間隔は **60 分** です。

デフォルトのログ記録間隔をそのまま使用するか、間隔を変更するには、intouch.ini ファイルに 2 つのパラメータを追加します。

- **ForceLogging**

ForceLogging は、ログ記録間隔の長さを分単位で指定します。**ForceLogging** の値は、5 ～ 120 分の間で設定できます。デフォルトは、**ForceLogging=60** です。

- **ForceLogCurrentValue**

ForceLogCurrentValue は、最後にログに記録された値のログ デッドバンド内にあっても一定のログ間隔で現在のタグ変数値をログするよう指定します。0 に設定すると、最後にログに記録された値が再度ログに記録されます。デフォルトは、**ForceLogging=0** です。

以下の例は、2 つのログ記録パラメータが含まれる intouch.ini ファイルの例を示しています。

```
WinFullScreen=1
WinWidth=808
AlarmBufferSize=5000
ForceLogging=5
ForceLogCurrentValue=1
```

この例では、タグ変数は 5 分ごとに履歴ログ ファイルに書き込まれます。

履歴ログの頻度を変更するには

1. WindowMaker と WindowViewer を閉じます。
2. intouch.ini ファイルを InTouch アプリケーションと同じフォルダから見つけます。
3. intouch.ini ファイルを編集します。
4. **ForceLogging** ステートメントに 5 ～ 120 の間の値を挿入します。
5. **ForceLogCurrentValue=1** ステートメントを挿入します。
6. 変更を保存し、intouch.ini ファイルを閉じます。
7. WindowViewer を再起動します。

ランタイムでの履歴ログの開始と停止

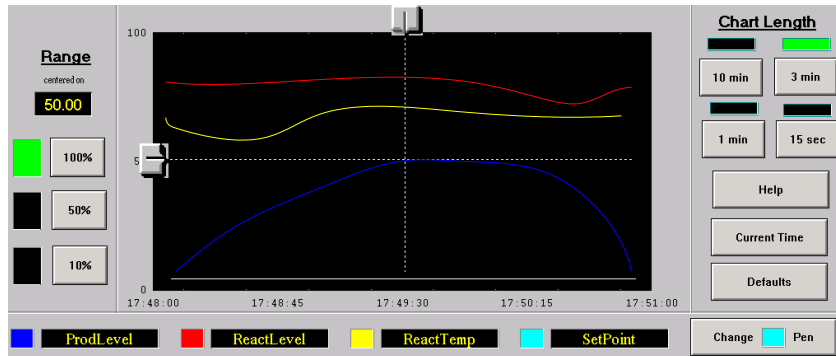
アプリケーションの稼働中は、WindowViewer の [システム] メニューのコマンドを実行して、履歴ログを手動で停止したり、再起動したりすることができます。

- [履歴データ ログの停止] コマンドを実行すると、アプリケーションの現在のセッション中はログ記録を停止します。アプリケーションを手動で再起動するまで、現在のセッション中はログ記録は停止したままとなります。
- [履歴データ ログの停止] オプションを使用して手動で停止した後で、[履歴データ ログの再起動] コマンドを実行すると、ログ記録が再開されます。

また、アプリケーションにボタンを追加し、**\$HistoricalLogging** システム タグ変数が含まれる QuickScript を記述することによっても、履歴ログを起動したり、停止したりすることができます。**\$HistoricalLogging** に値 1 が割り当てられると、ログ記録が開始されます。**\$HistoricalLogging** に値 0 が割り当てられると、ログ記録が停止します。**\$HistoricalLogging** システム タグ変数の詳細については、「[システム タグ変数](#)」を参照してください。

タグデータのトレンド

トレンドを作成して、InTouch アプリケーションから収集されたデータを自動表示できます。WindowMaker には、履歴トレンドやリアルタイムトレンドを作成できるユーティリティとウィザードのセットが用意されています。以下の図は、InTouch 平均/散布トレンドの例を示しています。



また、複数のトレンドコントロールを使用することもできます。トレンドコントロールを使用すれば、トレンドに表示するデータやトレンドのデータ表示方式を選択できます。

リアルタイムトレンドや履歴トレンドを設定できます。どちらのトレンドタイプにも、トレンドデータの収集間隔と表示方式の設定オプションが用意されています。

InTouch トレンドのタイプ

履歴トレンドには、過去に収集されたログデータと InTouch データレポジトリに保存されたログデータが表示されます。

分散履歴システムを使用すれば、アクセス可能なネットワークノードにあるすべての InTouch 履歴ログファイルから履歴データを検索することができます。履歴トレンドの検索機能を拡張する分散履歴システムでは、リモートログデータベースまでカバーされます。

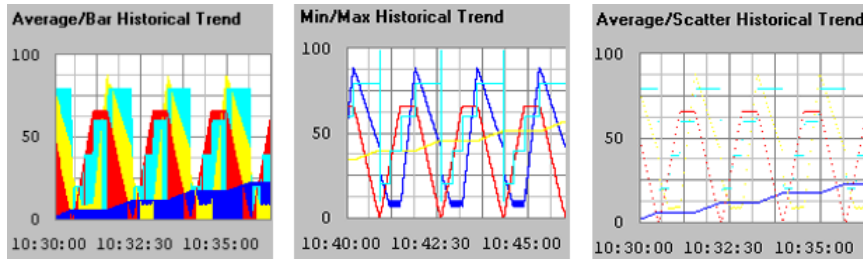
リアルタイムトレンドは、比較的短い間隔で生じるデータの表示を連続して更新します。WindowMaker のリアルタイムトレンドツールを使用することにより、ウィンドウ内にトレンドオブジェクトを作成できます。またオプションの 16 ペントレンドツールをインストールすれば、最大で 16 のタグ変数のデータを表示するリアルタイムトレンドを作成することもできます。

履歴トレンドの理解

履歴トレンドには、過去のデータの連続したセグメントが表示されます。リアルタイムトレンドとは異なり、履歴トレンドはスクリプトまたはユーザー操作によってのみ更新されます。

履歴トレンドには、最大で 8 タグ変数のデータのグラフが表示できます。トレンドペンに対してタグ変数を割り当てることにより、履歴トレンドに表示するデータが指定されます。

以下の図に 3 タイプの InTouch 履歴トレンドを示します。



- 平均／棒履歴トレンドには、一定時間ごとのデータ ポイントの平均値が棒グラフで表示されます。
- 最小／最大履歴トレンドには、表示時間全体にわたる値の変化が工学値のパーセントを縦軸に取って表示されます。変化量よりも時間の流れと変化率に重点が置かれています。
- 平均／散布履歴トレンドには、各トレンド間隔ごとのデータ ポイントの平均値が表示されます。

スクータと呼ばれるグラフィック スライダを作成することにより、トレンド内の現在のスクータの位置に応じた詳細なトレンドデータにアクセスできます。たとえば、データが表示されているトレンドの領域にスクータを移動すると、その位置のトレンドグラフに表示されているすべてのデータベース値の時刻と値を表示できます。

また、スクータ間や最大値と最小値などのデータ間隔を拡大／縮小表示するボタンやスライダを作成することもできます。グラフ全体、またはスクータ間の領域の平均や標準偏差を表示することもできます。

履歴トレンドをスクロールする時間も、任意に設定できます。カスタム スケールを作成して **.MinEU** と **.MaxEU** ドットフィールドにリンクすることにより、すべてのデータ セットをそのデータの工学単位で表示するトレンドを作成することができます。

リアルタイムトレンドの理解

リアルタイムトレンドには、現在実行中の InTouch アプリケーションのデータが表示されます。リアルタイムトレンドは継続的に更新されます。リアルタイムトレンドには、最大で 4 つのローカル タグ変数か式に関連する現在のデータ値が表示されます。

以下の操作を実行できます。

- リアルタイムトレンドの作成
- トレンドに対してタグ変数を選択する
- トレンドの表示時間と更新間隔の指定
- トレンドの表示オプションの設定

保存済みタグの値の履歴トレンド表示

以下のいずれかの WindowMaker ユーティリティを使用して、履歴トレンドを作成できます。

- 履歴トレンドツール
- 履歴トレンドウィザード
- 16 ペントレンドウィザード (オプション)

さらに、ActiveFactoryトレンドを組み込んで Historian データベースに保存された InTouch 履歴トレンドデータを表示できます。

履歴トレンド オブジェクトの使用

WindowMaker 履歴トレンド ツールを使用して、トレンドを作成して設定することができます。以下の操作を実行できます。

- 履歴トレンドの作成
- トレンドに対してタグを選択する
- トレンドの時間範囲と更新間隔の指定
- トレンドの表示オプションの設定

履歴トレンドの作成

履歴トレンド ツールを使用することにより、ウィンドウ内にトレンドオブジェクトを作成できます。新しく作成した履歴トレンドオブジェクトには、InTouch のデフォルト設定値が適用されます。

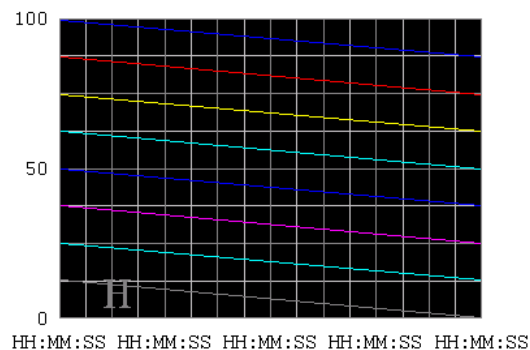
履歴トレンドの設定を変更した後は、WindowMaker では、最後の設定値が新しいトレンドオブジェクトの初期設定値として使用されます。

ウィンドウ枠の中であれば、どんなサイズのトレンドグラフでも表示できます。

履歴トレンドを作成するには

1. WindowMaker で、履歴トレンドを作成するウィンドウを開きます。
2. [描画] メニューの [トレンド] グループで [履歴] をクリックします。
3. 履歴トレンドを作成するウィンドウの領域にマウスを移動します。マウスを斜めにドラッグして長方形を作成し、所望のトレンドサイズにします。

ウィンドウに履歴トレンドオブジェクトが表示されます。



4. 必要に応じて、オブジェクトのハンドルを使用してトレンドの高さと幅を調整します。

履歴トレンドに表示するタグの設定

履歴トレンドペンによって、指定期間のログデータのグラフが作成されます。トレンドペンは、履歴データを収集するタグに割り当てられます。

履歴トレンドがサポートするペンの数は 8 本までです。

注記: WindowViewer は必ず閉じてください。そうでないと、ペンボックスを選択できません。

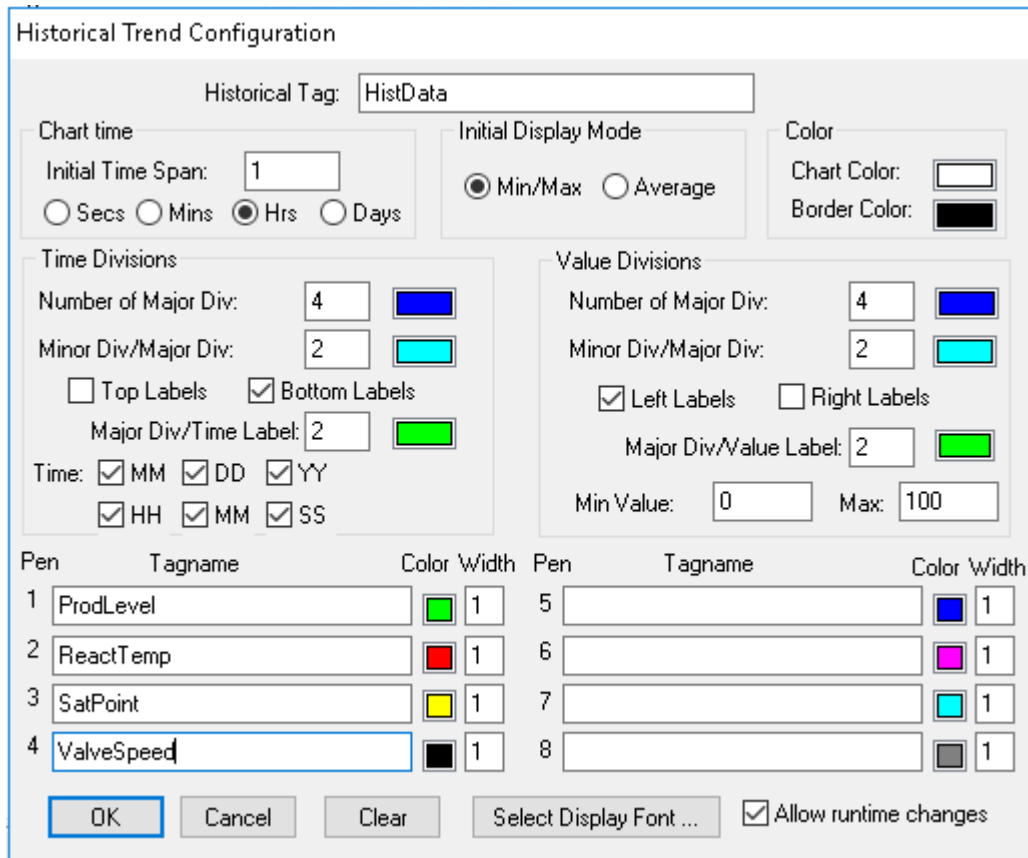
プロバイダが設定されている場合は、リモート履歴プロバイダからタグを選択できます。リモート履歴プロバイダの設定の詳細については、『AVEVA™ InTouch HMI アプリケーション配置ガイド』の「アプリケーションの分散」を参照してください。

注記: IndustrialSQL Server の履歴プロバイダを設定して履歴トレンドデータを視覚化することもできます。他のチャート オプションを使用してさらに用途を広げるには、ActiveFactoryトレンドツールを使用して IndustrialSQL Server データベースに保存された InTouch 履歴データのトレンドを作成してください。

履歴トレンドに表示するタグを設定するには

1. ウィンドウ内のトレンドオブジェクトをダブルクリックします。

[履歴トレンドの設定] ダイアログ ボックスが表示されます。



The dialog box is titled "Historical Trend Configuration". It contains several sections for configuring the trend display:

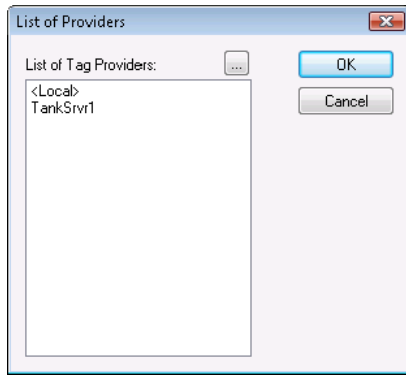
- Historical Tag:** A text box containing "HistData".
- Chart time:** Includes "Initial Time Span:" with a value of "1" and radio buttons for "Secs", "Mins", "Hrs" (selected), and "Days".
- Initial Display Mode:** Radio buttons for "Min/Max" (selected) and "Average".
- Color:** "Chart Color:" and "Border Color:" with corresponding color selection boxes.
- Time Divisions:** Includes "Number of Major Div:" (4), "Minor Div/Major Div:" (2), checkboxes for "Top Labels" and "Bottom Labels" (checked), "Major Div/Time Label:" (2), and checkboxes for "Time:" (MM, DD, YY, HH, MM, SS).
- Value Divisions:** Includes "Number of Major Div:" (4), "Minor Div/Major Div:" (2), checkboxes for "Left Labels" (checked) and "Right Labels", "Major Div/Value Label:" (2), "Min Value:" (0), and "Max:" (100).
- Pen and Tagname table:** A table with two columns: "Pen" and "Tagname". It lists 8 entries. The first four are: Pen 1 (ProdLevel, green), Pen 2 (ReactTemp, red), Pen 3 (SatPoint, yellow), and Pen 4 (ValveSpeed, black). The next four are empty for Pen 5 through 8.
- Buttons:** "OK", "Cancel", "Clear", "Select Display Font ...", and a checked checkbox for "Allow runtime changes".

2. [履歴タグ] ボックスに、トレンドに使用するタグを入力します。

タグは履歴トレンド型として定義する必要があります。InTouch アプリケーションの各履歴トレンドに対して、異なる履歴トレンドのタグを割り当ててください。

タグ名ディクショナリに現在定義されていないタグを入力した場合は、タグの作成を確認するメッセージがダイアログ ボックスに表示されます。[OK] を選択してタグを定義すると、[タグ名ディクショナリ] ダイアログ ボックスが表示されます。このタグは 128 文字をサポートしません。

3. [タグ名] 領域で、1 つ以上の [ペン] ボックスに既存のローカル タグまたはリモート タグを指定します。
4. 既存のローカル タグからリモート タグを直接指定するには、[ペン] ボックス内をクリックしてタグの名前を入力します。
5. タグを参照して指定するには
 - a. [ペン] ボックス内をダブルクリックします。[プロバイダのリスト] ダイアログ ボックスが表示されます。



- b. このペンに使用するタグプロバイダを選択します。
 - c. **[OK]** をクリックすると、選択されたプロバイダのタグ リストのダイアログ ボックスが表示されます。
 - d. リストのタグをダブルクリックして選択します。
6. タグが割り当てられた各ペンの横にあるカラー ボックスをダブルクリックすると、カラー パレットが開きます。ペンに使用する色をクリックします。
 7. **[幅]** ボックスに、トレンドに表示される各ペンの線幅をピクセル単位で入力します。
 8. 履歴トレンドのペンに割り当てる各タグについて、手順 3 から 7 を繰り返します。
 9. **[実行中の変更可能]** チェック ボックスをオンにすると、アプリケーションの実行中にオペレータが履歴トレンドの設定を変更できるようになります。

ランタイム時の履歴トレンドの更新の詳細については、「[ランタイム時のトレンド設定の変更](#)」を参照してください。

履歴トレンドの時間範囲の設定

履歴トレンドの時間範囲を設定できます。

履歴トレンドの時間範囲を設定するには

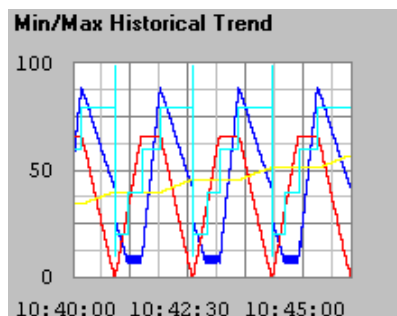
1. トレンドオブジェクトをダブルクリックして **[履歴トレンドの設定]** ダイアログ ボックスを表示します。
2. **[チャート時間]** 領域の **[初期時間範囲]** に、トレンドの横軸 (X 軸) に表示する時間長を入力します。
3. 時間単位を選択します。[秒]、[分]、[時]、[日] の中から選択します。

たとえば、**[初期時間範囲]** に 8 を入力してから **[時]** を選択すると、トレンドに表示される時間範囲は 8 時間になります。
4. **[初期表示モード]** 領域では、トレンドを表示するウィンドウが **WindowViewer** で最初に表示されたときに表示する履歴トレンドのタイプを選択します。

初期表示モード	説明
---------	----

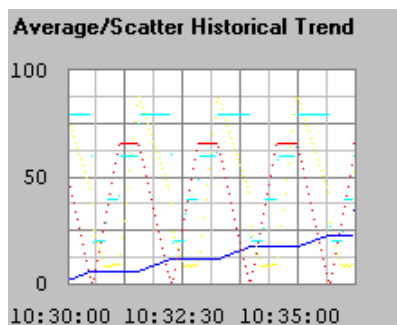
最小/最大

チャートには、時間範囲全体にわたる工学単位スケールのパーセンテージの変化が垂直線で表示されます。



平均

トレンド時間セグメント内の各ピクセルは、セグメントの全期間におけるタグの平均値を示したものです。



5. 履歴トレンドの表示方式を設定するには、「[履歴トレンドの表示オプションの設定](#)」に進んでください。

履歴トレンドの表示オプションの設定

履歴トレンドの表示方式を設定することができます。

履歴トレンドの表示オプションを設定するには

1. テトレンドオブジェクトをダブルクリックします。[履歴トレンドの設定] ダイアログ ボックスが表示されます。
2. 色オプションの設定には、以下の手順を実行します。
 - [配色] 領域で [背景] フィールドをクリックして、カラーパレットを開きます。
 - パレット内でトレンドの背景色をクリックします。
デフォルトの背景色は白です。背景色を別の色にすると、トレンドの印刷速度が遅くなります。
 - [輪郭色] を選択してカラーパレットを開きます。
 - パレット内でトレンドの輪郭色をクリックします。
3. 時間軸の目盛りオプションを設定するには、以下の手順を実行します。
 - [時間軸の目盛り] 領域の [主目盛り数] に、トレンドの主目盛り数を入力します。
主目盛りはトレンドの横軸（時間軸）に表示されます。主目盛りを分割する間隔は最長で 65536 秒、つまり 18 時間 12 分 16 秒までです。

- 主目盛り線の色を選択します。
- **【主目盛り間の分割数】** に、各主目盛り内の小目盛りの数を入力します。
時間軸の小目盛りの数は主目盛り内で均等に割り切れる必要があります。たとえば、主目盛りを 60 秒に設定して **【主目盛り間の分割数】** に 2 を入力すると、小目盛りは 30 秒に設定されます。
- 小目盛り線の色を選択します。
- **【上ラベル】** または **【下ラベル】** を選択して、トレンドに表示する時間ラベルの位置を指定します。
- 時間ラベルを使用する場合は、**【主目盛り/時間ラベル】** ボックスに時間ラベルあたりの時間軸の主目盛り線の数を入力します。
- 時間軸目盛りのラベルの色を選択します。
- 主目盛り線のラベルに表示する時間単位を選択します。

月 (MM)	時 (HH)
日 (DD)	分 (MM)
年 (YY)	秒 (SS)

4. **【縦軸の目盛り】** 領域で、トレンドの縦軸の表示を設定します。

【縦軸の目盛り】 オプションはの設定方法は、**【時間軸の目盛り】** オプションと同じです。縦軸では、すべてのタグの工学単位に基づいて、トレンドに表示するデータ値の範囲が指定されます。

5. **【OK】** をクリックして設定変更を保存してから、**【履歴トレンドの設定】** ダイアログ ボックスを閉じます。

ランタイム時のトレンド設定の変更

履歴トレンドの設定で **【実行中の変更可能】** オプションを選択すると、履歴トレンドの実行中に設定を変更できます。オペレータは、表示されているウィンドウからトレンドを選択してダイアログ ボックスを表示させてからトレンドの設定を変更します。

ランタイム中に履歴トレンドを設定するには

1. 実行中に履歴トレンドをクリックします。**【履歴トレンドの設定】** ダイアログ ボックスが表示されます。

2. [表示開始時間] 領域で、履歴トレンドデータの収集期間の開始日時を入力します。

3. [表示モード] 領域で、履歴トレンドチャートのタイプを選択します。

トレンドの表示モードは描画スピードに影響を及ぼします。トレンドの描画スピードを決める主要な要素は、トレンドに表示される線の長さです。線が長くなるほどトレンドの生成時間が長くなります。

線幅も同様に描画スピードに影響します。線幅が広いと描画速度は著しく遅くなります。[最大／最小] と [平均／散布]トレンドは、[平均／棒]トレンドよりも短い時間で生成されます。

4. [表示時間] 領域で、トレンド表示される時間間隔を入力してから時間単位を選択します。

たとえば、2を入力してから[時]を選択すると、トレンドの時間間隔は2時間になります。

5. [縦軸表示] 領域に、履歴トレンドの縦軸に表示される工学値の範囲をパーセント単位で入力します。

トレンドのスケールは、トレンド表示されているタグの工学値の範囲をパーセントの範囲で定義したものです。値の範囲は0から100までです。たとえば、選択したタグの工学値の範囲に対して40から60パーセントの範囲で変化するトレンドが必要な場合は、[最小値]と[最大値]のそれぞれの範囲フィールドに40と60を入力します。

6. [タグ] 領域で、タグを割り当てるペン番号をクリックします。

[タグの選択] ダイアログボックスに、履歴トレンドペンへの割り当てが可能なタグのリストが表示されます。

7. 以下のステートメントを QuickScript またはボタンに使用することにより、オペレータはチャートを更新できます。

```
Hist_TrendTag.UpdateTrend = 1
```

8. 以下のいずれかの関数を QuickScript またはボタンに使用します。

```
HTUpdateToCurrentTime(Hist_Tag);
```

```
HTScrollLeft(Hist_Tag,Percent);
```

```
HTScrollRight(Hist_Tag,Percent);
```

```
HTZoomIn(Hist_Tag,LockString);
```

```
HTZoomOut(Hist_Tag,LockString);
```

```
HTSetPenName(Hist_Tag, PenNum, Tagname);
```

トレンド関数を含むスクリプトの使用の詳細については、「[スクリプトの使用による履歴トレンドウィザードのコントロール](#)」を参照してください。

9. 以下のトレンドタグ ドットフィールドを変更します。

.ChartStart

.ChartLength

.MaxRange

.MinRange

.Pen1-.Pen8

履歴トレンドを含むドットフィールドの使用の詳細については、「[ドットフィールドの使用による履歴トレンドのコントロール](#)」を参照してください。

ドットフィールドの使用による履歴トレンドのコントロール

ドットフィールドを使用すれば、ランタイム実行中に履歴トレンドを管理できます。

.DisplayMode ドットフィールド

.DisplayMode ドットフィールドでは、タグ変数の値の表示に使用されるトレンド形式を指定します。

カテゴリ

履歴

使用法

```
tag_name.DisplayMode
```

パラメータ

tag_name

任意の履歴トレンドのタグ変数

データタイプ

アナログ型（読み取り／書き込み）

有効値

- 1 = 各サンプル期間の最大値と最小値を表示します（デフォルト）。
- 2 = 散布履歴トレンドの各サンプル期間の平均値を表示します。
- 3 = 棒グラフ履歴トレンドの各サンプル期間の平均値を表示します。

例

このステートメントでは、「HistTrend_Tag」で表される履歴トレンドの値に棒グラフ履歴トレンドの形式が指定されます。

```
HistTrend_Tag.DisplayMode=3;
```

参照項目

.ChartLength、.ChartStart

.MinRange ドットフィールド

.MinRange ドットフィールドでは、各タグ変数の履歴トレンド表示に使用される、タグ変数の工学値範囲に対する最小パーセントが指定されます。

カテゴリ

履歴

使用法

```
tag_name.MinRange
```

パラメータ

tag_name

任意の履歴トレンドのタグ変数

備考

履歴トレンドには、複数のタイプのタグ変数を同時に表示することができます。異なるタイプのタグ変数には別の工学単位が割り当てられるため、工学単位で値の範囲の下限値と上限値を指定するのは困難です。そのため、最小値と最大値は各タグ変数の工学値範囲に対するパーセントで表示しています。したがって履歴トレンドは、タグ変数の実際の工学値範囲にかかわらず、タグ変数に固有の工学値のパーセント表示で示されます。

データタイプ

実数型（読み取り／書き込み）

有効値

.MaxRange と **.MinRange** ドットフィールドの範囲は 0 から 100 です。**.MinRange** は **.MaxRange** よりも小さい値にする必要があります。これらのドットフィールドに 0 未満か 100 を超える値が割り当てられると、自動的に 0 または 100 に変更されます。**.MinRange** が **.MaxRange** 以上の場合は、トレンドにデータは表示されません。

例

以下のドットフィールドステートメントは、履歴トレンドの最小パーセントを履歴トレンドのタグ変数がとり得る工学値範囲の 25 パーセントに設定します。

```
HistTrend.MinRange=25
```

参照項目

.ChartStart、**.ChartLength**、**.DisplayMode**、**.EngUnits**、**.MinEU**、**.MaxEU**、**.MaxRange**、**.MinRaw**、**.MaxRaw**、**.RawValue**

.MaxRange ドットフィールド

.MaxRange ドットフィールドでは、各タグ変数の履歴トレンド表示に使用される、タグ変数の工学値の範囲に対する最大パーセントが指定されます。

カテゴリ

履歴

使用法

```
tag_name.MaxRange
```

パラメータ

tag_name

任意の履歴トレンドのタグ変数

備考

履歴トレンドには、複数のタイプのタグ変数を同時に表示することができます。タグ変数には異なる工学単位が割り当てられるため、工学単位で値の範囲の下限值と上限値を指定するのは困難になります。そのため、最小値と最大値は各タグ変数の工学値範囲に対するパーセントで表示しています。したがって、タグ変数の実際の工学値範囲にかかわらず、履歴トレンドはタグ変数の工学値のパーセント表示で示されます。

データタイプ

実数型（読み取り／書き込み）

有効値

.MaxRange と .MinRange ドットフィールドの範囲は 0 から 100 です。.MinRange は .MaxRange よりも小さい値にする必要があります。これらのドットフィールドに 0 未満か 100 を超える値が割り当てられると、自動的に 0 または 100 に変更されます。.MinRange が .MaxRange 以上の場合は、トレンドにデータは表示されません。

例

以下のドットフィールドステートメントは、履歴トレンドの最大パーセントを履歴トレンドのタグ変数がとり得る工学値範囲の 75 パーセントに設定します。

```
HistTrend.MaxRange=75
```

参照項目

.ChartStart、.ChartLength、.DisplayMode、.EngUnits、.MinEU、.MaxEU、.MinRange、.MinRaw、.MaxRaw、.RawValue

.UpdateCount ドットフィールド

.UpdateCount ドットフィールドでは、履歴トレンドの更新回数がカウントされます。.UpdateCount ドットフィールドは、関数のトリガとして使用できます。

カテゴリ

履歴

使用法

```
HistTrendTag.UpdateCount
```

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ変数

データタイプ

整数型（読み取り専用）

有効値

任意の正の整数値

例

以下の例では、右スクータの現在位置で Pen1 の値を取得するために **HTGetValueAtScooter()** 関数を使用しています。関数の引数に対して変更が行われると、その関数の値が再度求められます。更新が終了するたびに **.UpdateCount** が増分し、このステートメントも更新されます。

```
MyRealTag=HTGetValueAtScooter( MyHistTrendTag,MyHistTrendTag.UpdateCount, 2,  
MyHistTrendTag.ScooterPosRight, 1, "PenValue");
```

参照項目

.UpdateInProgress、.UpdateTrend

.UpdateInProgress ドットフィールド

.UpdateInProgress ドットフィールドには、履歴トレンド更新処理の現在の状況が示されます。履歴取得処理中にはこのドットフィールドに **1** がセットされ、それ以外の場合は **0** がセットされます。

カテゴリ

履歴

使用法

```
HistTrendTag.UpdateInProgress
```

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ変数

備考

履歴トレンドが新しいデータを要求するたびに、このドットフィールド値は **1** にセットされます。処理が終了すると、**.UpdateInProgress** は **0** にリセットされます。**.UpdateInProgress** は、履歴トレンドに関連する関数の中で使用することができます。

ユーザーが、現在の表示期間外にトレンドをスクロールした場合には、履歴データの取得に時間を要することがあります。**.UpdateInProgress** ドットフィールドを使用すれば、要求したデータの取得中であることをユーザーに知らせることができます。応答がなければ、トレンドを更新中であることがユーザーにはわかりません。

データタイプ

論理型（読み取り専用）

有効値

0 = 更新は行われていません

1 = 更新中

例

.UpdateInProgress ドットフィールドは通常、履歴トレンドの文字オブジェクトやスクロール ボタンの表示可能リンクの式としてよく使用されます。以下のメッセージ型値の表示アニメーションリンクを指定して **.UpdateInProgress** ドットフィールドを使用すると、データの取得中にウィンドウに「ビジー」を表示することができます。

```
DText(HistTrend1.UpdateInProgress, "Busy", "Ready")
```

参照項目

.UpdateCount、.UpdateTrend

.UpdateTrend ドットフィールド

.UpdateTrend ドットフィールドは、履歴トレンドの更新をトリガします。ボタンアクションスクリプトで**.UpdateTrend** ドットフィールドを使用すると、ランタイム中にユーザーは手動でトレンドを更新できます。

カテゴリ

履歴

使用法

HistTrendTag.UpdateTrend

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ変数

備考

通常、履歴トレンドは自動的に更新されません。特定のタグ変数に対するチャートを更新して現在の値を表示するには、**.ChartStart** か **.ChartLength** のいずれかのドットフィールドを変更する必要があります。このドットフィールドをボタンアクションスクリプトで使用すれば、ユーザーはランタイム中にチャートを更新することができます。また、履歴トレンドに関連付けられている別のドットフィールドが変化する場合には、**QuickScript** でこのドットフィールドを使用できます。

.UpdateTrend ドットフィールドには **1** だけをセットできます。

データタイプ

論理型（書き込み専用）

有効値

1

例

この例では、**MyHistTrendTag** タグ変数に関連付けられた履歴トレンドを現在の全パラメータ値で更新するようにトリガします。

```
MyHistTrendTag.UpdateTrend=1;
```

.ChartLength ドットフィールド

.ChartLength ドットフィールドでは、履歴トレンドに表示される時間長が指定されます。

カテゴリ

履歴

使用法

HistTrendTag.ChartLength

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ変数

備考

.ChartLength の値により、チャートの長さが秒単位で指定されます。この長さは、履歴トレンドのチャートに現在表示されている時間として定義されます。具体的には、履歴トレンドのチャートから表示時間を求めるには次の計算式が使用されます。

表示時間 = (チャートの右端の日付／時間スタンプ) - (チャートの左端の日付／時間スタンプ)

日付／時間スタンプは 1970 年 1 月 1 日の午前 0 時からの経過時間を秒数で表しているため、計算結果はチャートの左端と右端の間に表示される秒単位の時間になります。

.ChartLength の値を増減させる場合には、必ず秒単位で時間を表します。したがって、現在の .ChartLength を 2 時間減らすには、計算を実行する前に時間単位を秒単位に変換する必要があります。次に例を示します。

(2 時間) * (60 分／時間) * (60 秒／分) = 7200 秒

データタイプ

整数型 (読み取り／書き込み)

有効値

任意の正の整数値

例

以下の例では、履歴トレンドの長さが 1 時間に設定されます。

HtTag.ChartLength=3600 {60 分 * 60 秒／分};

以下の例では、トレンドが 50 パーセント分だけ左にスクロールされます。

HtTag.ChartStart=HtTag.ChartStart - HtTag.ChartLength / 2;

以下の例では、トレンドが 10 パーセント分だけ左にスクロールされます。

HtTag.ChartStart=HtTag.ChartStart - (.10 * HtTag.ChartLength);

参照項目

.ChartStart

.ChartStart ドットフィールド

.ChartStart ドットフィールドは、履歴トレンドの開始 (左端) 日付／時間スタンプの値を設定または確認するために使用されます。

カテゴリ

履歴

使用法

HistTrendTag.ChartStart

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ変数

備考

この読み取り／書き込みドットフィールドは、履歴トレンドの開始 (左端) 日付／時間スタンプの値を設定または確認するために使用されます。.ChartStart ドットフィールドは、1970 年 1 月 1 日の午前 0 時

からの経過時間を秒数で表します。この開始時点は、履歴トレンドの先頭の日付／時間スタンプとして定義されます。

データタイプ

整数型（読み取り／書き込み）

有効値

任意の正の整数値

例

次のステートメントは、チャートを右側に 1 分（60 秒）スクロールします。

```
HtTagname.ChartStart=HtTagname.ChartStart + 60;
```

参照項目

.ChartLength

.Pen1-8 ドットフィールド

.Pen1-8 ドットフィールドでは、履歴トレンド ペンにログ済みタグが割り当てられます。

カテゴリ

履歴

使用法

```
HistTrendTag{.Pen1 | .Pen2 | .Pen3 | .Pen4 | .Pen5 | .Pen6 | .Pen7 | .Pen8};
```

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ

備考

以下の形式で .Pen1-8 ドットフィールドを使用して、トレンド ペンにタグを割り当てます。

```
HistTrend.PenX = Tag_Name.TagID
```

ここで、X は 1 から 8 までの整数です。

可能であれば、HTSetPenName() または HTGetPenName() 関数の使用をお勧めします。

注記: .PenX ドットフィールドに割り当てられるのは、ローカル タグだけです。Provider.tag のような表記法は使用できません。この表記法は、HTSetPenName() 関数でのみ使用できます。

このドットフィールドの機能については、パートに分かれて画面表示される履歴トレンドウィザードが参考になります。

データ タイプ

タグ ID（読み取り/書き込み）

有効値

このドットフィールドのデータ タイプは **TagID** です。そのため、.Pen1-8 ドットフィールドにはタグのハンドルだけが割り当てられます。.Pen1-8 ドットフィールドにタグの名前を直接割り当てることはできません。以下の構文を使用して、.Pen1-8 ドットフィールドに対してタグの .TagID ドットフィールドを割り当てる必要があります。

```
HistTrendTag.Pen1=LoggedTag.TagID;
```

一般的に、TagID 型のタグは、別の TagID 型のタグとしか等式化できません。そのため .TagID ドットフィールドの拡張を他のタグに追加しない限り、他のタグ タイプとは混合できません。

.Pen1-8 ドットフィールドは読み取り/書き込みですが、ドットフィールドの値は画面に直接表示できません。

例

以下の例では、履歴トレンドのタグに関連付けられた履歴トレンドの .Pen5 ドットフィールドに対して、新しいタグが割り当てられます。 .TagID ドットフィールドを .Pen5 ドットフィールドに割り当てるには、.TagID ドットフィールドをログ済みタグの名前に追加しなければなりません。

```
HistTrendTag.Pen5=PumpPress.TagID;
```

上記の例によって、HistTrendTag.Pen5 に割り当てられたタグの名前が明確になりました。各トレンドペンに割り当てられたタグを示す凡例を作成しておく、オペレータへの有用な情報になります。

メッセージ表示リンクでは、HistTrendTag.Pen5 に割り当てられたタグを示すことはできません。 .Pen5 ドットフィールドの実際の値は WindowViewer 内のメモリ ロケーションを表す整数値で、表示の目的にはあまり有用でないためです。そのため、Pen05 という新しい TagID 型のタグを作成する必要があります。前のステートメントに以下のステートメントを追加します。

```
Pen05=HistTrendTag.Pen5;
```

最初の例では、PumpPress タグが HistTrendTag の ペン 5 に割り当てられます。この例では、PumpPress タグの TagID になった HistTrendTag の Pen5 の値に Pen05 が割り当てられます。

.Pen1-8 ドットフィールドは、トレンド表示に選択されたペンに関連付けられているタグへのポインタです。 .Pen1-8 ドットフィールドのデータ タイプは、.TagID という特殊なものです。割り当てを作成した後、TagID タグの .Name ドットフィールドを使用して、タグの名前を表示できます。

.TagID ドットフィールド

.TagID ドットフィールドを Pen1 ～ Pen8 ドットフィールドと共に使用して、履歴トレンドペンにタグ変数を割り当てることができます。

カテゴリ

履歴タグ変数。

使用法

```
tag_name.TagID
```

パラメータ

tag_name

論理型、整数型、実数型、間接論理型、または間接アナログ型の任意のタグ変数

備考

.TagID ドットフィールドはタグ変数のハンドルを可能にするもので、主に履歴トレンドのペンにタグ変数を割り当てるために使用されます。

データタイプ

タグ変数 ID 型（読み取り専用）

例

以下は、.TagID ドットフィールドを使用して履歴トレンドのペン 6 に PumpRPM タグ変数を割り当てる例です。


```
HistTrendTag.Pen6=PumpRPM.TagID;
```

参照項目

.Pen1-.Pen8

.ScooterLockLeft ドットフィールド

.ScooterLockLeft ドットフィールドでは、ユーザーが履歴トレンド上で右スクータを左スクータの現在位置よりもさらに左側に移動できるかどうか指定されます。

カテゴリ

履歴

使用法

```
HistTrendTag.ScooterLockLeft
```

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ変数

備考

一般的に、ユーザーには右スクータを左スクータの現在位置よりも左側に移動させないようにする必要があります。左スクータがロックされている場合、右スクータを左スクータよりさらに左側に移動すると、右スクータの位置は強制的に左スクータの位置と等しくなります。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = False 履歴トレンド上で右スクータを左スクータの現在位置よりもさらに左側に移動できます。

1 = True 履歴トレンド上で右スクータを左スクータの現在位置よりもさらに左側に移動できません。

例

以下の例では、履歴トレンド上で右スクータを左スクータの現在位置よりもさらに左側に移動できないようにします。

```
HistTrendTag.ScooterLockLeft=1;
```

参照項目

.ScooterPosRight、.ScooterPosLeft、.ScooterLockRight

.ScooterLockRight ドットフィールド

.ScooterLockRight ドットフィールドでは、ユーザーが履歴トレンド上で左スクータを右スクータの現在位置よりもさらに右側に移動できるかどうか指定されます。

カテゴリ

履歴

使用法

```
HistTrendTag.ScooterLockRight
```

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ変数

備考

一般的にユーザーには、左スクータを右スクータの現在位置よりも右側に移動させないようにする必要があります。右スクータがロックされている場合、左スクータを右スクータよりさらに右側に移動すると、左スクータの位置は強制的に右スクータの位置と等しくなります。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = False 履歴トレンド上で左スクータを右スクータの現在位置よりもさらに右側に移動できます。

1 = True 履歴トレンド上で左スクータを右スクータの現在位置よりもさらに右側に移動できません。

例

以下の例では、履歴トレンド上で左スクータを右スクータの現在位置よりもさらに右側に移動できないようにします。

```
HistTrendTag.ScooterLockRight=1;
```

参照項目

.ScooterPosRight、.ScooterPosLeft、.ScooterLockLeft

.ScooterPosLeft ドットフィールド

.ScooterPosLeft は、左スクータの位置を履歴トレンド上で動的にコントロールします。

カテゴリ

履歴

使用法

```
HistTrendTag.ScooterPosLeft
```

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ変数

備考

この読み取り／書き込みのドットフィールドは、左スクータの位置を動的にコントロールします。このドットフィールドを **QuickScript** 関数に使用すると、左スクータの現在位置を取得したり、このドットフィールドに新しい値を割り当てて左スクータをトレンド上の別の位置に動かしたりできます。

このドットフィールドは通常、**HTGetValue()** 関数と組み合わせて使用されます。この関数では、履歴トレンドのスクータの位置の指定のほかに、どの履歴トレンドをクエリーするのかを指定する必要があります。

データタイプ

実数型（読み取り／書き込み）

有効値

0.0～1.0; ここで、0.0 は履歴トレンドチャートの一番左端で、1.0 は履歴トレンドチャートの一番右端。

例

以下の例では、左スクータの位置が変更されます。MyHistTrendTag タグ変数に現在関連付けられている履歴トレンドチャートで、チャートの左端からチャート全体の長さの 34 パーセントの位置に左スクータを移動します。

```
MyHistTrendTag.ScooterPosLeft=.34;
```

以下のステートメントでは、QuickScript の HTGetValueAtScooter() 関数によって左スクータの現在位置におけるペン 1 の値が取得されます。関数の引数リスト内で値が変更されると、その関数の値が再度求められます。また、左スクータの位置が変わるたびに、このステートメントは更新されます。

```
MyRealTag=HTGetValueAtScooter (MyHistTrendTag,MyHistTrendTag.UpdateCount, 1,  
MyHistTrendTag.ScooterPosLeft, 1, "PenValue");
```

参照項目

.ScooterPosRight、.ScooterLockLeft、.ScooterLockRight

.ScooterPosRight ドットフィールド

この読み取り／書き込みの .ScooterPosRight ドットフィールドは、右スクータの位置を動的にコントロールします。

カテゴリ

履歴

使用法

```
HistTrendTag.ScooterPosRight
```

パラメータ

HistTrendTag

トレンド名が割り当てられた履歴トレンドのタグ変数

備考

この読み取り／書き込みのドットフィールドは、右スクータの位置を動的にコントロールします。このドットフィールドを QuickScript 関数に使用すると、右スクータの現在位置を取得できます。また、このドットフィールドに新しい値を割り当てて、右スクータをトレンド上の別の位置に動かすこともできます。

ドットフィールドは通常、HTGetValue() 関数と組み合わせて使用されます。この関数では、履歴トレンドのスクータの位置の指定のほかに、どの履歴トレンドをクエリーするのかを指定する必要があります。

データタイプ

実数型（読み取り／書き込み）

有効値

0.0～1.0; ここで、0.0 は履歴トレンドチャートの一番左端で、1.0 は履歴トレンドチャートの一番右端。

例

以下のステートメントでは、右スクータに新しい位置が指定されます。**MyHistTrendTag** タグ変数に関連付けられている履歴トレンドチャートで、チャートの左端からチャート全体の長さの **34 パーセント** の位置に右スクータを移動します。

```
MyHistTrendTag.ScooterPosRight=.34;
```

以下のステートメントでは、**HTGetValueAtScooter()** QuickScript 関数を使用して右スクータの新たな位置におけるペン 1 の値が取得されます。関数のパラメータ リスト内で変数が変更されると、その関数の値が再度求められます。また、右スクータの位置が変わるたびに、このステートメントは更新されます。

```
MyRealTag=HTGetValueAtScooter(MyHistTrendTag, MyHistTrendTag.UpdateCount, 2,  
MyHistTrendTag.ScooterPosRight, 1, "PenValue");
```

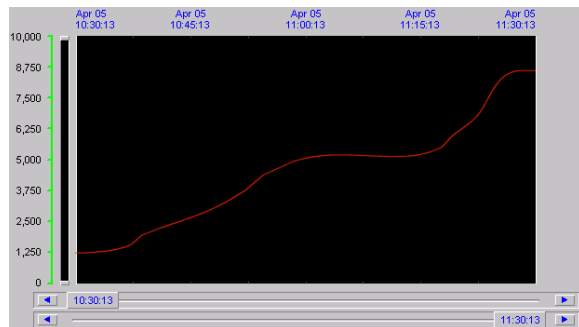
参照項目

.ScooterPosLeft、.ScooterLockLeft、.ScooterLockRight

履歴トレンド ウィザードの使用

履歴トレンドウィザードを使用して、自動的に履歴トレンドを作成することができます。手動で履歴トレンドペンにタグ変数を割り当てなくても、ウィザードが標準値を使用して自動的に履歴トレンドを設定します。

履歴トレンドウィザードで作成した標準のトレンドを以下に示します。このトレンドには、トレンドの特定位置のデータを表示したり、トレンドデータの範囲を選択してズームインするために使用するスクータと呼ばれるスライダが含まれています。



履歴トレンドにズームや移動機能またはペン コントロールを追加するには、[トレンドズーム / スクロール パネル] ウィザードと [トレンドペン凡例] ウィザードを使用します。

履歴トレンドを作成して設定できます。以下の操作を実行できます。

- ウィザードで履歴トレンドを作成する
- トレンドに対してタグ変数を選択する
- 履歴トレンドの表示時間を設定する
- QuickScript でトレンドをコントロールする

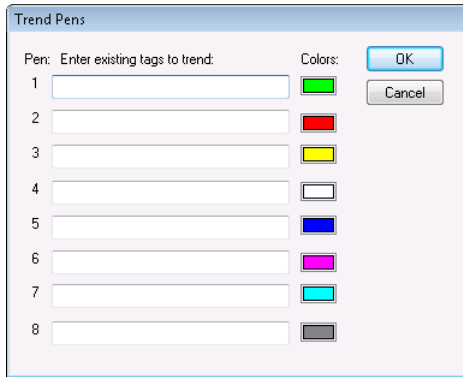
履歴トレンド ウィザードによるトレンドの作成

履歴トレンドウィザードの自動機能を使用して標準の履歴トレンドを作成できます。ウィザード オプションを使用して手動で履歴トレンドを設定する方法については、別のセクションに記載します。

履歴トレンドグラフ ウィザードでトレンドペンにタグを割り当てる作業は、履歴/リアルタイム ツールの手順と似ています。

履歴トレンドグラフ ウィザードでタグを割り当てるには

- 履歴トレンドをダブルクリックすると、
[履歴トレンドグラフ ウィザード] ダイアログ ボックスが表示されます。
- [ペン] をクリックします。
[トレンド ペン] ダイアログ ボックスが表示されます。



- [ペン] ボックスに既存のローカル タグの名前を入力します。入力できる最大文字数は 49 文字です。

注記: WindowViewer は必ず閉じてください。そうでないと、ペン ボックスを選択できません。

[ペン] ボックス内をダブルクリックすると、[タグの選択] ダイアログ ボックスが表示され、アプリケーションの [ログ データ] オプションが割り当てられたタグのリストが表示されます。[タグの選択] ダイアログ ボックスでタグを選択することにより、タグをペンに割り当てることができます。

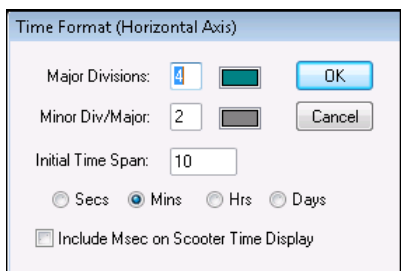
- ペンの色をデフォルト色から変更する場合は、各ペンの横にあるカラー ボックスをクリックして別の色を選択します。変更しない場合は、この手順を行わずにデフォルト色のままとします。
- [OK] をクリックして、[トレンド ペン] ダイアログ ボックスを閉じます。
- [OK] をクリックし、[履歴トレンドグラフ ウィザード] ダイアログ ボックスを閉じます。

履歴トレンドの表示時間の設定

[履歴トレンドグラフ ウィザード] ダイアログ ボックスには、履歴トレンド ウィザードで作成されたトレンドに表示される表示時間を手動で設定するオプションが用意されています。履歴トレンド ウィザードによるデフォルト設定をそのまま使用しない場合は、手動でトレンド時間の設定を変更できます。

履歴トレンドの表示時間の設定を変更するには

- 履歴トレンドをダブルクリックすると、[履歴トレンドグラフ ウィザード] ダイアログ ボックスが表示されます。
- [横軸] をクリックします。[時間の形式] ダイアログ ボックスが表示されます。



3. 時間の形式を設定するには、以下の操作を行います。
 - a. [主目盛り] ボックスに、トレンドの横軸（時間軸）に表示する主目盛りの数を入力します。
 - b. [主目盛り間の分割数] ボックスに、各主目盛り内の小目盛りの数を入力します。
 - c. [初期値] ボックスに、トレンドの横軸の表示時間長を入力します。履歴トレンドウィザードで作成されたトレンドは、**WindowViewer** でのアプリケーションの実行中に更新することができます。ユーザーは、トレンドの時間長を変更できますが、履歴トレンドの開始時には常に[時間の形式] ダイアログ ボックスで設定した時間間隔が使用されます。
 - d. トレンドの時間単位を [秒]、[分]、[時]、[日] の中から選択します。
 - e. オプションにより、スクータの時間表示をミリ秒単位まで含めることができます。現在時刻にミリ秒単位を追加して表示するスクータ スライダの例を以下に示します。

11:30:13.000

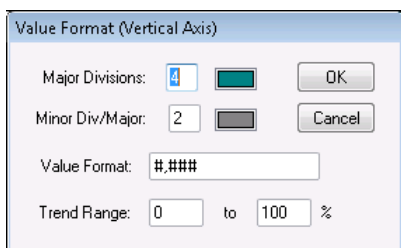
4. [OK] をクリックして、[時間の形式] ダイアログ ボックスを閉じます。
5. [OK] をクリックし、[履歴トレンドグラフ ウィザード] ダイアログ ボックスを閉じます。

表示オプションの設定

[履歴トレンドグラフ ウィザード] ダイアログ ボックスには、履歴トレンドの縦軸の単位を設定するオプションが用意されています。トレンドの縦軸に表示される主目盛りと小目盛りを手動で設定することができます。

履歴トレンドグラフ ウィザードで表示オプションを設定するには

1. 履歴トレンドをダブルクリックします。
[履歴トレンドグラフ ウィザード] ダイアログ ボックスが表示されます。
2. [値] をクリックします。
[値の形式] ダイアログ ボックスに、トレンドの縦軸（値軸）の設定オプションが表示されます。



3. 値の形式を設定するには、以下の手順を実行します。

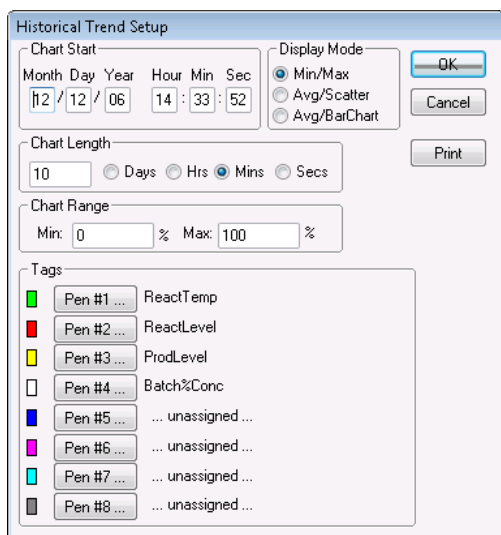
- a. **【主目盛り】** ボックスに、トレンドの縦軸に表示する主目盛りの数を入力します。カラー ボックスをクリックしてカラー パレットを表示して、縦軸の主目盛り線に指定する色をクリックします。
 - b. **【主目盛り間の分割数】** ボックスに、縦軸の各主目盛り内に表示する小目盛りの数を入力します。カラー ボックスをクリックしてカラー パレットが表示して、縦軸の小目盛り線に指定する色をクリックします。
 - c. **【値の形式】** ボックスに、トレンドの縦軸（値軸）に表示する数値の形式を入力します。デフォルトの数値形式は#,### です。
 - d. **【トレンドの範囲】** ボックスに、トレンドに表示するタグの工学単位に対する下限と上限のパーセントを入力します。
4. **【OK】** をクリックして、**【値の形式】** ダイアログ ボックスを閉じます。
 5. **【OK】** をクリックし、**【履歴トレンドグラフ ウィザード】** ダイアログ ボックスを閉じます。

ランタイム中の設定変更

履歴トレンドの設定で**【実行中の変更可能】** オプションを選択した場合は、ユーザーはランタイム中に履歴トレンドの設定をさまざまに変更できます。

ランタイム中に履歴トレンドを設定するには

1. WindowViewer でトレンドをクリックします。**【履歴トレンドの設定】** ダイアログ ボックスが表示されます。



2. **【表示開始時間】** 領域で、表示を開始する日時を入力します。
3. **【表示モード】** 領域で、履歴トレンドのタイプを選択します。
4. **【表示時間】** 領域で、トレンドに表示する時間の長さを入力してから時間の単位を選択します。
5. **【縦軸表示】** 領域で、トレンドの縦軸に表示される工学値のスケールをパーセント単位で入力します。
6. **【タグ変数】** 領域で、各 **【ペン#】** をクリックしてトレンドペンにタグ変数を割り当てます。**【タグ変数を選択してください】** ダイアログ ボックスが表示されて、ログが可能なタグ変数が表示されます。

7. トレンド ペンに割り当てるタグ変数名をダブルクリックします。
8. **[OK]** をクリックして、ランタイム中のトレンド変更を保存します。

スクリプトの使用による履歴トレンドウィザードのコントロール

QuickScript の関数をトレンドオブジェクトやアニメーションリンクの式と共に使用して、ランタイム中に履歴トレンドをコントロールできます。たとえば、QuickScript を使用して、トレンドを現在時刻に更新したり、タグ変数をトレンドペンに再割り当てしたり、チャートにペンを接続したり、グリッドを再描画したり、スクータを削除または再表示したりすることができます。

トレンドを現在時刻に更新

最新のタグ変数データを表示するために履歴トレンドを更新するスクリプトを作成することができます。

HTUpdateToCurrentTime() 関数

HTUpdateToCurrentTime() 関数は、現在の時刻を終了時刻に一致させてデータを取得して表示します。開始時刻は、終了時刻からチャートの幅を引いたものとなります。

カテゴリ

履歴

構文

```
HTUpdateToCurrentTime(Hist_Tag);
```

引数

Hist_Tag

履歴トレンド名が割り当てられた履歴トレンドのタグ変数

例

以下のステートメントでは、現在の時刻の **Trend1** 履歴タグ変数のデータを取得して表示します。

```
HTUpdateToCurrentTime("Trend1");
```

現在時刻が午後 3 時 4 分でトレンドの幅が 60 秒の場合、更新された終了時刻は午後 3 時 4 分になります。また、更新された開始時刻は午後 3 時 3 分になります。

トレンド設定の変更

以下のスクリプト関数を使用して、履歴トレンドのペンに割り当てられたタグ変数を変更できます。

- [HTSelectTag\(\) 関数](#)
- [HTSetPenName\(\) 関数](#)

HTSelectTag() 関数

HTSelectTag() 関数を使用すると **[タグ変数を選択してください]** ダイアログボックスが開き、ユーザーはトレンドペンに別のタグ変数を割り当てられます。

注記: タグ変数を選択してください ダイアログボックスには、タグ変数ディクショナリで **[ログデータ]** オプションが選択されて、履歴ログ用に定義されたタグ変数のみがリスト表示されます。

カテゴリ

履歴

構文

```
HTSelectTag();
```

備考

HTSelectTag() 関数は、タグ変数ディクショナリで **[ログ データ]** オプションが選択されているタグ変数のみを表示します。ただし、タグ ブラウザのフィルタを使用すれば、表示するタグのセットを絞り込むことができます。たとえば「A」で始まるすべてのタグ変数を指定できます。選択されたタグ変数を戻り値とするこの関数を、ペンにタグ変数を割り当てる関数のパラメータとして使用することができます。

例

以下の QuickScript は、**[タグ変数を選択してください]** ダイアログ ボックスを WindowViewer に表示します。ユーザーはリストからタグ変数を選択できます。選択したタグ変数は、HistTrend という名前の履歴オブジェクトのペン 1 に割り当てられます。

```
HTSetPenName("HistTrend",1,HTSelectTag());
```

参照項目

HTSetPenName()

HTSetPenName() 関数

HTSetPenName() 関数は、トレンド ペンに別のタグ変数を割り当てます。

カテゴリ

履歴

構文

```
HTSetPenName(Hist_Tag, PenNum, Tagname);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

PenNum

トレンドのペン番号 (1~8) を示す整数値か整数型タグ変数。

Tagname

ペンに割り当てる新しいタグ変数名

備考

この QuickScript 関数は、ランタイム中に分散履歴プロバイダのタグ変数を追加する唯一の方法です。

ペン名のリファレンスに入力できる最大文字数は 49 文字です。

トレンド ペンの割り当てを解除しようとする、以下のエラー メッセージが表示されることがあります。

```
VIEW /UpdateData:Invalid DBS.TAGNAME handle - 0
```

このエラーは、*histprovider.tag_name* の形式でリモートタグにすでに割り当てられたペンを解除しようとしたことが原因です。この問題を解決するには、**[ログ データ]** オプションが選択されたローカルタグ変数を作成する必要があります。さらに、以下のスクリプトを使用してペンの割り当てを解除します。

```
HTSetPenName( "HistTrend", 1, "localtag" );
```

```
{ローカルでログ済みのタグ変数---localtag にペンを割り当てる}  
HistTrend.Pen1=None;  
{ペンの割り当てを解除する}
```

ここで、None はタグ変数 ID 型のタグ変数です。

例

以下のステートメントは、Trend1 のペン 3 に OutletPressure タグ変数を割り当てます。

```
HTSetPenName("Trend1",3,"OutletPressure");
```

以下のステートメントは、Trend1 の TrendPen4 に HistPrv1.Tag1 タグ変数を割り当てます。

```
HTSetPenName("Trend1",TrendPen4,"HistPrv1.Tag1");
```

参照項目

HTSelectTag()

トレンドと履歴データ関連情報の取得

履歴トレンドの情報を履歴トレンドの実行中に取得するスクリプトを作成することができます。次の関数を使用します。

- [HTGetPenName\(\) 関数](#)
- [HTGetTimeAtScooter\(\) 関数](#)
- [HTGetTimeStringAtScooter\(\) 関数](#)
- [HTGetValue\(\) 関数](#)
- [HTGetValueAtScooter\(\) 関数](#)
- [HTGetValueAtZone\(\) 関数](#)
- [HTScrollLeft\(\) 関数](#)
- [HTScrollRight\(\) 関数](#)
- [HTZoomIn\(\) 関数](#)
- [HTZoomOut\(\) 関数](#)

HTGetPenName() 関数

HTGetPenName() 関数は、履歴トレンドの特定のペン番号に現在割り当てられているタグ変数名を返します。

カテゴリ

履歴

構文

```
MessageResult=HTGetPenName(Hist_Tag,UpdateCount, PenNum);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

UpdateCount

トレンドの **.UpdateCount** ドットフィールドを示す整数値。この引数の値は、関数の値を再度求めるためのデータ変更トリガとしての機能を果たします。

PenNum

トレンドのペン番号（1～8）を示す整数値か整数型タグ変数。

例

以下のステートメントでは、Trend1トレンドのペン2に割り当てられたタグ変数名が取得され、メッセージ型タグ変数の **TrendPen** にその名前を格納します。

```
TrendPen=HTGetPenName("Trend1", Trend1.UpdateCount,2);
```

HTGetTimeAtScooter() 関数

HTGetTimeAtScooter() は、*ScootNum* と *ScootLoc* 引数で指定したスクータ位置のサンプルの時刻を、1970年1月1日午前0時から数えた秒数で返します。

カテゴリ

履歴

構文

```
IntegerResult=HTGetTimeAtScooter(Hist_Tag, UpdateCount,ScootNum,ScootLoc);
```

引数**Hist_Tag**

トレンド名が割り当てられた履歴トレンドのタグ変数

UpdateCount

トレンドの **.UpdateCount** ドットフィールドを示す整数値。

ScootNum

左または右スクータを示す整数値：

1 = 左スクータ

2 = 右スクータ

ScootLoc

トレンド上の **.ScooterPosRight** または **.ScooterPosLeft** の位置の値を示す実数値。

備考

引数の **UpdateCount**、**ScootNum**、および **ScootLoc** に割り当てられた値が変わると、この式の値が再度求められます。これにより、新しいデータの検索後やスクータの移動後にこの式が評価されて値が更新されます。

例

以下のステートメントでは、Trend1トレンドの現在の左スクータ位置の値に対応する時刻が秒単位で取得されます。

```
HTGetTimeAtScooter("Trend1",Trend1.UpdateCount,1, Trend1.ScooterPosLeft);
```

HTGetStringAtScooter() 関数

HTGetStringAtScooter() 関数は、特定のスクータの位置のサンプルに対応する時刻／日付を含む文字列を返します。

カテゴリ

履歴

構文

```
MessageResult=HTGetTimeStringAtScooter(Hist_Tag, UpdateCount, ScootNum, ScootLoc, Format_Text);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

UpdateCount

トレンドの.UpdateCount ドットフィールドを示す整数値。

ScootNum

左または右スクータを示す整数値：

1 = 左スクータ

2 = 右スクータ

ScootLoc

トレンド上の .ScooterPosRight または .ScooterPosLeft の位置の値を示す実数値。

Format_Text

時刻／日付に使用する形式を指定する文字列。以下の Format_Text 文字列を受け入れることができます。

"Date"、"Time"、"DateTime"、"DOWShort"（たとえば Wed）、および "DOWLong"（たとえば Wednesday）

備考

引数の UpdateCount、ScootNum、および ScootLoc に割り当てられた値が変わると、この式の値が再度求められます。これにより、新しいデータの検索後やスクータの移動後にこの式が評価されて値が更新されます。文字列の形式によって返り値の内容が決定されます。

例

以下のステートメントでは、Trend1 トレンドの現在の右スクータ位置の値に対応する日付と時刻が取得されます。この値は、メッセージ型タグ変数の NewRightTimeString に "Time" 形式で格納されます。

```
NewRightTimeString=HTGetTimeStringAtScooter ("Trend1",Trend1.UpdateCount,2, Trend1.ScooterPosRight,"Time");
```

HTGetValue() 関数

HTGetValue() 関数は、トレンドの指定ペンに対して要求したタイプの値を返します。

カテゴリ

履歴

構文

```
RealResult=HTGetValue(Hist_Tag,UpdateCount, PenNum,ValType_Text);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

UpdateCount

トレンドの.UpdateCount ドットフィールドを示す整数値。

PenNum

トレンドのペン番号（1～8）を示す整数値か整数型タグ変数。

ValType_Text

戻り値のタイプを示す文字列：

PenAverageValue = トレンド全体の平均

PenMaxValue = トレンド全体の最大値

PenMinValue = トレンド全体の最小値

PenMaxEU = トレンド全体の工学値最大値

PenMinEU = トレンド全体の工学値最小値

PenStdDev = トレンド全体の標準偏差

備考

この関数は、要求した値を実数として返します。

例

以下のステートメントでは、PumpPress トレンドから取得された ペン 2 データ の標準偏差が得られます。この値は、メモリ実数型タグ変数 LeftHemisphereSD に保存されます。

```
LeftHemisphereSD=HTGetValue("PumpPress", PumpPress.UpdateCount,2,"PenStdDev");
```

HTGetValueAtScooter() 関数

HTGetValueAtScooter() 関数は、指定したスクータ位置、トレンド、およびペン番号のサンプルに対して要求したタイプの値を返します。UpdateCount 引数により、関数処理の終了後に式が評価されて値が求められます。

カテゴリ

履歴

構文

```
RealResult=HTGetValueAtScooter(Hist_Tag, UpdateCount,ScootNum,ScootLoc,PenNum,  
ValType_Text);
```

引数**Hist_Tag**

トレンド名が割り当てられた履歴トレンドのタグ変数

UpdateCount

トレンドの.UpdateCount ドットフィールドを示す整数値。

ScootNum

左または右スクータを示す整数値：

1 = 左スクータ

2 = 右スクータ

ScootLoc

トレンドの .ScooterPosRight または .ScooterPosLeft ドットフィールドを示す実数値。

PenNum

ペン番号（1～8）を示す整数値か整数型タグ変数。

ValType_Text

戻り値のタイプを示す文字列：

PenValue = スクータ位置の値。

PenValid = 無効の場合は 0、有効の場合は 1。

HTGetValueAtScooter() 関数に ValType_Text 引数を使用する場合は、上記の有効なタイプのいずれかを使用します。

例

以下の関数の戻り値は、Trend1 トレンドのペン 3 の右スクータ現在位置に対応する値が実際のサンプル値の場合は 1 で、無効値の場合は 0 です。

```
HTGetValueAtScooter("Trend1",Trend1.UpdateCount, 2,Trend1.ScooterPosRight,3, "PenValid");
```

HTGetValueAtZone() 関数

HTGetValueAtZone() 関数は、指定ペンのトレンドの左右スクータ間のデータに対する要求したタイプの値を返します。

カテゴリ

履歴

構文

```
RealResult=HTGetValueAtZone(Hist_Tag,UpdateCount,  
Scoot1Loc,Scoot2Loc,PenNum,ValType_Text);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

UpdateCount

トレンドの .UpdateCount ドットフィールドを示す整数値。関数を評価して値を求めるためのトリガとしてのみ使用されます。

Scoot1Loc

トレンドの .ScooterPosLeft ドットフィールドを示す実数値。関数を評価して値を求めるためのトリガとしてのみ使用されます。

Scoot2Loc

トレンドの .ScooterPosRight ドットフィールドを示す実数値。関数を評価して値を求めるためのトリガとしてのみ使用されます。

PenNum

トレンドのペン番号（1～8）を示す整数値か整数型タグ変数。

ValType_Text

戻り値のタイプを示す文字列

PenAverageValue = スクータ間ゾーンの平均値

PenMaxValue = スクータ間ゾーンの最大値

PenMinValue = スクータ間ゾーンの最小値

PenMaxEU = スクータ間ゾーンの工学値最大値

PenMinEU = スクータ間ゾーンの工学値最小値

PenStdDev = スクータ間ゾーンの標準偏差

備考

戻り値は、指定したタイプの計算値を示す実数です。Scoot1Loc と Scoot2Loc 引数に指定した定数は結果に影響を与えるものではなく、関数を評価して値を更新するためのトリガに使用されるだけです。

Scoot1Loc と Scoot2Loc 引数に指定した値とは関係なく、この関数ではトレンドタグ変数の .ScooterPosLeft と .ScooterPosRight ドットフィールドの値が直接使用されます。

例

以下のステートメントでは、Trend1 トレンドの ペン 1 の左右スクータ間のデータに対する平均値が計算されます。この値は、メモリ実数型タグ変数の AvgValue に格納されます。

```
AvgValue=HTGetValueAtZone("Trend1", Trend1.UpdateCount,Trend1.ScooterPosLeft,  
Trend1.ScooterPosRight,1,"PenAverageValue");
```

トレンドのパンとズームを行う

ランタイム中に履歴トレンドの特定のデータを選択する関数を含む QuickScript を作成できます。

HTScrollLeft() 関数

HTScrollLeft() 関数は、トレンドの表示時間全体に対するパーセントを指定して、トレンドの開始時刻を現在の開始時刻より早い時刻に設定します。その結果、トレンドの表示時間全体に対するパーセントで指定分だけ早い時刻にチャートが左スクロールされます。

カテゴリ

履歴

構文

```
HTScrollLeft(Hist_Tag,Percent);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

Percent

左にスクロールする間隔をチャートの表示時間に対するパーセント（0.0 ～ 100.0）で示した実数値。

例

以下のステートメントは、PumpPress トレンドの幅全体の 10 パーセント分だけ時刻／日付を左スクロールします。

```
HTScrollLeft("PumpPress",10.0);
```

現在の表示開始時刻が 午後 12 時 00 分 00 秒で表示幅が 60 秒の場合、この関数の実行後の新しいトレンドの開始時刻は午前 11 時 59 分 54 秒になります。

HTScrollRight() 関数

HTScrollRight() 関数は、トレンドの表示幅に対するパーセントを指定して、トレンドの開始時刻を現在の開始時刻より遅い時刻に設定します。その結果、トレンドの表示幅に対するパーセントで指定分だけチャートの時刻／日付が右スクロールされます。

カテゴリ

履歴

構文

```
HTScrollRight(Hist_Tag,Percent);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

Percent

チャートを右にスクロールするパーセント (0.0 ~ 100.0) を示す実数値

例

以下のステートメントは、PumpPressトレンドを 20 パーセント分だけ右スクロールします。

```
HTScrollRight("PumpPress",20.0);
```

現在の表示開始時刻が 午後 12 時 00 分 00 秒で表示幅が 60 秒の場合、この関数の実行後の新しいトレンドの開始時刻は午後 12 時 00 分 12 秒になります。

HTZoomIn() 関数

HTZoomIn() 関数は、新しいチャート幅と開始時刻を計算します。トレンドのスクータがトレンドの左右の端にある場合、新しいチャート幅は古いチャート幅の半分になります。新しい開始時刻は、*LockString* 引数の値に基づいて計算されます。

スクータがトレンドの左右の端にない場合は、HTZoomIn() はスクータで定義されるゾーンに対してトレンドをズームして、*LockString* 引数は無視されます。

カテゴリ

履歴

構文

```
HTZoomIn(Hist_Tag,LockString);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

LockString

ズームのタイプを示す文字列：

StartTime	ズーム前の開始時刻を維持する
Center	ズーム前の中央時刻を維持する
EndTime	ズーム前の終了時刻を維持する

備考

スクータがトレンドの左右の端にない場合は、新しいチャート幅は .ScooterPosLeft と .ScooterPosRight の位置の間の時間になります。この場合、*LockString* の値は使用されません。チャート幅の最小値は 1 秒間です。ズーム後のスクータ位置の設定は、.ScooterPosLeft=0.0 および .ScooterPosRight=1.0 になります。

例

下記のステートメントは、Trend1トレンドの表示を 2 倍にズームして、トレンドの開始時刻を維持します。Trend1.ScooterPosRight は 1.0、Trend1.ScooterPosLeft は 0.0 になります。ズーム前の開始時刻が午後 1 時 25 分 00 秒でチャート幅が 30 秒の場合、新しい開始時刻は午後 1 時 25 分 00 秒のままで、新しいチャート幅は 15 秒になります。

```
HTZoomIn("Trend1","StartTime");
```

HTZoomOut() 関数

HTZoomOut() 関数は、新しいチャート幅と開始時刻を計算します。新しいチャート幅は、古いチャート幅に 2 を掛けたものです。新しい開始時刻は、**LockString** 引数の値に基づいて計算されます。

カテゴリ

履歴

構文

```
HTZoomOut(Hist_Tag, LockString);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

LockString

ズームのタイプを示す文字列：

StartTime = ズーム前の開始時刻を維持する

Center = ズーム前の中央時刻を維持する

EndTime = ズーム前の終了時刻を維持する

備考

現在のスクータの位置は **HTZoomOut()** に影響しません。この関数のズームが終了すると、新しいスクータ位置の設定は **.ScooterPosLeft=0.0** および **.ScooterPosRight=1.0** になります。

例

以下のステートメントは、**Volume** トレンドの表示を 2 倍にズームアウトして、トレンドの中央時刻を維持します。ズーム前の開始時刻が午後 2 時 15 分 00 秒でチャート幅が 30 秒の場合、新しい開始時刻は午後 2 時 14 分 45 秒になります。チャート幅は 60 秒で、トレンドの中央時刻は午後 2 時 15 分 15 秒のままです。

```
HTZoomOut("Volume", "Center");
```

トレンドの印刷

PrintHT() 関数をスクリプトで使用して、**WindowViewer** の画面に現在表示されている履歴トレンドを印刷できます。

PrintHT() 関数

PrintHT() 関数は、現在画面に表示されている履歴トレンドを印刷します。**PrintHT()** 関数は通常、履歴トレンドウィンドウに配置されている画面ボタンと関連付けられます。ユーザーはこのボタンをクリックして、表示されている履歴トレンドを現在の値で印刷します。

カテゴリ

履歴

構文

```
PrintHT(Trend_Tag);
```

引数

Trend_Tag

履歴トレンドタグ変数。

例

この例では、現在画面に表示されている PumpPress 履歴トレンドが印刷されます。

```
PrintHT(PumpPress);
```

トレンドのトラブルシューティング

正しく取得されたデータが履歴トレンドに表示されているかを確認する QuickScript を作成できます。トレンドのトラブルシューティングには、HTGetLastError() 関数を使用します。

HTGetLastError() 関数

HTGetLastError() 関数をスクリプトに使用して、特定の履歴トレンド ペンの最終データの取り込み中にエラーが発生したかを確認できます。

カテゴリ

履歴

構文

```
[Result=]HTGetLastError(Hist_Tag,UpdateCount, PenNum);
```

引数

Hist_Tag

トレンド名が割り当てられた履歴トレンドのタグ変数

UpdateCount

トレンドの **.UpdateCount** ドットフィールドを示す整数値。

PenNum

トレンドのペン番号（1～8）を示す整数値か整数型タグ変数。

Result

特定のペンに対する最終のスクリプト関数呼び出しの結果を示す、タグ変数に割り当てられた整数値。

0 = エラーなし

1 = 一般サーバー エラー

2 = 古い要求

3 = ファイル エラー

4 = サーバーの未ロード

5 = 関数に指定したトレンド／ペンが存在しない

6 = トレンドタグ変数がサーバーに存在しない

7 = 関数に指定したペン番号が無効（1～8 の範囲外）

8 = ペン番号にタグ変数が割り当てられていない、またはログ済みでないタグ変数が割り当てられている

例

以下のステートメントでは、Trend1 トレンドのペン 3 に対する最終のデータ取り込みのステータスが取得され、整数型タグ変数 **ResultCode** にその結果が割り当てられます。

```
[ResultCode=]HTGetLastError("Trend1", Trend1.UpdateCount,3);
```

アニメーションのアナログ値の表示 QuickScript では、以下のステートメントを使用します。

```
HTGetLastError("Trend1",Trend1.UpdateCount,3);
```

トレンドでのリアルタイム値の表示

リアルタイムトレンドを作成するには2つの方法があります。リアルタイムトレンドオブジェクトには、データの選択、時間範囲の設定、グラフの外観の指定を行うためのコントロールの標準セットがあります。さらに InTouch には、リアルタイムトレンドおよび履歴トレンドを作成するためのオプションコントロールとして 16 ペントレンドウィザードも用意されています。16 ペントレンドウィザードでのリアルタイムトレンドの作成の詳細については、『AVEVA™ InTouch HMI 管理ガイド』の「[16 ペントレンドの作成](#)」を参照してください

リアルタイムトレンドオブジェクトの使用

リアルタイムトレンドを作成して、アプリケーションで現在の値を表示できます。

リアルタイムトレンドの作成

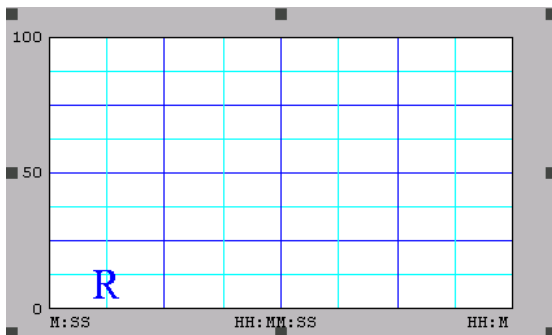
リアルタイムトレンドツールを使用して、ウィンドウ内にトレンドオブジェクトを作成できます。WindowMaker では、ウィンドウに新しく貼り付けたリアルタイムトレンドオブジェクトにはデフォルト設定値が適用されます。リアルタイムトレンドの設定を行うとその後は、新しいリアルタイムトレンドオブジェクトの初期設定として WindowMaker は最後の設定値を使用します。

ウィンドウ枠の中であれば、どんなサイズのトレンドグラフでも表示できます。

リアルタイムトレンドを作成するには

1. [描画] メニューの [トレンド] グループで [リアルタイム] をクリックします。
2. リアルタイムトレンドを作成するウィンドウの領域の上にマウスを移動します。マウスを斜めにドラッグして、目的のトレンドのサイズの長方形を作成します。

ウィンドウにリアルタイムトレンドオブジェクトが表示されます。



3. 必要に応じて、オブジェクトのハンドルを使用してトレンドの高さと幅を調整します。

リアルタイムトレンドに表示するタグ変数の選択

リアルタイムトレンドペンによって、ローカルタグ変数や1つ以上のローカルタグ変数が含まれる式の現在データのグラフが作成されます。リアルタイムトレンドにタグ変数データを表示するペンの設定を行います。

リアルタイムトレンドタグ変数を設定するには

1. ウィンドウ内のトレンドオブジェクトをダブルクリックします。[リアルタイムトレンドグラフの設定] ダイアログボックスが表示されます。

2. [式] 領域に、ローカル タグ変数か 1 つ以上のローカル タグ変数を含む式の名前を入力します。
[ペン] ボックス内をダブルクリックすると、[タグ変数を選択してください] ダイアログボックスが表示され、アプリケーションに定義されたタグ変数のリストが表示されます。[タグ変数を選択してください] ダイアログボックスでタグ変数を選択することにより、タグ変数をペンに割り当てることができます。
3. タグ変数が割り当てられた各ペンの横にあるカラー ボックスをクリックすると、カラー パレットが開きます。
4. ペンに割り当てる色をクリックします。
5. [ペン幅] ボックスに、トレンドに表示される各ペンの線幅をピクセル値で入力します。
線幅に 1 よりも大きい値を選択すると、トレンドの更新や印刷に要する時間が長くなります。
6. トレンドがアクティブ ウィンドウに表示されている間だけトレンドを更新するには、[メモリ内常駐時のみ更新] チェック ボックスをオンにします。
このオプションを選択しないと、ウィンドウが閉じている場合でも常にトレンドは更新されます。トレンドが常時更新されていると、システムのパフォーマンスが低下します。
7. [リアルタイムトレンドグラフの設定] ダイアログ ボックスを開いたまま、[「リアルタイムトレンドの時間範囲と更新率の設定」](#)に記載されている次の手順に進みます。

リアルタイムトレンドの時間範囲と更新率の設定

リアルタイムトレンドの時間範囲と更新率を設定できます。

リアルタイムトレンドの時間範囲と更新率を設定するには

1. トレンドオブジェクトをダブルクリックします。[リアルタイムトレンドの設定] ダイアログボックスが表示されます。
2. [時間] 領域の [時間範囲] ボックスに、トレンドの横軸 (X 軸) に表示する時間長を入力します。
3. トレンド時間の単位を選択します。
 - 秒
 - 分
 - 時

たとえば、[時間範囲] ボックスに 30 と入力してから [分] を選択すると、チャートに表示される時間範囲は 30 分になります。

4. [サンプル] 領域の [間隔] ボックスに、トレンドの式を評価してチャートを更新する間隔の値を入力します。
5. 間隔の単位を選択します。

- ミリ秒
- 秒
- 分
- 時

たとえば、[間隔] に 10 を入力してから [秒] を選択すると、リアルタイムトレンドは 10 秒ごとに更新されます。

6. [リアルタイムトレンドの設定] ダイアログボックスを開いたまま、「[リアルタイムトレンドの表示オプションの設定](#)」に記載されている次の手順に進みます。

リアルタイムトレンドの表示オプションの設定

リアルタイムトレンドの表示方式を設定することができます。

リアルタイムトレンド表示オプションを設定するには

1. トレンドオブジェクトをダブルクリックします。[リアルタイムトレンドグラフの設定] ダイアログボックスが表示されます。
2. [配色] 領域で色を設定します。以下のいずれかを実行します。
 - [背景の色] ボックスをクリックしてカラーパレットを開きます。トレンドの背景色を選択します。
デフォルトの背景色は白です。背景色を別の色にすると、トレンドの印刷速度が著しく遅くなります。
 - [輪郭線の色] ボックスをクリックしてカラーパレットを開きます。トレンドの輪郭色を選択します。
3. [時間軸の目盛り] 領域で、時間軸の目盛りを設定します。以下の操作を行います。
 - [主目盛り数] ボックスに、トレンドの主目盛り数を入力します。主目盛りは、トレンドの横軸（時間軸）に表示されます。

時間軸の主目盛り数は、[主目盛り間の分割数] の値で割り切れる必要があります。たとえば、目盛り数 20 は、[主目盛り間の分割数] の値 4 で割り切れます。

Time Divisions

Number of Major Div: 4 [Color Selection]

Minor Div/Major Div: 2 [Color Selection]

☐ Top Labels ☒ Bottom Labels

Major Div/Time Label: 2 [Color Selection]

HH:MM:SS Display: ☒ HH ☒ MM ☒ SS

- 主目盛り線の色を選択します。
- [主目盛り間の分割数] ボックスに、各主目盛り内に表示される小目盛りの数を入力します。

時間軸の小目盛りの数は主目盛り間隔内で均等に割り切れる必要があります。たとえば、主目盛り間隔を 60 秒に設定して【主目盛り間の分割数】に 2 を入力すると、時間軸の小目盛り間隔は 30 秒になります。

- 小目盛り線の色を選択します。
- 【上ラベル】 か 【下ラベル】 チェック ボックスをオンにして、トレンドの時間ラベルの表示位置を指定します。

時間ラベルをトレンドの上下に表示するように両方のオプションをオンにできます。両方のオプションをオフのままにすると、トレンドの横軸に時間ラベルは表示されません。

- 時間ラベルを使用する場合は、【主目盛り／時間ラベル】に時間ラベルあたりの時間軸の主目盛り線の数を入力します。主目盛り数は、【主目盛り間の分割数】の値で割り切れる必要があります。

時間軸目盛りのラベルの色を選択します。

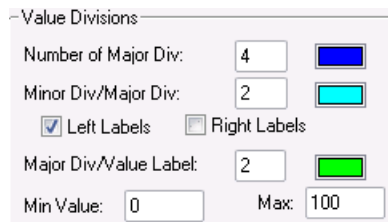
- 時間軸の主目盛りラベルに表示される時間単位を選択します。

時 (HH)

分 (MM)

秒 (SS)

4. 【縦軸の目盛り】領域で、トレンドの縦軸の表示を設定します。



【縦軸の目盛り】オプションの設定は、【時間軸の目盛り】オプションの設定と似ています。y-軸の主目盛りと小目盛りは、時間ではなくデータ値の大きさを表します。縦軸では、すべてのタグ変数の工学単位に基づいて、トレンドに表示するデータ値の範囲が指定されます。

主目盛り値と小目盛り値に小数点を表示するには、【最小値】と【最大値】オプションに実数値を入力します（例：0.00 ～ 100.00）。

5. 【表示フォント】をクリックします。【フォント】ダイアログボックスが表示されて、トレンドに表示されるテキストのフォント、スタイル、サイズの設定オプションが表示されます。
6. 【OK】をクリックします。

ランタイム中のトレンド印刷

トレンドを高速印刷できるようにするにはいくつかの要素があります。第 1 の要素は、印刷ページ上のトレンドのサイズです。また印刷速度は、選択した表示モードによっても異なります。【最大／最小】と【平均／散布】トレンドは、【平均／棒】トレンドよりも短い時間で印刷されます。また、線が長く太いほど印刷速度が遅くなります。

印刷速度に影響する別の要因には、トレンドの背景色があります。ほとんどの場合、背景に色をつけずに白の背景で印刷するほうが高速です。

トレンド印刷オプションの設定

トレンドの印刷方法を決めるオプションを設定できます。

履歴トレンドの印刷を設定するには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[履歴ログ] をクリックします。
[履歴ログ] 設定画面が表示されます。
3. [印刷制御] タブを選択します。

Historical logging Historian logging **Printing control**

Default % of page to print on: 50 ▼ ▲ % Max consecutive time to spend printing: 500 ▼ ▲ msec

Font: Microsoft Sans Serif ▼ Size: 12 ▼ Time to wait between printing: 2000 ▼ ▲ msec

☐ Always use color when printing

4. トレンドを印刷するページの割合を [履歴トレンドグラフの印刷割合] ボックスで指定します。
たとえば、50 と入力すると、縦横ともページの半分に印刷されます。50 パーセントでトレンドを印刷すると、全ページ印刷よりもはるかに高速になります。

PrintWindow() QuickScript 関数を使用して印刷することもできます。

5. [連続印刷処理時間] ボックスに、処理タイム スライスミリ秒単位を入力します。
タイム スライスは、印刷モジュールをバックグラウンドで実行してトレンドを印刷するためにコンピュータ処理に割り当てられる時間を表します。トレンド印刷に利用できるタイム スライスの時間を長くすれば高速印刷できますが、コンピュータ上で実行されている他の処理が遅くなります。
6. [印刷処理の実行間隔] ボックスに、印刷モジュールがプロセッサ タイム スライス間で待機する時間をミリ秒単位で入力します。
プロセッサ タイム スライス間の待機期間が短くなると、トレンドをより迅速に印刷できます。
7. [フォント] ドロップダウンリストから、トレンドに表示する [印字フォント] を選択します。
8. カラー印刷の [印刷時に常に色を使用] チェック ボックスをオンにします。
9. [保存] をクリックします。

他の InTouch ノードや Historian Server の履歴タグ値の表示

リモート保存されたデータを使用して履歴トレンドを作成するには、InTouch 履歴プロバイダリストにリモートプロバイダを登録する必要があります。このリストには、各履歴プロバイダの名前とネットワーク位置が指定されます。履歴トレンドペンがリモート履歴プロバイダのタグを指し示すと、そのたびにこの名前が参照されます。

リモート履歴プロバイダを定義して、リモート位置に保存されたタグのデータに履歴トレンド ペンを割り当てることができます。以下の操作を実行できます。

- リモート履歴プロバイダを設定する
- リモート履歴プロバイダのデータを表示するようにペンを設定する
- タグブラウザを使用して、リモート履歴プロバイダに保存されたタグのデータにペンを割り当てる
- QuickScript を使用して、リモート タグにペンを割り当てる

リモート履歴プロバイダのデータの使用の詳細については、『AVEVA™ InTouch HMI アプリケーション配置ガイド』の「アプリケーションの分散」を参照してください。

Historian Server での InTouch HMI の使用

Historian Server は、産業用アプリケーション向けに特に設計されたリアルタイムのリレーショナルデータベースです。Historical Logger を設定して、InTouch 履歴データを AVEVA 履歴サーバー データベースの Historian に保存できます（オプション）。

注記: リモート履歴プロバイダでの InTouch HMI の設定の詳細については、『AVEVA™ InTouch HMI アプリケーション配置ガイド』の「アプリケーションの分散」を参照してください。

Historian を使用して履歴データを保存する場合、データベースへの接続を指定するために WindowMaker の分散名前マネージャを使用する必要があります。

Historian Server データベースへの接続を設定するには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[分散名前マネージャ] をクリックします。
[分散アラーム] のタブおよび [分散履歴] のタブを含む [分散名前マネージャ] 設定画面が表示されます。
3. [分散履歴] タブで [+] アイコンをクリックして履歴プロバイダを追加します。または、Alt+A を押します。
[履歴プロバイダの追加] ダイアログが表示されます。

Add history provider

Provider name
TankFarmServer

☐ InTouch provider UNC name

☒ Historian

AVEVA history provider

Provider name: TankFarmServer

Data source: Galaxy01 Credentials: Credential1

Test connection

Cancel Add

4. [プロバイダ名] ボックスに、新しい履歴プロバイダの名前を入力します。
プロバイダ名には、16 文字以下の英数字を使用できます。
5. [Historian] を選択します。
 - a. [データ ソース] ボックスに、Historian Server がインストールされているサーバーのノード名を入力します。
 - b. [資格情報] ドロップダウンで認証に使用する資格情報を選択します。

注記: スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーションマネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報マネージャから取得されます。詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[資格情報マネージャの操作](#)」を参照してください。

- a. [接続テスト] をクリックして、Historian Server データベースへの接続を確認します。データベースへの接続が成功したかどうかを通知するメッセージが表示されます。
6. [追加] をクリックして、ダイアログ ボックスを閉じます。[履歴プロバイダ] リストに Historian Server ノードが表示されます。
7. [保存] をクリックします。

リモートトレンドデータを表示するためのペンの設定

履歴トレンドには、ローカルとリモートの両方の履歴プロバイダからのタグ変数データを表示できます。リモート履歴プロバイダのデータを表示するようにトレンドペンを指定することができます。

リモート履歴プロバイダのタグ変数を表示するには

- 履歴トレンドをダブルクリックして[履歴トレンドグラフの設定] ダイアログ ボックスを表示します。
- 各ペンの[タグ変数] ボックスに、リモート履歴プロバイダへのリファレンスを入力します。リモート履歴プロバイダへのリファレンスの形式は以下のとおりです。

history_provider_name.tag_name

例:

TankFarm1.Pump1RPM

履歴トレンドの各ペンは、異なるリモート履歴プロバイダを参照することができます。

- [OK] をクリックして、設定変更を保存します。

注: .TagID ドットフィールドは、リモート履歴プロバイダのタグリファレンスには使用できません。

タグブラウザを使用してリモート履歴プロバイダにペンを割り当てる

タグブラウザを使用してリモート履歴プロバイダのタグ変数データにトレンドペンを割り当てるための手順を以下に説明します。タグブラウザを使用してタグ変数を選択することにより、各タグ変数名を手入力する必要がなくなり、エラー発生の確率を低減できます。

アクセス名に指定するリモート ノード名は、タグ変数を含むノードの実際の名前である必要はありません。ただし、アクセス名を作成して、リモート履歴プロバイダをタグソースとして定義する必要があります。I/O 型タグ変数のアクセス名の作成については、「[アクセス名の設定](#)」([アクセス名の設定](#))を参照してください。

リモート履歴プロバイダをタグソースとして定義するには

- 履歴プロバイダのあるノード名を指定するアクセス名を作成します。
- 履歴トレンドをダブルクリックして[履歴トレンドグラフの設定] ダイアログ ボックスを開きます。
- ペンの[タグ変数] 入力ボックスをダブルクリックすると、[タグ変数を選択してください] ダイアログ ボックスが表示されます。
- [タグソースの定義] をクリックして、リモート履歴プロバイダをタグソースとして定義します。
- [タグソース] の矢印をクリックして新しいリモート履歴プロバイダのタグソースをリストから選択するか、[ツリービュー] ボタンをクリックしてツリービュー ペインからタグソースを選択します。[タグ変数を選択してください] ダイアログ ボックスが更新されて、選択したリモート履歴プロバイダのタグ変数が表示されます。
- 履歴ペンに割り当てるタグ変数を選択して、[OK] をクリックします。[履歴トレンドグラフの設定] ダイアログ ボックスに戻り、選択したタグ変数がペンの[タグ変数] ボックスに[アクセス名: アイテム名] の形式で表示されます。
- アクセス名の部分を[分散名前マネージャ] で定義した履歴プロバイダ名で置き換えます。

たとえば、HistPrv1.tag_name

この手順は面倒なように見えますが、タグブラウザで履歴プロバイダをタグソースとして定義しておく、別のタグ変数入力フィールドをダブルクリックした際に、タグブラウザでタグ変数名をダブルクリックしてアクセス名の部分を履歴プロバイダ名で置き換えるだけで済むようになります。

履歴トレンドをランタイム中に変更できるよう設定されている場合は、**WindowViewer** でユーザーがペン ボタンをクリックしてタグ変数を変更しようとするときタグブラウザが表示されますが、変更できるのはローカル アプリケーションのタグ変数だけです。

QuickScript を使用してリモート履歴プロバイダにペンを割り当てる

InTouch アプリケーションの実行中に、リモート履歴プロバイダのタグ変数データを表示するようにトレンド ペンを設定できます。HTSetPenName() 関数を使用して、リモート履歴プロバイダのタグ リファレンスを指定する QuickScript を作成してください。次に例を示します。

```
HTSetPenName("HistTrendTag", 1, "HistPrv1.Boiler1");
```

この例では、HistPrv1 リモート履歴プロバイダの **Boiler1** タグ変数の値を表示する履歴トレンドのペン番号に **1** が指定されます。

ランタイム中の **履歴トレンド グラフの設定** ダイアログ ボックスと **Pen** ドットフィールドは、リモート履歴プロバイダに対してはサポートされていません。

ユーザー定義型

ユーザー定義型（UDT）は、個々のユーザー定義メンバーで構成される階層データ構造です。各メンバーは、整数、ブール、文字列、その他の一般的なデータ型、またはその他の UDT として独自のデータ型を持つことができます。

UDT を使用すると、複数のレベルの入れ子構造を作成するオプションを提供することによってエンジニアリングの労力を削減できます。例えば、反応炉を表現する個々のタグのセットを作成する代わりに InTouch で 1 つの反応炉 UDT を作成できます。

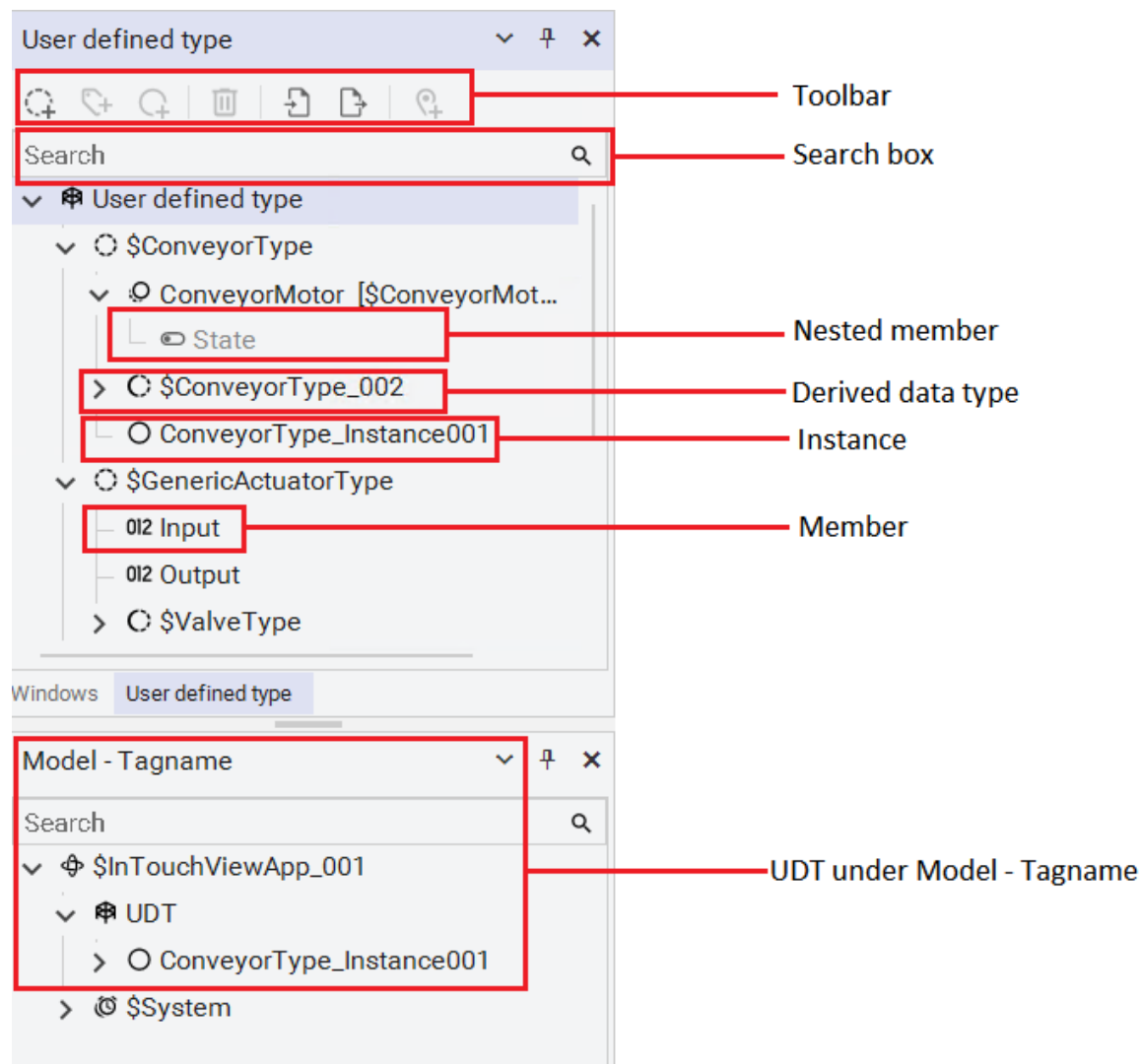
UDT の概要

InTouch HMI は、以下の UDT サポートを提供します。WindowMaker キャンバスの左側に [ユーザー定義型] ペインが表示されます。これには以下が含まれます。

- [検索] ボックス - 目的のデータ型、派生データ型、インスタンス、またはメンバーを検索できます。
- ツールバー - ツールバーには以下のオプションが含まれます。
 - 新規データ型を追加
 - 新規メンバーを追加
 - 新規インスタンスを追加
 - 削除
 - データ型をインポート
 - データ型をエクスポート
 - 場所メンバーを追加

ユーザー定義型階層には、これらのアイテムを以下の順序で含めることができます。

1. **直接メンバー** - 現在のユーザー定義型内で定義されたメンバー。
2. **継承メンバー** - その他のデータ型から継承されたメンバー。これらはグレイアウトされて表示されます。
3. **派生データ型** - ベース データ型から派生したデータ型（ベース データ型は読み取り専用で InTouch で事前定義されています）。これには、派生元のベース データ型の配下のすべてのメンバーが含まれます。既存のアイテムとは別に、メンバーを派生データ型に追加できます。ベース データ型と派生データ型には、デフォルトで \$ 記号の接頭辞が付けられています。派生型の名前を変更した場合、名前の最初の文字として \$ が引き続き使用されます。
4. **インスタンス** - インスタンスはユーザー定義型の物理または仮想オブジェクトを表します。グラフィック アニメーションまたはスクリプトでユーザー定義型を使用するには、インスタンスを作成する必要があります。インスタンス タグ メンバーは、InTouch 基本タグです。



UDT の仕様

ユーザー定義型（UDT）は、以下の仕様に準拠する必要があります。

名前の長さ

- データ型、派生データ型、およびインスタンスの名前の最大長は 32 文字です。
- メンバー データ型とメンバーの名前の最大長は 32 文字です。

有効な名前

- インスタンス名には、英数字に加えて、\$（ドル記号）、#（ポンド記号）、_（アンダースコア）の特殊文字を含めることができます。
- \$ 文字は最初の文字として使用できません。

最大のネストレベル

- 最大で 6 つのネストされた UDT レベルを使用できます。

たとえば、"Datatype1.Datatype2.Datatype3.Datatype4.Datatype5.Datatype6.Member1" の場合を考えてみます。

構文の要素を以下に示します。

- "Datatype1.Datatype2.Datatype3.Datatype4.Datatype5.Datatype6" は 6 つのネストされた UDT レベルです。
- "Member1" は、メモリ整数型などの基本 InTouch タグタイプです。

注記: 循環参照はサポートされません。

手動でのユーザー定義型の作成

ユーザー定義型の作成および管理は、WindowMaker の [ユーザー定義型] ペインで行うことができます。独立した複数の操作のセットを使用して、個々の UDT を作成し、複数の UDT で構成される階層構造を構築できます。以下のトピックでは、これらの操作について説明します。

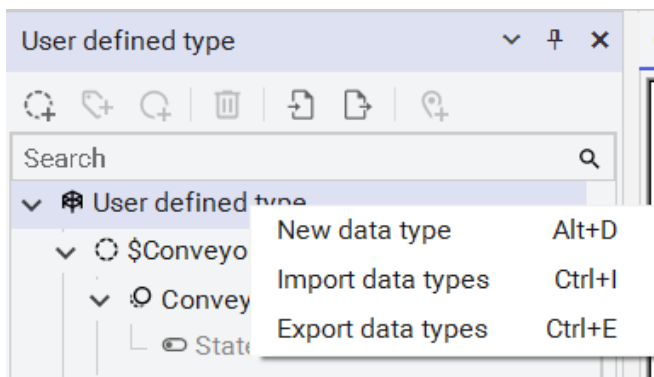
- [新しいデータ型の作成](#)
- [新しいメンバーの作成](#)
- [新しい派生データ型の作成](#)
- [新しいインスタンスの作成](#)
- [別のデータ型の下でのメンバーの入れ子](#)
- [インポートによるユーザー定義型のセットの構築](#)
- [データ型のエクスポート](#)
- [ユーザー定義型の管理](#)
- [ユーザー定義型の一括編集](#)

注記: このセクションで示される UDT 名、メンバー名、およびプロパティ名は説明のみを目的としています。型とメンバーには実際のアプリケーションに適切な名前を付けてください。


新しいデータ型の作成

新しいデータ型を作成するには

1. [ユーザー定義型] ペインで、[ユーザー定義型] を右クリックし、コンテキストメニューから [新規データ型] を選択するか、ツールバーの [新規データ型] アイコンをクリックします。



または

ツールバーの [新規データ型]  をクリックします。

または

ショートカットキー **Alt+D** を使用します。

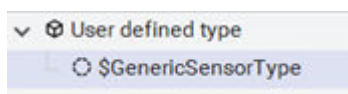
データ型が作成されます。

デフォルトのデータ型名は **\$Data type_00n** です。*n* の初期値は **1** です。データ型の名前の先頭には **\$** 記号が付きます。**\$** 記号を削除することはできません。データ型の名前には、英数字に加えて **\$**、**#**、および **_** (アンダースコア) 文字を使用できます。名前の最大長は **32** 文字です。

2. データ型の名前を変更するには、データ型を右クリックして **[名前の変更]** を選択します。

例:

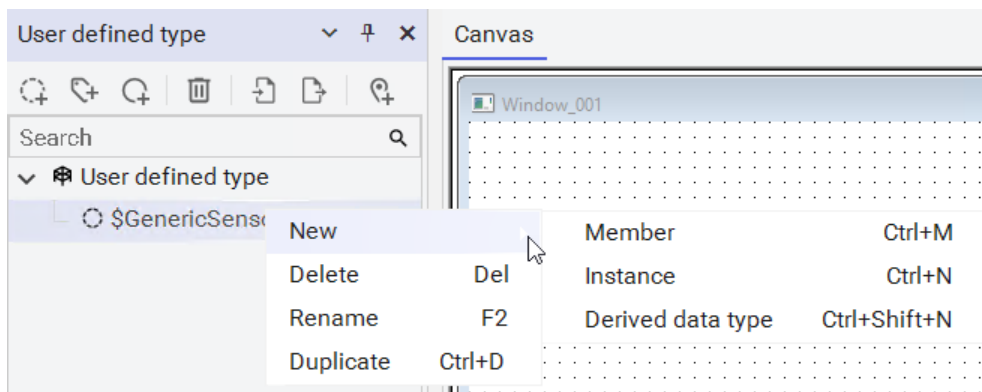
新しいデータ型の名前を **GenericSensorType** に変更します。




新しいメンバーの作成

メンバー作成するには

1. **[ユーザー定義型]** ペインでデータ型を右クリックし、コンテキストメニューから **[新規作成]** して **[メンバー]** を選択するか、**[新規メンバー]** アイコンをクリックします。



または

ツールバーの [新規メンバー]  をクリックします。

または

ショートカットキー **Ctrl+M** を使用します。

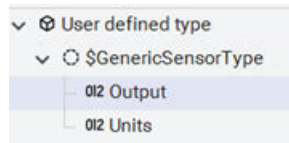
新しいメンバー **Member001** が作成されます。メンバーの名前には、英数字に加えて **\$**、**#**、および **_** (アンダースコア) 文字を使用できます。メンバー名の最大長は **32** 文字です。

2. メンバーの名前を変更するには、メンバーを右クリックして **[名前の変更]** を選択します。

例:

次のような基本 InTouch タグ タイプの 2 つのメンバーを追加できます。

- 出力 - 型: 整数
- 単位 - 型: メッセージ (文字列)



右側のペインの [プロパティ] グリッドでタグのプロパティを設定することもできます。

例:

出力メンバー タグの「初期値」を 50 に設定します。

Details	
Initial value	50
Eng units	
Min value	-32768
Max value	32767

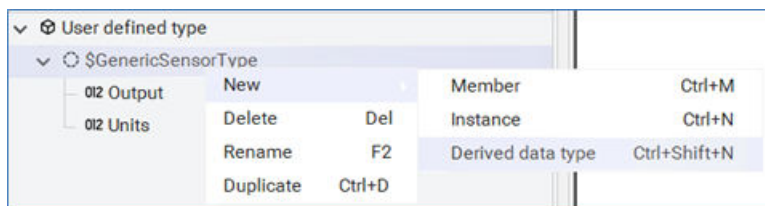
プロパティの詳細については、「[プロパティ グリッドでのユーザー定義型の編集](#)」を参照してください。

新しい派生データ型の作成

データ型を派生させて、そのすべてのメンバーを継承できます。

派生データ型を作成するには

1. [ユーザー定義型] ペインで目的のデータ型を右クリックして、コンテキスト メニューから [新規] を選択し、[派生データ型] を選択します。



または

ショートカット キー **Ctrl+Shift+N** を使用します。

新しい派生データ型が作成されます。

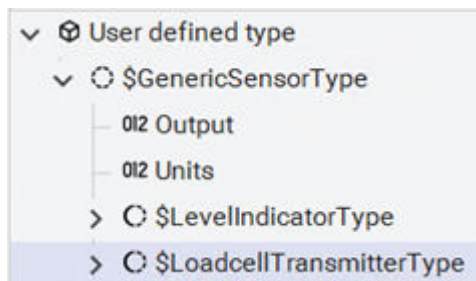
派生データ型のデフォルト名は <親データ型の名前>_00n です。n の初期値は 1 です。たとえば、\$Pump から派生データ型を作成した場合、最初の派生データ型のデフォルト名は "\$Pump_001" です。派生データ型の名前には \$ 記号の接頭辞が付けられます。この \$ 記号を削除することはできません。派生データ型の名前には、英数字に加えて \$、#、および _ (アンダースコア) 文字を使用できます。名前の最大長は 32 文字です。

ベース データ型のメンバーが派生データ型に追加されます。[名前] および [タイプ] のプロパティを除く親データ型のメンバーのプロパティを変更できます。新しいメンバーを派生データ型に追

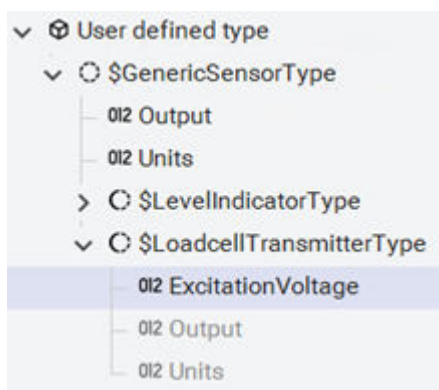
加できます。継承メンバーはグレイアウトされます。派生データ型のプロパティは、[プロパティ] ペインで変更できます。

例:

GenericSensorType から 2 つの派生データ型を作成し、LevelIndicatorType と LoadcellTransmitterType という名前を付けます。



次に、新しいメンバーを LoadcellTransmitterType データ型に追加し、整数型の ExcitationVoltage という名前を付けます。

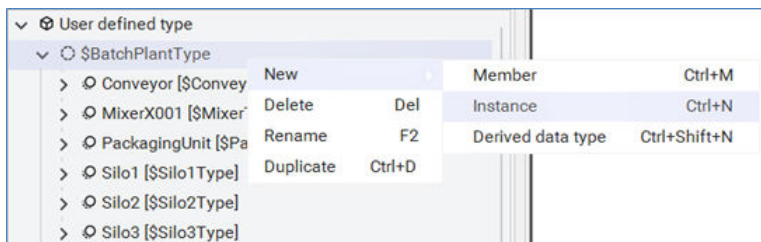


派生データ型は、親のすべてのデータメンバーを継承し、グレイアウトとして表示されます。


新しいインスタンスの作成

インスタンスを作成するには

1. [ユーザー定義型] ペインで目的のデータ型を右クリックして、コンテキストメニューから [新規] を選択し、[インスタンス] を選択します。ツールバーの [新規インスタンス] アイコンをクリックすることもできます。



または

ツールバーの [新規インスタンス]  をクリックします。

または

ショートカット キー **Ctrl+N** を使用します。

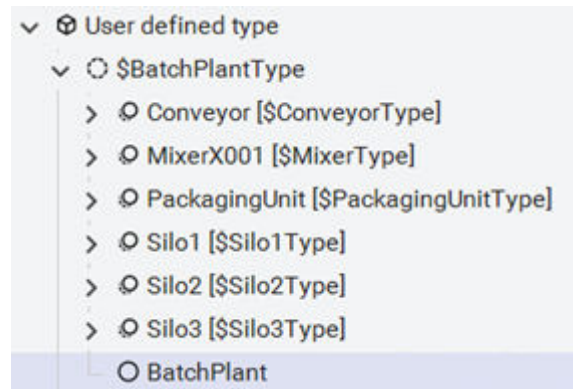
新しいインスタンスが作成されます。

2. UDT の名前を変更します。インスタンスのデフォルト名は <データ型名>_00*n* です。*n* の初期値は 1 です。インスタンス名には \$ 文字の接頭辞は付きません。インスタンスの名前には、英数字に加えて \$、#、および _ (アンダースコア) 文字を使用できます。名前の最大長は 32 文字です。

ベース データ型のメンバーがインスタンスに追加されます。新しいメンバーをインスタンスに追加することはできません。継承メンバーはグレーで表示されます。インスタンスのプロパティは、[プロパティ] ペインで変更できます。[名前] および [タイプ] のプロパティを除くメンバーのプロパティを変更できます。1 つのデータ型または派生データ型の配下のすべてのメンバーは、同じデータ型または派生データ型の配下のインスタンスに属します。どのメンバーがどのインスタンスに属しているかは [モデル-タグ名] ペインで確認できます。すべてのインスタンスは、そのメンバと共に [モデル-タグ名] ペインの [UDT] に表示されます。

例:

\$BatchPlantType からインスタンスを作成して BatchPlant という名前を付けます。

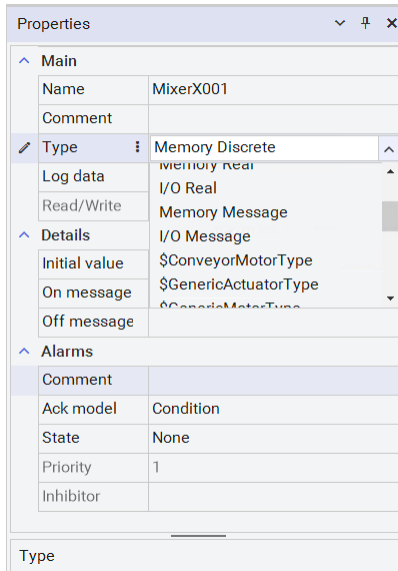


別のデータ型の下でのメンバーの入れ子

UDT のネストは、別のデータ型のメンバーを作成することによって行います。[プロパティ] ペインで 1 つのデータ型のメンバーを別のデータ型にポイントできます。入れ子を使用すると、タグを作成する時間と労力を削減できます。最大 6 つのレベルの UDT の入れ子がサポートされています。

別のデータ型の下でメンバーを入れ子にするには

1. 入れ子にするメンバーを選択します。
2. [プロパティ] ペインの [タイプ] フィールドで、入れ子を作成するデータ型または派生データ型を選択します。[型] フィールドにデータ型の名前を入力してデータ型をフィルタすることもできます。



3. 選択したデータ型または派生データ型の下でメンバーが入れ子になります。この名前は変更できません。たとえば、入れ子にされたメンバーの名前を "InletValve" に変更します。

注記: この名前は、データ型のもう 1 つのインスタンスではありません。これは、データ型インスタンスのデータ型メンバーを参照するもう 1 つの方法です。

例:

"InletValve" は実際のインスタンスではありません。これはタグ名ディクショナリのルートには表示されず、クロスリファレンスユーティリティのタグ階層のルートに表示されません。入れ子にされたメンバーをインスタンスにマップして、スクリプトとアニメーションで参照することができます。

UDT の最大の入れ子レベルは 6 です。以下の例を考えてみます。

Datatype1.Datatype2.Datatype3.Datatype4.Datatype5.Datatype6.Member1

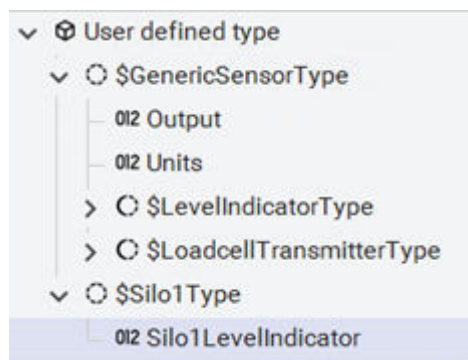
この例の詳細を以下に示します。

- "Datatype1.Datatype2.Datatype3.Datatype4.Datatype5.Datatype6" は UDT の 6 つのレベルです。
- "Member1" は、メモリ整数型などの基本 InTouch タグタイプです。

注記: 循環参照はサポートされません。

例:

新しいデータ型を作成して **Silo1Type** という名前を付け、**Silo1LevelIndicator** というメンバーを追加します。



[プロパティ] グリッドから、プロパティ [タイプ] を別のデータ型に変更できます。LevelIndicatorType を選択したとします。

Main	
Name	Silo1LevelIndicator
Comment	
Type	Memory Integer
Log data	I/O Real
Retentive parameters	Memory Message
Read/Write	I/O Message
Details	GenericSensorType
Initial value	LevelIndicatorType
	LoadcellTransmitterType

ネストされたメンバーが追加されます。



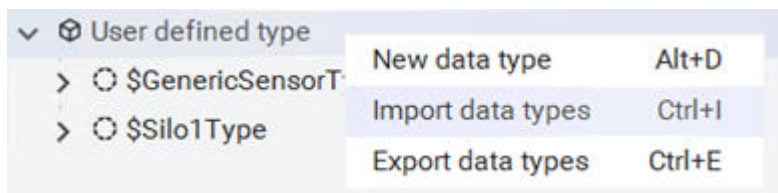
インポートによるユーザー定義型のセットの構築

以前に作成した UDT のセットを含む .json ファイルをアプリケーションにインポートすることによって UDT のセットを効率的に構築できます。

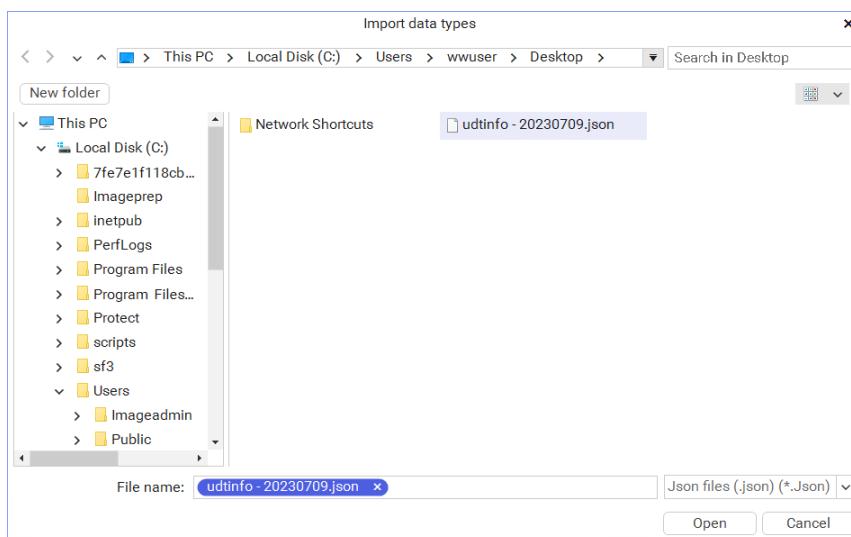
データ型をインポートするには

1. [ユーザー定義型] ペインを開きます。

2. [ユーザー定義型] を右クリックしてコンテキストメニューから [データ型をインポート] を選択し、ツールバーの [インポート] アイコンをクリックします。



3. データ型設定の詳細を含む .json ファイルにナビゲートし、目的のファイルを選択して [開く] を選択します。



4. [ユーザー定義型をインポート] ダイアログ ボックスで以下の操作を行います。
- 既存のデータ型と同じ名前のデータ型のインポートをスキップする場合、[スキップ: コピーしない] を選択します。
 - データ型の名前が既存のデータ型と同じときに既存のデータ型を上書きして置き換える場合、[上書き: 既存のデータ型を置換] を選択します。

Import user defined types

Data types with same name as existing data types

- ☐ Skip: Do not copy
- ☒ Overwrite: Replace existing data types regardless

Cancel

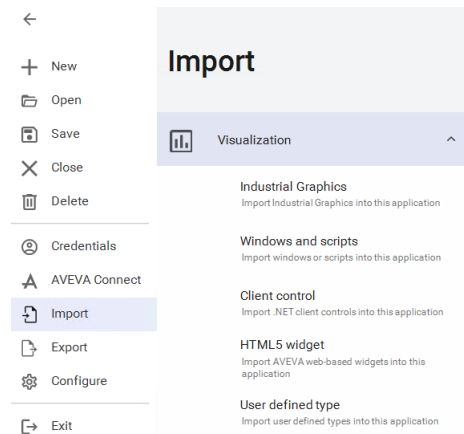
Ok

5. [OK] をクリックします。

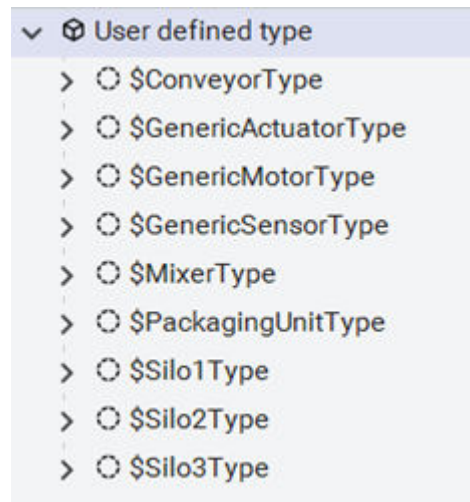
前の手順で選択したオプションに従って、.json ファイルで使用可能なすべてのデータ型とその他のアイテムが表示されます。

注記: 上書きされたインスタンス値は、ファイルをインポートするときに保持されません。

[ファイル] メニューの [インポート] > [表示] > [ユーザー定義型] オプションを使用してデータ型をインポートすることもできます。



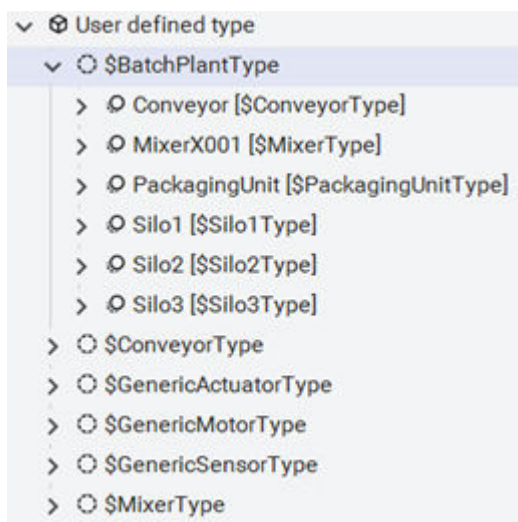
ファイルをインポートすると、UDT 階層が表示されます。



階層には他の UDT を追加できます。その場合、「BatchPlantType」という新しいデータ型を作成してメンバーを追加し、以下に示すように、各メンバーのプロパティグリッドで型を変更します。

- Conveyor - 型 \$ConveyorType
- MixerX001 - 型 \$MixerType
- PackagingUnit - 型 \$PackagingUnitType
- Silo1 - 型 \$Silo1Type
- Silo2 - 型 \$Silo2Type
- Silo3 - 型 \$Silo3Type

結果の階層に新しいメンバーが反映されます。



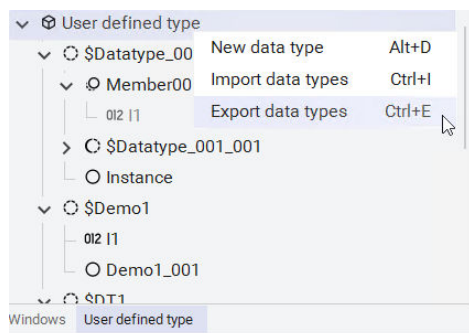
データ型のエクスポート

その他のアプリケーション、システムのその他のコンピュータ、または別のシステムのその他のコンピュータで使用するためにデータ型をエクスポートできます。

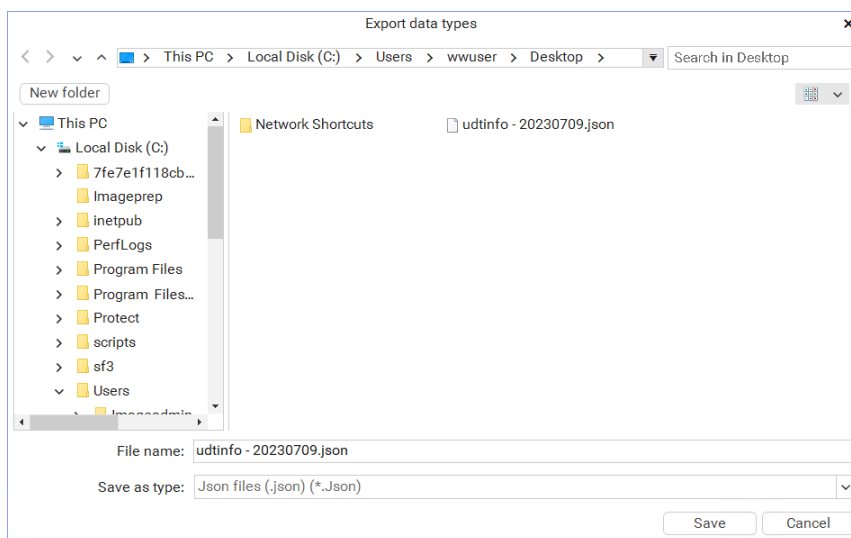
データ型をインエクスポートするには

1. [ユーザー定義型] ペインを開きます。

[ユーザー定義型] を右クリックしてコンテキストメニューから [データ型をエクスポート] を選択するか、ツールバーの [エクスポート] アイコンをクリックします。



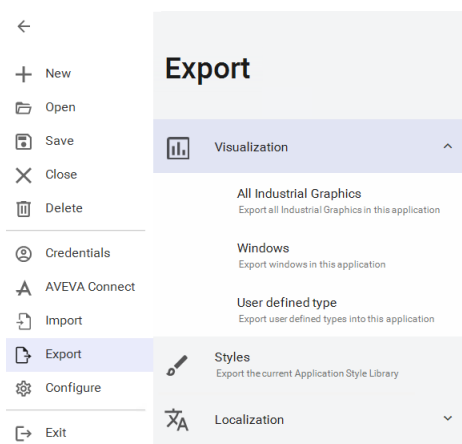
[データ型のエクスポート] ウィンドウが開きます。



2. 必要に応じてファイル名を変更し、**【保存】**を選択します。

選択した場所に .json ファイルが保存されます。エクスポートしたファイルは、外部の JSON エディタを使用して編集し、元のアプリケーションまたは別のアプリケーションにインポートできます。

【ファイル】メニューの【エクスポート】>【表示】>【ユーザー定義型】オプションを使用してデータをエクスポートすることもできます。



注記: エクスポート中、すべての UDT が JSON ファイルに含まれます。

ユーザー定義型の管理

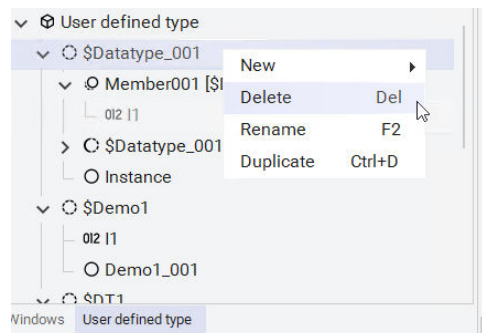
前のトピックで説明されている UDT を作成および更新する操作に加えて、基本的な管理操作を実行できます。

メンバーを削除するには

- 削除するメンバーを右クリックして、コンテキストメニューから**【削除】**をクリックします。ツールバーの**【削除】**アイコンを選択してメンバーを削除することもできます。複数のメンバーを選択するには **Ctrl** キーを使用します。

データ型、派生データ型、またはインスタンスを削除するには

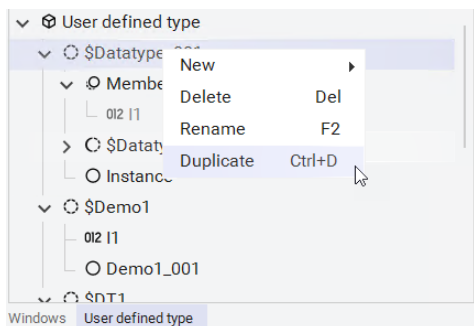
データ型、派生データ型、またはインスタンスを右クリックしてコンテキストメニューから **削除** を選択するか、ツールバーの **削除** アイコンをクリックします。データ型およびそのすべてのメンバーが削除されます。



注記: データ型を削除するには、配下のインスタンスと派生データ型を削除する必要があります。


データ型を複製するには

- **[ユーザー定義型]** ペインでベースデータ型を右クリックして、コンテキストメニューから **複製** を選択します。すべてのメンバーを含むベースデータ型のコピーが作成されます。



注記: 複製できるのはデータ型と派生データ型だけです。インスタンスとメンバーを複製することはできません。

場所属性を追加するには

1. **[ユーザー定義型]** ペインを開きます。
2. データ型または派生データ型を選択して、ツールバーの **場所を追加**  アイコンをクリックします。

緯度と経度の新しい属性がデータ型または派生データ型に追加されます。場所属性の使用の詳細については、このセクションの「[MapApp](#)」トピックを参照してください。

ユーザー定義型の一括編集

JSON ファイルを使用して複数の UDT を一括編集できます。エクスポートされた JSON ファイルを編集するには、一般的な JSON エディタを使用して UDT を追加、削除、または編集します。

UDTを一括で編集または作成するには

1. 既存の複数の UDT を JSON ファイルにエクスポートします。UDT をエクスポートする方法の詳細については、「[データ型のエクスポート](#)」を参照してください。
2. JSON ファイルをテキスト エディタまたは JSON エディタで開きます。
3. 既存の UDT をコピーしてファイルに貼り付けて UDT を編集または作成します。
4. 必要に応じて、その他のプロパティの名前の変更または編集を行います。
5. JSON ファイルを保存します。
6. JSON ファイルを WindowMaker の [ユーザー定義型] ペインにインポートします。インポートの詳細については、「[インポートによるユーザー定義型のセットの構築](#)」を参照してください。

例:

```

▼ {
  Version : 12
  ▼ Templates : [ 1 item ]
    ▼ 0 : {
      Name : DT1
      Comments : value
      ▼ Members : [ 1 item ]
        ▼ 0 : {
          Name : IODisc
          TagType : AtomicTag
          Source : Reference
          TagComment : value
          AlarmGroup : 1
          LogData : False
          LogEvents : False
          LocalTag : false
          TypeOf : Discrete
          ▶ IoConfig : { 2 props }
          ▶ AlarmConfiguration : { 3 props }
        }
      ]
      ▼ Overrides : [ 0 items ]
    ]
  ]
  ▼ Derived Templates : [ 0 items ]
  ]
  ▼ Instances : [ 3 items ]
    ▼ 0 : {
      Name : DT1_001
      Comments : value
      TypeOf : DT1
      ▼ Overrides : [ 3 items ]
        ▼ 0 : {
          Name : IOInt
          ContextName : value
          DataType : Int
          ▼ Val : {
            DataType : Int
            DefaultValue : value
          }
        }
      ]
    }
  ]
}

```

ユーザー定義型の表示

ユーザー定義型は次の 2 つのいずれかを使用して表示できます。

- [\[ユーザー定義型\] ビュー](#)
- [\[モデル-タグ名\] ビュー](#)

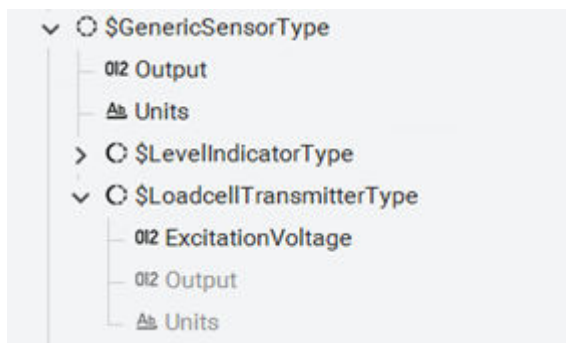
[ユーザー定義型] ビュー

[ユーザー定義型] ビューには、ユーザー定義型、メンバー タグとメンバー データ型、派生データ型、およびインスタンスのリストが表示されます。

- データ型には "\$" 文字の接頭辞が付きます。

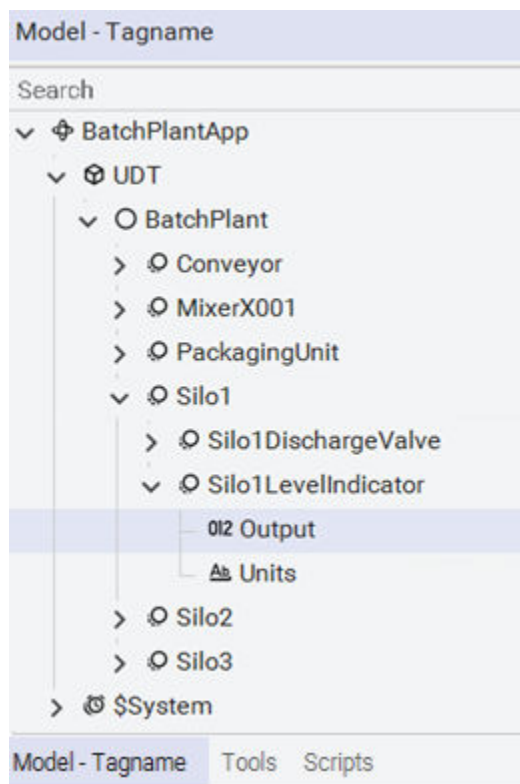
- メンバー データ型には、「[\$<データ型名>]」が追加されます。
- すべてのメンバーと派生データ型は、完全に展開してメンバーを表示できます。
- このビューでは、インスタンス名を展開することはできません。
- 各データ型に対して、すべての子アイテムは以下の示す順番で表示されます。グループ内では、アイテムはアルファベット順に並べ替えられます。
 - a. 直接メンバー
 - b. 継承メンバー
 - c. 派生データ型
 - d. インスタンス

例:



[モデル - タグ名] ビュー

[モデル - タグ名] ビューは、UDT インスタンス ビューが追加された InTouch タグ ビューです。UDT インスタンスとそのメンバーを表示できます。



プロパティ グリッドでのユーザー定義型の編集

プロパティを更新するには

UDT のプロパティは [プロパティ] グリッドで表示および変更できます。選択したアイテムのプロパティは、WindowMaker の右側の [プロパティ] ペインに表示されます。

- 各プロパティをクリックしてプロパティを選択および更新します。対応するプロパティの説明が [プロパティ] ペインの下部に表示されます。
- 行った変更は自動的に保存されます。
- プロパティのフィールドは、[タイプ] プロパティ フィールドで選択したオプションに基づいて変化します。
- [タイプ] プロパティ フィールドには、UDT がその他のデータ型と共に表示されます。
- プロパティのリストが更新され、選択した [タイプ] に関連付けられたプロパティが反映されます。
- [名前] プロパティと [タイプ] プロパティを除き、メンバーが定義されているすべてのタグ プロパティ値を編集できます。

メンバー データ型

メンバーを別のデータ型に変更する場合、以下のプロパティだけが表示されます。この例では、\$BatchPlantType > Silo1 メンバーを選択できます。

Properties	
^ Main	
Name	Silo1
Comment	
Type	Silo1Type

メンバー タグの編集

メンバーの型を基本 InTouch タグに変更すると、関連するすべてのタグ プロパティが編集可能になります。

たとえば、\$GenericSensorType > Units という名前の UDT を選択し、型をメモリ論理型に変更します。プロパティ グリッドが更新され、この理論型タグのすべての関連タグ プロパティが表示されます。

Properties	
^ Main	
Name	Units
Comment	
Type	Memory Discrete
Alarm group	\$System
Log data	<input type="checkbox"/>
Log events	<input type="checkbox"/>
Retentive value	<input type="checkbox"/>
Read/Write	Read Write
^ Details	
Initial value	False
On message	
Off message	
^ Alarms	
Comment	
Ack model	Condition
State	None
Priority	1
Inhibitor	

無効な値と必須フィールド

プロパティの無効な値は赤いアイコンで示されます。プロパティ値を更新し、その他のいずれかのユーザー定義型をクリックすると、プロパティ名の前に赤いエラー アイコンが表示されます。エラー アイコンの上にカーソルを置くと、エラーの理由を示すツールヒントが表示されます。

Properties

Comment	
Type	Memory Integer
Alarm group	\$System
Log data	<input type="checkbox"/>
Log events	<input type="checkbox"/>
Read/Write	Read Write

^ Details

Initial value	0
Eng units	
Min value	: 100
Max value	: 200
Deadband	0
Log deadband	0

^ Alarms

Comment	
Ack model	Condition
LoLo	Enabled
Enable	<input checked="" type="checkbox"/>
Value	10
Priority	1
Inhibitor	

Low Disabled

Value

値が必要なプロパティは赤いフラグで示されます。データ型レベルで値を入力する必要はありません。しかし、インスタンス レベルで値を入力または上書きすることが推奨されます。

例:

Properties

Name	Member001
Comment	
Type	: I/O Discrete
Alarm group	\$System
Log data	<input type="checkbox"/>
Log events	<input type="checkbox"/>
Read/Write	Read Write

^ Details

Initial value	: False
Conversion	Direct
On message	
Off message	
Access name	
Name as item name	<input type="checkbox"/>
Item name	

^ Alarms

Comment	
Ack model	Condition
State	None
Priority	0
Inhibitor	

Type

値の上書きと変更の反映

[名前] プロパティと [タイプ] プロパティを除き、メンバーが定義されているデータ型レベルのすべてのタグプロパティ値を編集できます。

例 1:

1. \$GenericSensorType > Output を選択します。
2. 初期値を 50 に変更します。
3. 次に、\$LevelIndicatorType > Output を選択します。
4. この初期値も 50 であることを確認します。
5. \$GenericSensorType > Output を選択します。
6. 初期値を 60 に変更します。

\$LevelIndicatorType > Output に戻ります。この初期値も 60 に更新されていることを確認します。これは変更の反映の一例です。

7. 初期値を上書きすることもできます。初期値を 75 に設定します。
初期値が太字になり、上書きされたことが示されます。

Details	
Initial value	75
Eng units	

8. \$GenericSensorType > Output を選択して初期値を 65 に変更します。
このインスタンスでは値が上書きされているので、値は \$LevelIndicatorType に反映されません。

例 2:

この例は、前の例 1 の延長です。

1. [モデル - タグ名] ビューで、BatchPlant > Silo1 > Silo1LevelIndicator > Output を選択します。初期値は 75 です。
これは \$LevelIndicatorType データ型から継承されたものです。
2. 25 の上書き値を入力します。
3. BatchPlant > Silo2 > Silo2LevelIndicator > Output を選択して、初期値を 55 で上書きします。
4. BatchPlant > Silo3 > Silo3LevelIndicator > Output を選択します。初期値を 80 で上書きします。

変更の反映

データ型の変更は [ユーザー定義型] に即時反映されます。

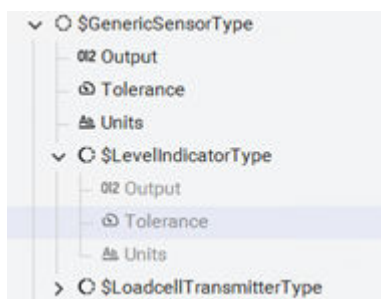
新規メンバーを追加

新規メンバーの追加は、派生データ型、メンバー データ型、およびインスタンスに反映されます。

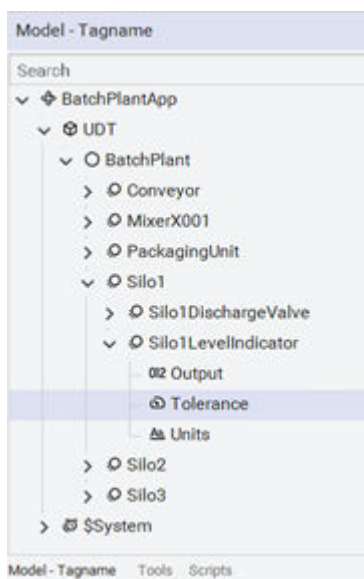
例:

1. \$GenericSensorType を選択します。
2. メモリ実数型の "Tolerance" という新しいメンバーを追加します。

この新しいメンバーは、その派生データ型 `$LevelIndicatorType` で使用可能になります。



この新しいメンバーは、インスタンス メンバー データ型でも使用可能です。



メンバー型の変更

既存のメンバーの変更は、派生データ型、メンバー データ型、およびインスタンスにも直ちに反映されます。

例:

前の例の延長:

1. `$GenericSensorType > Tolerance` に移動します。
2. プロパティ グリッドで、データ型をメモリ実数型からメモリ メッセージ型に変更します。
`$LevelIndicatorType > Tolerance` もメモリ メッセージのデータ型に更新されます。

既存の機能の UDT サポート

注記: 例に示す UDT 名、メンバー名、およびプロパティは、説明目的でのみ使用されています。型とメンバーには実際のアプリケーションに適切な名前を付けてください。

UDT は以下の InTouch 機能でサポートされています。

- [Intellisense](#)

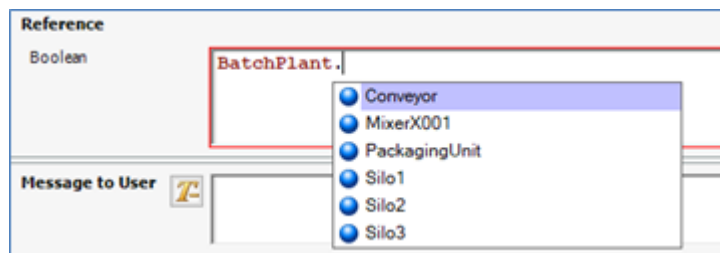
- [アニメーション](#)
- [スクリプト](#)
- [参照の変更](#)
- [UDT メンバーの参照](#)
- [インスタンスの UDT メンバーの産業用グラフィック エディタ キャンバスへのドラッグアンドドロップ](#)
- [アニメーションとスクリプト](#)
- [アラームおよびアラーム クライアント コントロール](#)
- [履歴およびトレンドクライアント](#)
- [I/O タグとしての UDT メンバーの設定](#)
- [所有オブジェクトとしての UDT インスタンスの使用](#)
- [Web Client](#)
- [MapApp](#)
- [クロス リファレンス](#)
- [未使用タグの削除](#)
- [スーパータグとの共存](#)
- [ライセンス](#)
- [ランタイム Tag Viewer](#)

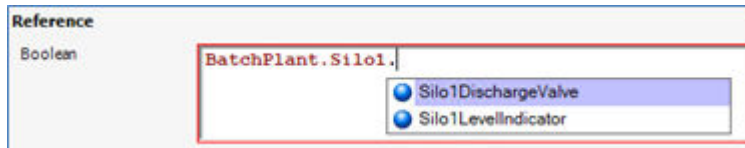
Intellisense

UDT インスタンスの Intellisense には、そのレベルだけのすべてのメンバーとメンバー データ構造が一覧表示されます。産業用グラフィック エディタは、スクリプト、アニメーション、およびカスタム プロパティで Intellisense をサポートします。ネイティブ InTouch はスクリプト作成でのみ Intellisense をサポートします。

- 産業用グラフィック: Intellisense は、スクリプト、アニメーション、およびカスタム プロパティでサポートされます。
- ネイティブ InTouch: Intellisense はネイティブ InTouch スクリプトでのみサポートされます。

例:





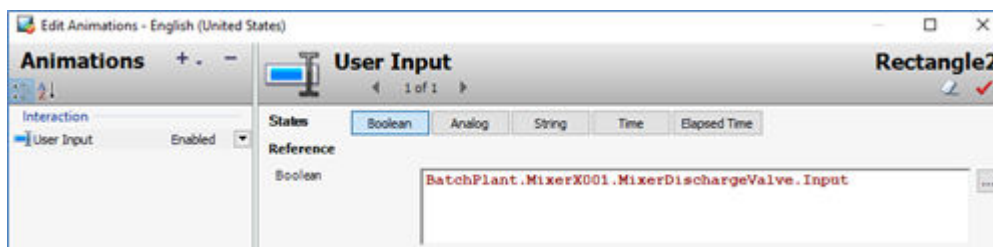
アニメーション

産業用グラフィックアニメーションとネイティブ InTouch アニメーションで UDT を設定できます。

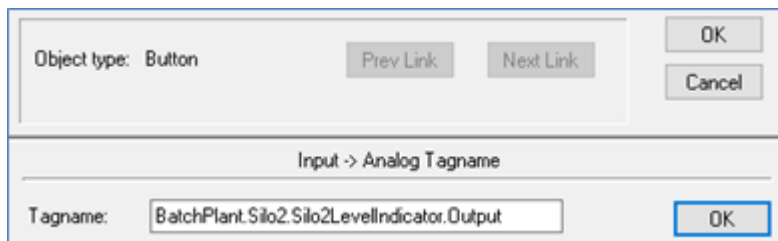
注記: 値の表示アニメーションを含む UDT のタグの変更の動作はリモート タグと同じです。完全な参照を UDT タグ、またはドット フィールドを含む UDT タグで置き換える必要があります。

例:

産業用グラフィックアニメーションで UDT を設定します。



ネイティブ InTouch アニメーションで UDT を設定します。

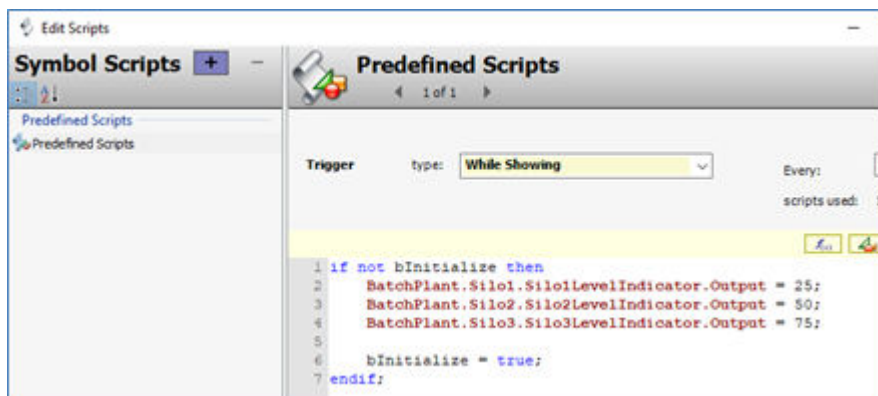


スクリプト

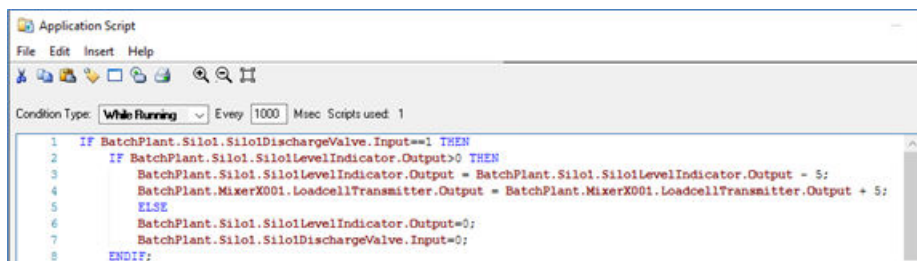
産業用グラフィック スクリプトとネイティブ InTouch スクリプトで UDT を設定できます。

例:

産業用グラフィック スクリプトで UDT を設定します。



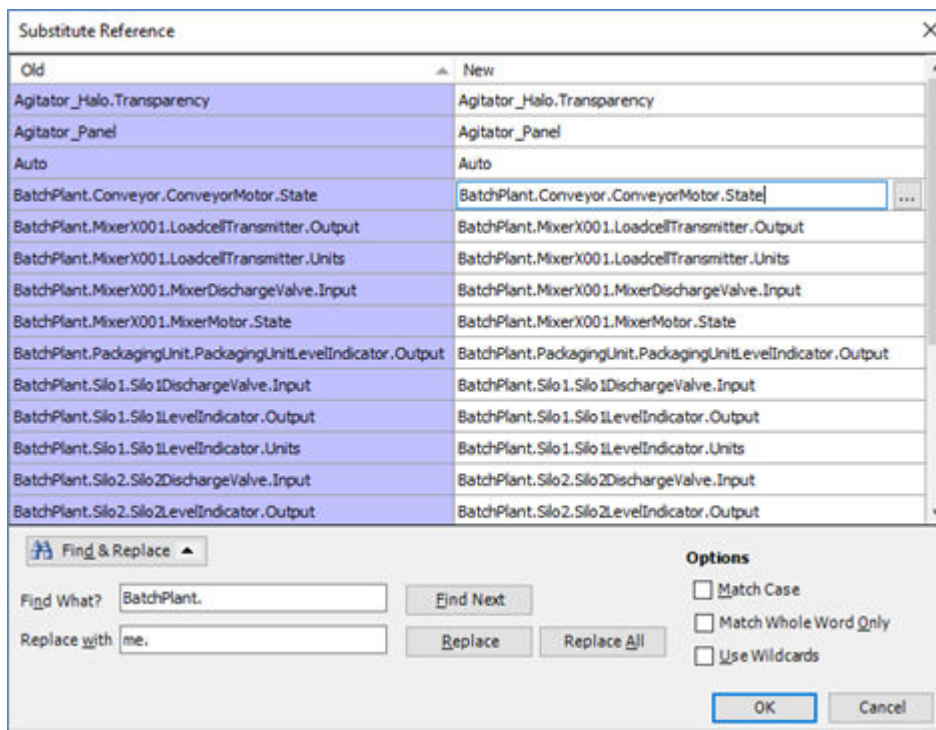
ネイティブ InTouch スクリプトで UDT を設定します。



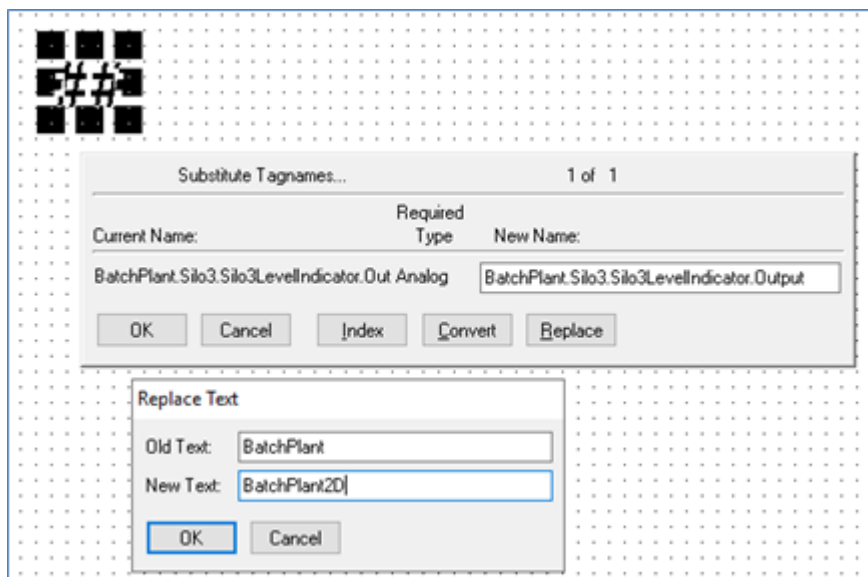
参照の変更

産業用グラフィック エディタと IN ネイティブ InTouch の両方で、その他の InTouch タグと同じ方法で UDT インスタンス メンバー タグ リファレンスを置換できます。

産業用グラフィックの参照の変更の例を以下に示します。



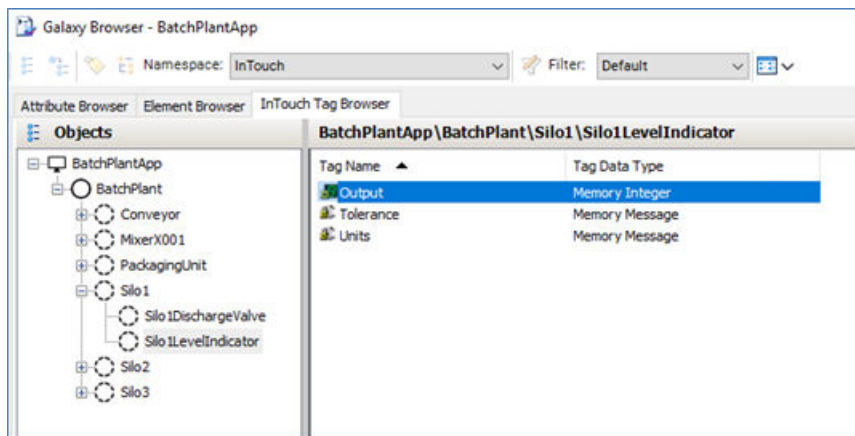
ネイティブ InTouch での参照の変更の例を以下に示します。



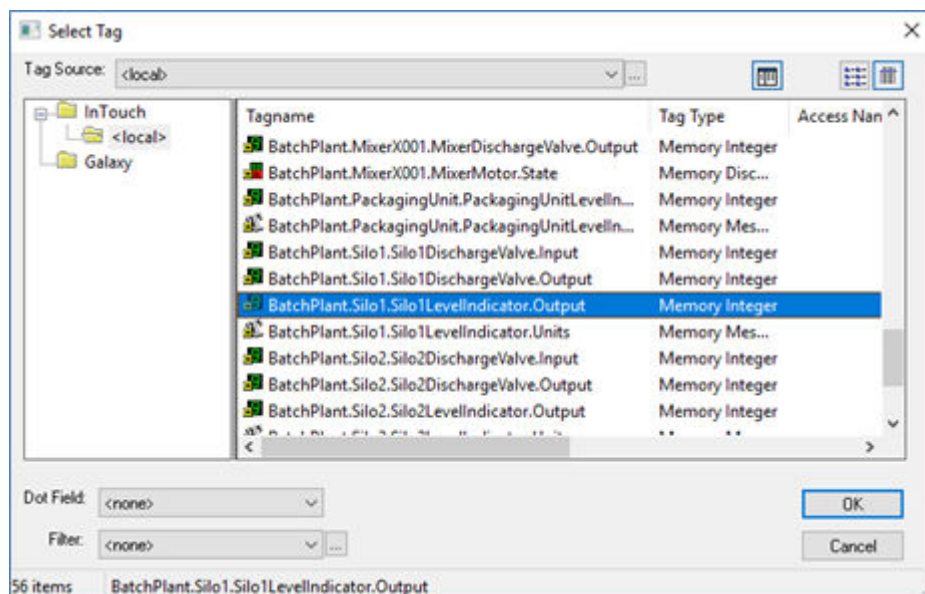
UDT メンバーの参照

産業用グラフィック エディタでアニメーションを設定する場合、またはスクリプトを使用する場合、省略記号ボタンを使用して **[Galaxy ブラウザ]** を開き、UDT メンバーを参照します。ネイティブ InTouch のアニメーションまたはスクリプト オプションで、**タグブラウザ**を開いて UDT メンバーを参照します。

産業用グラフィック エディタでの参照の例を以下に示します。



ネイティブ InTouch での参照の例を以下に示します。

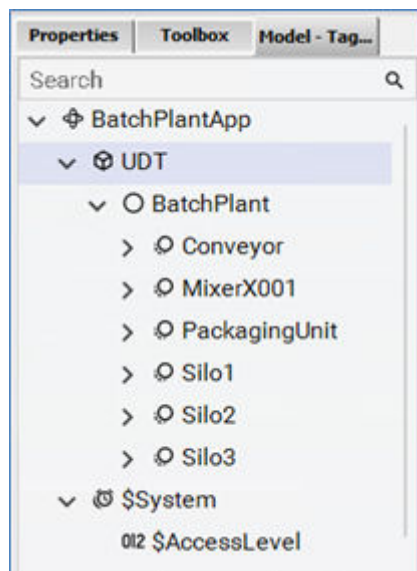


インスタンスの UDT メンバーの産業用グラフィック エディタ キャンバスへのドラッグ アンド ドロップ

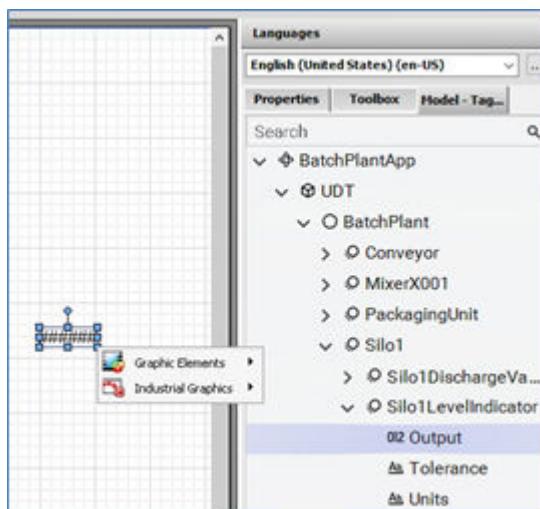
産業用グラフィック エディタの [モデル - タグ名] ペインにも UDT インスタンスが表示されます。産業用グラフィック エディタで 1 つまたは複数の UDT メンバーをキャンバスにドラッグ アンド ドロップしてアニメーションを作成できます。

例:

[モデル - タグ名] でドラッグ アンド ドロップします。



産業用グラフィック エディタでドラッグ アンド ドロップします。



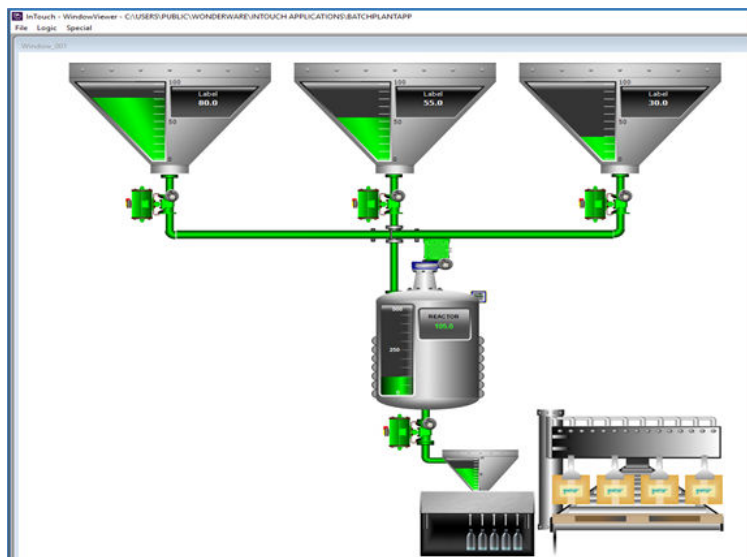
アニメーションとスクリプト

グラフィックに以下の設定が含まれている場合、UDT インスタンス メンバーは **WindowViewer** でサポートされます。

- 産業用グラフィックと InTouch ネイティブ グラフィックのアニメーションおよびスクリプト
- または
- ネイティブ InTouch のスクリプト

例:

WindowViewer では、以下を表示できます。

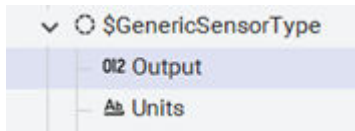


アラームおよびアラーム クライアント コントロール

インスタンスの UDT メンバーは、その他の InTouch タグと同じ方法でアラームをサポートします。アラーム プロパティは、[プロパティ] グリッドで更新できます。 **WindowViewer** で、アラームの生成とアラーム クライアント コントロールでの表示を行っているインスタンスのメンバーを表示できます。

例: UDT メンバーのアラームの設定

1. [ユーザー定義型] ビューで \$GenericSensorType > Output メンバーを選択します。



2. [プロパティ] グリッドで、次の図に示すように、このタグのアラーム プロパティを更新します。



アラーム プロパティが UDT インスタント レベルまで反映されます。以下の UDT メンバーでは、前の例に示されているように、アラームを設定できます。

- BatchPlant.Silo1.Silo1LevelIndicator.Output
- BatchPlant.Silo2.Silo2LevelIndicator.Output
- BatchPlant.Silo3.Silo3LevelIndicator.Output

次の図は、**WindowViewer deno** UDT のアラームのサポートの例を示します。



ログデータの有効化の例

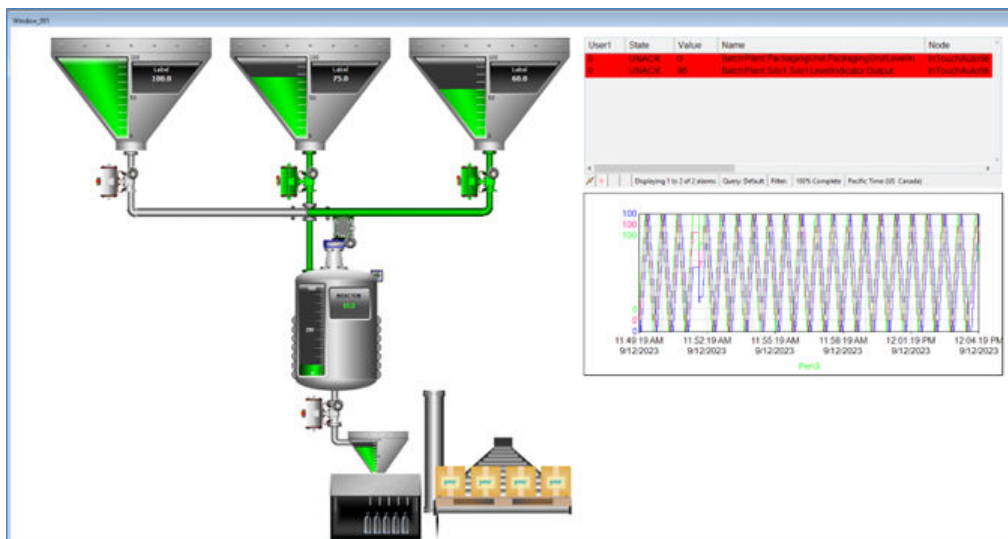
- 012 Output
- Ab Units

-

1. バックステージに移動します ([ファイル] > [設定] > [履歴ログ])

2. 指定したディレクトリが存在しない場合、ファイルエクスプローラを使用して作成します。
3. WindowViewer を開き、ランタイムに移動し、LGH ファイルへのデータ履歴の記録を検証します。

次の図は、LGH ファイルへの履歴データを記録する UDT メンバーを示し、トレンドクライアントはリアルタイムおよび履歴データをプロットしています。履歴データは、トレンドにバックフィルされます。



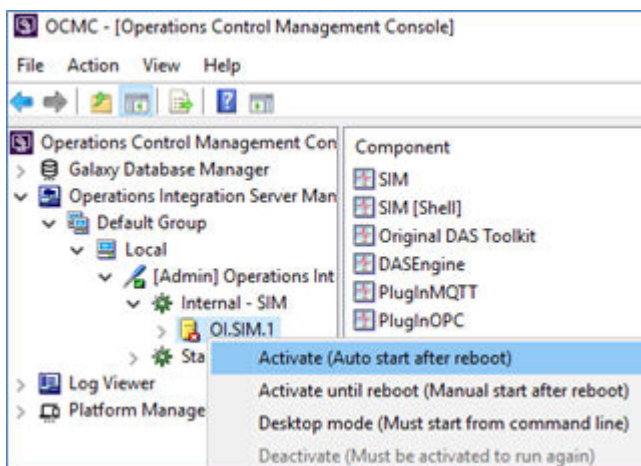
I/O タグとしての UDT メンバーの設定

その他の InTouch タグと同じ方法で UDT メンバーを I/O タグとして設定できます。[プロパティ] グリッドで、[タイプ]、[アクセス名]、および [アイテム名] フィールドを指定してメンバーを I/O タグとして設定します。

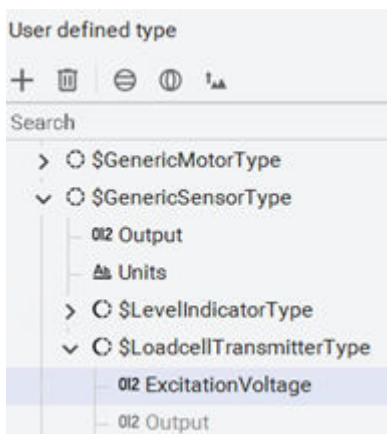
例: I/O タグとしての UDT メンバーの設定

1. ローカル Simulator サーバーのアクセス名 "SIM" を作成します。

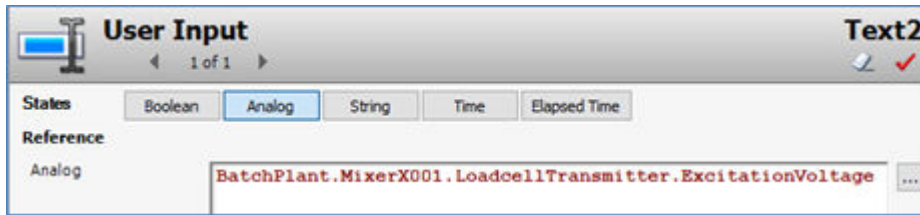
2. Operations Control Management Console（OCMC）から SIM サーバーをアクティブ化します。



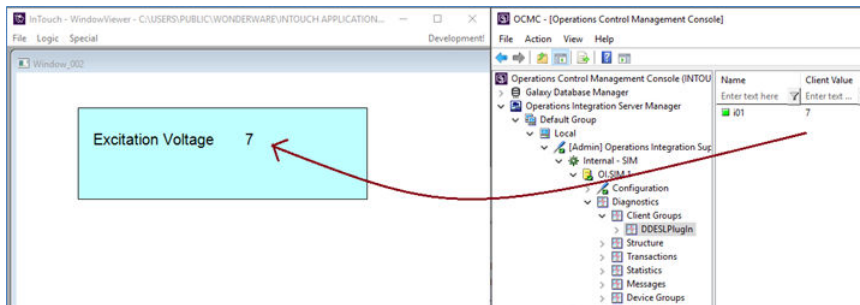
3. ExcitationVoltage 例メンバーを選択して設定します。



4. プロパティを設定してメンバーを I/O タグにします。インスタンス レベルでアクセス名およびアイテム名を上書きすることもできます。
 - a. 型を I/O 整数型に設定します。
 - b. SIM へのアクセス名を設定します（ドロップダウンリストから選択できます）。
 - c. アイテム名を i01 に設定します。
5. グラフィックを作成し、以下の図に示す参照を含むアニメーションを追加します。



- 作成したグラフィックを埋め込んでランタイムで表示します。値が I/O 参照にバインドされていることを確認します。



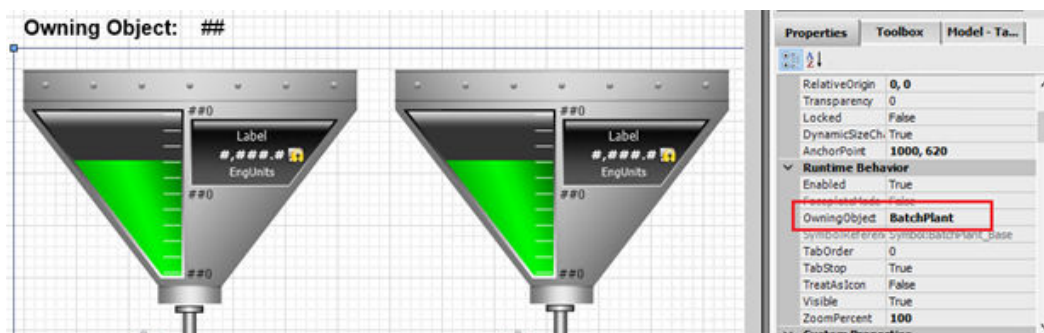
所有オブジェクトとしての UDT インスタンスの使用

埋め込まれたグラフィックの所有オブジェクトを設定して UDT インスタンスをポイントすることができます。ランタイム時、**me.** 相対参照は UDT インスタンスによって置き換えられます。ランタイム時に所有オブジェクトを別の UDT インスタンスに更新することもできます。

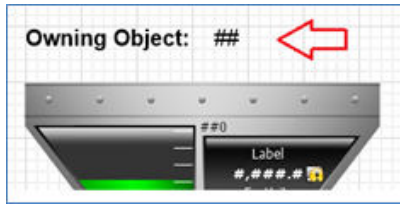
例: 所有オブジェクトとしての UDT インスタンスの使用

- 「視覚化」フォルダで、ReactorDemo Symbols > MainDisplays に移動します。
- BatchPlant_OwningObj グラフィックを編集します。
- 埋め込まれたグラフィック BatchPlant_RelativeRef1 を選択します。

プロパティグリッドで、OwningObject 属性が UDT インスタンスである BatchPlant に設定されていることを確認します。



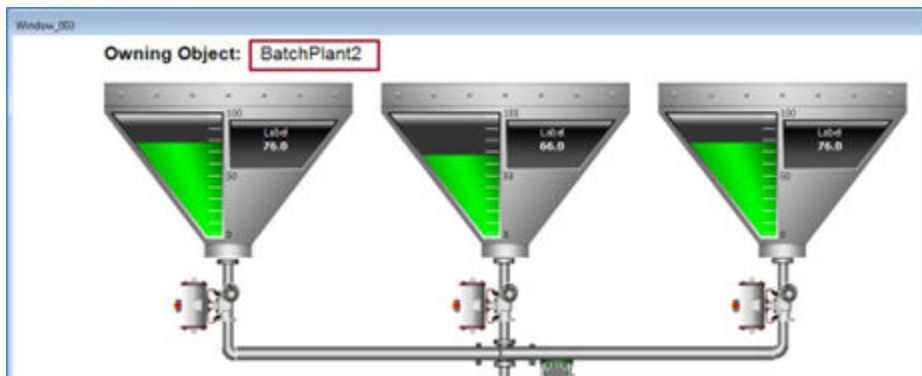
BatchPlantType から別の UDT を作成し、ランタイム時に入力アニメーションを使用して別の UDT insutansu に切り替えることができます。



例: ランタイム時の所有オブジェクトから別の **UDT** インスタンスへの切り替え

1. BatchPlant_OwningObj グラフィックを別の枠ウィンドウに埋め込みます。
2. WindowViewer を閉じます（実行している場合）。
3. WindowViewer ランタイムに切り替えます。
4. "me." 総体参照が BatchPlant にバインドされていることを確認します。

所有オブジェクトを別のインスタンスに変更できます。

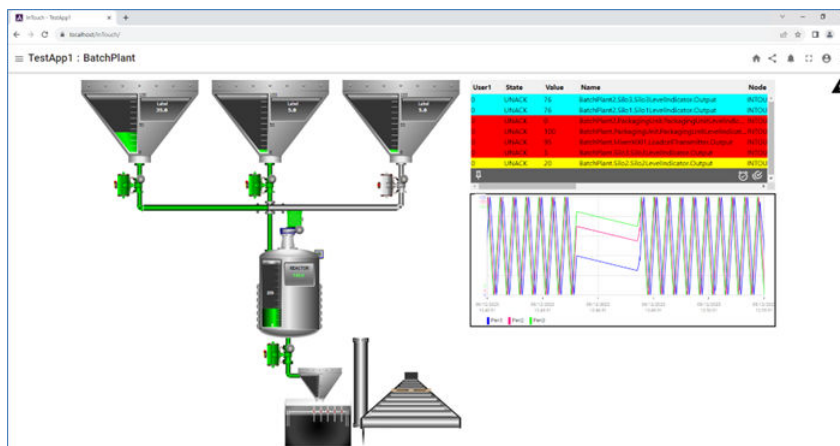
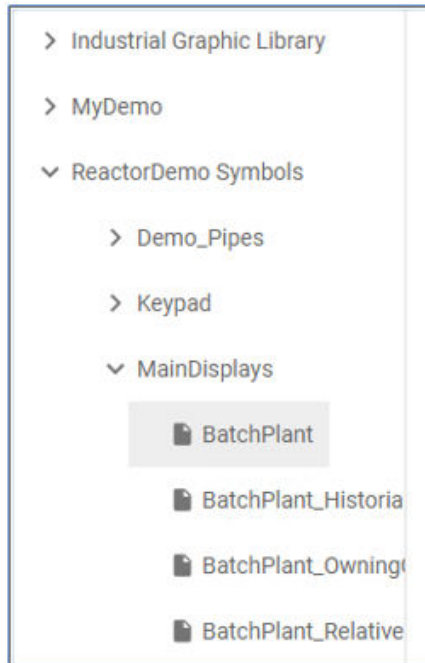


Web Client

インスタンスの UDT メンバーは、Web Client でその他の InTouch タグと同じように機能します。

例:

WindowViewer でアプリケーションが実行している状態で、WindowMaker から Web Client を起動し、メニューから BatchPlant グラフィックを開きます。



MapApp

インスタンスの UDT メンバーは、MapApp でその他の InTouch タグと同じように動作します。

場所メンバーの追加の例

1. この例の "BatchPlantType" を選択し、ツールバーの「場所を追加」を使用して新しいメンバーの経度および緯度を追加します。



2. プロパティ グリッドで以下の値を設定します。

経度 = 33

緯度 = -114

マップ設定の構成例

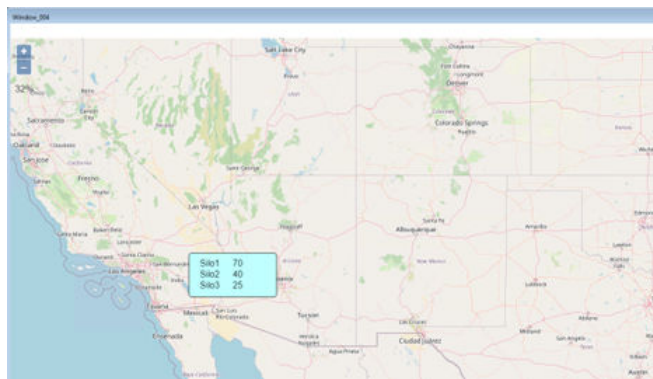
1. [視覚化] フォルダで、ウィジェットに移動して [Map_App] をクリックします。
2. グラフィック "MapSym1" を "BatchPlant"に関連付けます。MapSym1 は、この UDT インスタンス BatchPlant に対して MapApp で表示するグラフィックです。

MapApp 設定では、インスタンスおよびそのインスタンスのすべてのメンバー（メンバー データ型）がリストされます。

Sources Zoom Layers Locations						
Instance	Graphic	Layer	Latitude	Longitude	Position	
▶ BatchPlant	MapSym1	Default	33	-114	bottom-center	
BatchPlant.Conveyor	MapDefaultSym	Default			bottom-center	
BatchPlant.Conveyor.Conveyor...	MapDefaultSym	Default			bottom-center	
BatchPlant.MixerX001	MapDefaultSym	Default			bottom-center	

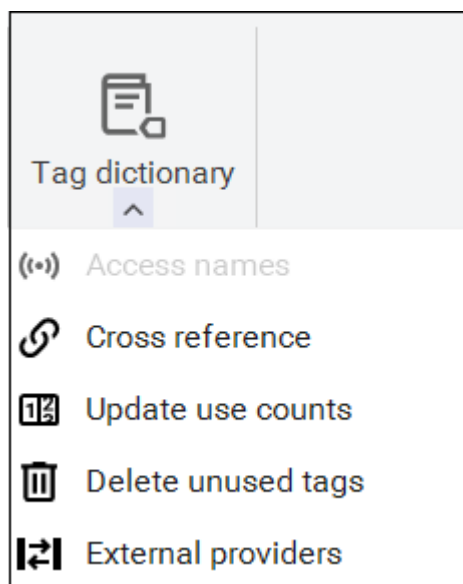
ランタイムでの MapApp の使用例

- グラフィック「MapAppMain」を別の枠ウィンドウに埋め込みます。WindowViewer を閉じ（既に実行している場合）、ランタイムに切り替えます。

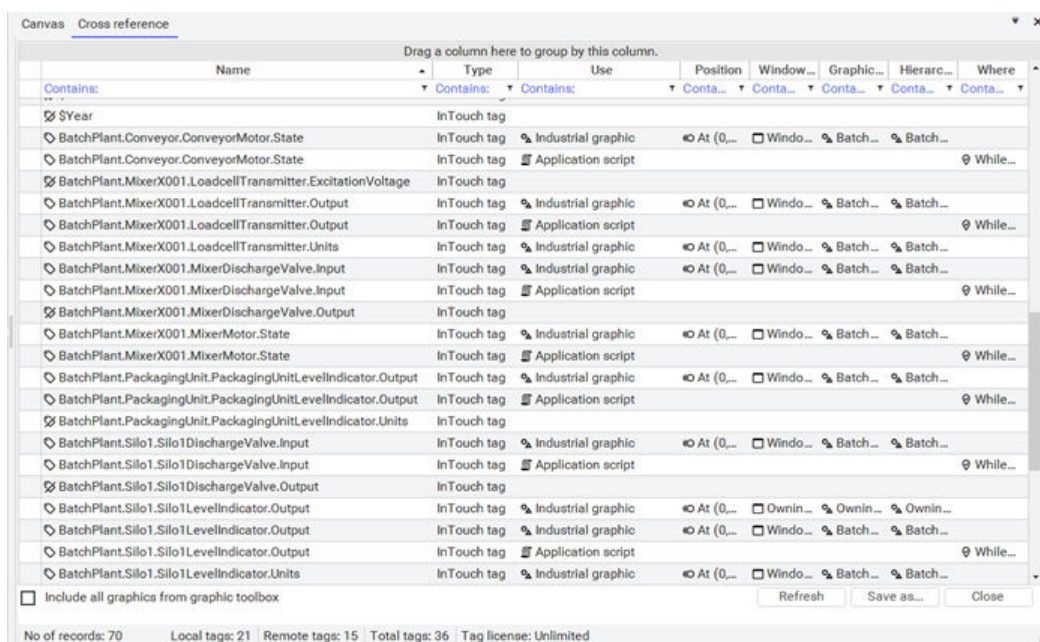


クロス リファレンス

クロス リファレンス ユーティリティを使用して、インスタンスの UDT メンバーが使用されている場所を見つけることができます。WindowMaker リボンの [タグディクショナリ] をクリックして [クロス リファレンス] を選択します。



クロス リファレンスの例:

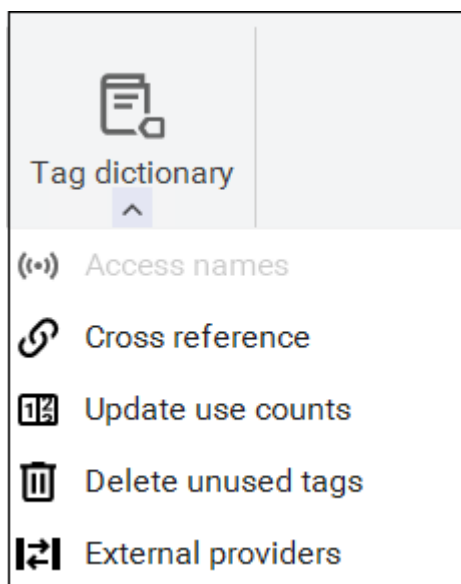


未使用タグの削除

インスタンスの未使用の UDT メンバーは、[未使用タグの削除] ページに表示され、そこから削除することができます。

例:

1. \$BatchPlantType の 2 つのインスタンスを作成し、それぞれに BatchPlant2 と BatchPlant3 という名前を付けます。
2. [未使用タグの削除] を選択します。



未使用タグ BatchPlant2 と BatchPlant3 を削除できます。BatchPlant は使用中なので表示されません。

Canvas		Delete unused tags	
	Delete tag		
	<input type="checkbox"/>	Contains:	
	<input checked="" type="checkbox"/>	BatchPlant2	
	<input checked="" type="checkbox"/>	BatchPlant3	
	<input type="checkbox"/>	Tag1	
	<input type="checkbox"/>	Tag2	
	<input type="checkbox"/>	Tag3	

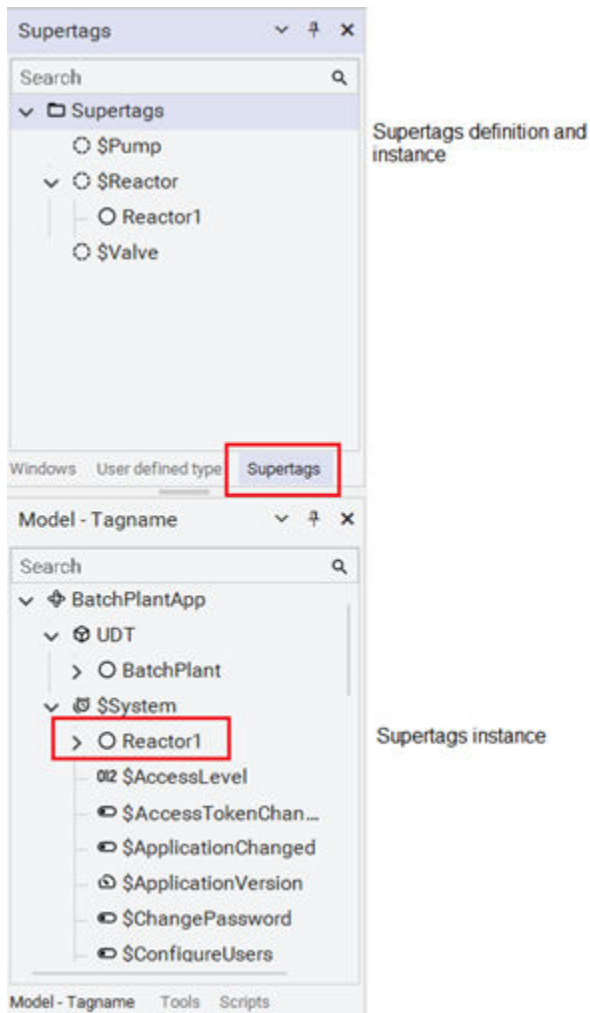
スーパータグとの共存

UDT はスーパータグと共存できます。スーパータグと UDT を同じ名前で作成できます。スーパータグ インスタンスと UDT インスタンスは、スクリプト、アニメーション、アラーム、およびトレンドで連動することも、個別に機能することもできます。

共存の動作を以下に示します。

- デフォルトでは、スーパータグ ビューはオフになっています。
 - これは、新しいアプリケーションだけでなく、最新のリリースに移行されたアプリケーションにも該当します。
 - 必要な場合、スーパータグ ビューに戻すことができます。
- スーパータグでは引き続き "\" 構文が使用されます。
- スーパータグは、引き続き **System Platform 2023** と同様に機能します。
- スーパータグ定義とユーザー定義型の間に名前の衝突はありません。
 - 同じ名前でスーパータグと UDT を作成できます ("Reactor" など)。
- スーパータグ インスタンスと UDT インスタンスは、スクリプト、アニメーション、アラーム、およびトレンドで連動することも、個別に機能することもできます。

- スーパータグでは所有オブジェクトはサポートされません。したがって、スーパータグでは "me." はサポートされません。



ライセンス

インスタンスの各 UDT メンバーは 1 つのローカル タグとしてカウントされます。UDT データ型または派生データ型はローカル タグの使用にカウントされません。

例:

次の例では、ローカル タグの合計数は 21 です。

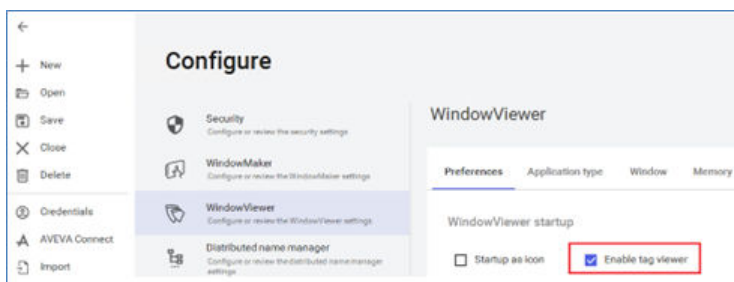
Update use counts	
Local tags:	21
Remote tags:	15
Total tags:	36
Tag license:	Unlimited

ランタイム Tag Viewer

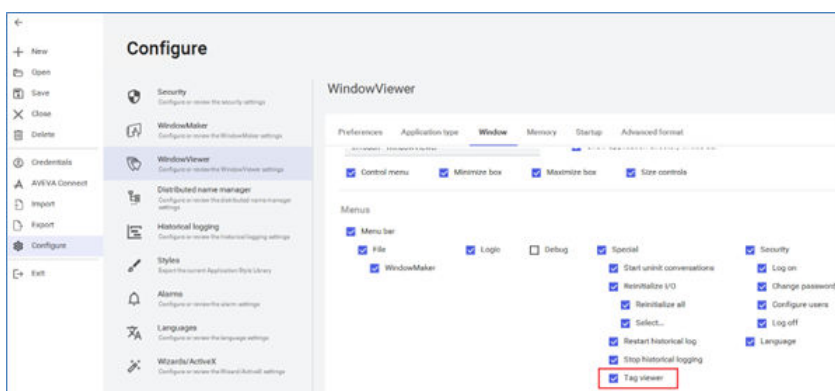
ランタイム Tag Viewer は UDT メンバーをサポートします。

Tag Viewer を開く前に、以下のことを確認します。

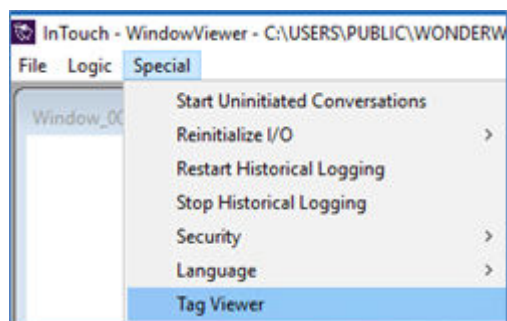
- [ファイル] > [設定] > [WindowViewer] > [環境設定] で [Tag Viewer の有効化] チェック ボックスが選択されていること。



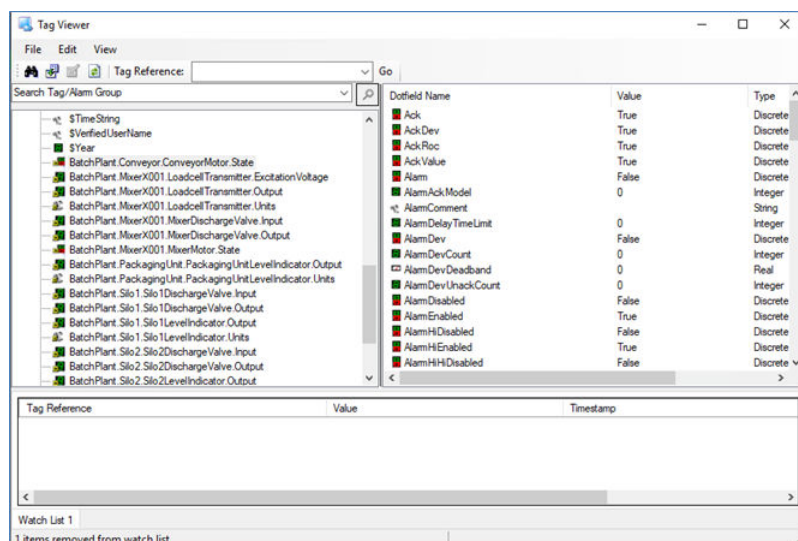
- [ファイル] > [設定] > [WindowViewer] > [ウィンドウ] で [Tag Viewer] チェック ボックスが選択されていること。



WindowViewer でアプリケーションを実行し、Tag Viewer を起動します。



Tag Viewer に UDT インスタンス メンバーが表示されます。



UDT の制限事項

現在のリリースでは、以下の機能は UDT でサポートされていません。

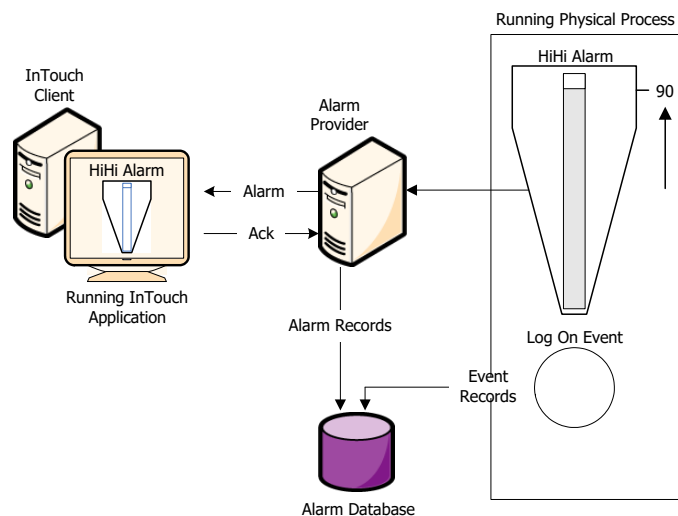
- ActiveX コントロール プロパティでの UDT タグの使用。
- ウィザードでの UDT タグの使用（ボタン、クロック、枠など）。
- トレンドでの UDT タグの使用（履歴トレンド）。
- UDT タグでの SmartSymbol のインスタンス参照の上書き。
- SQL アクセス マネージャ バインド リストでの UDT タグの使用。
- レシピ マネージャでの UDT タグの使用。
- ネイティブ ユーザー入力アニメーションの最大および最小フィールドでの UDT タグの使用。
- InTouchProxy オブジェクトからのタグ ブラウザ。
- アラーム コメントでの UDT タグの使用。
- 言語の切り替えでの UDT タグの使用。
- タグの保持プロパティ。

章 10 アラーム

アラームとイベントを生成してオペレータにプロセスの動作状況を通知する InTouch アプリケーションを作成できます。

- 問題を引き起こす可能性のあるプロセス状態に関する警告が、アラームによってランタイム オペレータに警告されます。通常は、プロセス値が定義済みのしきい値を超えた場合にトリガするようにアラームを設定します。オペレータは、アラームを確認する必要があります。
- イベントは、通常のシステム ステータス メッセージを表します。オペレータが InTouch アプリケーションにログオンするなど、システム条件が発生するときに典型的なイベントとなります。オペレータは、イベントを確認する必要はありません。

以下の図は、アプリケーションが実行されている間、InTouch HMI でアラームとイベントが処理される方法を示しています。アラームとイベントのデータは、アラーム データベースに保存されます。



イベント監視用のタグを設定できます。イベントメッセージは、タグの値が変更するたびにアラーム システムにログ記録されます。イベントメッセージには、値がどのように変更されたかに加えて、オペレータ、I/O、スクリプト、またはシステムのいずれによってその変更が開始されたのかという情報が含まれます。

アラームの設定

アラームを設定するには、簡単にアラーム条件を持つタグを設定します。

必要に応じて、以下の操作も実行できます。

- アラーム階層を定義します。
- アラームを無効にし、抑止します。
- アラーム コメントを設定します。
- その他のアラームとイベントのプロパティを設定します。

アラーム階層の定義

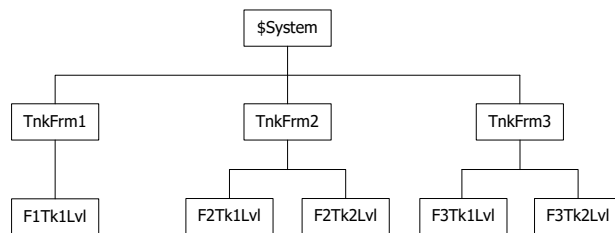
各 InTouch アラームはアラーム グループに属します。関連するアラームをグループに整理することによって、オペレータは簡単にアラームをフィルタしたり、表示したり、確認したりできるようになります。詳細については、「[アラーム グループ](#)」を参照してください。

分散アラーム システムでは、アラーム グループ リストの基本としてアラーム グループが使用されます。分散アラーム システム グループ リストの作成の詳細については、「[アラーム グループ リスト ファイルの作成](#)」を参照してください。

アラーム グループの作成

アラーム グループの作成を開始する前に、アラーム グループを構成する方法と、アラーム グループ名を準備する必要があります。一貫したグループ命名規則を使用することによって、階層内でのグループの論理的な順序が制約されます。

以下の図で、階層内の同じレベルのグループに割り当てられた名前が類似していることに注意してください。



また、下位グループの名前に親の名前の一部を含めることによって、下位グループが親グループを参照していることにも注目してください。たとえば、第 3 レベルのアラーム グループの名前 **F1Tk1Lvl** にはグループ名の **F1** プリフィックスを含めることによって、そのアラーム グループの親である **TnkFrm1** を参照しています。階層内の異なるレベルのアラーム グループ間での親子関係を提示する命名規則を開発します。

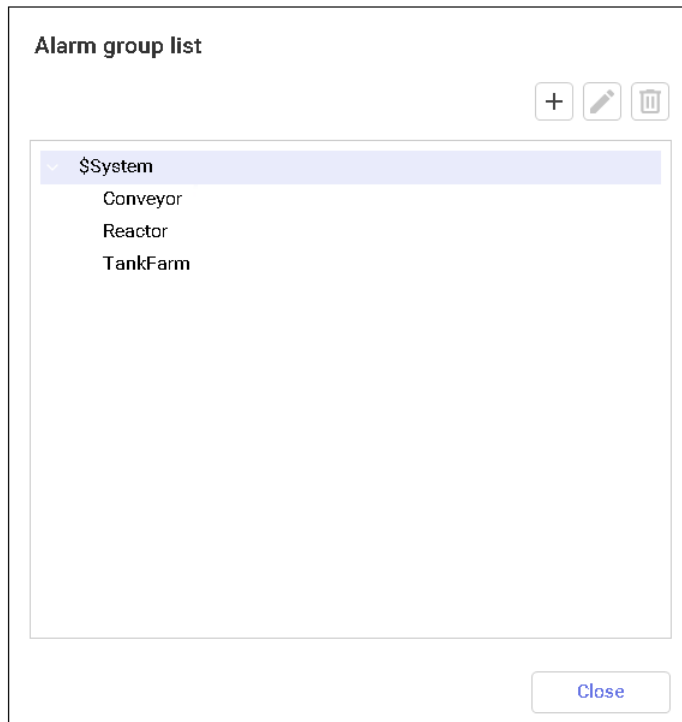
注記: アラーム グループは InTouch ライセンスではタグとして数えられないものの、データベースではタグとして数えられます。したがって、アラーム グループと実際のタグの合計数は、ユーザーの InTouch ライセンスで設定されている最大数を超えることはできません。

アラーム グループ名は、以下の要件を満たす必要があります。

- 名前は 32 文字以内である必要があります。
- 名前は英数文字 (A~Z、a~z または 0~9) で開始する必要があります。
- 名前には以下のキーボード文字 (@、#、\$、%、&、-、_、?、!、\) を含めることができ、名前の 2 番目の位置から使用を開始できます。
- 名前にハイフン (-) が含まれている場合、その名前は英字で開始する必要があります。
- 名前に空白文字を含めることはできません。
- 名前には少なくとも英字を 1 つ使用する必要があります。

アラーム グループを作成するには

1. [ホーム] メニューの [アラーム] グループで [アラーム グループ] をクリックします。
[アラーム グループ リスト] ウィンドウが表示されます。



2. **[+]** アイコンをクリックして追加するか、**Alt+A** キーを押します。
[アラーム グループの追加] ダイアログ ボックスが表示されます。

3. **[グループ名]** ボックスに、新しいアラーム グループの名前を入力します。
4. **[コメント]** ボックスに、新しいアラーム グループのオプションのコメントを 49 文字以内で入力します。
5. アラーム グループを別の親グループに割り当てるには:

- a. リストから親グループ名を選択します。InTouch アプリケーションに初めて定義されるアラームグループである場合、このグループは親の \$System グループに自動的に割り当てられます。
 - b. リストから新しい親グループを選択します。
6. [追加] をクリックします。
[アラーム グループ リスト] ウィンドウが表示され、リストに追加された新しいアラーム グループが表示されます。
 7. [閉じる] をクリックします。

アラーム グループの変更

アラーム グループは、以下の操作を実行するために変更できます。

- 名前を変更する
- 関連付けられたコメントを変更する
- 別のグループに再割り当てする

アラーム グループを変更するには

1. [ホーム] メニューの [アラーム] グループで [アラーム グループ] をクリックします。
[アラーム グループ リスト] ウィンドウが表示されます。
2. 変更するアラーム グループを選択し、[編集] アイコンをクリックするか、Alt+E を押します。
[アラーム グループの編集] ウィンドウが表示されます。

Edit alarm group

Group name
TankFarm

Comments

Parent group
\$System

Cancel

Save

3. アラーム グループの名前やコメントを変更します。
4. アラーム グループを別の親グループに割り当てるには:
 - a. [親グループ] ドロップダウン リストから新しい親グループを選択します。
5. [保存] をクリックします。
6. [OK] をクリックします。

アラーム グループの削除

アラーム グループを削除して、階層からそのグループを削除できます。削除されたアラーム グループに属すアラームやタグは、階層内で削除されたグループのすぐ上の親グループに自動的に再割り当てされます。また、削除された子グループも削除されたグループのすぐ上の親グループに再割り当てされます。

アラーム グループを削除するには

1. [ホーム] メニューの [アラーム] グループで [アラーム グループ] をクリックします。
[アラーム グループリスト] ウィンドウが表示されます。
2. アラーム グループを選択して [削除] アイコンをクリックするか、Alt+D キーを押します。メッセージが表示されたら、[はい] をクリックします。
3. [閉じる] をクリックします。

アラーム状況を使用したタグ変数の設定

アラームのタグ変数は、アラームのタイプと 1 つまたは複数のアラームしきい値を指定することによって設定できます。タグ変数の値が定義されたしきい値に達すると、アラームが発生します。タグ変数値のアラーム状況への移行情報はすべて、分散アラーム システムに報告されます。

論理値アラームの設定

論理値アラームは、論理型タグ変数に対応します。アラーム状況が論理型タグ変数の **true**（オン、はい、1）状態または **false** 状態（オフ、いいえ、0）に対応するかどうかを設定できます。

論理型タグ変数のアラーム状況を設定するには

1. タグ変数ディクショナリを開きます。
2. 既存の論理型タグ変数を選択するか、新しい論理型タグ変数を作成します。
3. [タグ変数ディクショナリ] ダイアログ ボックスの一番上にある [アラーム] または [両方] のいずれかをクリックして、論理値アラームの詳細設定ダイアログ ボックスを表示します。

The screenshot shows a dialog box for configuring an alarm. It has several sections: 'ACK Model' with radio buttons for 'Condition' (selected), 'Event Oriented', and 'Expanded Summary'; 'Alarm Comment' with a text input field; 'Alarm State' with radio buttons for 'On', 'Off', and 'None' (selected); 'Priority' with a numeric input field set to '1'; and 'Alarm Inhibitor' with a dropdown menu.

4. [確認モデル] 領域で、タグ変数のアラーム確認モデルを選択します。
 - 確認の時間までにアラーム状況である、またはサブステートであるすべての移行を確認でカウントするには、[条件] をクリックします。これがデフォルトの確認モデルです。
 - 確認をアラーム状況またはサブステートへの特定の移行に対してのみ行うには、[イベント指向] をクリックします。最新のトランザクションを参照する場合のみ確認が受け入れられます。
 - 確認をアラーム状況やサブステートへの移行、または平常状態への復帰に対してのみ行うには、[拡張サマリ] をクリックします。平常状態からの移行はすべて、新しい「平常への復帰」(RTN) グループの開始としてマーク付けされます。RTN グループ内でのすべての移行は、RTN グループ全体が確認されたと認識されるまで、個別に確認される必要があります。
5. [アラーム コメント] ボックスに、最大 131 文字のアラーム コメントを入力します。

注意: [アラーム コメント] ボックスには、二重引用符文字 (") を含めないでください。二重引用符を含むアラーム コメントをフェッチする場合、区切り文字として二重引用符を使用するプロセスが失敗します。

1. [アラーム状況] 領域で、アクティブなアラーム状況の論理型タグ変数値を**オン**または**オフ**にするかを選択します。
2. [優先度] ボックスで 1 ～ 999 のアラーム優先度番号を割り当てます。デフォルトの優先度番号は、最高のアラーム優先度を示す 1 です。
3. オプションで、論理値アラームのアラーム抑止タグ変数を割り当てます。
 - a. [アラーム抑止タグ変数] ボックスで、ボタンをクリックして、定義済みのタグ変数リストを持つ [タグ変数を選択してください] ダイアログ ボックスを表示します。
 - b. リストからタグ変数を選択し、[OK] をクリックします。抑止タグ変数として選択したタグ変数の名前が、[アラーム抑止タグ変数] ボックスに表示されます。

アラームの抑止の詳細については、「[アラームの抑止](#)」を参照してください。

4. [保存] をクリックします。
5. [閉じる] をクリックして、[タグ変数ディクショナリ] ダイアログ ボックスを閉じます。

値型アラームの設定

値型アラームは、整数型または実数型のタグ変数に関連付けられています。タグ変数値が事前に設定された LoLo から HiHi までの範囲のしきい値から移行する場合に発生するアラームを設定できます。アラーム状況が任意のタグ変数の値やそのアラームの関連付けられた優先度に対応するかどうかを設定できます。

値型アラームを設定するには

1. タグ変数ディクショナリを開きます。
2. 既存の実数型または整数型のタグ変数を選択するか、新しいタグ変数を作成します。
3. [タグ変数ディクショナリ] ダイアログ ボックスの一番上にある [アラーム] または [両方] のいずれかをクリックして、アラームの詳細設定ダイアログ ボックスを表示します。

ACK Model: <input checked="" type="radio"/> Condition <input type="radio"/> Event Oriented <input type="radio"/> Expanded Summary				Alarm Comment: Reactor level				
	Alarm Value	Priority	Alarm Inhibitor		Alarm Value	Priority	Alarm Inhibitor	Value Deadband
<input type="checkbox"/> LoLo	0	1	<input type="checkbox"/>	<input checked="" type="checkbox"/> High	1800	1	<input type="checkbox"/>	0
<input checked="" type="checkbox"/> Low	200	1	<input type="checkbox"/>	<input type="checkbox"/> HiHi	180	1	<input type="checkbox"/>	

4. [確認モデル] 領域で、タグ変数のアラーム確認モデルを選択します。
 - 確認の時間までにアラーム状況である、またはサブステートであるすべての移行を確認でカウントするには、[条件] をクリックします。これがデフォルトの確認モデルです。
 - 特定のアラーム状況やサブステートへの移行に対してのみ確認を行うには、[イベント指向] をクリックします。確認は最後のトランザクションを参照している場合にのみ受け入れられます。
 - 確認をアラーム状況やサブステートへの移行、または平常状態への復帰に対してのみ行うには、[拡張サマリ] をクリックします。平常状態からの移行はすべて、新しい「平常への復帰」(RTN) グループの開始としてマーク付けされます。RTN グループ内でのすべての移行は、RTN グループ全体が確認されたと認識されるまで、個別に確認される必要があります。

5. [アラーム コメント] ボックスに、最大 131 文字のデフォルト コメントを入力します。このコメントは、タグ変数の `.AlarmComment` ドットフィールドに割り当てられます。

注意：[アラーム コメント] ボックスには、二重引用符文字 (") を含めないでください。二重引用符を含むアラーム コメントをフェッチする場合、区切り文字として二重引用符を使用するプロセスが失敗します。

1. タグ変数の値が限界値を超えたかどうかを検出するために使用するアラームのタイプ (**LoLo**、**Lo**、**Hi**、**HiHi**) を選択します。
2. [アラーム値] ボックスに、アラーム タイプに対する限界値を入力します。
たとえば、**LoLo** アラームおよび **Lo** アラームの場合、タグ変数の値が [アラーム値] より小さい場合にアラーム状況となります。**Hi** アラームおよび **HiHi** アラームの場合、タグ変数の値が [アラーム値] を超えた場合にアラームが発生します。しきい値には、実数値を使用できます。
3. [デッドバンド] ボックスに、アラーム状態から移行する前に、タグ変数値がアラーム値に対してどのくらいの範囲内であればならないかを工学単位数で指定します。
たとえば、アラーム状況から平常状態に戻るには、タグ変数値はアラームしきい値の範囲内に戻るだけでなく、指定したデッドバンド値以上の値に戻る必要があります。このデッドバンドは、タグ変数値がしきい値に近似であり、継続的にアラーム状態になったり、アラーム状態から離れたり変動している場合に、繰り返しの警告によってアラームが発生されることを避けるために使用されます。
4. オプションで、タグ変数のアラーム タイプ (**LoLo**、**Lo**、**Hi**、**HiHi**) に対してアラーム抑止タグ変数を割り当てます。
 - a. [アラーム抑止タグ変数] 領域で、ボタンをクリックして、定義済みのタグ変数リストを持つ [タグ変数を選択してください] ダイアログ ボックスを表示します。
 - b. リストからタグ変数を選択し、[OK] をクリックします。抑止タグ変数として選択したタグ変数の名前が、[アラーム抑止タグ変数] ボックスに表示されます。

抑止タグ変数の詳細については、「[アラームの抑止](#)」を参照してください。

5. [保存] をクリックします。
6. [閉じる] をクリックして、[タグ変数ディクショナリ] ダイアログ ボックスを終了します。

偏差アラームの設定

偏差アラームは、整数型または実数型のタグ変数に関連付けられます。現在のタグ変数値を目標値と比較することによって、アラームを設定できます。その後で、その差異の絶対値がタグ変数値の範囲のパーセンテージとして表される 1 つまたは複数のしきい値と比較されます。

たとえば、以下の値ではタグ変数の小偏差アラームと大偏差アラームの条件を設定しています。

最小値 = -1000

最大値 = 1000

小偏差 % = 10

大偏差 % = 15

基準値 = 500

これらの値を例として使用して、小偏差アラームと大偏差アラームの値は以下のように計算されます。

1. タグ変数の値範囲の合計を計算します。

$$1000 - (-1000) = 2000$$

2. タグ変数の値範囲の合計に小偏差と大偏差のパーセンテージを乗算します。

$$2000 \times 0.10 = 200 = \text{小偏差しきい値}$$

$$2000 \times 0.15 = 300 = \text{大偏差しきい値}$$

3. 基準値に対して、小偏差しきい値および大偏差しきい値を加算および減算します。

$$500 - 200 = 300 = \text{小偏差下限値}$$

$$500 + 200 = 700 = \text{小偏差上限値}$$

$$500 - 300 = 200 = \text{大偏差下限値}$$

$$500 + 300 = 800 = \text{大偏差上限値}$$

偏差アラームを設定するには

1. タグ変数ディクショナリを開きます。
2. 既存の実数型または整数型のタグ変数を選択するか、新しいタグ変数を作成します。
3. [タグ変数ディクショナリ] ダイアログ ボックスの一番上にある [アラーム] または [両方] のいずれかをクリックして、アラームの詳細設定ダイアログ ボックスを表示します。

	% Deviation	Target	Priority	Alarm Inhibitor	Deviation Deadband %
<input type="checkbox"/> Minor Deviation	0	0	1		0
<input type="checkbox"/> Major Deviation	0		1		

4. アナログ型タグ変数の値が指定した基準値に対して大偏差であるか小偏差であるかを判断するために使用する偏差（[小偏差] および [大偏差]）を選択します。
5. [% 偏差] ボックスに、小偏差または大偏差のアラーム状況を発生させるための、アナログ型タグ変数の基準値からの逸脱パーセンテージを入力します。これはタグ変数の範囲に対するパーセンテージとして表されます。I/O 型タグ変数の場合、タグ変数の詳細設定ダイアログ ボックスに入力した [工学値最小値] と [工学値最大値] によって範囲が定義されます。メモリ型タグ変数の場合、範囲は最小値と最大値によって定義されます。
6. [基準値] ボックスに、小偏差と大偏差のパーセンテージの基準となるタグ変数の参照値を入力します。
7. [偏差デッドバンド %] ボックスに、タグ変数がアラーム状況から逸脱する前にタグ変数値が低下する必要のあるしきい値からの偏差パーセンテージを入力します。
8. [保存] をクリックします。
9. [閉じる] をクリックして、[タグ変数ディクショナリ] ダイアログ ボックスを閉じます。

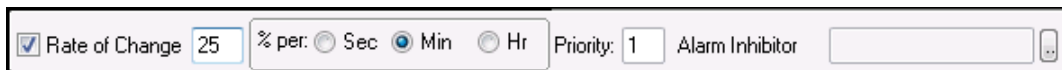
変化率アラームの設定

変更率アラームは、測定間隔内でアラームの絶対値が指定された制限を超えたときを検出します。値が変化するたびに、タグ変数は変化率アラームについてテストされます。変化率は、以前のタグ値、以前の変更が行われた時間、現在のタグ値、および現在の時間を使用して計算されます。

特定の時間範囲内で計算された変化率は、タグに対して指定された変化率のパーセンテージ制限と比較されます。計算された変化率がパーセンテージ制限より大きい場合、アラーム状況がタグに設定されます。変化率アラームは、タグの変化率がアラーム制限を下回るまで有効なままになります。

変化率アラームを設定するには

1. タグ変数ディクショナリを開きます。
2. 既存の実数型または整数型のタグ変数を選択するか、新しいタグ変数を作成します。
3. [タグ変数ディクショナリ] ダイアログ ボックスの一番上にある [アラーム] または [両方] のいずれかをクリックして、アラームの詳細設定ダイアログ ボックスを表示します。以下の図は、変化率アラームに適用されるオプションのみを示しています。



4. [変化率] ボックスをオンにします。
5. [%] ボックスに、変化しきい値の最大許容パーセンテージを入力します。
6. 時間間隔の単位を、[秒]、[分]、または [時] の中から選択します。
7. [優先度] ボックスに、アラーム優先度の値を 1 ～ 999 の範囲で入力します。
8. オプションで、変化率アラームのアラーム抑止タグ変数を割り当てます。
 - a. [アラーム抑止タグ変数] 領域で、ボタンをクリックして、定義済みのタグ変数リストを持つ [タグ変数を選択してください] ダイアログ ボックスを表示します。
 - b. リストからタグ変数を選択し、[OK] をクリックします。抑止タグ変数として選択したタグ変数の名前が、[アラーム抑止タグ変数] ボックスに表示されます。

アラームの抑止の詳細については、「[アラームの抑止](#)」を参照してください。

9. [保存] をクリックします。
10. [閉じる] をクリックして、[タグ変数ディクショナリ] ダイアログ ボックスを閉じます。

アラームの無効化

.AlarmEnabled ドットフィールドまたは AlarmDisabled ドットフィールドを使用して、タグ変数のすべてのアラームを一度に無効にしたり、有効にしたりできます。サブステートを持つアラームに対しては、各サブステートを個別に無効にできます。たとえば、アナログ値のアラームでは Hi を有効にし、HiHi を無効にすることができます。

ランタイム中、アラーム プロバイダでは、無効であるアラームやサブステートに対してアラームは生成されません。アラームが無効であるか有効であるかの変更は、ランタイムで行うことができます。

アラームが無効から有効の状態に移行するたびに、チェック ロジックにより、アラーム プロバイダによってそのアイテムをアラーム状況にする必要があるかどうか判断されます。

アイテムがアラーム状況の間、アラームが無効となるかアクティブに抑止される場合、そのアイテムは強制的に別の（有効な）状態となります。どの状態となるべきかは、どの状態が使用可能であるかどうか、またそれらの状態が無効になっていないかどうかにより異なります。この処理は、アラーム タイプやしきい値に基づいて、アラーム プロバイダによって実行されます。

アラームの抑止

オプションとして、アラームがアクティブ状態に移行することを防ぐ抑止アラーム タグ変数を各アラームまたはアラーム サブステートに割り当てることができます。

- 抑止タグ変数の値が **True** (0 以外または **Null** 以外) となり、その状態が維持されると、アラームは抑止されます。
- 同様に、抑止アラーム タグ変数が **False** (0 または **Null**) となり、その状態が維持されると、アラームは抑止されません。

抑止タグ変数は **WindowMaker** でのみ変更できます。また、抑止タグ変数の値は、ランタイムで変更できます。

抑止タグ変数は個別のアラーム サブステートに割り当てることができます。各サブステートは異なるタグ変数で抑止できます。また、一部のサブステートに抑止タグ変数を割り当てない状態で残すこともできます。

抑止されているアラーム (タグ変数が **True**) は、確認を待機していません。アラームがサブステートを持つ場合、引き続き使用可能であるサブステートでのみ確認を待機している状態となります。

アラームやサブステートは、個別に無効化したり、抑止すること (またはその両方) ができます。アラームは、それが有効になっており、アクティブに抑止されていない場合しかアクティブになりません。

アラームやサブステートに抑止タグ変数が割り当てられていない場合、常に **False** である抑止タグ変数を持つ場合と同じ結果になります。

移行によってアラームがアクティブに抑止されている状態から変化するたびに、チェック ロジックにより、**InTouch** でそのアイテムをアラーム状況にする必要があるかどうか判断されます。

アイテムがアラーム状況の間、アラームがアクティブに抑止される場合、そのアイテムは強制的に別の (有効な) 状態にされます。どの状態になるかは、どの状態が使用可能かどうか、またその使用可能な状態も無効になっていないかどうかによって異なります。この処理は、アラームのタイプやしきい値などに基づいて、**InTouch** によって処理されます。

確認を待機するアラーム (またはアラーム サブステート) がアクティブに抑止状態になった場合、アイテムを強制的に別の (有効な) 状態に設定する必要があります。アイテムがアラーム状況であるかに関しては、**InTouch** によって適切な状態が決定される必要があります。

アラーム抑止タグ変数は、使用カウントとライセンス制限に含まれます。

アラーム抑止タグ変数の名前を取得するには、以下の読み取り専用のタグ変数ドットフィールドを使用します。

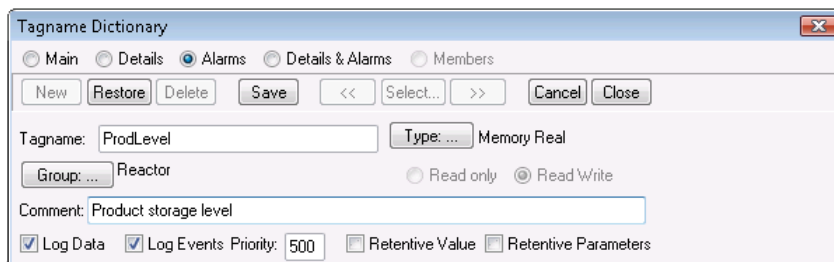
- AlarmDscInhibitor
- AlarmLoLoInhibitor
- AlarmLoInhibitor
- AlarmHiHiInhibitor
- AlarmHiInhibitor
- AlarmMajDevInhibitor
- AlarmMinDevInhibitor
- AlarmRocInhibitor

これらのフィールドはタグ変数の名前を返します。したがって、InTouch QuickScript で間接型タグ変数リファレンスでこの名前を使用して、アラーム抑止タグ変数の現在の値を特定したり、アラーム抑止タグ変数の値を変更したりできます。これにより、ランタイムでアラームのグループを強制的に有効にしたり、アクティブに抑止したりできます。

個々のタグのイベント プロパティの設定

タグを定義してイベントの監視を行う場合、タグの値が変化するたびにイベント メッセージがアラーム システムにログ記録されます。イベント メッセージでは、値がどのように変更されたかがログ記録されます。たとえば、オペレータ、I/O、QuickScript、またはシステムのいずれかによって変更が発生したかが記録されます。

1. タグ名ディクショナリを開きます。
2. イベントとしてログ記録されるデータに関連付ける既存のタグを選択するか、新しいタグを作成します。
3. [イベント ログ] を選択します。[優先度] ボックスが使用可能になります。[優先度] に入力する値によって、タグのイベント優先度レベルが決定されます。



4. [優先度] ボックスに、イベント優先度としての数字を 1 ～ 999 の範囲で割り当てます。1 が最も高いイベント優先度であり、999 が最も低い優先度です。
5. [保存] をクリックします。
6. [閉じる] をクリックして、[タグ名ディクショナリ] ダイアログ ボックスを閉じます。

アラームとイベントのグローバル設定

アプリケーションによって生成されるすべてのアラームやイベントに適用される以下のグローバル設定を構成できます。

- 内部アラーム メモリ (バッファ) のサイズ。
- アラームが通常状態に戻ったことが確認を意味するかどうか。詳細については、[「タグ値が平常に復帰するときの自動確認の使用」](#)を参照してください。
- イベント ロギング。
- WindowViewer が再起動されるときに、アラームの有効化状態を保持するかどうか。
- アラームの確認コメントを一般のアラーム コメントとして使用するかどうか。詳細については、[「アラーム コメントと確認コメントの使用」](#)を参照してください。
- アラーム クライアント コントロール グリッドに LATCHED アラームを表示するかどうか。詳細については、[「ラッチ済み状態の有効化」](#)を参照してください。

アラーム バッファ サイズの設定

分散アラーム システムでの通信の大部分は、ノード間で送信されるアラーム クエリーとアラーム レコードにより構成されます。ノード内では、アラーム クエリーとアラーム レコードはアラーム バッファとも呼ばれる InTouch 内部アラーム メモリに保持され、ネットワーク トラフィックが最小限に抑えられています。アラーム バッファ サイズは、サマリまたは履歴アラーム クエリーに対してノードが保存できるアラームの最大数です。新しいレコード用の容量を作成するため、アラーム バッファは最も古いレコードを削除します。

メモリに保存されているアラーム イベントのみがアプリケーション ウィンドウに表示できます。InTouch アプリケーションでアラーム ステータスが表示されない場合、バッファ サイズを 1 に設定して、ノード メモリを節約できます。

アラーム バッファに大きな値を割り当てると、ノードのパフォーマンスに影響が与えられる可能性があります。分散アラーム システムの場合、デフォルト値の 500 が推奨されます。

アラーム バッファ サイズを設定するには

1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[アラーム] をクリックします。
[アラーム] 設定画面が表示されます。

General

Alarm buffer size: entries

☒ RTN implies ACK ☐ Alarm Enable retentive
☒ Events enabled ☐ Retain ACK comment as alarm comment
☐ Alarm Latch enabled

3. [アラーム バッファ サイズ] ボックスで、サマリまたは履歴クエリー用のメモリ アラーム バッファに保存できるアラーム エントリの最大数を入力します。
4. [保存] をクリックします。

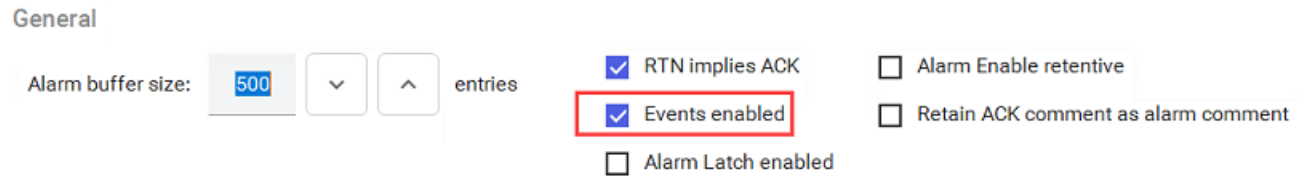
イベントの有効化

イベントを有効にして、アプリケーション内にログ記録できます。イベントは、オペレータ操作、QuickScript、または I/O から発生するアプリケーション データの認識可能な変更を表します。

タグの [イベント ログ] プロパティは、タグに関連付けられたイベントが内部アラーム メモリに保存される前、またはアラーム データベースにログ記録される前に、タグ名ディクショナリから設定される必要があります。タグのイベント ロギングの指定の詳細については、[「個々のタグのイベント プロパティの設定」](#)を参照してください。

イベントを有効にするには

1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[アラーム] をクリックします。
[アラーム] 設定画面が表示されます。



General

Alarm buffer size: 500 entries

☒ RTN implies ACK

☒ Events enabled

☐ Alarm Enable retentive

☐ Retain ACK comment as alarm comment

☐ Alarm Latch enabled

3. [イベント有効] チェック ボックスをオンにして、InTouch アプリケーションの実行中に発生するすべてのイベントをログに記録します。
4. [保存] をクリックします。

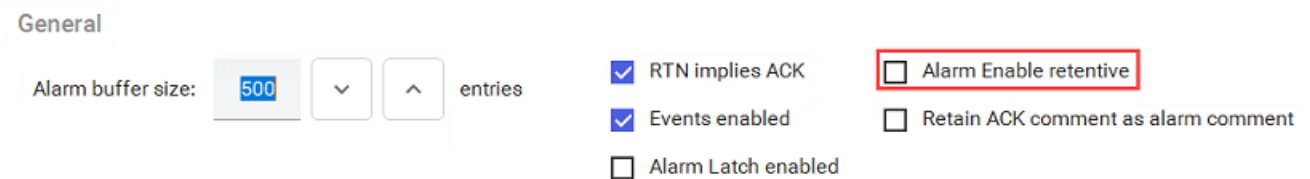
アラームの有効化の保持

InTouch アプリケーションが停止され再起動されるとき、タグの **.AlarmEnabled** ドットフィールドに割り当てられている現在の値を保持するように選択できます。

.AlarmEnabled ドットフィールドに割り当てられている値によって、アラームとイベントのロギングがオンまたはオフに切り替わります。**.AlarmEnabled** ドットフィールドは、タグまたはアラーム グループに割り当てることができます。**.AlarmEnabled** ドットフィールドがアラーム グループに割り当てられると、指定したアラーム グループ内でそのタグに関連付けられているすべてのアラームがログ記録されるかどうか決定されます。

アラームの有効化を保持するには

1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[アラーム] をクリックします。
[アラーム] 画面が表示されます。



General

Alarm buffer size: 500 entries

☒ RTN implies ACK

☒ Events enabled

☒ Alarm Enable retentive

☐ Retain ACK comment as alarm comment

☐ Alarm Latch enabled

3. [アラーム イベントの保持] チェック ボックスをオンにして、InTouch アプリケーションの再起動時に **.AlarmEnabled** ドットフィールドの現在の状態を初期値として保持します。
4. [保存] をクリックします。

ラッチ済み状態の有効化

LATCHED (ラッチ済み) 状態を有効にして、アラーム クライアント コントロール グリッドで LATCHED アラームを表示できます。次の場合、アラームは LATCHED 状態になります。

- アラームが UNACK_RTN 状態から確認された場合。
または
- アラームが ACK 状態から通常状態に戻った (RTN) 場合。

ラッチ済み状態を有効にするには

1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[アラーム] をクリックします。
[アラーム] 設定画面が表示されます。

General

Alarm buffer size:

500



entries

☒ RTN implies ACK☐ Alarm Enable retentive☒ Events enabled☐ Retain ACK comment as alarm comment☐ Alarm Latch enabled

3. [アラーム ラッチ有効] チェック ボックスをオンにして、アラーム クライアント コントロール グリッドに LATCHED アラームを表示します。
4. [保存] をクリックします。

アラーム グループ リスト ファイルの作成

アラーム グループ リストを作成するには、分散名前マネージャを使用します。その後で、ローカル ノードおよびリモート ノードからの既存のアラーム グループをリストに追加します。

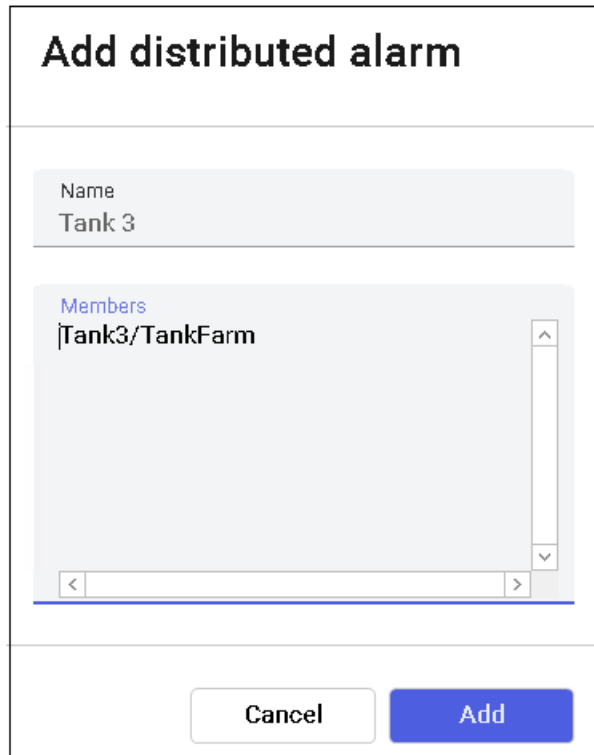
以下の表には、分散名前マネージャからアラーム グループを指定するための構文が示されています。

ノード	アラーム グループ構文
ローカル	\InTouch!Group_Name または、 .Group_Name
リモート	\\Node_Name\InTouch!Group_Name または、 Node_Name.Group_Name

これらの例の場合、Node_Name は InTouch リモート ノードの名前です。Group_Name は、アラーム グループの名前です。アラーム グループ リストが定義されているローカル ノードでアラーム グループが定義されている場合は、前にピリオドを付けることによってアラーム グループ名を簡単に入力できます。たとえば、".Group_Name" と入力します。

アラーム グループ リストを作成するには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[分散名前マネージャ] をクリックします。
[分散名前マネージャ] 設定画面が表示されます。
3. [分散アラーム] タブで [追加] (+) アイコンをクリックします。
[分散アラームの追加] ダイアログが表示されます。



Add distributed alarm

Name
Tank 3

Members
[Tank3/TankFarm]

Cancel Add

4. [メンバー] ボックスに、クエリーに含める InTouch ノードとアラーム グループのリストを入力します。

ノード名とアラーム グループ名は、標準グループ エントリ 構文またはピリオドを使用したショートカット エントリを使用して入力できます。ショートカット エントリは、アラーム グループ リストを保存する際に標準グループ エントリに変換されます。

注記: Node.Group および .Group の構文は、この設定ダイアログ ボックスでのみ使用できます。アラーム表示の設定やアラーム QuickScript 関数では有効ではありません。

5. [追加] をクリックして、このリストをアラーム グループ ファイルに追加します。
メンバーの構文が自動的に変換されます。
6. [保存] をクリックします。
7. アラーム グループ リストの名前を **Alarm Viewer** コントロールのクエリーに追加します。これで、リストで指定したグループすべてのアラームが **Alarm Viewer** コントロールで表示されるようになります。

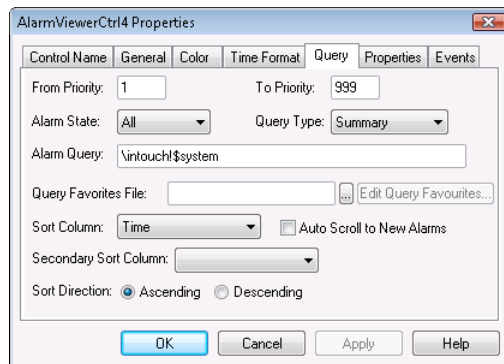
アラーム クエリー

アラーム クエリーは以下のいずれかを取得します。

- InTouch 内部アラーム メモリまたはアラーム データベースからのアラームとイベント（履歴アラーム）。
- InTouch 内部アラーム メモリからの現在のアラーム（サマリ アラーム）

InTouch アラーム コントロールを設定する際、クエリー ソースを指定します。クエリ結果をフィルタするためのクエリ オプションを選択することもできます。

以下の図は、Alarm Viewer ActiveX コントロールの [クエリー] タブを示しています。

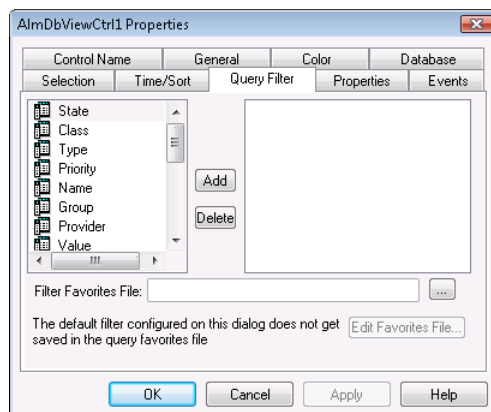


この例では、以下の条件によって選択されたアラーム データを表示するアラーム表示を作成します。

- アラーム優先度 (1～999)
- アラーム状態 (すべて、確認、または未確認)
- クエリー タイプ (サマリまたは履歴)
- アラーム グループ (ローカルまたはリモート データ ソース)

クエリーは「クエリー設定」ファイルと呼ばれる .xml ファイルに保存できます。ランタイム時、このファイルに保存された選択条件を使用して別のクエリーを実行することによって、アラーム表示を新しいアラーム データで更新できます。

その他の InTouch アラーム ActiveX コントロールはより広範囲のクエリー条件を提供しています。以下の図は、Alarm DB Viewer コントロールの [クエリー フィルタ] タブを示しています。



クエリーは、ダイアログ ボックスの左ペインに表示されているリストからのアラーム属性またはイベント属性を選択することによって作成します。その後、値を選択した属性に割り当てます。最後に、ブール型演算子を使用して属性を組み合わせ、クエリー フィルタ条件を設定します。

アラーム メモリからアラーム レコードおよびイベント レコードを選択するためのクエリー関数またはドットフィールドを含む QuickScript を記述できます。次の Alarm Viewer コントロール ステートメントでは、ApplyQuery() メソッドを使用して、アラーム メモリに対してクエリーを実行しています。

```
#AlarmViewerCtrl11.ApplyQuery ("\\InTouch!$System",500,600,"All", "Historical");
```

このステートメントは、500 ～ 600 の間の優先度で「\\InTouch!\$System」で指定されているすべての履歴アラームを取得します。選択したアラーム レコードは Alarm Viewer コントロール画面に表示されます。

アラーム クエリーの例

ローカル ノードでは、この構文の後にアラーム クエリーを使用します。

```
\Provider!AlarmGroup
```

次に例を示します。

```
\InTouch!$System
```

リモート ノードには次のクエリー構文を使用します。

```
\\NodeName\Provider!AlarmGroup
```

たとえば、MyNode1 というノードでは次のようになります。

```
\\MyNode1\InTouch!$System
```

Galaxy からのアラームのクエリーを実行するには、["Galaxy" ではなく "Galaxy_<Galaxy 名>" を使用する] チェック ボックスが選択されている状態で、次の構文を使用します。この構文では、特定のコンピュータ上の特定の領域にあるオブジェクトの特定のアラーム名からアラームを取得します。アラーム名は、属性名またはアラーム プリミティブ名の場合があります。Galaxy 名は、アラーム ウィンドウの [プロバイダー] カラムに表示されます。

```
\\NodeName\Galaxy_GalaxyName!AreaName!ObjectName.AlarmName
```

次の構文では、特定の領域からすべてのアラームを取得します。

```
\Galaxy_GalaxyName!AreaName
```

次の構文では、2 つの領域からアラームを取得します。

```
\Galaxy_GalaxyName!Area1 \Galaxy_GalaxyName!Area2
```

次の構文では、指定したコンピュータ ノードの Platform から指定した領域内のすべてのアラームを取得します (デフォルト)。

```
\\NodeName\Galaxy_GalaxyName!AreaName
```

単一のワイルドカードを使用して、指定した領域内の複数のアラーム名に一致させることもできます。次の構文では、領域「AreaName」にあり「Tank」という文字で始まるすべてのオブジェクトからすべてのアラームを取得します。

```
\Galaxy_GalaxyName!AreaName!Tank*
```

次の構文では、「AreaName」領域にあるすべてのオブジェクトから「Hi」という名前が付いているすべてのアラームを取得します。

```
\Galaxy_GalaxyName!AreaName!*.Hi
```

詳細な InTouch クエリー情報の取得

以下の表には、各 InTouch クエリー ソースのクエリーに関する詳細情報のリファレンスが一覧表示されています。

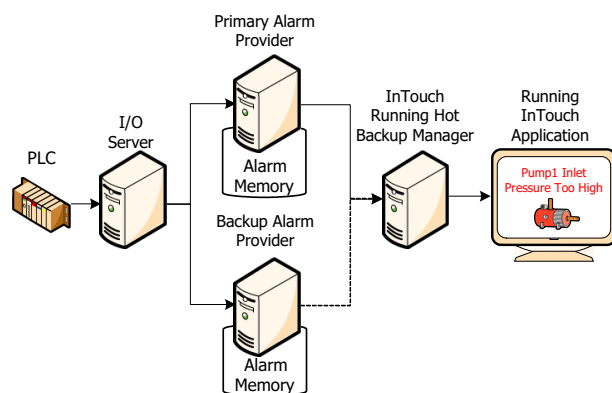
クエリ ソース	参照先
Alarm DB Logger Manager	ログするアラームの設定

クエリ ソース	参照先
Alarm Printer ユーティリティ	印刷するアラームの設定
Alarm Viewer コントロール	表示するアラームの設定
Alarm Tree Viewer コントロール	表示するプロバイダとグループの設定
Alarm Pareto コントロール	分析するアラームの設定
分散アラーム表示オブジェクト	表示するアラームの設定
Hot Backup Manager	アラーム レコード マッピング ファイルの作成

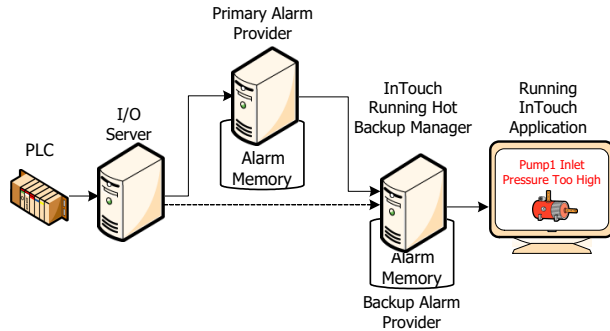
アラーム冗長性による工場セキュリティの強化

InTouch 分散アラーム システムは通知を発行し、ネットワーク内のリモート ノードで実行中のアプリケーションからアラーム確認を受け取ります。アラーム プロバイダ アプリケーションでは、メモリ内にアラーム データが保存されます。アラーム コンシューマ アプリケーションはその他のノードでクライアントとして実行して、アラーム プロバイダからのアラームをリモートでクエリーしたり、表示したり、確認したりします。

重複するアラーム プロバイダを作成するには、**Alarm Hot Backup Manager** を使用します。以下の図は、**Hot Backup Manager** がセカンダリの現在のアラーム リポジトリをバックアップ プロバイダとして使用する方法を示しています。



次の図に示すように、**Hot Backup Manager** とバックアップ プロバイダを同じノード上で実行することもできます。



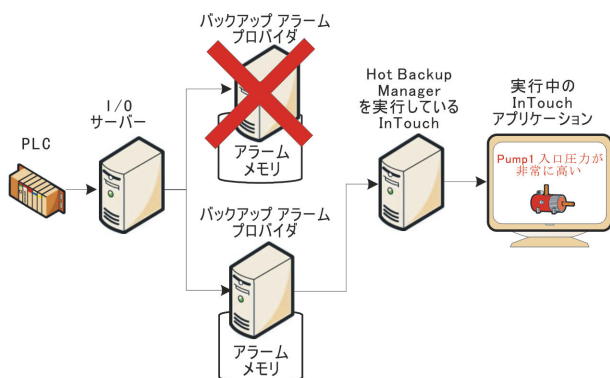
Hot Backup の理解

Hot Backup には、プライマリとバックアップ（**Hot Backup ペア**）という 2 つのアラーム プロバイダをポイントする 1 つの名前（**Hot Backup ペア名**）が提供されています。InTouch HMI アラームのコンシューマは、この 1 つの **Hot Backup** ペア名を参照して、プライマリまたはバックアップアラーム プロバイダのいずれかからアラームを取得できます。次のアラーム クライアントは、**Hot Backup** ペアのクエリーをサポートします。

- InTouch HMI Alarm DB Logger Manager
- InTouch HMI Alarm Printer
- InTouch HMI Alarm Viewer コントロール
- アラーム クライアント コントロール

両方のプロバイダ ノードが正常に動作している場合、アラーム コンシューマはプライマリ プロバイダからアラーム データを受け取ります。ただし、プライマリ プロバイダが失敗した場合、アラーム コンシューマは代わりにバックアップからアラーム データを受け取ります。プライマリ プロバイダが使用できる場合でもバックアッププロバイダを使用するように **Hot Backup Manager** を設定することもできます。このバックアップ オプションを選択しない場合、プライマリ プロバイダが使用可能な場合、プライマリ プロバイダに自動的に切り替わります。詳細については、「[Hot Backup ペアの作成](#)」を参照してください。

以下の図は、プライマリ アラーム プロバイダが失敗した後も引き続きアラーム コンシューマがアラームを受け取る様子を示しています。アラーム コンシューマは引き続き **Hot Backup** ペアを参照しますが、バックアッププロバイダがアラーム データを提供します。



Hot Backup のアラーム プロバイダの指定

以下のアラーム プロバイダのどれでも指定することができます。

- InTouch

InTouch をアラーム プロバイダとして指定する場合、プライマリ ノードとバックアップ ノードの両方に対してプロバイダが InTouch である必要があります。ただし、バックアップ ノードのアラーム グループは、プライマリ ノードとは異なるものにできます。

- Galaxy

Galaxy または Galaxy_<Galaxy 名> アラーム プロバイダを指定するには、IDE で AppEngine 冗長性を設定する必要があります。

- Galaxy_<Galaxy 名>

Galaxy 名に対する有効な文字は、英数字と特殊記号 \$、#、および _ です。

Galaxy_<Galaxy 名> を指定する場合、プライマリ ノードとバックアップ ノードの両方に対して Galaxy 名が同じである必要があります。

Galaxy または Galaxy_<Galaxy 名> をアラーム プロバイダとして指定する場合、バックアップ ノードのプロバイダは Galaxy または Galaxy_<Galaxy 名> にできます。

バックアップ ノードのアラーム グループは、プライマリ ノードと同じである必要があります。

注意：システムは、2 つの異なる Galaxy から生成されたアラームの Hot Backup ペアをサポートしません。

Hot Backup Manager について

Hot Backup Manager では、プライマリ プロバイダとバックアップ プロバイダの間でアラーム確認が同期化されます。アラームがプライマリ プロバイダで確認される場合、同じアラームがバックアップ プロバイダでも同時に確認されます。

Hot Backup Manager :

- 設定ユーティリティを提供して、バックアップ ペアを作成します。
- 設定ユーティリティを提供して、Hot Backup ペアのプライマリ プロバイダとバックアップ プロバイダの間でアラーム レコードをマップします。
- InTouch アラーム バックアップ ペアの間でアラーム確認を同期化します。
- 分散アラーム システムが開始および停止するときに、Hot Backup システムに属するすべてのノード間で通信が確立されます。

TSE セッションでの Alarm Hot Backup Manager の使用

[ツール] > [アプリケーション] を使用することで、異なる WindowMaker の Terminal Services Edition (TSE) セッションで、1 つの Alarm Hot Backup Manager のみを起動できます。

[スタート] > [プログラム] ショートカットを使用することで、各 TSE セッション内で 1 つの Alarm Hot Backup Manager を起動できます。最後に保存された設定は、以前の設定を上書きします。

Hot Backup ペアのアラームのクエリーは、ランタイム時に TSE セッション内でサポートされています。

Hot Backup ペアの設定

Alarm Hot Backup Manager では、プロバイダ アプリケーションを実行している 2 つのホスト ノードからバックアップ ペアが作成されます。Hot Backup Manager は、WindowMaker から起動できます。Hot Backup ペアを設定するには :

- Hot Backup ペアを作成します。
- アラーム レコードのキー フィールドを設定します。
- アラーム レコードのキー フィールドをマップします。
- アラーム レコードマップを Hot Backup Manager にインポートします。

Hot Backup ペアの作成

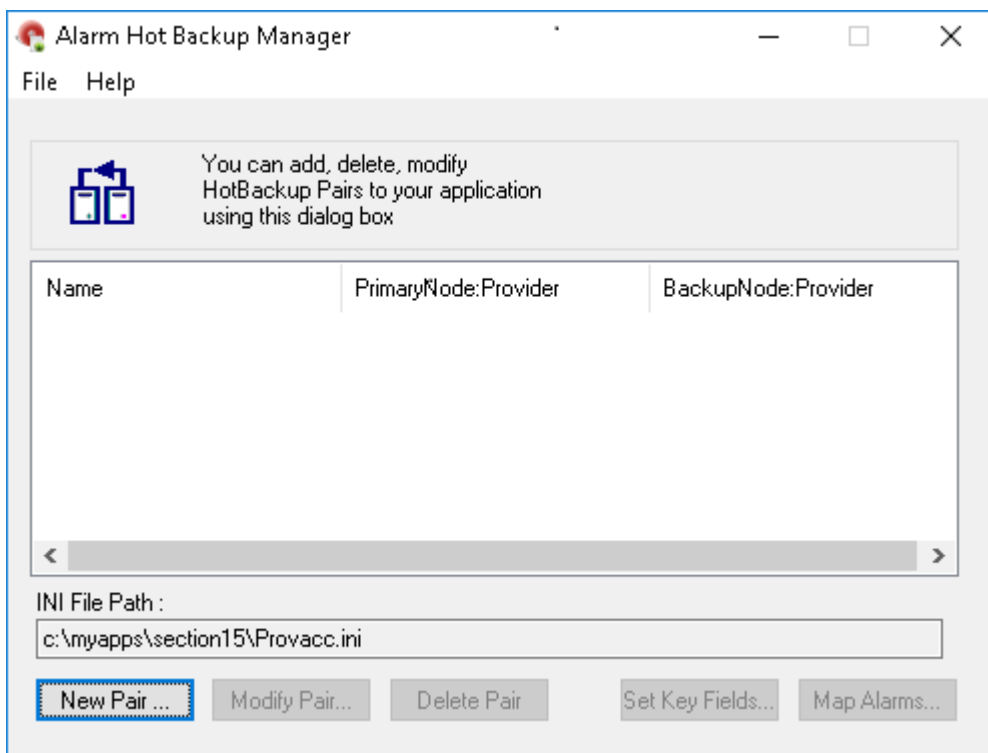
Hot Backup ペアを設定するには:

- Hot Backup ペアに名前を割り当てます。
- プライマリ アラーム プロバイダを識別します。
- バックアップ アラーム プロバイダを識別します。

設定情報を含む Provacc.ini ファイルも指定できます。

Hot Backup ペアを設定するには

1. Alarm Hot Backup Manager を開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm Hot Backup Manager] をダブルクリックします。



2. [ファイル] メニューで、[開く] をクリックします。Provacc.ini ファイルを選択し、[OK] をクリックします。

デフォルトでは、Alarm Hot Backup Manager は最後に開いた InTouch アプリケーション フォルダにある Provacc.ini ファイルを確認します。InTouch アプリケーション フォルダにある Provacc.ini ファイルを使用する必要があります。それ以外の場合は、別に指定したフォルダに Provacc.ini ファイルのコピーを作成し、それを Hot Backup Manager で使用するために選択します。

3. [ペアの作成] をクリックします。[新規ペアの追加] ダイアログ ボックスが表示されます。

The screenshot shows the 'Add New Pair' dialog box. It has a title bar with the text 'Add New Pair' and a close button (X). The dialog is divided into several sections. The first section is 'Hot Backup Pair Name' with a text input field. The second section is 'Primary Node' with three fields: 'Name' (text input), 'Provider' (dropdown menu showing 'InTouch'), and 'Group' (text input). The third section is 'Backup Node' with three fields: 'Name' (text input), 'Provider' (dropdown menu showing 'InTouch'), and 'Group' (text input). The fourth section is 'Backup Option' with a checkbox labeled 'Switch back to Primary Node only when the Backup Node is down.' At the bottom of the dialog are two buttons: 'OK' and 'Cancel'.

4. [Hot Backup ペアの名前] ボックスに、新しいバックアップ ペアの一意の名前を入力します。
- ペア名には、32 文字以下の英数字を使用できます。ペア名には、ドル記号 (\$)、ポンド記号 (#)、およびアンダースコア (_) 文字を使用できます。
5. [メイン ノード] 領域で、プライマリ ノードを設定します。以下の手順を実行します。
- [名前] ボックスに、プライマリ プロバイダ アプリケーションを実行中のコンピュータのノード名を入力します。ノード名は、Hot Backup Manager に固有である必要があります。存在しないノード名を入力した場合、またはノード名が別の Hot Backup ペアで使用されている場合、エラーメッセージが表示されます。
 - [プロバイダ] ボックスで、ドロップダウン リストからアラーム プロバイダを選択します。デフォルトは InTouch です。
 - [グループ] ボックスに、プライマリ プロバイダからのアラームをクエリーするアラーム グループの名前を入力します。
6. [バックアップ ノード] 領域で、バックアップ ノードを設定します。以下の手順を実行します。
- [名前] ボックスに、バックアップ プロバイダ アプリケーションを実行中のコンピュータのノード名を入力します。これは Hot Backup Manager が実行されているのと同じノードでもかまいません。

- b. Galaxy または Galaxy_<GalaxyName> のアラーム プロバイダを指定した場合、[プロバイダ] ボックスでバックアップ プロバイダを選択します。

Galaxy または Galaxy_<GalaxyName> をプライマリ ノード アラーム プロバイダとして指定した場合、バックアップ ノードのプロバイダは Galaxy または Galaxy_<GalaxyName> である必要があります。

InTouch をプライマリ ノード アラーム プロバイダとして指定した場合、バックアップ ノードのプロバイダは InTouch である必要があります。

- c. [グループ] ボックスに、バックアップ プロバイダからのアラームをクエリーするアラーム グループの名前を入力します。

Galaxy または Galaxy_<GalaxyName> をアラーム プロバイダとして指定した場合、バックアップ ノード グループはプライマリ ノード グループと同じである必要があります、それは編集できません。

Configure Hot Backup Pair

Hot Backup Pair Name

GalaxyPair

Primary Node

Name : Primary

Provider : Galaxy

Group : Area_001

Backup Node

Name : Backup

Provider : Galaxy_Demo01

Group : Area_001

Backup Option

☐ Switch back to Primary Node only when the Backup Node is down.

OK Cancel

7. プライマリ ノードが使用可能なときでもバックアップ ノードを継続して使用するには、[バックアップ ノードが使用できない場合にのみプライマリ ノードに切り替える] を選択します。（このオプションは英語以外のオペレーティング システムではテストされていません。）

デフォルトでは、このオプションは選択されていません。

8. [OK] をクリックします。
9. [ファイル] メニューで、[保存] をクリックします。

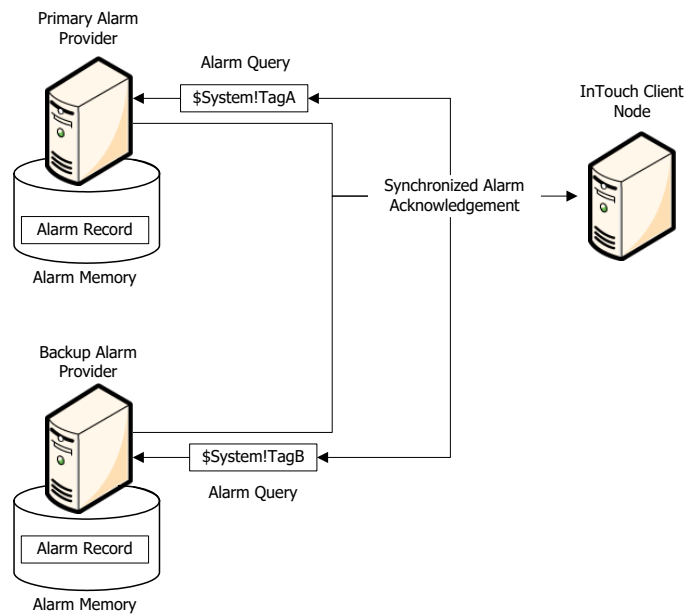
10. WindowMaker を再起動します。

Hot Backup ペアのアラーム キー フィールドの設定

プライマリ プロバイダとバックアップ プロバイダの間でアラーム確認を同期化するには、タグ変数アラーム レコード フィールドの組み合わせを識別する必要があります。フィールドのこの組み合わせによって、各プロバイダの現在のアラーム リポジトリに保存されるペアのアラーム レコードに固有のマッピング キーが生成されます。

InTouch アラーム プロバイダに対してのみ、アラーム キー フィールドを設定、およびアラームをマップできます。

以下の図は、標準クエリー内のアラーム レコード フィールドに基づく同期化されたアラーム確認要求を示しています。



マッピング キーは、デザイン時およびランタイムのアラーム レコードの組み合わせで構成できます。デザイン時アラーム レコードは、タグ変数ディクショナリから定義される場合、タグ変数のアラーム プロパティに基づいています。

たとえば、アラーム名フィールドはプライマリおよびバックアップのノードアプリケーションで定義されるタグ変数の名前を使用するため、デザイン時に認識されています。**QuickScript** またはオペレータアクションによって、アプリケーションの実行中にレコードとして保存されるアラーム プロパティが定義または変更されます。

マップ キーは、デザイン時およびランタイムのアラーム レコード フィールドの組み合わせから作成できます。マップ キーは、各プロバイダの現在のアラーム リポジトリから **1** つのレコードのみを選択する必要があります。キー フィールドでは、固有のクエリーが作成される必要があります。

Hot Backup Manager を使用して、アラーム レコード フィールドからマッピング キー リストを作成します。

注意： Galaxy または Galaxy_<Galaxy 名> アラーム プロバイダを指定する場合、[キーフィールドの設定] と [アラームのマップ] オプションの両方が無効です。

Hot Backup ペアのアラーム フィールド マッピング リストを作成するには

1. Alarm Hot Backup Manager を開きます。以下の操作を行います。
 - a. [ツール] ビューで、[アプリケーション] を展開します。
 - b. [Alarm Hot Backup Manager] をダブルクリックします。
2. リストから Hot Backup ペアを選択します。
3. [キー フィールドの設定] をクリックします。[キー フィールドの選択] ダイアログ ボックスが表示されます。



4. [アラーム レコード フィールド] 領域で、マッピング キー リストに含めるアラーム レコード フィールドを選択します。

選択したアラーム レコード フィールドは、[選択されたフィールド] リスト ボックスに表示されます。
5. 選択したアラーム レコード フィールド用に [デザイン時] または [ランタイム] を選択します。
6. [OK] をクリックします。
7. WindowMaker を再起動します。

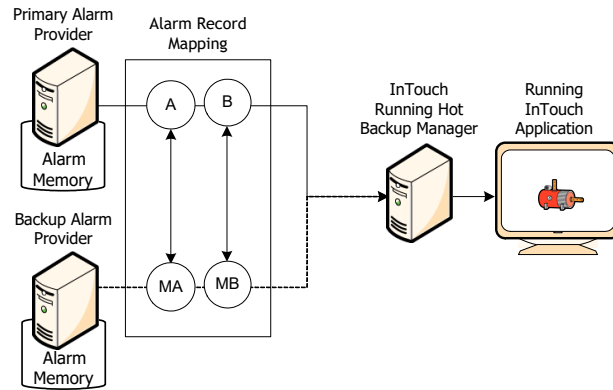
アラーム レコード マッピング ファイルの作成

プライマリとバックアップのアラーム プロバイダが異なるアプリケーションを実行しているときはいつでも、Hot Backup ペアのアラーム レコードをマップする必要があります。アラーム レコード マッピングでは、プライマリ プロバイダとバックアップ プロバイダの異なるアラーム レコード間の対応が確立されます。たとえば、割り当てられた InTouch タグ変数名に基づいてアラーム レコードをマップできます。名前は異なる可能性がありますが、アラーム レコードは、2つのプロバイダ アラーム リポジトリの間で論理的に一貫性を保っています。

注意：プライマリ プロバイダとバックアップ プロバイダの両方が同じアプリケーションを実行している場合は、アラーム レコード マッピング ファイルの作成は必要ありません。マッピング ファイルが提供されていない場合、分散アラーム システムでは、プライマリ プロバイダとバックアップ プロバイダが同じアラーム レコードを使用して同じアプリケーションを実行していると想定されます。

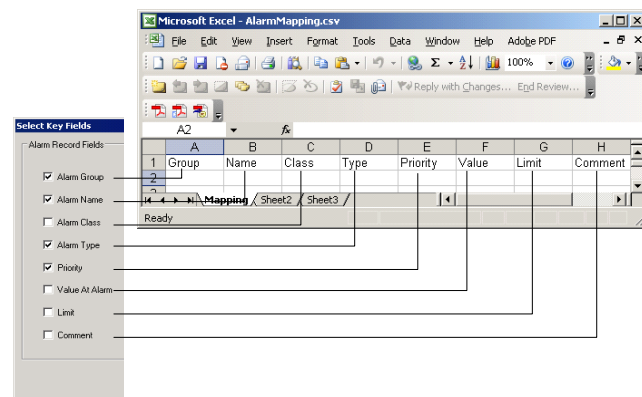
マッピングによって、アラーム確認は異なるアプリケーションを実行中のプロバイダの間で同期化できます。分散アラーム システムがプロバイダのアラームを確認する際、別のプロバイダでどのアラームを確認するべきなのかも把握しています。

以下の図は、**Hot Backup** ペアの 2 つのプロバイダ間のアラーム レコード マッピングを示しています。この例では、プライマリ プロバイダの **A** アラーム レコードと **B** アラーム レコードがバックアップ プロバイダの対応するアラーム レコードにマップされます (**MA** と **MB**)。

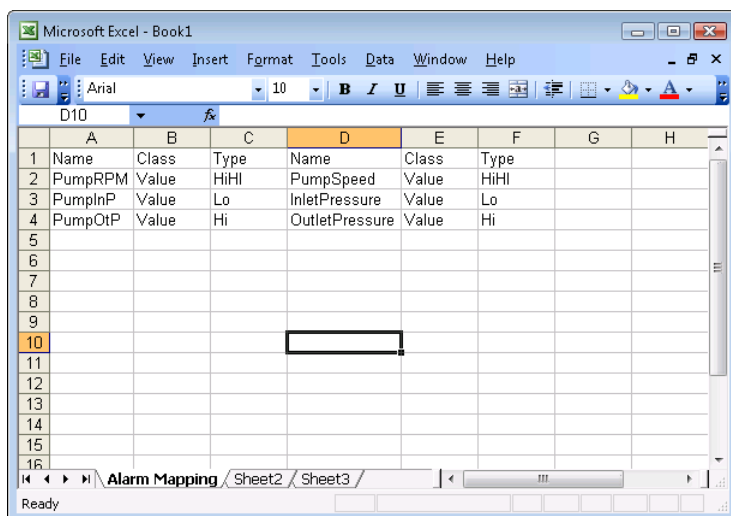


Hot Backup Manager は、**Microsoft Excel** またはメモ帳のようなテキスト エディタで作成するカンマ区切り値 (CSV) ファイルからアラーム レコード マップをインポートします。マッピング ファイルには、プライマリ プロバイダとバックアップ プロバイダの対応するアラーム レコードを関連付けるアラーム レコード フィールドの順序付きリストが含まれます。

タグ変数アラーム レコード フィールドをマッピング ファイルのヘッダとして指定する必要があります。ファイル内のヘッダの順序は、**[キー フィールドの選択]** ダイアログ ボックスに表示されるアラーム レコード フィールドと一致する必要があります。以下の図は、**[キー フィールドの選択]** ダイアログ ボックスのアラーム レコード フィールドの順序に一致する **Excel** ファイルのカラム ヘッダを示しています。



マッピング キーを生成するために使用されるアラーム フィールド レコードの選択されたヘッダのみを含むマッピング ファイルを作成できます。以下の図は、アラーム名、クラス、およびタイプのヘッダのみを含む **Excel** ファイルを示しています。ヘッダを追加するとき、ヘッダの順序は、**[キー フィールドの選択]** ダイアログ ボックスのアラーム レコード フィールドの順序と常に一致する必要があります。



カラムの左セットでプライマリ プロバイダのアラーム フィールドレコードを指定します。同じように、カラムの右セットでバックアッププロバイダの同じレコードを指定します。

マッピング ファイ

ルカラム ヘッド アラーム フィールド レコードに割り当てられる値

グループ タグ変数が割り当てられたアラーム グループの名前。アラームグループ名には空白を含めることはできません。

名前 アラーム レコードがマップされるタグ変数の名前。タグ変数名に空白を含めることはできません。

クラス タグ変数に割り当てられるアラームのクラス。
可能なクラスの値は以下のとおりです。

- VALUE - 値アラーム
- DEV - 偏差アラーム
- ROC - 変更率アラーム
- DSC - 論理値アラーム

タイプ アラーム クラスに関連付けられるアラーム条件のタイプ

- LOLO、LO、HI、HIHI - 値アラーム
- MinDev と MajDev - 偏差アラーム
- ROC - 変更率アラーム
- DSC - 論理値アラーム

優先度 アラーム条件に割り当てられる優先度。優先度は 1 ～ 999 の数である必要があります。

値 以下の注記を参照してください。

しきい値 以下の注記を参照してください。

マッピング ファイル
ル カラム ヘッド

アラーム フィールド レコードに割り当てられる値

コメント 以下の注記を参照してください。

アラーム値、しきい値、およびコメントのカラム：

- 特定のノードの特定のレコードに対する「クラス」または「タイプ」の値が不明である場合、「アラーム値」と「しきい値」のカラム値には、Null 以外の値であればどれでも使用できます。
- 特定のノードの特定のレコードに対する「クラス」の値が Value、Dev、または ROC である場合、「アラーム値」と「しきい値」のカラム値は 1234567890.-+eE の文字のみを受け入れることができます。
- 特定のノードの特定のレコードに対する「タイプ」の値が LOLO、LO、HI、HIHI、MinDev、MajDev、または ROC である場合、「アラーム値」と「しきい値」のカラムには 1234567890.-+eE の文字のみを受け入れることができます。
- 特定のノードの特定のレコードに対する「クラス」または「タイプ」のうちいずれかの値が DSC である場合、「アラーム値」と「しきい値」のカラム値は Null 以外の値であればどれでも使用できます。
- 「コメント」カラムの値には、制限はありません。
- マッピング ファイルのすべてのレコードは、固有のものである必要があります。Hot Backup Manager では、インポート プロセスの間に重複レコードが省略されます。インポート プロセスが完了した後で、詳細を表示できます。

グループ、アラーム名、および優先度など、アラーム レコードのフィールド値を組み合わせ、アラーム レコードを固有に識別する「合成マッピング キー」を生成できます。

InTouch アラーム プロバイダは、アラームを生成したタグ変数の名前と、[アラーム名] フィールドを同じにします。したがって、Hot Backup ペアが指定されている場合、アラーム グループ名とタグ変数名の組み合わせを使用して、マッピング キーを生成できます。

たとえば、

プロバイダ ノード バックアップ ノード

\$System!TagA	\$System!TagB
---------------	---------------

プロバイダが固有フィールドとしてアラーム名フィールドとコメント フィールドを一緒に持つ場合、マッピング キーはアラーム名とコメントの組み合わせにすることができます。

プロバイダ ノード	バックアップ ノード
-----------	------------

tagA!CommentA	tagB!CommentB
---------------	---------------

3 番目のプロバイダのその他のフィールドの組み合わせに対しても同じです。

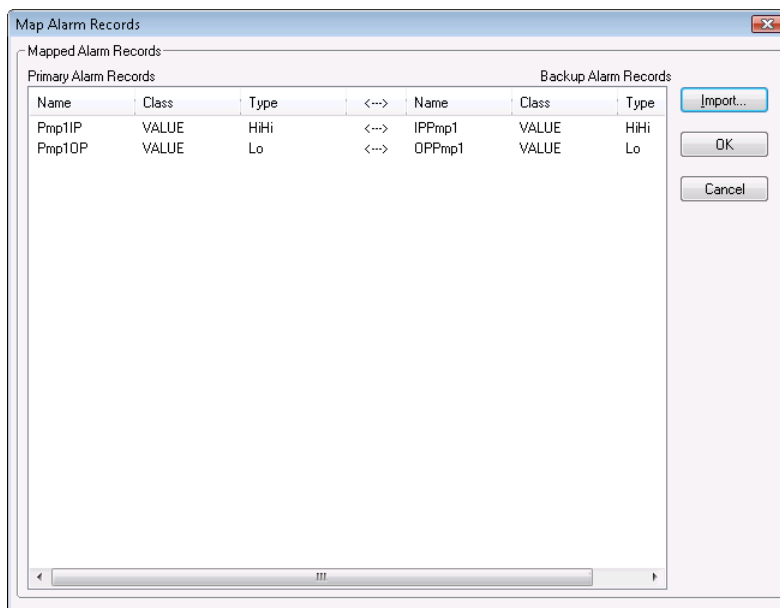
アラーム レコード マッピング ファイルのインポート

アラーム レコード マッピング ファイルの内容を Hot Backup Manager にインポートできます。

マッピング ファイルをインポートするとき、[アラーム クラス] フィールドと [アラーム タイプ] フィールドの間の検証は行われません。

アラーム レコードをマッピング ファイルからインポートするには

1. Alarm Hot Backup Manager を開きます。以下の操作を行います。
 - a. [ツール] ビューで、[アプリケーション] を展開します。
 - b. [Alarm Hot Backup Manager] をダブルクリックします。
2. リストから Hot Backup ペアを選択します。
3. [アラームのマップ] をクリックします。[アラーム レコードのマップ] ダイアログ ボックスが表示されます。



4. [インポート] をクリックします。[開く] ダイアログ ボックスが表示されます。マッピング ファイルを選択し、[開く] をクリックします。

Hot Backup Manager がファイルからのレコードのインポートを開始します。
5. すべてのマッピング レコードがインポートされたら、[OK] をクリックします。
6. [ファイル] メニューで、[保存] をクリックします。
7. WindowMaker を再起動します。

アラーム マッピング ファイルインポート問題のトラブルシューティング

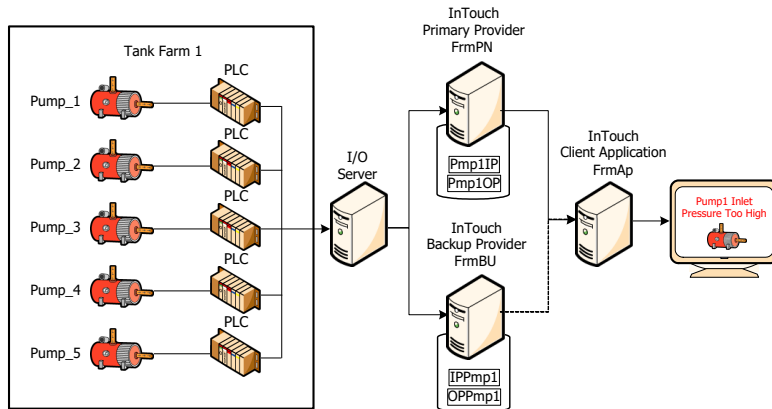
以下の状況では、ファイルのインポートが妨げられます。

- 必要なカラム数に、インポート ファイルのすべてのレコードの値が入力されている必要があります。レコードに対して、値が多すぎても少なすぎてもいけません。
- インポート ファイルの見出しは、[キー フィールドの選択] ダイアログ ボックスと同じ見出しでなければならず、同じ順序でなければなりません。

インポートされるレコードに間違っただけのエントリがあると、その特定のレコード番号を省略するか、インポート プロセス自体を中止するよう求めるメッセージが表示されます。

Hot Backup ペアの例

このセクションでは、アラーム バックアップ ペアを設定する一般的な作業シナリオについて説明します。以下の図は、InTouch タンク ファーム アプリケーションの **Hot Backup** ペアの設定を示しています。この例では、InTouch アプリケーションがアラーム状況としてポンプ圧力を監視しています。



3 台のコンピュータはすべて InTouch HMI を実行しています。Hot Backup ペアには、プライマリ プロバイダとして **FrmPN**、バックアップ プロバイダとして **FrmBU** が含まれています。これらの 2 つのノードは、InTouch 分散アラーム システム内でアラーム プロバイダとして機能します。

Hot Backup Manager は **FrmAp** ノードで実行します。InTouch クライアント アプリケーションは **FrmAp** で実行され、Hot Backup ペアの 2 つのプロバイダからアラームを取り込みます。

FrmPN を実行している InTouch アプリケーションは、ポンプ入口と出口の圧力用に 2 つのサマリ アラームを生成します。2 つのアラームは、**TnkFrm1** アラーム グループに属します。**FrmBU** で実行している InTouch アプリケーションは、ポンプ圧力用に 2 つの論理的に等しいアラームを生成します。

冗長な Hot Backup ペアを設定するときは、以下の操作を行います。

- Hot Backup ペアを作成します。
- アラーム レコードのキー フィールドを設定します。
- アラーム レコードマッピング ファイルを作成します。
- アラーム レコードマッピング ファイルをインポートします。

Hot Backup ペアを作成するには

1. WindowMaker で InTouch アプリケーションを開きます。この例では、アプリケーションは **FrmAp** ノードで実行します。
2. Alarm Hot Backup Manager を開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm Hot Backup Manager] をダブルクリックします。
3. [ペアの作成] をクリックします。[新規ペアの追加] ダイアログ ボックスが表示されます。
4. 次の図に示すように、[新規ペアの追加] ダイアログ ボックスのオプションを入力します。

Hot Backup Pair Name

BUPair

Primary Node

Name : FrmPN

Provider : InTouch

Group : \$System

Backup Node

Name : FrmBN

Provider : InTouch

Group : \$System

Backup Option

☐ Switch back to Primary Node only when the Backup Node is down.

OK Cancel

5. [OK] をクリックして、[Alarm Hot Backup Manager] ダイアログ ボックスに戻ります。

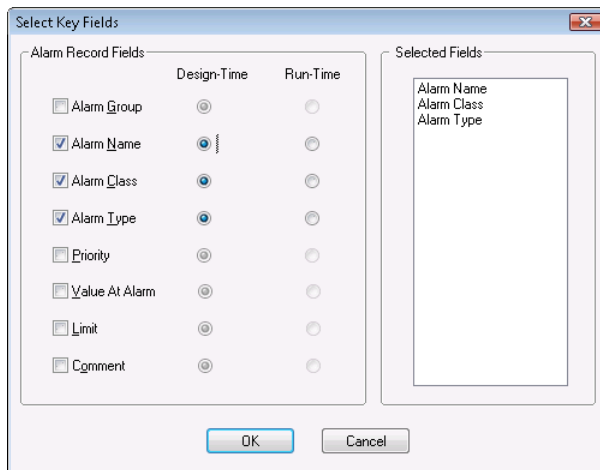
6. [Alarm Hot Backup Manager] ダイアログ ボックスが InTouch で表示されます。

Hot Backup ペアを作成する最初の手順が完了しました。次に、以下の手順に従って、各プロバイダのアラーム リポジトリに保存されているペアのアラーム レコードに固有のマッピング キーを生成します。

アラーム レコード キー フィールドをマップするには

1. リストで Hot Backup ペアを選択します。
2. [キー フィールドの設定] をクリックします。[キー フィールドの選択] ダイアログ ボックスが表示されます。

次の図に示すように、[キー フィールドの選択] ダイアログ ボックスのオプションを入力します。プライマリ プロバイダとバックアップ プロバイダの間のアラームは論理的には一貫していますが、異なる名前が割り当てられており、異なるアラーム グループに属します。各プロバイダのアラーム リポジトリに保存されているレコードへの固有のマッピング キーは、[デザイン時] オプションとして、[アラーム グループ]、[アラーム名]、[アラーム クラス]、および [アラーム タイプ] を選択することによって生成できます。



3. [OK] をクリックします。メッセージが表示されたら、[はい] をクリックします。

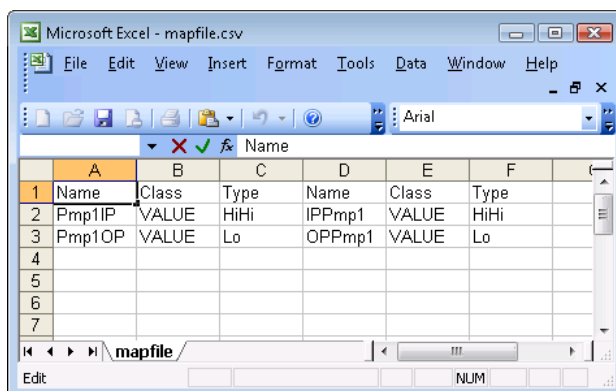
アラーム レコードマッピング キーを作成する 2 番目の手順が完了しました。

ここでは、3 つのノードすべてが InTouch アプリケーションを実行しています。2 つのプロバイダ ノードは同等のアラームを生成しますが、異なるタグ変数を使用しています。プライマリ プロバイダは、ポンプの入口と出口の圧力が高すぎる場合、2 つのサマリ アラームを生成します。バックアップ プロバイダは、同じポンプ圧力アラーム状況に対して 2 つの論理的に同一であるアラームを生成します。次に、以下の手順に従って、各プロバイダのアラーム リポジトリに保存されている等しいレコードを関連付けるマッピング ファイルを作成します。

アラーム マッピング ファイルを作成するには

1. Excel またはメモ帳のようなテキスト エディタで .csv ファイルを作成します。
2. ファイルヘッダの名前を [キー フィールドの選択] ダイアログ ボックスのアラーム レコード フィールド オプションと同じ順序で入力します。

この例では、ファイルヘッダをアラーム グループ、アラーム名、アラーム クラス、およびアラーム 状況のタイプ別に順序付ける必要があります。
3. ファイルの各行で 2 つのプロバイダ間のアラームをマップします。
4. 以下の Excel ファイルの例では、ヘッダとアラーム状況を Hot Backup ペアの 2 つのプロバイダに対して指定する方法を示しています。マッピング ファイルを、クライアント ノードで実行している Hot Backup Manager にアクセス可能な場所に保存します。



アラーム レコードマッピング ファイルを作成する 3 番目の手順が完了しました。

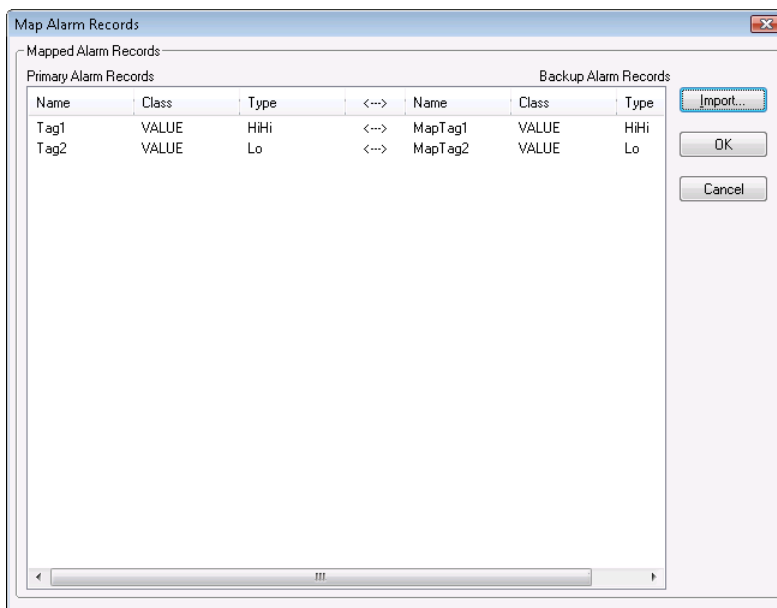
最後の手順で、アラーム マッピング ファイルの内容を **Hot Backup Manager** にインポートします。この例では、クライアント アプリケーションは 2 つのアラーム プロバイダ間でどのポンプ圧力アラーム レコードを確認すべきかを把握しています。

アラーム レコード マッピング .csv ファイルをインポートするには

1. **Alarm Hot Backup Manager** を開きます。

以前に作成したアラーム バックアップ ペアを一覧表示する必要があります。

2. **[アラームのマップ]** をクリックします。**[アラーム レコードのマップ]** ダイアログ ボックスが表示されます。
3. **[インポート]** をクリックします。**[ファイルを開く]** ダイアログ ボックスが表示されます。
4. マッピング ファイルを選択し、**[開く]** をクリックします。**[アラーム レコードのマップ]** ダイアログ ボックスが開き、ファイルからのアラーム マッピング レコードが一覧表示されます。



5. **[OK]** をクリックします。**Hot Backup Manager** がファイルからのレコードのインポートを開始します。
6. すべてのマッピング レコードがインポートされたら、**[OK]** をクリックします。

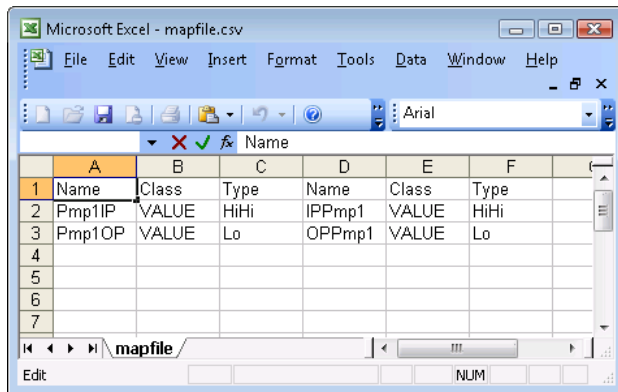
これで **Hot Backup** アプリケーションを実行できます。

確認同期化の例

例：

- アラーム名、アラーム クラス、およびアラーム タイプを、デザイン時キー フィールドとして選択します。
- ランタイム フィールドとして、アラーム グループが選択されています。

以下のアラーム レコード マップ ファイルは、**Microsoft Excel** を使用して作成されました。



The screenshot shows a Microsoft Excel window titled 'mapfile.csv'. The spreadsheet contains a table with 7 rows and 6 columns (A-F). The first row (row 1) has headers: A: Name, B: Class, C: Type, D: Name, E: Class, F: Type. Rows 2 and 3 contain data for Pmp1IP and Pmp1OP respectively. Row 2: A: Pmp1IP, B: VALUE, C: HiHi, D: IPPmp1, E: VALUE, F: HiHi. Row 3: A: Pmp1OP, B: VALUE, C: Lo, D: OPPmp1, E: VALUE, F: Lo. Rows 4-7 are empty.

	A	B	C	D	E	F
1	Name	Class	Type	Name	Class	Type
2	Pmp1IP	VALUE	HiHi	IPPmp1	VALUE	HiHi
3	Pmp1OP	VALUE	Lo	OPPmp1	VALUE	Lo
4						
5						
6						
7						

Pmp1IP アラームは IPPmp1 アラームにマップされます。両方とも、VALUE のクラスおよび HiHi のタイプです。

Pmp1OP アラームは OPPmp1 アラームにマップされます。両方とも、VALUE のクラスおよび Lo のタイプです。

- Pmp1IP の HiHi アラームがプライマリ アラーム ノードで確認されるとき、セカンダリ プロバイダ ノードの IPPmp1 の HiHi アラームにも確認が表示されます。ただし、両方のノードのアラーム グループ名が同じままである必要があります。
- Pmp1OP の Lo アラームがプライマリ アラーム ノードで確認されるとき、セカンダリ プロバイダ ノードの OPPmp1 の Lo アラームにも確認が表示されます。ただし、両方のノードのアラーム グループ名が同じままである必要があります。

確認の同期化は、デザイン時とランタイムのマッピングが一致した場合にのみ発生します。

マッピング用に、デザイン時とランタイムのアラーム レコード フィールドの任意の組み合わせを選択できます。ただし、マッピングの結果が複数のリファレンスにならないことを確認してください。

たとえば、クラスと優先度など、2つのアラーム レコード フィールドが選択されると、1 つ以上のアラームが条件に一致する可能性が高くなります。このような場合、Hot Backup の同期化が機能する保証はありません。確認を反映させるプロセスにおいて、条件に一致するランダムなアラームまでが確認される一方で、他の一致するアラームが未確認のままになることがあります。

Hot Backup ペアについての注記

- Hot Backup は、InTouch 7.11 以降のクライアントでのみ使用できます。
- 分散アラーム表示オブジェクトで Hot Backup ペアをクエリーして、その後で再びプライマリ プロバイダを別にクエリーすると、分散アラーム表示オブジェクトに重複したレコードが表示されます。
- 1 つのプロバイダを、1 つ以上の Hot Backup ペアのプライマリ プロバイダまたはセカンダリ プロバイダとして設定しないでください。
- プライマリ プロバイダのレコードが確認され、(確認が行われたときにダウン状態であった) セカンダリ プロバイダを後で起動した場合、確認されたレコードのタイムスタンプはプライマリ プロバイダのレコードと同一です。
- Hot Backup ペアをクエリーしているアラーム コンシューマでは、個別のノード名がプロバイダとして表示されます。
- マッピング用に、デザイン時とランタイムのアラーム レコード フィールドの任意の組み合わせを選択できます。ただし、マッピングの結果が複数の参照にならないことを確認してください。

- [値] と [しきい値] の各キー フィールドをマッピングするとき、これらの値は小数点第 4 位までに四捨五入され、マップされます。
- デザイン時とランタイムのマッピングの特定の組み合わせを持たないアラーム レコードでは、デフォルトのランタイム マッピングが使用されます。

アラーム監査記録の作成

InTouch アラーム プロバイダがオペレーティング システムまたは ArchestrA 認証のいずれかを使用する場合にアラームが発生すると、アラーム表示の [ユーザー フルネーム] カラムには、ユーザーがログオンしていると想定して、ユーザーのフルネームが含まれます。

たとえば、ユーザー ID が JohnS、フルネームが John Smith というユーザーが PLANT_FLOOR ドメインに登録されている場合、[ユーザー フルネーム] カラムには John Smith が含まれます。アラームが後で確認され、確認を実行するノードがオペレーティング システムまたは ArchestrA セキュリティを使用する場合、[ユーザー フルネーム] カラムは更新され、確認ユーザーのフルネームが表示されます。それ以外の場合、アラーム表示にはコンピュータ名と \$Operator タグの内容が連結されて表示されます。

InTouch セキュリティには、オペレータのフルネームとアラーム確認を含めることができます。これは、アラーム検出に関するレコードでも可能です。ほとんどの組織では、ログイン ID はユーザーのフルネームではなく、略語や役割分類です。

プロバイダやコンシューマ InTouch ノードに対してオペレーティング システム認証を設定する場合：

- アラームが生成され、確認が実行されると、アラーム表示でフルネームが表示されます。
- アラームが生成され、確認が実行されると、アラームプリンタでフルネームが印刷されます。
- Alarm DB Logger では、各アラーム レコードの [Operator] フィールドと [AckOperator] フィールドにドメイン名、ログオンユーザー ID、およびフルユーザー名が記録されます。これにより、組織内に同じフルネームの従業員が 2 人存在する場合でも、個別に識別できます。
- [ユーザー] フィールドには、DomainName\UserName の形式でユーザー アカウントが表示されます。

認証を要求するアラームの確認

一部の業種では、特定のタイプのアラームを確認する際に、明示的な「署名」またはユーザー認証を要求します。InTouch は、このタイプの認証操作を行うために、アラーム クライアント コントロールまたはスクリプト機能を利用します。アラームを確認するのに特定のユーザー署名を要求するかどうかを決定する主要な要素は、アラームの優先度です。

アラームは、ユーザーが確認するのに認証を要求するように設定することができます。認証を要求するようにアラームを設定するとは、ランタイム時に、アラームを確認するためユーザーが自分の資格情報を入力する必要があることを意味しています。システムがスマートカードリーダーを使用するように設定されている場合には、ユーザーはスマートカードの資格情報を入力することができます。

このセクションでは、認証を要求する WindowViewer で、アラームを確認する方法について説明します。

アラームの確認に認証を要求するようアラーム クライアントを設定する方法の詳細については、アラーム クライアント コントロールのガイドを参照してください。

SignedAlarmAck() 関数を使用して、アラームを確認するための認証の動作を設定する方法の詳細については、『産業用グラフィック エディタ ユーザー ガイド』の「グラフィック スクリプトの追加と維持」の章を参照してください。

認証を要求する WindowViewer のアラームの確認について

認証を要求するよう設定されたアラームの確認には、次の事柄が必要です。

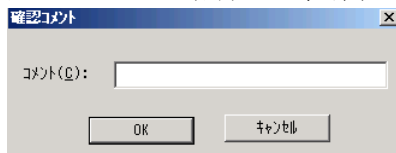
- Galaxy に対してセキュリティが有効にされている必要があります。
- 配置された InTouchViewApp に対してセキュリティが有効にされている必要があります。
- InTouchViewApp は、少なくとも 1 つ、アラームとイベントを表示し、アラームの確認のために署名（認証）を要求するよう設定され、アラームの優先度範囲の最小値と最大値が設定されているアラーム コントロールを含んでいる必要があります。
- ランタイム オペレータは Galaxy 内で設定された適切なユーザー名、パスワード、および権限を持っている必要があります。オペレータは、WindowViewer を実行するのに十分な権限を持っている必要があります。

現在誰が InTouch アプリケーションのランタイム ユーザーとしてログインしているかには関わりなく、アラームの確認のために署名を要求するよう設定されたアラームでは、常にユーザー認証が必要とされます。この操作はログオンしているユーザーのセッションには影響を与えません。

アラーム確認のランタイム時の基本的な動作

オペレータが 1 つか複数のアラームを選択して、それらを確認しようと試みます。

- アラーム クライアントは、アラームが確認署名を要求するよう設定されているかどうかをチェックします。設定されていた場合には、選択されたアラームのうち、設定された優先度範囲の中に入っているものがあるかどうかをチェックします。入っていたものがある場合には、選択されたアラームすべてを確認するための署名を要求します。
- 署名が要求されていない場合には、ポップアップの確認ダイアログが表示されます。そのように設定されていた場合には、確認コメント用のテキスト ボックスが表示されます。

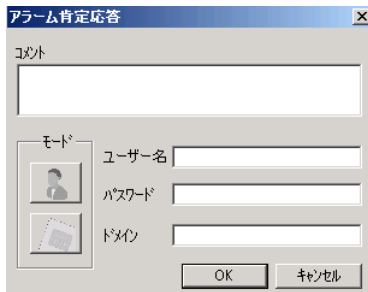


- オペレータのアカウント情報が有効であれば、アラーム クライアントは、選択されたアラームすべてを確認しようと試みます。
- アラーム クライアント グリッドでは、証明を行ったユーザーは、選択されたアラームすべてを確認した人物として識別されます。
- アラーム クライアントは、それが署名された確認であることを示すため、確認コメントの先頭に「署名された確認」というプレフィックスを付けます。
- 選択されたアラームのうち、その優先度が要求されている範囲に入っていないものがなかった場合には、アラーム クライアントは、それが標準の確認であることを示すため、確認コメントの先頭に「標準の確認」というプレフィックスを付けます。

認証を要求するアラームを確認するには

1. WindowViewer で、1 つまたは複数の未確認のアラームを選択します。右クリックして、実行する確認操作を選択します。いずれかのアラームが署名を要求するよう設定されていた場合には、**[確認**

アラーム] ダイアログ ボックスが表示されます。

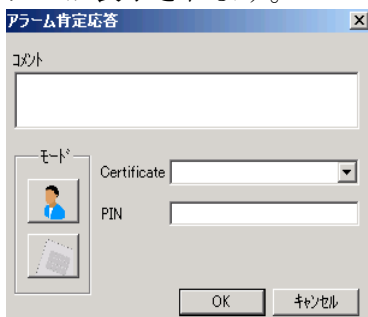


スマート カードが有効となった場合には、スマート カードまたはパスワード認証の一方を選択するための **【モード】** ボタンは無効になります。

2. コメントを入力するように設定されていた場合には、**【コメント】** ボックスにコメントを入力します。
3. ネットワーク ユーザー アカウントを使用して認証する場合には、ユーザー アカウントのオプションが表示されます。

以下の操作を行います。

- a. **【ユーザー名】** ボックスに、ユーザー名を入力します。デフォルトでは現在ログオンしているユーザーの名前が表示されます。現在ログオンしているユーザーがない場合、空白のボックスが表示されます。
 - b. **【パスワード】** ボックスに、ユーザー名に関連付けられているパスワードを入力します。
 - c. **【ドメイン】** ボックスに、ドメイン名を入力します。セキュリティ モードが **ArchestrA Galaxy** になっている場合には、表示されるドメイン名は **ArchestrA** になります。変更することはできません。
 - d. **【OK】** をクリックします。
4. スマート カードを使用して認証する場合には、スマート カードの **【確認アラーム】** ダイアログ ボックスが表示されます。



スマート カードを使用して認証するには、以下の操作を行います。

- a. **【証明書】** リストで、自分のスマート カード証明書を選択します。証明書のリストは、ドメイン名／ユーザー名の形式で表示されます。証明書のリストが表示されるには、その時点でコンピュータに接続されているリーダーにスマート カードが挿入されている必要があります。デフォルトでは現在ログオンしているユーザーの証明書が表示されます。**【セキュリティ書き込み】** ダイアログ ボックスが開いているときにカードの抜き差しを行った場合、証明書のリストは自動的に更新されます。

b. [PIN] ボックスに、スマート カードの PIN を入力します。

c. [OK] をクリックします。

代わりに名前、パスワード、およびドメイン名で認証する場合には、[モード] ボタンをクリックします。手順 3 に移動します。

分散アラーム表示オブジェクトの操作

InTouch HMI のこのバージョンには、分散アラーム表示オブジェクトが含まれており、InTouch バージョン 7 以前で開発されたアプリケーションをサポートします。分散アラーム表示オブジェクトは従来のオブジェクトなので、InTouch HMI のバージョン 7 より新しいバージョンでアラーム表示を行う場合は、この代わりに Alarm Viewer コントロールを使用する必要があります。

InTouch HMI のこのバージョンには、分散アラーム表示オブジェクトが含まれており、InTouch バージョン 7 以前で開発されたアプリケーションをサポートします。分散アラーム表示オブジェクトは従来のオブジェクトなので、InTouch HMI のバージョン 7 より新しいバージョンでアラーム表示を行う場合は、この代わりに Alarm Viewer コントロールを使用する必要があります。

分散アラーム表示オブジェクトの操作

InTouch HMI のこのバージョンには、分散アラーム表示オブジェクトが含まれており、InTouch バージョン 7 以前で開発されたアプリケーションをサポートします。分散アラーム表示オブジェクトは従来のオブジェクトなので、InTouch HMI のバージョン 7 より新しいバージョンでアラーム表示を行う場合は、この代わりに Alarm Viewer コントロールを使用する必要があります。

InTouch HMI のこのバージョンには、分散アラーム表示オブジェクトが含まれており、InTouch バージョン 7 以前で開発されたアプリケーションをサポートします。分散アラーム表示オブジェクトは従来のオブジェクトなので、InTouch HMI のバージョン 7 より新しいバージョンでアラーム表示を行う場合は、この代わりに Alarm Viewer コントロールを使用する必要があります。

分散アラーム表示オブジェクトについて

分散アラーム表示オブジェクトには、ローカルアラームとリモートアラームの両方を表示する単一表示の機能が提供されています。

Date	Time	State	Class	Type	Prio...	Name	Group	Provider
02/14	15:32	UNACK_RT...	VALUE	HI	1	ReactLevel	Reactor	InTouch
02/14	15:32	UNACK	VALUE	HI	1	ReactTemp	Reactor	InTouch
02/14	15:32	UNACK_RT...	VALUE	HI	1	ProdLevel	Reactor	InTouch

Update Successful Default Query

この表示オブジェクト機能には、組み込みスクロールバー、サイズ変更可能なカラム、複数のアラーム選択、ステータスバー、ショートカットメニュー、およびアラーム優先度と状況に基づく色が含まれています。

分散アラーム表示オブジェクトには、アラーム表示の外観（表示される情報を含む）、さまざまなアラーム条件に使用される色、表示するアラームグループとアラーム優先度レベルを設定できるプロパティが含まれています。

表示オブジェクトの詳細については、「[ランタイムでの分散アラーム表示オブジェクトの使用](#)」を参照してください。

分散アラーム表示オブジェクトのガイドライン

分散アラーム表示オブジェクトを使用するときは、以下のガイドラインを確認してください。

- 関連付けられている QuickScript 関数を変更する表示を識別できるように、各表示には識別子が必要です。この識別子は、[アラーム設定] ダイアログ ボックスの [表示名] ボックスに入力しますが、各表示に対して固有である必要があります。
- ウィンドウ コントロール オブジェクトやグラフィック オブジェクトなど、他の InTouch オブジェクトと表示が重ならないようにする必要があります。WindowMaker で分散アラーム表示オブジェクトをクリックし、表示の「ハンドル」を確認することによって、簡単に確認できます。ハンドルは画面の他のオブジェクトと接触できません。
- 表示は慎重に使用する必要があります。1つの画面に多数の表示を配置すると、システムのパフォーマンスが低下する可能性があります。可能な限り、ウィンドウの表示数を制限し、追加の表示に対しては新しいウィンドウを呼び出します。

デザイン時の分散アラーム表示オブジェクトの設定

以下を設定できます。

- ステータス バー、スクロール バーなどの一般機能
- カラムとソート順
- アラーム レコードを取得するために使用するアラーム クエリー
- アラーム レコードが表示される時間形式
- 表示されるアラーム レコードのフォントと色
- カラムのサイズ変更、アラームの選択、ショートカット メニューへのアクセスなどランタイム ユーザーが表示で実行できること

分散アラーム表示オブジェクトの作成

ウィザードの場合と同じように、分散アラーム表示オブジェクトを作成します。

分散アラーム表示オブジェクトを作成するには

1. [描画] メニューの [挿入] グループで [ウィザード] をクリックします。
[ウィザードの選択] ダイアログ ボックスが表示されます。
2. ウィザードのリストで、[アラームの表示] を選択します。
3. [分散] をダブルクリックします。アラーム表示 ウィザードをダブルクリックします。ダイアログ ボックスが閉じて、カーソルが「貼り付け」モードの状態のウィンドウが再表示されます。
4. 分散アラーム表示オブジェクトを貼り付ける位置をウィンドウ内でクリックします。ウィザードのサイズを変更するには、選択ハンドルをドラッグします。

これで、表示を設定する準備が整いました。

グリッドの表示プロパティの設定

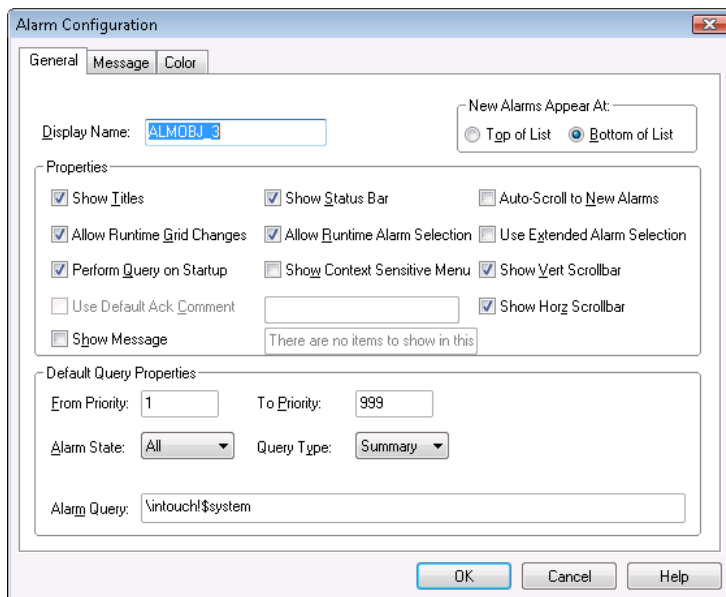
表示グリッドの場合は、以下の内容を設定できます。

- タイトル バー、ステータス バー、スクロール バー
- 新しいアラームがグリッドに表示される場所、自動スクロールの有無

- 表示するアラーム レコードがない場合に表示されるデフォルト メッセージ

グリッドの外観を設定するには

1. 分散アラーム表示オブジェクトを右クリックし、[プロパティ] をクリックします。[アラーム設定] ダイアログ ボックスが表示されます。



2. [表示名] ボックスに、アラーム表示の名前を入力します。この名前は、使用される各アラーム表示に対して固有である必要があります。この名前はアラーム確認やクエリーなど、このオブジェクトを参照するタスクの実行に対して、システムを通じて使用されます。
3. [新規アラームの表示箇所] 領域で、オブジェクト内で新しいアラームを表示する場所を設定します。
 - 最新のアラームをリストの一番上に表示するには、[リストの一番上] をクリックします。
 - 最新のアラームをリストの一番下に表示するには、[リストの一番下] をクリックします。
4. [プロパティ] 領域で、タイトル バー、ステータス バー、およびスクロール バーを設定します。以下のいずれかを実行します。
 - アラーム メッセージタイトル バーを表示するには、[タイトルの表示] チェック ボックスをオンにします。
 - ステータス バーを表示するには、[ステータス バーの表示] チェック ボックスをオンにします。
 - 垂直スクロール バーを表示するには、[垂直スクロールバーの表示] チェック ボックスをオンにします。
 - 水平スクロール バーを表示するには、[水平スクロールバーの表示] チェック ボックスをオンにします。
5. 表示するアラーム レコードがない場合、デフォルト メッセージを表示するには、[メッセージの表示] チェック ボックスをオンにします。ボックスにメッセージを入力します。
6. 選択を自動的に新規アラームに移動させるには、[新規アラームに自動スクロール] チェック ボックスをオンにします。新しいアラームは、表示オブジェクト内に現在表示されていないアラームです。

7. [OK] をクリックします。

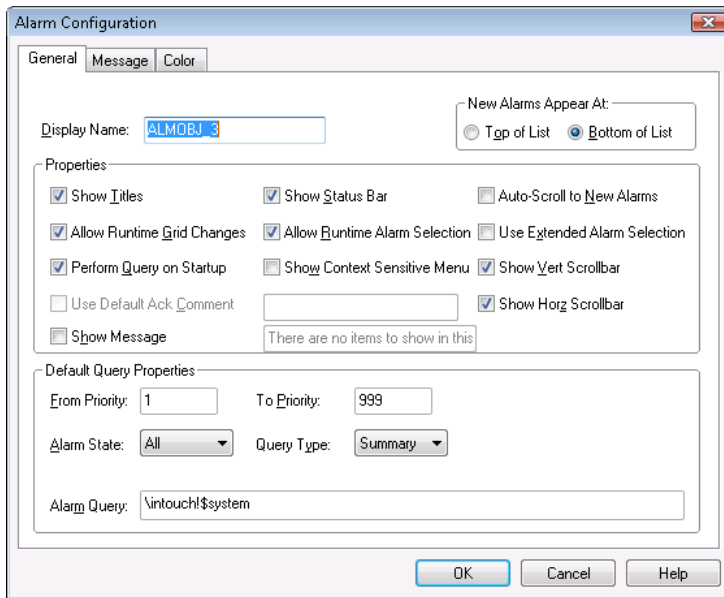
ランタイムでユーザーがアクセスできる機能の管理

ランタイム ユーザーが、カラム設定を変更し、アラームを選択し、ショートカットメニューを開けるようにできます。

注意： スクリプト関数を使用して、ショートカットメニューに表示されるコマンドを実行できます。

ランタイム機能を設定するには

1. 分散アラーム表示オブジェクトを右クリックし、[プロパティ] をクリックします。[アラーム設定] ダイアログ ボックスが表示されます。



2. ランタイムで利用できるオプションを設定します。以下のいずれかを実行します。
 - ユーザーがカラム設定を変更できるようにするには、[実行中のグリッド変更可能] チェック ボックスをオンにします。
 - ショートカットメニューを有効にするには、[コンテキストメニューを表示] チェック ボックスをオンにします。
 - ユーザーがアラームを選択できるようにするには、[実行中にアラームの選択可能] チェック ボックスをオンにします。
 - Ctrl キーまたは Shift キーを押さえたままマウスで選択することによって、ユーザーが複数のアラームを選択できるようにするには、[拡張アラーム選択の使用] チェック ボックスをオンにします。デフォルトでは、アラームを簡単にクリックして選択を切り替えます。

3. [OK] をクリックします。

表示するアラームの設定

分散アラーム表示オブジェクトを設定し、以下の内容に基づいてアラームを表示できます。

- 優先度
- 確認済み、未確認などの状況

- サマリまたは履歴などのタイプ

アラーム クエリーを設定するときは、テキストのみを使用します。タグ変数は使用できません。クエリーの例は以下のとおりです。

アラーム グループへのフルパス：

\\Node\\InTouch!Group

ローカル アラーム グループへのフルパス

\\InTouch!Group

別のグループ リスト：

GroupList

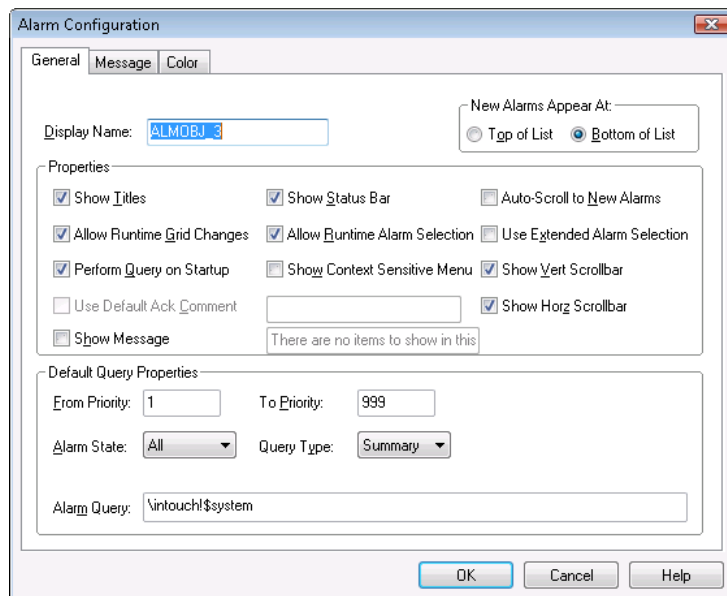
複数のクエリーを実行するには、各クエリーをスペースで区切ります。次に例を示します。

\\InTouch!Group GroupList

デフォルトのクエリー プロパティは、**[起動時にクエリー実行]** チェック ボックスをオンにした場合、または almDefQuery() 関数が実行される場合のみ使用できます。

表示するアラームを設定するには

1. 分散アラーム表示オブジェクトを右クリックし、**[プロパティ]** をクリックします。**[アラーム設定]** ダイアログ ボックスが表示されます。



2. **[起動時にクエリー実行]** チェック ボックスをオンにします。
3. **[クエリー属性のデフォルト]** 領域で、オブジェクトのデフォルト クエリーを設定します。
 - **[優先度始点]** ボックスに、デフォルトの最小アラーム優先度を入力します。
 - **[優先度始点]** ボックスに、デフォルトの最大アラーム優先度を入力します。
 - **[アラーム状況]** リストで、クエリーするデフォルトのアラーム状況をクリックします（全て、未確認、確認）。

- [クエリー タイプ] リストで、[サマリ] または [履歴] のいずれかのアラーム タイプをクリックします。
- [アラーム クエリー] ボックスに、最初のアラーム クエリーを入力します。

4. [OK] をクリックします。

デフォルトのアラーム コメントの設定

オペレータがアラームを確認するときに使用されるデフォルトのコメントを設定できます。デフォルトのコメントを設定しない場合、オペレータがアラームを確認するときに、オペレータがコメントを入力できるダイアログ ボックスが表示されます。このダイアログ ボックスはテキストを入力することも、空白のまま残すこともできます。

デフォルトのコメントを設定するには

1. 分散アラーム表示オブジェクトを右クリックし、[プロパティ] をクリックします。[アラーム設定] ダイアログ ボックスが表示されます。

Alarm Configuration

General Message Color

Display Name: ALMOBJ.3

New Alarms Appear At:
☐ Top of List ☒ Bottom of List

Properties

☒ Show Titles ☒ Show Status Bar ☐ Auto-Scroll to New Alarms
☒ Allow Runtime Grid Changes ☒ Allow Runtime Alarm Selection ☐ Use Extended Alarm Selection
☒ Perform Query on Startup ☐ Show Context Sensitive Menu ☒ Show Vert Scrollbar
☐ Use Default Ack Comment ☐ Show Message
☒ Show Horiz Scrollbar

Default Query Properties

From Priority: 1 To Priority: 999
Alarm State: All Query Type: Summary
Alarm Query: \intouch\system

OK Cancel Help

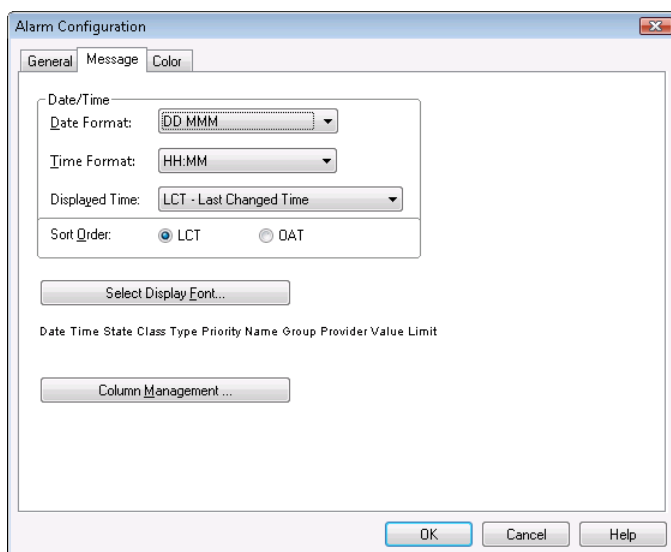
2. [コンテキスト メニューを表示] チェック ボックスをオンにします。
3. [デフォルトの確認コメントを使用] チェック ボックスをオンにし、ボックスにコメント テキストを入力します。
4. [OK] をクリックします。

アラーム レコードの時間形式の設定

オリジナルのアラーム時刻です。アラームが開始した日付／時間スタンプです。タグ変数が I/O 型タグ変数であり、そのサーバーがタイムスタンプを渡すことができる場合は、I/O Server からのタイムスタンプです。

アラーム表示時間形式を設定するには

1. 分散アラーム表示オブジェクトを右クリックし、[プロパティ] をクリックします。[アラーム設定] ダイアログ ボックスが表示されます。
2. [メッセージ] タブをクリックします。



3. [日付] リストで、日付の形式をクリックします。使用できる形式は以下のとおりです。

Selection	内容	Selection	内容
DD MMM	28 2	MM/DD	02/28
DD MMM YYYY	28 2 2007	MM/DD/YY	02/28/07
DD/MM	28/07	MMM DD	2 28
DD/MM/YY	28/02/07	MMM DD YYYY	2 28 2007
YY/MM/DD	07/02/28	YYYY/MM/DD	2007/02/28

4. [時間] リストで、時間の形式をクリックします。このリストの値は、時間の形式を指定するためのテンプレートとして使用します。たとえば、時間を 10:24:30 AM と指定するには、HH:MM:SS AP を使用します。テンプレートの文字は以下のとおりです。

文字	説明
AP	AM／PM 形式を選択します。たとえば、午後 3 時は 15:00 と表示されます。 AP が指定されていない場合、時間はデフォルトで 24 時間制で表示されます。たとえば、午後 3 時は 15:00 と表示されます。
HH	アラーム／イベントが発生した時間を示します。
MM	アラーム／イベントが発生した分を表示します。
SS	アラーム／イベントが発生した秒を表示します。
SSS	アラーム／イベントが発生したミリ秒を表示します。

5. [ソート順] 領域で、アラームをオブジェクトに保存する順序を設定します。

- **[OAT]** をクリックして、アラーム発生の日付／時間スタンプである元のアラーム時間を使用します。
- **LCT** をクリックして、最後のアラーム変更時間を使用します。これはアラームのインスタンスに対して発生した最新の状態変化（アラームの開始、サブステートの変化、平常への復帰、または確認）の日付／時間スタンプです。

6. **[OK]** をクリックします。

アラーム レコードのフォントの設定

コントロール内のレコードと見出しのフォントを設定します。

フォントのプロパティを設定するには

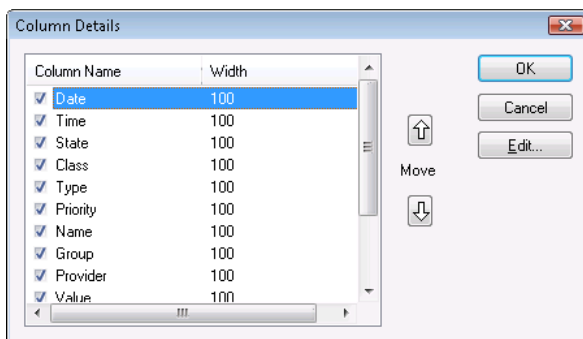
1. 分散アラーム表示オブジェクトを右クリックし、**[プロパティ]** をクリックします。**[アラーム設定]** ダイアログ ボックスが表示されます。
2. **[メッセージ]** タブをクリックします。
3. **[表示フォントの選択]** をクリックします。Windows 標準の **[フォント]** ダイアログ ボックスが表示されます。
4. フォントを設定して、**[OK]** をクリックします。

アラーム レコードのカラムの設定

表示するカラムを選択し、カラム順を指定し、カラムの名前と幅を設定できます。

表示カラムの詳細を設定するには

1. 分散アラーム表示オブジェクトを右クリックし、**[プロパティ]** をクリックします。**[アラーム設定]** ダイアログ ボックスが表示されます。
2. **[メッセージ]** タブをクリックします。
3. **[カラムの詳細]** をクリックします。**[カラムの詳細]** ダイアログ ボックスが表示されます。

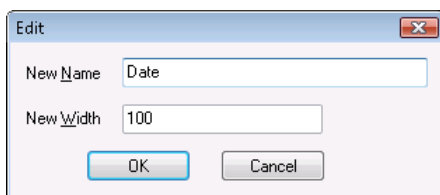


4. 分散アラーム表示オブジェクトのカラムを表示するには、**[カラム名]** リストのチェック ボックスをオンにします。少なくとも 1 つのカラムを選択する必要があります。以下の表は、カラムについて説明しています。

カラム	説明
日付	[メッセージ] タブで、選択した形式の日付を表示します。

カラム	説明
時刻	[メッセージ] タブで、選択した形式の時間を表示します。
状態	アラームの状況を表示します。
クラス	アラームのカテゴリを表示します。
タイプ	アラーム タイプを表示します。
優先度	アラーム優先度を表示します。
名前	アラーム／タグ変数を表示します。
グループ	アラームのグループ名を表示します。
プロバイダ	アラーム プロバイダ名を表示します。
値	アラームが発生した際のタグ変数値を表示します。
しきい値	タグ変数のアラームしきい値を表示します。
オペレータ	アラーム状況に関連付けられているログオンしたユーザーの ID を表示します。
コメント	タグ変数のコメントを表示します。このコメントは、データベースでタグ変数のアラームが定義されたときに [アラーム コメント] ボックスに入力されたものです。

5. カラムの順序を移動するには、カラム名を選択してから [移動] 矢印ボタンを使用してください。
[カラムの詳細] ダイアログ ボックスの一番上にあるカラム名が、アラーム表示で一番左に表示されます。
6. カラム名と幅を編集するには、カラム名を選択してから [編集] をクリックします。選択したカラムの [編集] ダイアログ ボックスが表示されます。



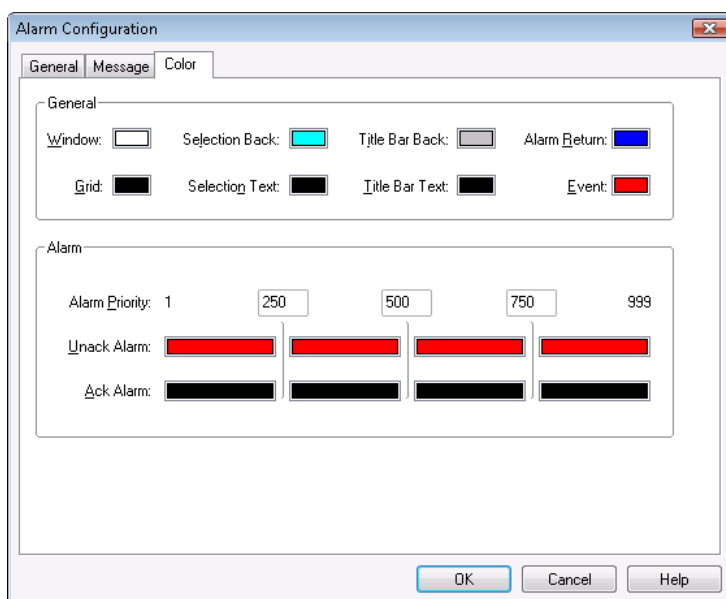
- a. デフォルトのカラム名以外にカラム名を表示するには、[新しい名前] ボックスに新しい名前を入力します。
 - b. [新しい幅] ボックスに、デフォルトのカラム幅を入力します。カラム幅は、1 ～ 999 ピクセルの範囲で設定できます。デフォルトのカラム幅は 100 ピクセルです。
 - c. [編集] ダイアログ ボックスの [OK] をクリックします。
7. [カラムの詳細] ダイアログ ボックスの [OK] をクリックします。
 8. [適用] をクリックします。

アラーム レコードの色の設定

背景色や選択したレコードの色など分散アラーム表示オブジェクトのさまざまな部分の色を設定できます。さまざまな範囲のアラーム レコードのさまざまな色も設定できます。

アラーム表示の色を設定するには

1. 分散アラーム表示オブジェクトを右クリックし、[プロパティ] をクリックします。[アラーム設定] ダイアログ ボックスが表示されます。
2. [色] タブをクリックします。



3. [表示項目] 領域で、それぞれの色のボックスをクリックし、InTouch カラー パレットを開きます。以下のそれぞれに使用する色をクリックします。

オプション	説明
ウィンドウ	表示背景色を設定します。
グリッド	グリッドの色を設定します。
背景の選択	反転表示されたテキスト背景色を設定します。
テキストの選択	反転表示されたテキストの色を設定します。
タイトルバーの背景	タイトルバーの背景色を設定します ([タイトルの表示] オプションをオンにした場合にのみ表示されます)。
タイトルバーのテキスト	タイトルバー テキストの色を設定します ([タイトルの表示] オプションをオンにした場合にのみ表示されます)。

オプション	説明
アラーム復帰	平常状態に復帰したアラーム（確認されずに平常に戻ったアラーム）の色を設定します。
イベント	イベントアラームの色を設定します。

4. [アラーム優先度] ボックスに、アラーム表示の変化境界値を入力します。
5. [未確認アラーム] と [確認アラーム] の色ボックスをクリックして、InTouch パレットを開きます。使用する色をパレットでクリックします。
6. [OK] をクリックします。

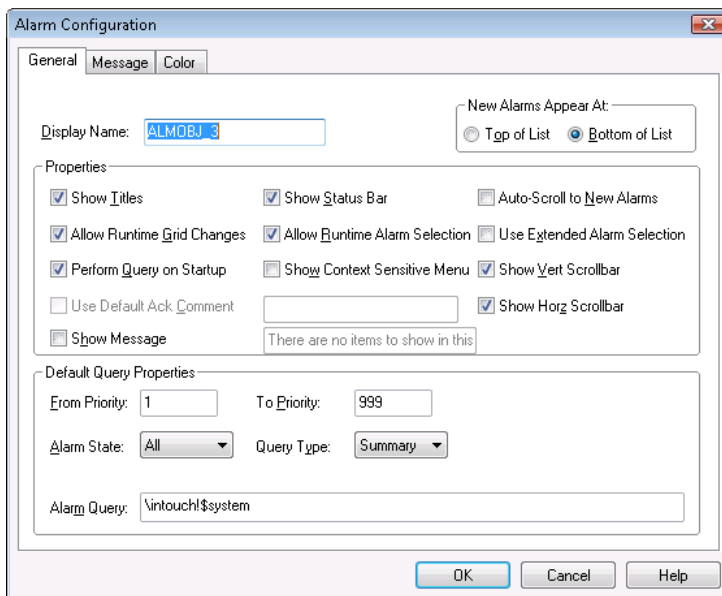
表示タイプの設定

分散アラーム表示オブジェクトには、アクティブなアラームのサマリまたは履歴アラームのリストを表示できます。

表示タイプはランタイムで動的に変更することもできます。これを行うには、たとえば、`almQuery()` スクリプト関数を実行します。`almQuery()` スクリプト関数では、指定した分散アラーム表示オブジェクト（たとえば、「AlmObj_1」）を指定の表示タイプ（たとえば、「サマリ」）に設定するパラメータが使用されます。詳細については、「[almQuery\(\) 関数](#)」を参照してください。

クエリーのデフォルトを設定するには

1. 分散アラーム表示オブジェクトを右クリックし、[プロパティ] をクリックします。[アラーム設定] ダイアログ ボックスが表示されます。



2. [クエリー タイプ] リストで、ランタイム デフォルトに使用するアラーム表示のタイプをクリックします。
3. [OK] をクリックします。

ローカルアラームを監視するための分散表示の使用

分散アラーム表示オブジェクトを使用すると、ローカルアラームとリモートアラームの両方の表示と確認を行うことができます。

ローカルアラームだけを監視するように表示を設定するには

1. 分散アラーム表示オブジェクトを右クリックし、[プロパティ] をクリックします。[アラーム設定] ダイアログ ボックスが表示されます。
2. [アラーム クエリー] ボックスに、以下を入力します。 `\InTouch!$System`
任意の有効なアラーム グループを `$System` の代わりに使用できます。 `\InTouch!$System` のみを含むアラーム グループ リストを定義し、直接リファレンスの代わりにこのグループ リストを使用することもできます。
3. 表示タイプの他のパラメータを設定し、アプリケーションが必要とするフィルタを設定します。

ランタイムでの分散アラーム表示オブジェクトの使用

分散アラーム表示オブジェクトでは、グリッドを使用して、アラーム メッセージを表示します。このグリッドを使用すると、カラム ハンドルを選択し、必要な幅にドラッグするだけで簡単にカラムの幅のサイズを変更できます。この機能はランタイム中でのみ使用できます。

グリッドのカラム変更は保存されないため、グリッドカラム変更を行い、アラーム表示が含まれているウィンドウを閉じると、再びそのウィンドウを開いたときにはグリッドカラムはデフォルトのカラム幅に戻ります。 **WindowMaker** でデフォルトのカラム幅を調整できます。

グリッドを使用すると、リスト ボックス内の 1 つまたは複数のアラームを選択できます。選択したアラームは、 `almAckSelect()` QuickScript 関数を使用して確認できます。分散アラーム表示オブジェクトを設定するときは、選択動作を定義して、切り替え選択（アイテムごと）または複数選択（Ctrl キーまたは Shift キーを押したまま、マウスクリックを使用して複数のアラームを選択）のいずれかを許可できます。ランタイム中の選択はオフに設定できます。

表示される各アラーム メッセージの色は、アラームの優先度や確認の状態に応じて、最大 8 色の中から選択できます。

設定時にオンにした場合は、スクロールバーを利用できます。

アラーム表示オブジェクトには、コントロールが設定される方法に基づいて、アラーム レコードを検索するコントロールがあります。

サイズ変更可能な表示カラム

分散アラーム表示オブジェクトはグリッドを使用して、アラーム メッセージを保持します。ランタイムで、カラムを選択し、ドラッグするだけで簡単にカラムの幅のサイズを動的に変更できます。また、縦のグリッド線をダブルクリックして、カラムのサイズを自動的に変更できます。この機能はランタイム中でのみ使用できます。カラムの幅のサイズ変更は、設定時にオンにする必要があります。

アラーム表示が含まれるウィンドウを閉じるとき、グリッドカラムの変更は保存されません。

複数の選択

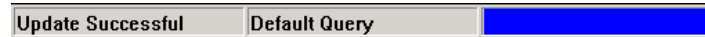
アラーム表示が設定される方法に基づいて、リスト ボックスの 1 つまたは複数のアラームを選択できます。

アラーム メッセージの色

アラーム メッセージの色は、アラームの優先度や確認の状態に応じて、最大 8 色の中から選択できます。

ステータス バー

アラーム オブジェクトが設定される方法に基づいて、ステータス バーには、ステータス メッセージ、現在のアラーム クエリー、および進行状況バーが表示されます。



ステータス バーの左部分には、コントロールの現在のステータスが表示されます。

これらのインジケータでは、表示クエリーの現在の状況の概要と分散アラーム表示オブジェクトで利用できる非表示機能についての詳細が提供されます。「停止」に設定されている場合、ステータス バーの右ペインが赤となります。「非表示」に設定されている場合、ステータス バーの左ペインが赤となります。

「非表示」に設定されている場合、「非表示中」という単語が左ペインに表示されます。

機能	説明
ステータス メッセージ	ステータス バーの左端にあるステータス メッセージには、現在のクエリー ステータスの詳細な説明が表示されます。
アラーム クエリー	アラーム クエリーでは、現在のアラーム クエリーが視覚的に表示されます。
進行状況バー	ステータス バーの右端にある進行状況バーには、現在のクエリーの進行状況が視覚的に表示されます。

ステータス メッセージは以下のとおりです。

ステータス メッセージ	状況／インジケータ	進行状況バー
なし	クエリーなし	なし
更新中	クエリー中	青／緑
更新が完了しました	クエリー終了	赤
非表示	クエリー名	青一色
停止	クエリー名	赤

ショートカット メニュー

アラーム オブジェクトが設定される方法に基づいて、オブジェクトを右クリックし、共通コマンドのショートカット メニューを開きます。

クリックするコマンド	目的
選択アラームを確認	選択したアラームを確認します。
その他を確認	表示内のすべてのアラームを確認するか、表示されているアラーム、選択したグループ、選択したタグ名、および優先度のみを確認します。
選択項目を非表示	選択したアラームを非表示にします。
その他を非表示	表示内のすべてのアラームを非表示にするか、表示されているアラーム、選択したグループ、選択したタグ名、および選択した優先度のみを非表示にします。
クエリー設定	[アラーム クエリー] ダイアログ ボックスを開き、使用するクエリーを選択できます。
統計	[アラームの統計] ダイアログ ボックスを開きます。
非表示	[アラーム非表示] ダイアログ ボックスを開きます。
停止	現在の表示を停止します。
選択項目を閉じる	選択したアラームを解除します。
その他を閉じる	表示内のすべてのアラームを解除するか、表示されているアラーム、選択したグループ、選択したタグ名、および選択した優先度のみを解除します。

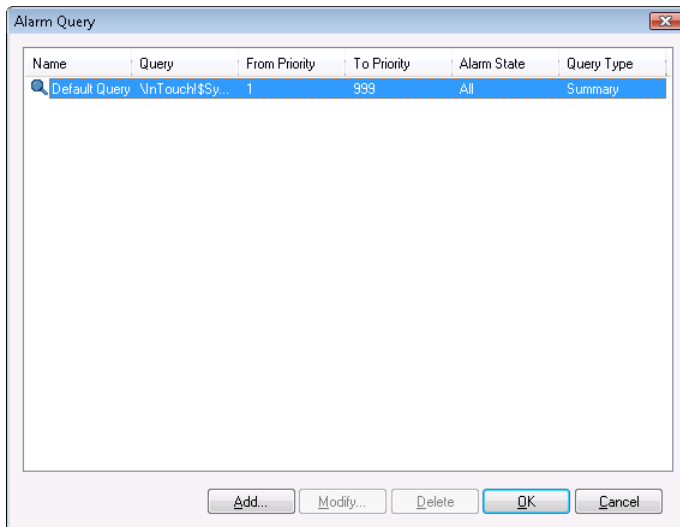
アラーム クエリーの選択と設定

ショートカット メニューの [クエリー設定] コマンドをクリックし、以前定義されたアラーム クエリーのリストからアラーム クエリーを素早く選択します。また、新しい名前付きクエリーを作成したり、既存のクエリーを編集したり、既存のクエリーを削除したりすることもできます。

注意: 分散アラーム表示に複数行アラーム クエリーが表示される場合、改行部分が文字化けします。ただし、機能には影響ありません。

表示用のアラーム クエリーを選択するには

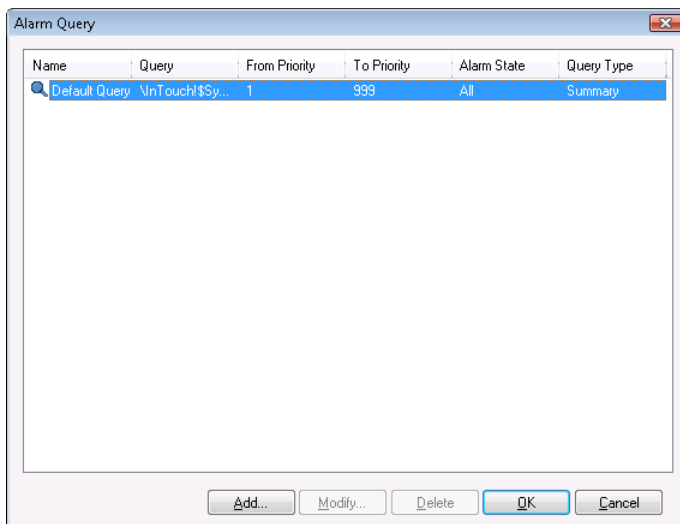
- ランタイムで、分散アラーム表示を右クリックし、[クエリー設定] をクリックします。[アラーム クエリー] ダイアログ ボックスが表示されます。



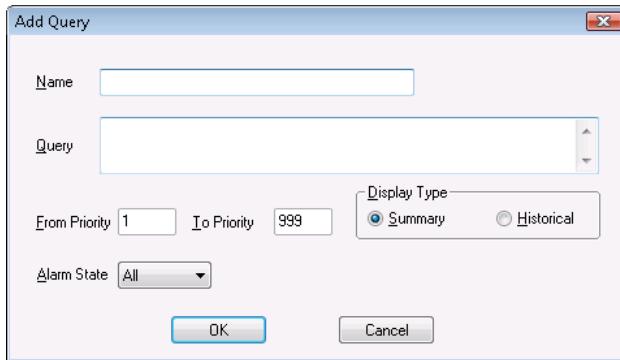
2. 現在定義されているクエリーのリストで、使用する名前付きクエリーを選択します。
3. [OK] をクリックします。分散アラーム表示オブジェクトには、選択したクエリーに関するアラーム情報が表示されます。

新しい名前付きクエリーを追加するには

1. ランタイムで、分散アラーム表示を右クリックし、[クエリー設定] をクリックします。[アラームクエリー] ダイアログ ボックスが表示されます。



2. [追加] をクリックします。[クエリーの追加] ダイアログ ボックスが表示されます。



3. クエリーを設定します。以下の操作を行います。
 - a. [名前] ボックスに、クエリーの名前を入力します。
 - b. [クエリー] ボックスに、実行する InTouch アラーム クエリーのセットを入力します。1 つまたは複数のアラーム プロバイダとグループを指定できます。
 - c. [優先度始点] ボックスに、最低アラーム優先度の値 (1 ～ 999) を入力します。[優先度終点] ボックスに、最高アラーム優先度の値 (1 ～ 999) を入力します。
 - d. [アラーム状況] リストで、アラーム クエリーに使用するアラーム状況をクリックします。
 - e. [表示タイプ] 領域で、表示するアラームのタイプをクリックします。詳細については、「[サマリアラームと履歴アラーム](#)」を参照してください。
4. [OK] をクリックして、[クエリーの追加] ダイアログ ボックスを閉じます。
5. [アラーム クエリー] ダイアログ ボックスで、[OK] をクリックします。

既存の名前付きクエリーを変更するには

1. ランタイムで、分散アラーム表示を右クリックし、[クエリー設定] をクリックします。[アラーム クエリー] ダイアログ ボックスが表示されます。
2. 現在定義されているクエリーのリストで、変更する名前付きクエリーを選択します。
3. [修正] をクリックします。[クエリーの修正] ダイアログ ボックスが表示されます。
4. 必要な修正を行ってから [OK] をクリックして、[クエリーの修正] ダイアログ ボックスを閉じます。
5. [アラーム クエリー] ダイアログ ボックスの [OK] をクリックします。

注意：変更されているアラーム クエリーを使用しているその他の分散アラーム表示オブジェクトには、変更は自動的に適用されません。

既存の名前付きクエリーを削除するには

1. ランタイムで、分散アラーム表示を右クリックし、[クエリー設定] をクリックします。[アラーム クエリー] ダイアログ ボックスが表示されます。
2. 現在定義されているクエリーのリストで、変更する名前付きクエリーを選択します。
3. [削除] をクリックします。メッセージが表示されたら、[はい] をクリックします。
4. [アラーム クエリー] ダイアログ ボックスの [OK] をクリックします。

注意：削除されているアラーム クエリーを使用しているその他の分散アラーム表示オブジェクトには、削除は自動的に適用されません。

関数とドットフィールドを使用した分散アラーム表示オブジェクトの制御

関数とドットフィールドを使用して、ランタイムで分散アラーム表示オブジェクトを制御できます。

返される関数のエラー番号のリストについては、「[エラーの説明](#)」を参照してください。

プロパティの取得と設定

プロパティには、`GetPropertyX()` 関数を使用してアクセスできます。ここで、**X** はデータ タイプ（論理型を表す **D**、整数型を表す **I**、メッセージ型を表す **M**）です。次に例を示します。

```
GetPropertyM(ControlName.Property, MsgTag)
```

`GetPropertyX()` 関数の詳細については、「[組み込み関数](#)」を参照してください。

`GetPropertyX()` 関数が含まれるスクリプトを実行するとき、プロパティ値は `MsgTag` に保存されます。複数の行が選択されている場合、`MsgTag` に割り当てられるプロパティは、複数選択の最初の行にあるタグ変数値です。

アラームの確認

分散アラーム表示オブジェクトは、クエリーできる任意のアラームを確認できます（サマリ表示のみ）。分散アラーム表示オブジェクトには、アラーム確認関数が含まれます。これらの関数は、ローカルアラームとアラーム グループを確認するために使用される **.Ack** ドットフィールドを補足します。これらの関数を使用すると、すべてのアラーム、表示されるアラーム、および選択したアラームを確認できます。

グループ メンバシップ、優先度、アプリケーション名、およびタグ変数名のような特性によって、アラームを確認することもできます。

- [almAckAll\(\) 関数](#)
- [関数とドットフィールドを使用した分散アラーム表示オブジェクトの制御](#)
- [almAckGroup\(\) 関数](#)
- [almAckPriority\(\) 関数](#)
- [almAckRecent\(\) 関数](#)
- [almAckTag\(\) 関数](#)
- [almAckSelect\(\) 関数](#)
- [almAckSelectedGroup\(\) 関数](#)
- [almAckSelectedPriority\(\) 関数](#)
- [almAckSelectedTag\(\) 関数](#)

almAckAll() 関数

現在のクエリー内のすべてのアラームを確認します。サマリ モードで分散アラーム表示オブジェクトに現在表示されていないアラームも含まれます。

カテゴリ

アラーム

構文

```
[Result=]almAckAll(ObjectName,Comment);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

Comment

アラーム確認コメントです。

例

```
MessageTag = "Acknowledge All by " + $Operator;  
almAckAll("AlmObj_1",MessageTag);
```

参照項目

Ack()、almAckGroup()、almAckTag()、almAckDisplay()、almAckRecent()、almAckSelect()、almAckSelectedGroup()、almAckSelectedPriority()、almAckSelectedTag()

almAckDisplay() 関数

サマリ モードで分散アラーム表示オブジェクトに現在表示されているアラームのみを確認します。

カテゴリ

アラーム

構文

```
[Result=]almAckDisplay(ObjectName,Comment);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

Comment

アラーム確認コメントです。

例

```
almAckDisplay("AlmObj_1","Display Acknowledgement");
```

参照項目

Ack()、almAckAll()、almAckGroup()、almAckTag()、almAckRecent()、almAckSelect()、almAckSelectedGroup()、almAckSelectedPriority()、almAckSelectedTag()

almAckGroup() 関数

指定したプロバイダとグループ名に一致する、指定されている分散アラームオブジェクトに表示されるすべてのアラームを確認します。

カテゴリ

アラーム

構文

```
[Result=]almAckGroup( "ObjectName", ApplicationName, GroupName, Comment);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

ApplicationName

たとえば、\\node1\Intouch などアプリケーション名です。

GroupName

たとえば、\$System など InTouch アラーム グループの名前です。

Comment

アラーム確認コメントです。

例

```
MessageTag = "Acknowledge group, Turbines, by " + $Operator;  
almAckGroup("AlmObj_1", "\\Intouch", "Turbine", MessageTag);
```

almAckPriority() 関数

アラームのアプリケーション名、アラーム グループ、および優先度範囲に一致する最後のクエリーの結果として、指定されている分散アラームオブジェクトに表示されるすべてのアラームを確認します。

カテゴリ

アラーム

構文

```
[Result=]almAckPriority(ObjectName, ApplicationName, GroupName, FromPri, ToPri, Comment);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

ApplicationName

たとえば、\\node1\Intouch などアプリケーション名です。

GroupName

たとえば、\$System などグループの名前です。

FromPri

アラーム優先度範囲の開始番号です。たとえば、100 です。

ToPri

アラーム優先度範囲の終了番号です。たとえば、900 です。

Comment

アラーム確認コメントです。

例

```
almAckPriority("AlmObj_1", "\\node1\Intouch", "Turbines", 10, 100, "Range 10 to 100  
acknowledged");
```

almAckRecent() 関数

最後に発生したアラームを確認します。

構文

```
[Result=]almAckRecent(ObjectName, Comment)
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

Comment

アラーム確認コメントです。

例

```
almAckRecent("AlmObj_1", $DateString);
```

almAckTag() 関数

最後のクエリーの結果として、指定されている分散アラームオブジェクトに表示されるすべてのアラームを確認します。アラームは、アプリケーション名、グループ名、タグ変数名、およびクエリーで指定された優先度範囲に一致する必要があります。

カテゴリ

アラーム

構文

```
[Result=]almAckTag("ObjectName", ApplicationName, GroupName, TagName, FromPri, ToPri, Comment);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

ApplicationName

アプリケーションの名前です。たとえば、\\node1\Intouch です。

GroupName

アラーム グループの名前です。たとえば、\$System です。

TagName

値がアラーム状況にあるタグ変数の名前です。

FromPri

アラーム優先度範囲の開始番号です。たとえば、100 です。

ToPri

アラーム優先度範囲の終了番号です。たとえば、900 です。

Comment

アラーム確認コメントです。

例

```
almAckTag("AlmObj_1", "\\node1\Intouch", "Turbines", "Valve1", 10, 100, "Acknowledged for Valve1");
```

参照項目

Ack()、almAckAll()、almAckGroup()、almAckDisplay()、almAckRecent()、almAckSelect()、almAckSelectedGroup()、almAckSelectedPriority()、almAckSelectedTag()

almAckSelect() 関数

サマリ モードで分散アラーム表示オブジェクトで選択したアラームのみを確認します。

カテゴリ

アラーム

構文

```
[Result=]almAckSelect(ObjectName,Comment);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

Comment

アラーム確認コメントです。

例

この例では、昼間勤務または夜間勤務の間に発生したアラームのみを確認します。

```
IF ($Hour >= 0 and $Hour < 8) THEN
    AckTag = "Night Shift";
ELSE
    AckTag = "Day Shift";
ENDIF;
almAckSelect ("AlmObj_1",AckTag);
```

参照項目

Ack()、almAckAll()、almAckGroup()、almAckTag()、almAckDisplay()、almAckRecent()、
almAckSelectedGroup()、almAckSelectedPriority()、almAckSelectedTag()

almAckSelectedGroup() 関数

指定された分散アラーム表示オブジェクト内で選択される 1 つまたは複数のアラームの同じグループ名を持つ同じプロバイダ名とグループ名のすべてのアラームを確認します。

カテゴリ

アラーム

構文

```
[Result=]almAckSelectedGroup(ObjectName,Comment);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

Comment

アラーム確認コメントです。

例

```
MessageTag = "Acknowledge selected groups by " + $Operator;
almAckSelectedGroup ("AlmObj_1", MessageTag);
```

参照項目

Ack()、almAckAll()、almAckGroup()、almAckTag()、almAckDisplay()、almAckRecent()、almAckSelect()、
almAckSelectedPriority()、almAckSelectedTag()

almAckSelectedPriority() 関数

指定された分散アラーム表示オブジェクト内で選択される 1 つまたは複数のアラームの同じ優先度値を持つ同じプロバイダ名とグループ名のすべてのアラームを確認します。優先度は、選択したアラームレコードの最低優先度および最大優先度から計算されます。

カテゴリ

アラーム

構文

```
[Result=]almAckSelectedPriority(ObjectName, Comment);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

Comment

アラーム確認コメントです。

例

```
MessageTag = "Acknowledge selected priorities by " + $Operator;  
almAckSelectedPriority ("AlmObj_1", MessageTag);
```

参照項目

Ack()、almAckAll()、almAckGroup()、almAckTag()、almAckDisplay()、almAckRecent()、almAckSelect()、almAckSelectedGroup()、almAckSelectedTag()

almAckSelectedTag() 関数

指定された分散アラーム表示オブジェクト内の 1 つまたは複数の選択したアラームと同じ優先度を持つ同じプロバイダ名およびグループ名からの同じタグ変数を持つすべてのアラームを確認します。この関数は、InTouch がアラーム プロバイダの場合にのみ機能します。

カテゴリ

アラーム

構文

```
[Result=]almAckSelectedTag(ObjectName, Comment);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

Comment

アラーム確認コメントです。

例

```
MessageTag = "Acknowledge selected tagnames by " + $Operator;  
almAckSelectedTag ("AlmObj_1", MessageTag);
```

参照項目

Ack()、almAckAll()、almAckGroup()、almAckTag()、almAckDisplay()、almAckRecent()、almAckSelect()、almAckSelectedGroup()、almAckSelectedPriority()

アラームの選択

分散アラーム表示オブジェクトからアラームを選択するスクリプトを作成できます。すべてのアラームを選択することも、選択したアラームのみを選択することも、現在のアラームのカウンタを取得することもできます。

- [almSelectAll\(\) 関数](#)
- [almUnselectAll\(\) 関数](#)
- [almSelectionCount\(\) 関数](#)
- [almSelectGroup\(\) 関数](#)
- [almSelectItem\(\) 関数](#)
- [almSelectPriority\(\) 関数](#)
- [almSelectTag\(\) 関数](#)

データ ソース、アラーム優先度、および InTouch タグ変数に基づいて、特定のアラームを選択することもできます。

almSelectAll() 関数

指定された分散アラーム表示オブジェクト内のすべてのアラームの選択を切り替えます。

カテゴリ

アラーム

構文

```
[Result=]almSelectAll(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

例

```
If $AccessLevel > 8000 THEN  
    almSelectAll("AlmObj_1");  
    almAckSelect("AlmObj_1", "Ack Selected by a Manager");  
ENDIF;
```

参照項目

almSelectItem()、almSelectGroup()、almSelectPriority()、almSelectTag()、almUnselectAll()

almUnselectAll() 関数

指定された分散アラーム表示オブジェクト内のすべての選択済みアラームを選択解除します。

カテゴリ

アラーム

構文

```
[Result=]almUnselectAll(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

例

```
If $AccessLevel == 9999 THEN
    almAckSelect("AlmObj_1", "Comment");{This alarm can be acknowledged by only
    Administrator}
ELSE
    almUnselectAll("AlmObj_1");
ENDIF;
```

参照項目

almSelectAll()、almSelectItem()、almSelectGroup()、almSelectPriority()、almSelectTag()

almSelectionCount() 関数

分散アラーム表示オブジェクトでオペレータによって選択されたアラーム数を返します。

カテゴリ

アラーム

構文

```
[Result=]almSelectionCount(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

例

AlarmCount タグ変数には、分散アラーム表示オブジェクトからオペレータによって選択されたアラーム数が割り当てられます。

```
AlarmCount = almSelectionCount("AlmObj_1");
```

almSelectGroup() 関数

表示の最後のクエリーの結果として、結果アラームに同じアラーム グループ名が含まれる、指定された分散アラーム表示オブジェクトによって含まれるすべてのアラームの選択を切り替えます。

カテゴリ

アラーム

構文

```
[Result=]almSelectGroup(ObjectName, ApplicationName,GroupName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

ApplicationName

アプリケーションの名前です。たとえば、\\node1\Intouch です。

GroupName

グループの名前です。たとえば、\$System です。

例

```
almSelectGroup("AlmObj_1","\\Intouch","Turbine");
```

参照項目

almSelectAll()、almSelectItem()、almSelectPriority()、almSelectTag()、almUnSelectAll()

almSelectItem() 関数

アラーム表示オブジェクト内で最後に選択または選択解除されたアイテムの選択を切り替えます。

構文

```
[Result=]almSelectItem(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

例

```
almSelectItem("AlmObj_1");
```

参照項目

almSelectAll()、almSelectGroup()、almSelectPriority()、almSelectTag()、almUnSelectAll()

almSelectPriority() 関数

表示の最後のクエリーの結果として、結果アラームが指定した優先度範囲内である、指定された分散アラーム表示オブジェクトのすべてのアラームの選択を切り替えます。

カテゴリ

アラーム

構文

```
[Result=]almSelectPriority( "ObjectName", ApplicationName, GroupName, FromPri, ToPri );
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

ApplicationName

アプリケーションの名前です。たとえば、\\node1\Intouch です。

GroupName

グループの名前です。たとえば、\$System です。

FromPri

アラームの開始優先度です。たとえば、100 または整数型タグ変数です。

ToPri

アラームの終点優先度です。たとえば、900 または整数型タグ変数です。

例

```
almSelectPriority("AlmObj_1","\\node1\Intouch", "Turbines",10,100);
```

参照項目

almSelectAll()、almSelectItem()、almSelectGroup()、almSelectTag()、almUnSelectAll()

almSelectTag() 関数

表示の最後のクエリーおよび指定されたタグ変数の結果として、指定された分散アラーム表示オブジェクトのすべてのアラームの選択を切り替えます。

カテゴリ

アラーム

構文

```
[Result=]almSelectTag (ObjectName, ApplicationName, GroupName, TagName, FromPri, ToPri);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

ApplicationName

アプリケーションの名前です。たとえば、\\node1\Intouch です。

GroupName

グループの名前です。たとえば、\$System です。

TagName

アラーム タグ変数の名前です。

FromPri

アラームの開始優先度です。たとえば、100 または整数型タグ変数です。

ToPri

アラームの終点優先度です。たとえば、900 または整数型タグ変数です。

例

```
almSelectTag("AlmObj_1","\\node1\Intouch","Turbines","Valve1",10,100);
```

参照項目

almSelectAll()、almSelectItem()、almSelectGroup()、almSelectPriority()、almUnSelectAll()

選択したアラーム レコードに関する情報の取得

選択したアラームに関する情報を返すスクリプトを作成できます。スクリプトで以下のドットフィールドおよび関数を使用します。

- [.AlarmTime](#) ドットフィールド
- [.AlarmDate](#) ドットフィールド
- [.AlarmName](#) ドットフィールド
- [.AlarmValue](#) ドットフィールド
- [.AlarmClass](#) ドットフィールド
- [.AlarmType](#) ドットフィールド
- [.AlarmState](#) ドットフィールド
- [.AlarmLimit](#) ドットフィールド
- [.AlarmPri](#) ドットフィールド

- [.AlarmGroupSel](#) ドットフィールド
- [.AlarmAccess](#) ドットフィールド
- [.AlarmProv](#) ドットフィールド
- [.AlarmOprName](#) ドットフィールド
- [.AlarmOprNode](#) ドットフィールド
- [.AlarmComment](#) ドットフィールド

.AlarmTime ドットフィールド

アラームが発生した時間を返します。アラームは、サマリ モードで分散アラーム表示オブジェクトで選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM( "ObjectName.AlarmTime",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

この例では、AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almTime** はメモリ メッセージ型タグ変数です。

```
GetPropertyM("AlmObj_1.AlarmTime",almTime);
```

押しボタン QuickScript で使用した場合、このステートメントはアラームの発生時刻を **almTime** タグ変数に返します。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmName、.AlarmOprName、.AlarmOprNode、.AlarmPri、.AlarmProv、.AlarmState、.AlarmType、.AlarmValue

.AlarmDate ドットフィールド

選択したアラームに関連付けられている日付を返します。サマリ モードで分散アラーム表示オブジェクトをクリックして、アラームを選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM("ObjectName.AlarmDate",TagName);
```


パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

押しボタン QuickScript で使用した場合、このステートメントは日付を **almDate** タグ変数に返します。

```
GetPropertyM("AlmObj_1.AlarmDate",almDate);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almDate** は選択したアラームに関連付けられているタグ変数の日付を取得するメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmComment、.AlarmLimit、.AlarmName、.AlarmOprName、.AlarmOprNode、.AlarmPri、.AlarmProv、.AlarmState、.AlarmTime、.AlarmType、.AlarmValue

.AlarmName ドットフィールド

選択したアラームに関連付けられているタグ変数の名前を返します。サマリ モードで分散アラーム表示オブジェクトをクリックして、アラームを選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM("ObjectName.AlarmName",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データ タイプ

文字列（読み取り専用）

例

押しボタン QuickScript で使用した場合、このステートメントはアラームの名前を **almName** に返します。

```
GetPropertyM("AlmObj_1.AlarmName",almName);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almName** は選択したアラームに関連付けられているタグ変数名を受け取るメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmOprName、.AlarmOprNode、.AlarmPri、.AlarmProv、.AlarmState、.AlarmTime、.AlarmType、.AlarmValue

.AlarmValue ドットフィールド

選択したアラームに関連付けられているタグ変数のアラーム値を返します。サマリ モードで分散アラーム表示オブジェクトをクリックして、アラームを選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM("ObjectName.AlarmValue,TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

備考

この関数は、メッセージ型タグ変数を使用して、数値を取得します。これは、GetProperty 関数が実数をサポートしていないためです。StringToReal() 関数を使用して、結果を実数型タグ変数に割り当てることができます。

例

押しボタン QuickScript で使用した場合、このステートメントは値を **almValue** に返します。

```
GetPropertyM("AlmObj_1.AlarmValue", almValue);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almValue** は選択したアラームに関連付けられるタグ変数のアラーム値を含むメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmOprName、.AlarmOprNode、.AlarmPri、.AlarmProv、.AlarmState、.AlarmTime、.AlarmType、.AlarmValue

.AlarmClass ドットフィールド

選択したアラームに関連付けられるタグ変数のアラーム クラスを返します。アラームは、サマリ モードで分散アラーム表示オブジェクトから選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM("ObjectName.AlarmClass",Tagname);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

以下のステートメントは、選択したアラームに関連付けられているアラーム クラスを返します。

```
GetPropertyM("AlmObj_1.AlarmClass",almClass);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almClass** は選択したアラームに関連付けられるタグ変数のアラーム クラスを含むメモリ メッセージ型タグ変数です。

押しボタン QuickScript で使用した場合、このステートメントはアラームのアラーム クラスを **almClass** タグ変数に返します。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmName、.AlarmOprName、.AlarmOprNode、.AlarmPri、.AlarmProv、.AlarmState、.AlarmTime、.AlarmType、.AlarmValue

.AlarmType ドットフィールド

選択したアラームに関連付けられているタグ変数のアラーム タイプを返します。サマリ モードで分散アラーム表示オブジェクトをクリックして、アラームを選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM("ObjectName.AlarmType",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

押しボタン QuickScript で使用される場合、このステートメントはオペレータがアラームを確認するときに、**almType** タグ変数に対して選択されたアラームのタイプを返します。

```
GetPropertyM("AlmObj_1.AlarmType",almType);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almType** は選択したアラームに関連付けられるタグ変数のアラーム タイプを含むメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmName、.AlarmOprName、.AlarmOprNode、.AlarmPri、.AlarmProv、.AlarmState、.AlarmTime、.AlarmValue

.AlarmState ドットフィールド

選択したアラームの状況を返します。サマリ モードで分散アラーム表示オブジェクトをクリックして、アラームを選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM( "ObjectName.AlarmState",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

押しボタン QuickScript で使用した場合、このステートメントは選択したアラームの状態を almState タグ変数に返します。

```
GetPropertyM("AlmObj_1.AlarmState",almState);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almState** は選択したアラームに関連付けられるタグ変数のアラーム状況を含むメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmName、.AlarmOperName、.AlarmOperNode、.AlarmPri、.AlarmProv、.AlarmTime、.AlarmType、.AlarmValue

.AlarmLimit ドットフィールド

選択したアラームに関連付けられているタグ変数のしきい値を返します。サマリ モードで分散アラーム表示オブジェクトをクリックして、アラームを選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM( "ObjectName.AlarmLimit",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

備考

この関数は、メッセージ型タグ変数を使用して、数値を取得します。これは、**GetProperty** 関数が実数をサポートしていないためです。**StringToReal()** 関数を使用して、結果を実数型タグ変数に割り当てることができます。

例

押しボタン QuickScript で使用した場合、このステートメントは選択したアラームのしきい値を **almLimit** タグ変数に返します。

```
GetPropertyM("AlmObj_1.AlarmLimit",almLimit);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almLimit** は選択したアラームに関連付けられるタグ変数のアラームしきい値を含むメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、**.AlarmAccess**、**.AlarmClass**、**.AlarmComment**、**.AlarmDate**、**.AlarmName**、**.AlarmOprName**、**.AlarmOprNode**、**.AlarmPri**、**.AlarmProv**、**.AlarmState**、**.AlarmTime**、**.AlarmType**、**.AlarmValue**

.AlarmPri ドットフィールド

選択したアラームに関連付けられているタグ変数の優先度（1 ～ 999）を返します。アラームは、サマリ モードで分散アラーム表示オブジェクトをクリックして選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM("ObjectName.AlarmPri", TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、**AlmObj_1** です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

押しボタン QuickScript で使用した場合、このステートメントはアラーム優先度を **almPriLvl** タグ変数に返します。

```
GetPropertyM("AlmObj_1.AlarmPri",almPri;
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almPri** は選択したアラームに関連付けられるタグ変数の優先度レベルを含むメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、**.AlarmAccess**、**.AlarmClass**、**.AlarmComment**、**.AlarmDate**、**.AlarmLimit**、**.AlarmName**、**.AlarmOprName**、**.AlarmOprNode**、**.AlarmProv**、**.AlarmState**、**.AlarmTime**、**.AlarmType**、**.AlarmValue**

.AlarmGroupSel ドットフィールド

選択したアラームに関連付けられているタグ変数のアラーム グループを返します。サマリ モードで分散アラーム表示オブジェクトをクリックして、アラームを選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM( "ObjectName.AlarmGroupSel",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

押しボタン QuickScript で使用した場合、このステートメントはアラーム グループの名前を almGroup タグ変数に返します。

```
GetPropertyM("AlmObj_1.AlarmGroupSel",almGroup);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almGroup** は選択したアラームに関連付けられるタグ変数のアラーム グループを含むメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、.AlarmGroup、.AlarmName

.AlarmAccess ドットフィールド

選択したアラームに関連付けられているタグ変数のアクセス名を返します。アラーム レコードは、サマリ モードで分散アラーム表示オブジェクトをクリックして選択する必要があります。

カテゴリ

アラーム

使用法

```
GetPropertyM("Objectname.AlarmAccess",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

押しボタン QuickScript で使用した場合、アラームに関連付けられるタグ変数のアクセス名を **almAccess** タグ変数に返します。

```
GetPropertyM("AlmObj_1.AlarmAccess",almAccess);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almAccess** は選択したアラームに関連付けられるタグ変数のアクセス名を含むメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、.AlarmClass、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmName、.AlarmOprName、.AlarmOprNode、.AlarmPri、.AlarmProv、.AlarmState、.AlarmTime、.AlarmType

.AlarmProv ドットフィールド

選択したアラームに関連付けられているタグ変数のアラーム プロバイダを返します。サマリ モードで分散アラーム表示オブジェクトをクリックして、アラームを選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM( "ObjectName.AlarmProv",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

押しボタン QuickScript で使用した場合、このステートメントはプロバイダ名を **almProv** タグ変数に返します。

```
GetPropertyM("AlmObj_1.AlarmProv", almProv);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almProv** は選択したアラームに関連付けられるタグ変数のプロバイダ名を含むメモリ メッセージ型タグ変数です。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmName、.AlarmOprName、.AlarmOprNode、.AlarmPri、.AlarmState、.AlarmTime、.AlarmType、.AlarmValue

.AlarmOprName ドットフィールド

選択したアラームを確認したログオン ユーザーの名前を返します。サマリ モードで分散アラーム表示オブジェクトをクリックして、アラームを選択する必要があります。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM( "ObjectName.AlarmOprName", TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

```
GetPropertyM("AlmObj_1.AlarmOprName", almOprName);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almOprName** はタグ変数に関連付けられるアラームに応答するユーザーの名前を含むメモリ メッセージ型タグ変数です。

押しボタン QuickScript で使用した場合、このステートメントはユーザーの名前を **almOprName** タグ変数に返します。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmName、.AlarmOprNode、.AlarmPri、.AlarmProv、.AlarmState、.AlarmTime、.AlarmType、.AlarmValue

.AlarmOprNode ドットフィールド

選択したアラームに関連付けられているタグ変数に対するオペレータ ノードを返します。アラームは、サマリ モードで分散アラーム表示オブジェクトをクリックして選択する必要があります。

ターミナル サービス環境でアラームが確認されると、オペレータ ノードは、対応するオペレータがターミナル サービス セッションを確立しているクライアント マシンの名前になります。ノード名を取得できない場合、ノードの IP アドレスが代わりに使用されます。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM( "ObjectName.AlarmOprNode", TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

```
GetPropertyM("AlmObj_1.AlarmOprNode", almOprNode);
```

AlmObj_1 は分散アラーム表示オブジェクトの名前で、**almOprNode** は選択したアラームに関連付けられるタグ変数のオペレータ ノードの名前を含むメモリ メッセージ型タグ変数です。

押しボタン QuickScript で使用した場合、このステートメントはユーザーのノードを almOprNode タグ変数に返します。

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmComment、.AlarmDate、.AlarmLimit、.AlarmName、.AlarmOprName、.AlarmPri、.AlarmProv、.AlarmState、.AlarmTime、.AlarmType、.AlarmValue

.AlarmComment ドットフィールド

タグ変数ではなく、アラームを記述する読み取り／書き込み文字列であるアラーム コメントを返します。デフォルトでは、新しいアプリケーションのコメントは空です。

ただし、古い InTouch アプリケーションが InTouch バージョン 7.11 以降に変換される場合、後方互換性のために、タグ変数コメントが .AlarmComment ドットフィールドにコピーされます。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM( "ObjectName.AlarmComment", TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

データタイプ

文字列（読み取り専用）

例

以下の例は、AlmObj_1 分散アラーム表示オブジェクトで選択したタグ変数のアラーム コメントが返され、almComment タグ変数に配置されます。

```
GetPropertyM("AlmObj_1.AlarmComment", almComment);
```

参照項目

GetPropertyM()、.AlarmAccess、.AlarmClass、.AlarmDate、.AlarmLimit、.AlarmName、.AlarmOprName、.AlarmOprNode、.AlarmPri、.AlarmProv、.AlarmState、.AlarmTime、.AlarmType、.AlarmValue

アラーム クエリーの設定

以下のクエリー関数を使用して、アラーム メモリからレコードを取得します。

- [almDefQuery\(\) 関数](#)
- [almQuery\(\) 関数](#)
- [almSetQueryByName\(\) 関数](#)

almDefQuery() 関数

デフォルト プロパティを使用してクエリーを実行し、指定された分散アラーム表示オブジェクトを更新します。

カテゴリ

アラーム

構文

```
[Result=]almDefQuery(ObjectName);
```

引数

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

備考

WindowMaker で分散アラーム表示オブジェクトを開発する間に、デフォルトのクエリー プロパティが指定されます。

例

```
almDefQuery("AlmObj_1");
```

参照項目

almQuery()、almSetQueryByName()

almQuery() 関数

指定された分散アラーム表示オブジェクトを更新するクエリーを実行して、指定したパラメータを使用します。

カテゴリ

アラーム

構文

```
[Result=]almQuery(ObjectName,AlarmList,FromPri,ToPri,State,Type);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

AlarmList

アラーム クエリー／名前マネージャのエイリアスを設定し、たとえば「\intouch!\$System」やメッセージ型タグ変数に対するクエリーを実行します。

FromPri

表示するアラームの開始優先度です。たとえば、100 または整数型タグ変数です。

ToPri

表示するアラームの終了優先度です。たとえば、900 または整数型タグ変数です。

State

表示するアラームのタイプを指定します。たとえば、「UnAck」またはメッセージ型タグ変数です。有効な状況は、All、UnAck、または Ack です。

タイプ

更新された表示に示されるアラーム レコードのタイプ :

"Hist" = 履歴アラーム

"Summ" = サマリ アラーム

例

このステートメントは、優先度が 500 ~ 600 の MyAlarmListGroup で指定されたすべての履歴アラームを取得します。アラームは AlmObj_1 アラーム表示に表示されます。

```
almQuery("AlmObj_1","MyAlarmListGroup",500,600,"All","Hist");
```

この例では、MyAlarmListGroup は、名前マネージャ セットアップから設定されたアラーム リストです。

参照項目

almDefQuery()、almSetQueryByName()

almSetQueryByName() 関数

ユーザー定義クエリー設定ファイルからのパラメータを使用して、分散アラーム表示オブジェクトの指定されたインスタンスに対して新しいアラーム クエリーを開始します。

カテゴリ

アラーム

構文

```
[Result=]almSetQueryByName(ObjectName, QueryName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

QueryName

クエリー設定を使用して作成されたクエリーの名前です。

備考

これは分散アラーム表示オブジェクトの特定インスタンスに対するクエリーです。画面上に、独自のクエリーを持つ表示がいくつか存在する可能性があります。

例

この例は、「Turbine Queries」という名前のクエリーからのパラメータを使用して、新しいクエリーを開始します。

```
almSetQueryByName("AlmObj_1","Turbine Queries");
```

参照項目

almQuery()、almDefQuery()

現在のクエリー プロパティの確認

以下のドットフィールドを使用して、アラーム メモリ クエリーのステータスを返します。

- [.AlarmGroup](#) ドットフィールド
- [.QueryType](#) ドットフィールド
- [.QueryState](#) ドットフィールド
- [.Successful](#) ドットフィールド

- [.PriFrom ドットフィールド](#)
- [.PriTo ドットフィールド](#)

.AlarmGroup ドットフィールド

分散アラーム表示オブジェクトを形成するために使用される現在のクエリーが含まれます。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyM( "ObjectName.AlarmGroup",TagName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意のメッセージ型タグ変数です。

備考

この読み取り専用ドットフィールドには、指定された分散アラーム表示オブジェクトで使用される現在のアラーム クエリーが含まれます。このクエリーは、アラーム グループのリストまたは直接のアラーム プロバイダ リファレンスの場合があります。

データタイプ

文字列（読み取り専用）

例

このステートメントは、AlmObj_1 分散アラーム オブジェクトによって使用される現在のアラーム クエリーを **CurrentQuery** タグ変数に返します。

```
GetPropertyM("AlmObj_1.AlarmGroup",CurrentQuery);
```

参照項目

GetPropertyM(), .AlarmGroupSel, .AlarmName

.QueryType ドットフィールド

現在のアラーム クエリーのタイプを表示します。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyI( "ObjectName.QueryType",TagName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意の整数型タグ変数です。

備考

この読み取り専用ドットフィールドには、指定された分散アラーム表示オブジェクトで使用する現在のクエリー タイプが含まれます。

データタイプ

整数型（読み取り専用）

有効値

1 = 履歴

2 = サマリ

例

以下のステートメントは、AlmObj_1 分散アラーム表示オブジェクトの現在のクエリー タイプを AlmQueryType タグ変数に返します。

```
GetPropertyI("AlmObj_1.QueryType",AlmQueryType);
```

参照項目

GetPropertyI()、.QueryState

.QueryState ドットフィールド

現在のアラーム状況クエリー フィルタを表示します。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyI( "ObjectName.QueryState",TagName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意の整数型タグ変数です。

備考

この読み取り専用ドットフィールドには、指定された分散アラーム表示オブジェクトによって使用される現在のクエリー フィルタが含まれます。

データタイプ

整数型（読み取り専用）

有効値

0 = 全て

1 = 未確認

2 = 確認

例

以下のステートメントは、AlmObj_1 分散アラーム表示オブジェクトの現在のクエリー フィルタを AlmQueryState タグ変数に返します。

```
GetPropertyI("AlmObj_1.QueryState", AlmQueryState);
```

参照項目

GetPropertyI(), .QueryType

.Successful ドットフィールド

現在のクエリーが成功したかどうかを示します。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyD( "ObjectName.Successful",TagName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

関数が処理される時、プロパティ値を保持する論理型タグ変数

備考

この読み取り専用ドットフィールドには、指定された分散アラーム表示オブジェクトの最後のクエリーの状況が含まれます。

データタイプ

論理型（読み取り専用）

有効値

0 = クエリーでエラー発生

1 = クエリー成功

例

以下のステートメントは、AlmObj_1 分散アラーム表示オブジェクトの最後のクエリーの状況を AlmFlag タグ変数に返します。

```
GetPropertyD("AlmObj_1.Successful",AlmFlag);
```

参照項目

GetPropertyD()

.PriFrom ドットフィールド

現在のクエリーで使用されるアラーム優先度範囲の最低値を返します。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyI("ObjectName.PriFrom", Tagname);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

任意の整数型タグ変数です。

データタイプ

整数型（読み取り専用）

例

以下のステートメントは、分散アラーム表示オブジェクト AlmObj_1 で使用されるクエリーの最小優先度値を、MinPri 整数型タグ変数に返します。

```
GetPropertyI("AlmObj_1.PriFrom",MinPri);
```

参照項目

GetPropertyI()、.PriTo、.AlarmPri

.PriTo ドットフィールド

現在のクエリーで使用されるアラーム優先度範囲の最大値が含まれます。

使用法

```
[ErrorNumber=]GetPropertyI("ObjectName.PriTo", Tagname);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

関数が処理されるときにプロパティ値を保持する整数型タグ変数です。

データタイプ

整数型（読み取り専用）

例

以下のステートメントは、分散アラーム表示オブジェクト AlmObj_1 によって使用されるクエリーの最大優先度値を、MaxPri 整数型タグ変数に返します。

```
GetPropertyI("AlmObj_1.PriTo",MaxPri);
```

参照項目

GetPropertyI()、.PriFrom、.AlarmPri

分散アラーム表示オブジェクトの更新の確認

以下のドットフィールドを使用して、分散アラーム表示オブジェクトにすべての現在アラームが含まれているか、または保留中の更新があるかどうかを確認します。

- [.ListChanged](#) ドットフィールド
- [.PendingUpdates](#) ドットフィールド

.ListChanged ドットフィールド

分散アラーム表示オブジェクト用の新規アラームまたは更新されたアラームがあるかどうかを示します。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyD("ObjectName.ListChanged",TagName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

関数が処理されるとき、プロパティ値を保持する論理型タグ変数

備考

この読み取り専用ドットフィールドには、分散アラーム表示オブジェクトで更新される必要のある変更があるかどうかについての状況が含まれます。このプロパティは、プロパティを読み取る際に自動的にリセットされます。

データ タイプ

論理型（読み取り専用）

有効値

0 = 表示オブジェクトに対する新しいアラームまたは更新がありません

1 = 表示オブジェクトに対する新しい更新があります

例

以下のステートメントは、AlmObj_1 分散アラーム表示オブジェクトの新規アラームまたは更新の状況を AlmDispStat タグ変数に返します。

```
GetPropertyD("AlmObj_1.ListChanged",AlmDispStat);
```

参照項目

GetPropertyD()

.PendingUpdates ドットフィールド

分散アラーム表示オブジェクトに対する保留中の更新の数を示します。表示が停止し、新しいアラームレコードが作成されたときには、一般的に保留中の更新があります。これらの更新は表示されませんが、保留中の更新のカウントは増加します。

カテゴリ

アラーム

使用法

```
[ErrorMessage=]GetPropertyI( "ObjectName.PendingUpdates", TagName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

関数が処理されるときにプロパティ値を保持する整数型タグ変数です。

備考

この読み取り専用ドットフィールドには、指定された分散アラーム表示オブジェクトの保留中の更新数が含まれます。ゼロより大きい任意の値は、分散アラーム表示オブジェクトに新規アラーム データがあることを示しています。この値は、オブジェクトがリフレッシュされるたびにリセットされます。

データタイプ

整数型（読み取り専用）

例

以下のステートメントは、AlmObj_1 分散アラーム表示オブジェクトの保留中の更新がある場合、整数 1 以上を AlarmPendingUpdates タグ変数に返します。

```
GetPropertyI( "AlmObj_1.PendingUpdates",AlarmPendingUpdates);
```

参照項目

GetPropertyI()

アラームの非表示

分散アラーム表示オブジェクトは、例外基準に一致するアラーム コンシューマで 1 つまたは複数のアラームを非表示にできます。アラームが例外基準に一致する場合、表示のインスタンスに表示されません。

QuickScript 関数を使用して、アラームを非表示にできます。

- [almSuppressAll\(\) 関数](#)
- [almUnsuppressAll\(\) 関数](#)
- [almSuppressDisplay\(\) 関数](#)
- [almSuppressGroup\(\) 関数](#)
- [almSuppressPriority\(\) 関数](#)
- [almSuppressTag\(\) 関数](#)
- [almSuppressSelected\(\) 関数](#)
- [almSuppressSelectedGroup\(\) 関数](#)
- [almSuppressSelectedPriority\(\) 関数](#)
- [almSuppressSelectedTag\(\) 関数](#)
- [almSuppressRetain\(\) 関数](#)
- [.SuppressRetain ドットフィールド](#)

almSuppressAll() 関数

サマリ モードで分散アラーム表示オブジェクトに現在表示されていないインスタンスも含め、現在のクエリーのアラームの現在および今後のすべてのインスタンスを非表示にします。

構文

```
[Result=] almSuppressAll(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

備考

この関数は almAckAll() と同様に機能します。画面を確認して、スクロールで上下に移動してアラームをすべて確認した場合に表示されるすべてのアラームを識別することによって、非表示にするアラームを識別します。

例

```
almSuppressAll("AlmObj_1");
```

参照項目

almSuppressGroup()、almSuppressTag()、almSuppressDisplay()、almSuppressPriority()、almSuppressRetain()、almSuppressSelected()、almSuppressSelectedGroup()、almSuppressSelectedPriority()、almSuppressSelectedTag()、almUnSuppressAll()

almUnSuppressAll() 関数

非表示のアラームをすべてクリアします。

構文

```
[Result=] almUnSuppressAll(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

例

```
almUnSuppressAll("AlmObj_1");
```

参照項目

almSuppressAll()、almSuppressGroup()、almSuppressTag()、almSuppressDisplay()、almSuppressPriority()、almSuppressRetain()、almSuppressSelected()、almSuppressSelectedGroup()、almSuppressSelectedPriority()、almSuppressSelectedTag()

almSuppressDisplay() 関数

サマリ モードで、分散アラーム表示オブジェクトで表示される現在および今後のアラームを非表示にします。

構文

```
[Result=]almSuppressDisplay(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

備考

この関数是对应する `almAckDisplay()` 関数と同様に機能します。現在表示されているすべてのアラームを識別して、非表示にするアラームを識別します。

例

```
almSuppressDisplay("AlmObj_1");
```

almSuppressGroup() 関数

指定したプロバイダとグループ名を持つ任意のアラームの現在および今後の発生を非表示にします。

構文

```
[Result=]almSuppressGroup(ObjectName, ApplicationName,GroupName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

ApplicationName

アプリケーションの名前です。たとえば、\\node1\InTouch です。

GroupName

アラーム グループの名前です。たとえば、\$System です。

例

```
almSuppressGroup( "AlmObj_1","\\InTouch","Turbines");
```

参照項目

`almSuppressAll()`、`almSuppressTag()`、`almSuppressDisplay()`、`almSuppressPriority()`、`almSuppressRetain()`、`almSuppressSelected()`、`almSuppressSelectedGroup()`、`almSuppressSelectedPriority()`、`almSuppressSelectedTag()`、`almUnSuppressAll()`

almSuppressPriority() 関数

同じプロバイダ名とグループ名を持つ指定された優先度範囲の任意のアラームの現在および今後の発生を非表示にします。

構文

```
[Result=]almSuppressPriority(ObjectName, ApplicationName, GroupName, FromPri, ToPri);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

ApplicationName

アプリケーションの名前です。たとえば、\\node1\InTouch です。

GroupName

グループの名前です。たとえば、\$System です。

FromPri

アラームの開始優先度です。たとえば、100 または整数型タグ変数です。

ToPri

アラームの終点優先度です。たとえば、900 または整数型タグ変数です。

例

```
almSuppressPriority("AlmObj_1","\\node1\Intouch", "Turbines",10,100);
```

参照項目

almSuppressAll()、almSuppressGroup()、almSuppressTag()、almSuppressDisplay()、almSuppressRetain()、almSuppressSelected()、almSuppressSelectedGroup()、almSuppressSelectedPriority()、almSuppressSelectedTag()、almUnSuppressAll()

almSuppressTag() 関数

同じプロバイダ名、グループ名、および優先度範囲を持つ指定したタグ変数に属する任意のアラームの現在および今後の発生を非表示にします。

カテゴリ

アラーム

構文

```
[Result=]almSuppressTag(ObjectName, ApplicationName, GroupName, TagName, FromPri, ToPri, AlarmClass, AlarmType);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

ApplicationName

アプリケーションの名前です。たとえば、\\node1\Intouch です。

GroupName

グループの名前です。たとえば、\$System です。

TagName

アラーム タグ変数の名前です。

FromPri

アラームの開始優先度です。たとえば、100 または整数型タグ変数です。

ToPri

アラームの終点優先度です。たとえば、900 または整数型タグ変数です。

AlarmClass

アラームのクラスです。たとえば、"Value" です。

AlarmType

アラームのタイプです。たとえば、"HiHi" です。

例

```
almSuppressTag("AlmObj_1","\\node1\Intouch", "Turbines","Valve1",10,100,"Value","LoLo");
```

参照項目

almSuppressAll()、almSuppressGroup()、almSuppressDisplay()、almSuppressPriority()、almSuppressRetain()、almSuppressSelected()、almSuppressSelectedGroup()、almSuppressSelectedPriority()、almSuppressSelectedTag()、almUnSuppressAll()

almSuppressSelected() 関数

サマリ モードで、分散アラーム表示オブジェクトで選択されるアラームの現在および今後の発生を非表示にします。

カテゴリ

アラーム

構文

```
[Result=]almSuppressSelected(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

備考

この関数は、**almAckSelect()** 関数と同様に機能します。表示オブジェクトで選択したアラームによってアラームを識別します。

例

```
almSuppressSelected("AlmObj_1");
```

参照項目

almSuppressAll()、almSuppressGroup()、almSuppressTag()、almSuppressDisplay()、almSuppressPriority()、almSuppressSelectedGroup()、almSuppressSelectedPriority()、almSuppressSelectedTag()、almUnSuppressAll()

almSuppressSelectedGroup() 関数

指定された分散アラーム表示オブジェクト内の同じプロバイダ名を持つ 1 つまたは複数の選択したアラームの同じグループに属すアラームの現在および今後の発生を非表示にします。

カテゴリ

アラーム

構文

```
[Result=]almSuppressSelectedGroup(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

備考

この関数は **almAckSelectedGroup()** 関数と同様に機能します。選択されたアラームを識別し、それらのアラームが属すグループを識別し、これらのグループからの今後の発生を非表示にします。

例

```
almSuppressSelectedGroup("AlmObj_1");
```

参照項目

almSuppressAll()、almSuppressGroup()、almSuppressTag()、almSuppressDisplay()、almSuppressSelected()、almSuppressSelectedPriority()、almSuppressSelectedTag()

almSuppressSelectedPriority() 関数

指定された分散アラーム表示オブジェクト内で、同じプロバイダ名およびグループ タグ変数を持つ 1 つまたは複数の選択されたアラームの同じ優先度に属すアラームの現在および今後の発生を非表示にします。

カテゴリ

アラーム

構文

```
[Result=]almSuppressSelectedPriority(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

備考

優先度は、選択したアラーム レコードの最低優先度および最大優先度から計算されます。

この関数は almAckSelectedPriority() 関数と同様に機能します。表示で選択されたアラームを識別し、これらのアラームの対応する優先度を識別し、同じ優先度を持つアラームの今後の発生を非表示にします。

例

```
almSuppressSelectedPriority("AlmObj_1");
```

参照項目

almSuppressAll()、almSuppressGroup()、almSuppressTagName()、almSuppressDisplay()、almSuppressSelected()、almSuppressSelectedGroup()、almSuppressSelectedTag()、almAckSelectedPriority()

almSuppressSelectedTag() 関数

同じプロバイダ名、グループ名、および優先度範囲を持つ 1 つまたは複数の選択されたアラームの同じタグ変数に属す任意のアラームの現在および今後の発生を非表示にします。

カテゴリ

アラーム

構文

```
[Result=]almSuppressSelectedTag(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

例

```
almSuppressSelectedTag("AlmObj_1");
```

参照項目

almSuppressAll()、almSuppressGroup()、almSuppressTag()、almSuppressDisplay()、almSuppressSelectedAlarm()、almSuppressSelectedGroup()、almSuppressSelectedPriority()、almAckSelectedTag()、almUnSuppressAll()

almSuppressRetain() 関数

後続のクエリーで発生したすべてのアラームを非表示にします。

カテゴリ

アラーム

構文

```
[Result=]almSuppressRetain(ObjectName,SuppressionRetainFlag);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

SuppressionRetainFlag

任意の論理型またはアナログ型のタグ変数、0 またはゼロ以外の値です。後続のクエリーに対しても非表示情報が保持される場合は TRUE、それ以外の場合は FALSE です。

備考

アラーム クエリーが変更されたときフラグが 0 である場合、非表示フィルタは削除されます。

例

```
almSuppressRetain("AlmObj_1", 0);
```

参照項目

almSuppressAll()、almSuppressGroup()、almSuppressTag()、almSuppressDisplay()、almSuppressPriority()、almSuppressSelected()、almSuppressSelectedGroup()、almSuppressSelectedPriority()、almSuppressSelectedTag()、almUnSuppressAll()

.SuppressRetain ドットフィールド

分散アラーム表示オブジェクトの非表示状態を保持する機能のステータスを読み取りまたは書き込みします。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyD( "ObjectName.SuppressRetain",TagName);  
[ErrorNumber=]SetPropertyD( "ObjectName.SuppressRetain",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

Tagname

スクリプトが処理されるとき、プロパティ値を保持する論理型タグ変数です。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = オフを保持します

1 = オンを保持します

例

以下のステートメントは、SupRtn 論理型タグ変数から、「AlmObj_1」の非表示保持のステータスを設定します。

```
SetPropertyD("AlmObj_1.SuppressRetain", SupRtn);
```

参照項目

GetPropertyD()、SetPropertyD()

アラーム表示のスクロール

以下の関数とドットフィールドを使用して、分散アラーム表示オブジェクト内のアラーム リストを垂直または水平にスクロールします。表示を停止することもできます。

- [almMoveWindow\(\) 関数](#)
- [.Freeze ドットフィールド](#)
- [.PrevPage ドットフィールド](#)
- [.NextPage ドットフィールド](#)

almMoveWindow() 関数

分散アラーム表示オブジェクトのアラーム リストを垂直または水平にスクロールします。

カテゴリ

アラーム

構文

```
[Result=]almMoveWindow(ObjectName,Option,Repeat);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

Option

実行するスクロール操作のタイプ :

タイプ	説明
LineDn	次の行へ移動
LineUp	前の行へ移動
PageDn	次のページへ移動
PageUp	前のページへ移動
Top	リストの一番上へ移動
Bottom	リストの一番下へ移動
PageRt	右ページへ移動
PageLf	左ページへ移動

タイプ	説明
Right	リストの最後へ移動（右側）
Left	リストの先頭へ移動（左側）

Repeat

操作が繰り返される回数です。

例

```
almMoveWindow("AlmObj_1", "Bottom", 0);  
almMoveWindow("AlmObj_1", "LineDn", 3);  
almMoveWindow("AlmObj_1", "PageUp", 0);
```

.Freeze ドットフィールド

.Freeze ドットフィールドは、停止ステータスを読み込むか、分散アラーム表示オブジェクトを停止または停止解除します。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyD("ObjectName.Freeze", TagName);  
[ErrorNumber=]SetPropertyD("ObjectName.Freeze", TagName);
```

引数**ObjectName**

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

関数が処理されるとき、プロパティ値を保持する論理型タグ変数

備考

分散アラーム表示オブジェクトの停止ステータスを含む、または変更する読み取り／書き込みドットフィールドです。アラーム表示オブジェクトが停止されたとき、表示されるアラームは更新することも、新しいアラームを追加することもできません。停止によって、アラームが点滅するかどうかに影響は与えられません。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = 停止がオフ

1 = 停止がオン

例

以下のステートメントは、AlmFreeze 論理型タグ変数からの値によって "AlmObj_1" の Freeze プロパティを設定します。

```
SetPropertyD("AlmObj_1.Freeze", AlmFreeze);
```

参照項目

GetPropertyD()、SetPropertyD()

.PrevPage ドットフィールド

分散アラーム表示オブジェクトを 1 ページ（アラームが全体に表示された 1 画面）上にスクロールします。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]SetPropertyD("ObjectName.PrevPage",0);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

備考

このプロパティが設定されると、分散アラーム表示オブジェクトで前のページが表示されます。リストの先頭に達した場合を除いて、前のページが表示された後で、変数が自動的に 1 に設定されます。リストの先頭に達した場合、値は 0 のままです。

データタイプ

論理型（読み取り／書き込み）

参照項目

GetPropertyD()、SetPropertyD()、.NextPage、.PageNum、.TotalPages

.NextPage ドットフィールド

分散アラーム表示オブジェクトを 1 ページ（アラームが全体に表示された 1 画面）下にスクロールします。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]SetPropertyD("ObjectName.NextPage",0);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

備考

このプロパティが設定されると、分散アラーム表示オブジェクトで次のページが表示されます。リストの末尾に達した場合を除いて、次のページが表示された後で、変数が自動的に 1 に設定されます。リストの先頭に達した場合、値は 0 のままです。

データタイプ

論理型（読み取り／書き込み）

参照項目

GetPropertyD()、SetPropertyD()、.PrevPage、.PageNum、.TotalPages

アラーム統計とカウンターの表示

以下の関数とドットフィールドを使用して、現在の分散アラーム表示オブジェクトに関する統計情報を表示します。

- [almShowStats\(\) 関数](#)
- [.PageNum ドットフィールド](#)
- [.TotalPages ドットフィールド](#)
- [.NumAlarms ドットフィールド](#)
- [.ProvidersReq ドットフィールド](#)
- [.ProvidersRet ドットフィールド](#)

almShowStats() 関数

指定した分散アラーム表示オブジェクトの [アラームの統計] ダイアログ ボックスを表示します。

カテゴリ

アラーム

構文

```
[Result=]almShowStats(ObjectName);
```

引数

ObjectName

アラーム オブジェクトの名前です。たとえば、AlmObj_1 です。

例

```
almShowStats("AlmObj_1");
```

.PageNum ドットフィールド

アラーム オブジェクトで表示されている現在のページ番号が含まれます。

カテゴリ

アラーム

構文

```
[ErrorNumber=]GetPropertyI("ObjectName.PageNum",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

分散アラーム表示オブジェクトから現在表示されているページ番号を保持する整数型タグ変数

備考

この読み取り専用ドットフィールドは、指定された分散アラーム表示オブジェクトで現在表示されているページ番号を返します。

データタイプ

整数型（読み取り専用）

例

以下のステートメントは、AlmObj_1 分散アラーム表示オブジェクトから現在表示されているページ番号を **AlarmPage** 整数型タグ変数に返します。

```
GetPropertyI("AlmObj_1.PageNum",AlarmPage);
```

参照項目

GetPropertyI(), .NextPage、.PrevPage、.TotalPages

.TotalPages ドットフィールド

分散アラーム表示オブジェクトの合計ページ数が含まれます。

カテゴリ

アラーム

構文

```
[ErrorNum=]GetPropertyI("ObjectName.TotalPages", TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

指定された分散アラーム表示オブジェクトに含まれるアラーム ページの合計数を取得する整数型タグ変数

備考

このドットフィールドは、指定された分散アラーム表示オブジェクトに含まれるアラーム ページの合計数を返します。1 ページは、指定時間に画面のオブジェクトに表示されるすべてのアラームと等しくなります。

データタイプ

整数型（読み取り専用）

例

以下のステートメントは、AlmObj_1 分散アラーム表示オブジェクトに含まれるページの合計数を **AlmTotalPages** 整数型タグ変数に返します。

```
GetPropertyI("AlmObj_1.TotalPages",AlmTotalPages);
```

参照項目

GetPropertyI(), .NextPage、.PrevPage、.PageNum

.NumAlarms ドットフィールド

分散アラーム表示オブジェクト内のアラーム数が含まれます。

カテゴリ

アラーム

構文

```
[ErrorNum=]GetPropertyI("ObjectName.NumAlarms", Tagname);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

指定された分散アラーム表示オブジェクトに登録されるアラームの現在数を保持する整数型タグ変数です。これには、表示されるアラームだけでなく、登録されているすべてのアラームが含まれます。

備考

この読み取り専用ドットフィールドは、分散アラーム表示オブジェクト内のアラームの合計数を返します。

データタイプ

整数型（読み取り専用）

例

以下のステートメントは、AlmObj_1 分散アラーム表示オブジェクトによって使用されるアラームの現在数を **AlarmCount** 整数型タグ変数に返します。

```
GetPropertyI("AlmObj_1.NumAlarms", AlarmCount);
```

参照項目

GetPropertyI()

.ProvidersReq ドットフィールド

指定された分散アラーム表示オブジェクトによって使用される現在のクエリーで必要とされるアラームプロバイダ数が含まれます。

カテゴリ

アラーム

構文

```
[ErrorNumber=]GetPropertyI( "ObjectName.ProvidersReq", TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

指定された分散アラーム表示オブジェクトに登録されるアラーム プロバイダの現在数を保持する整数型タグ変数です。これには、表示されるアラームだけでなく、登録されているすべてのアラームが含まれます。

データタイプ

整数型（読み取り専用）

例

以下のステートメントは、分散アラーム表示オブジェクト "AlmObj_1" によって使用される現在のクエリーで必要とされるアラーム プロバイダ数を返します。この値は、**TotalProv** 整数型タグ変数に書き込まれます。

```
GetPropertyI("AlmObj_1.ProvidersReq",TotalProv);
```

参照項目

GetPropertyI()、.ProvidersRet

.ProvidersRet ドットフィールド

指定された分散アラーム表示オブジェクトによって使用される現在のクエリーによって返されるアラーム プロバイダ数が含まれます。

カテゴリ

アラーム

使用法

```
[ErrorNumber=]GetPropertyI ("ObjectName.ProvidersRet",TagName);
```

パラメータ

ObjectName

分散アラーム表示オブジェクトの名前です。たとえば、AlmObj_1 です。

TagName

指定された分散アラーム表示オブジェクトにアラームを正常に返したアラーム プロバイダ数を保持する整数型タグ変数

備考

この読み取り専用ドットフィールドには、指定された分散アラーム表示オブジェクトによって使用される現在のクエリーによって返されるアラーム プロバイダ数が含まれます。

データタイプ

整数型（読み取り専用）

例

以下のステートメントは、AlmObj_1 分散アラーム表示オブジェクトにアラームを正常に返したアラーム プロバイダ数を返します。この値は、**RetProv** 整数型タグ変数に書き込まれます。

```
GetPropertyI("AlmObj_1.ProvidersRet",RetProv);
```

参照項目

GetPropertyI()、.ProvidersReq

エラーの説明

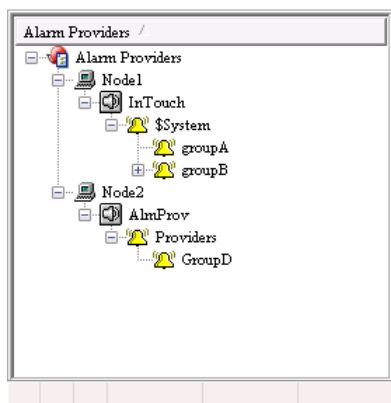
以下の表は、エラー番号について説明しています。この表に存在しない番号が返された場合、そのエラーは不明なエラーです。

エラー番号	説明
0	成功

エラー番号	説明
-1	一般エラー
-2	使用できるメモリが不十分です
-3	プロパティは読み取り専用です
-4	指定したアイテムはすでに存在します
-5	オブジェクト名が不明です
-6	プロパティ名が不明です

アラーム階層の表示

Alarm Tree Viewer ActiveX コントロールには、アラーム クエリーによって選択されたアラーム プロバイダのアラーム グループ階層が表示されます。Alarm Tree Viewer コントロールに表示されるアイテムには、アラーム プロバイダ、ノード、およびグループが含まれます。



Alarm Tree Viewer コントロールを使用することによって、Alarm Viewer コントロールの操作性が向上します。Alarm Tree Viewer コントロールでオペレータがアラーム プロバイダを選択する際に Alarm Viewer コントロールが新しいアラーム プロバイダをクエリーするようにスクリプトを作成できます。

Alarm Tree Viewer コントロールをどのように表示するか、またどのデータを表示するかを設定できます。詳細については、「[Alarm Viewer コントロールの設定](#)」を参照してください。

Alarm Tree Viewer コントロールの設定が終了したら、表示されているデータを以下のように変更できます。

- 名前によるデータのソート
- ツリーの更新
- 別のクエリーの実行

ActiveX コントロールの詳細については、「[ActiveX コントロール](#)」を参照してください。

Alarm Viewer コントロールの設定

Alarm Tree Viewer コントロールに対して、以下の内容を設定できます。

- 色を含む一般的なコントロール外観
- テキスト フォント
- 自動リフレッシュ
- ランタイムにユーザーがアクセスできる機能
- 表示するプロバイダとグループの選択
- カスタム保存クエリー
- アラーム グループのソート順

これらのオプションは、WindowMaker で Alarm Tree Viewer コントロールの実行中に設定できます。

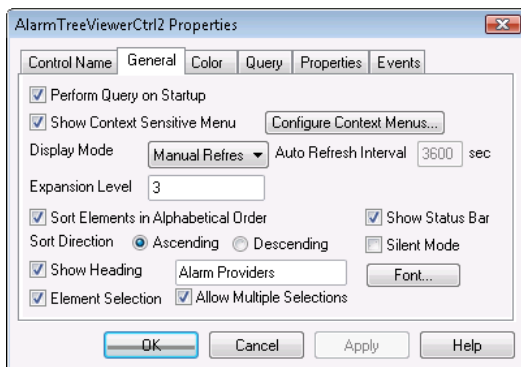
外観と色の設定

Alarm Tree Viewer コントロールの表示外観を設定する場合、以下の操作を行えます。

- ステータス バーを含めます。
- カラム ヘッダーを含めます。
- 表示要素の色を設定します。

外観を設定するには

1. Alarm Tree Viewer コントロールを右クリックして、[プロパティ] をクリックします。
[AlarmTreeViewCtrl2 のプロパティ] ダイアログ ボックスが表示されます。
2. [全般] タブをクリックします。

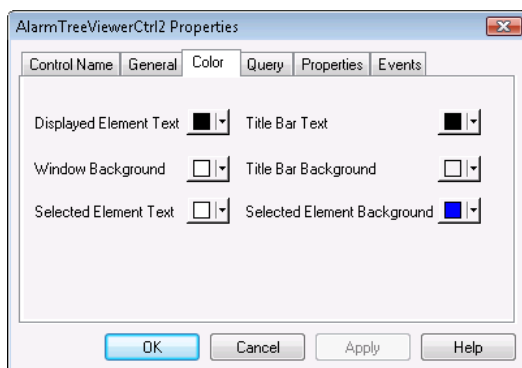


3. ランタイム ユーザーに対する Alarm Tree Viewer コントロールの表示方法を設定します。以下のいずれかを実行します。
 - デフォルトのクエリー プロパティを使用してツリーを自動的に更新するには、[起動時にクエリー実行] チェック ボックスをオンにします。それ以外の場合、ユーザーは [リフレッシュ] コマンドを実行して、ツリーを更新する必要があります。
 - ショートカット メニューを有効にするには、[コンテキスト メニューを表示] チェック ボックスをオンにします。メニューに表示するコマンドを設定するには、[コンテキスト メニューの設定] をクリックします。詳細については、[「ランタイムでユーザーがアクセスできる機能の管理」](#)を参照してください。
 - [表示モード] リストで、ツリーを更新する方法をクリックします。自動更新の場合は、[自動更新間隔] ボックスに更新間隔を入力します。範囲は 5 ～ 32767 秒です。

- **[拡張レベル]** ボックスに、ツリーの拡張レベル数を入力します。この設定により、コントロールを手動で更新するときに、アラーム ツリーが開くアラーム グループの分岐レベルが決定されます。値 **1** はプロバイダのみ表示し、値 **2** はプロバイダの直接アラーム グループを表示します。
- ツリー要素をアルファベット順にソートするには、**[アルファベット順に要素をソート]** チェック ボックスをオンにします。また、**[昇順]** または **[降順]** をクリックして、ソート順を設定します。
- 階層の上に見出しを表示するには、**[見出しの表示]** チェック ボックスをオンにします。ボックスには、見出しバーのテキストを入力します。
- **Alarm Tree Viewer** コントロールの一番下にステータス バーを表示するには、**[ステータス バーの表示]** チェック ボックスをオンにします。
- ツリーのフォント プロパティを設定するには、**[フォント]** をクリックします。Windows 標準の **[フォント]** ダイアログ ボックスが表示されます。
- ユーザーがツリーで要素を選択できるようにするには、**[要素の選択]** チェック ボックスをオンにします。
- **Ctrl** キーおよび **Shift** キーを使用して、1 つまたは複数の要素を選択できるようにするには、**[複数選択可]** チェック ボックスをオンにします。
- **Alarm Tree Viewer** コントロールでランタイムのエラー メッセージが表示されないようにするには、**[サイレントモード]** チェック ボックスをオンにします。エラー メッセージは、常にロガーに送信されます。

4. **[適用]** をクリックします。

5. **[色]** タブをクリックします。



1. パレットのボタンをクリックして、**Alarm Tree Viewer** コントロールの表示要素に色を割り当てます。

タイトルバーのテキスト、ウィンドウの背景、選択された要素テキスト、および選択された要素背景の色を設定できます。

2. **[適用]** をクリックします。

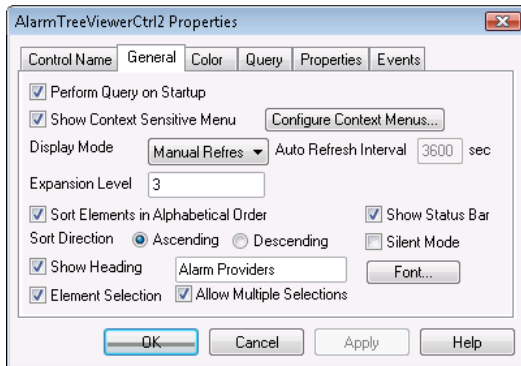
フォントの設定

Alarm Tree Viewer コントロールのテキスト外観を設定できます。

フォントを設定するには

1. **Alarm Tree Viewer** コントロールを右クリックして、**[プロパティ]** をクリックします。**[AlarmTreeViewerCtrl のプロパティ]** ダイアログ ボックスが表示されます。

2. **[全般]** タブをクリックします。



3. **[フォント]** をクリックします。Windows 標準の **[フォント]** ダイアログ ボックスが表示されます。フォントを設定して、**[OK]** をクリックします。
4. **[OK]** をクリックします。

自動更新の設定

Alarm Tree Viewer コントロールをランタイム時に自動更新するように設定できます。それ以外の場合、オペレータは手動で Alarm Tree Viewer コントロールを更新する必要があります。

自動更新に設定するには

1. Alarm Tree Viewer コントロールを右クリックして、**[プロパティ]** をクリックします。
[AlarmTreeViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. **[一般]** タブをクリックします。
3. **[表示モード]** リストで、ツリーを更新する方法 (**[手動更新]** または **[自動更新]**) をクリックします。自動更新の場合は、**[自動更新間隔]** ボックスにツリーの更新間隔を入力します。範囲は 5 ～ 32767 秒です。
4. **[適用]** をクリックします。

Alarm Tree View の表示更新のチューニング

Alarm Tree View の表示を更新するとき、ツリーに一覧表示されるアラーム プロバイダがクエリーに含まれたアラーム プロバイダに一致しないことがあります。これは、リモートアラーム プロバイダにアラーム グループの非常に大きな階層がある場合、またはアラーム プロバイダとのネットワーク接続が遅い場合に発生することがあります。

Alarm Tree View の表示を適切に更新するには、InTouch.ini ファイルに入力してチューニングできる 3 つの設定があります。これらの設定は、クエリーに対する完全な応答を待機するまでの時間、およびその間にアラーム クエリーを再試行する頻度を指定します。

デフォルトでは、これらの設定は InTouch.ini ファイルに含まれていないので、手動で入力してチューニングする必要があります。設定を明示的に InTouch.ini ファイルに入力しない場合、デフォルト値が使用されます。

AlarmTreeFastRetryMax

この設定は、クエリーが送信された直後に、そのツリー データ取得の最後の高速試行（毎秒 1 回）が実行されるまでの時間を決定します。値は秒単位です。

使用できる値：1 ～ 32767

デフォルト値：10

例：

```
[InTouch]  
AlarmTreeFastRetryMax=5
```

AlarmTreeSlowRetryInterval

高速再試行が完了した後、この設定は、ツリー データ取得の追加の再試行を実行する頻度（秒）を決定します。

使用できる値：1 ～ 32767

デフォルト値：5

例：

```
[InTouch]  
AlarmTreeSlowRetryInterval=10
```

AlarmTreeTotalRetryMax

この設定は、両方の種類の再試行を実行する最大期間（秒）を指定します。

使用できる値：1 ～ 32767

デフォルト値：30

例：

```
[InTouch]  
AlarmTreeTotalRetryMax=60
```

デフォルト設定の再試行動作

たとえば、デフォルトの再試行設定およびデフォルトの最大再試行期間（30 秒）を使用する場合は、以下の処理が行われます。

- 高速再試行間隔の間、10 秒間、1 秒ごとに取得の再試行が行われます。
- 低速再試行間隔の間、5 秒間、1 秒ごとに取得の再試行が行われます。

表示更新が完了したと見なされる場合

表示は、以下の時点まで引き続き更新されます。

- アラーム クエリーで指定されたすべてのプロバイダーへの接続が完了した時点
- すべてのプロバイダの階層ツリーが表示された時点

表示の更新回数は、クエリーが送信されるたびに変更された値またはデフォルト値にリセットされます。

ランタイム時における機能へのアクセスの管理

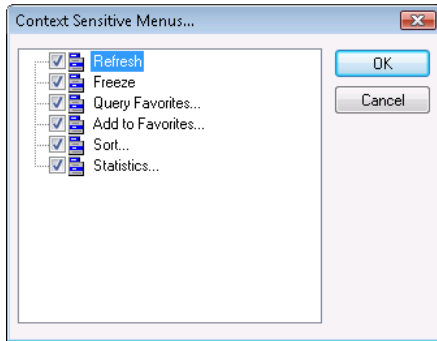
Alarm Tree Viewer コントロールには、オペレータがランタイム中にコントロールを右クリックして開くことができるショートカットメニューが含まれています。メニューに表示するコマンドは設定できます。

ランタイムのショートカットメニューを設定するには

1. Alarm Tree Viewer コントロールを右クリックして、[プロパティ] をクリックします。

[AlarmTreeViewCtrl のプロパティ] ダイアログ ボックスが表示されます。

2. **[全般]** タブをクリックします。
3. ショートカット メニューを有効にするには、**[コンテキストメニューを表示]** チェック ボックスをオンにします。
4. **[コンテキストメニューの設定]** をクリックします。
[コンテキストメニュー] ダイアログ ボックスが表示されます。



5. ショートカット メニューに表示する各コマンドのチェック ボックスをオンにします。ショートカット コマンドは少なくとも 1 つ選択する必要があります。

コマンド	説明
最新の情報への更新	Alarm Tree Viewer コントロールに表示されるデータを更新します。
停止	ツリーの停止/停止解除のモード切り替えを可能にします。
クエリー設定	使用できるリストからクエリー設定を選択するための、 [アラーム クエリー] ダイアログ ボックスを表示します。
設定ファイルに追加	[クエリーの追加] ダイアログ ボックスで、新しいクエリーを追加できます。
ソート	[ソート] ダイアログ ボックスを表示して、Alarm Tree Viewer コントロールのデータを昇順または降順にソートします。
統計	Alarm Tree Viewer コントロールに現在表示されている取得されたアラーム プロバイダのパーセンテージを表示する [アラームの統計] ダイアログ ボックスを表示します。

6. **[OK]** をクリックし、**[コンテキストメニュー]** ダイアログ ボックスを閉じます。
7. **[適用]** をクリックします。

表示するプロバイダとグループの設定

Alarm Tree Viewer コントロールに属するアラーム プロバイダとグループのアラーム クエリーを設定します。アラーム クエリーは、スペースで区切られたアラーム プロバイダのリストです。アラーム プロバイダの有効な構文は以下のとおりです。

アラーム プロバイダへのフルパス:

\\Node\ProviderName

ローカル アラーム プロバイダへのパス:

\ProviderName

複数のクエリーの場合、各クエリーをスペースで区切ります。次に例を示します。

\\InTouch \\Node17\InTouch \\MyNode\InTouch

デフォルトのアラーム クエリーは **\InTouch** です。アラーム クエリーに対してタグ変数を使用することはできません。

複数のアラーム グループをクエリーして、その後で 1 つまたは複数のグループの配置を解除した場合、**Alarm Tree Viewer** コントロールでは自動的に更新は行われず、これらのグループは表示から削除されません。また、アラーム プロバイダの登録を解除するには、アラーム プロバイダを停止して再起動する必要があります。

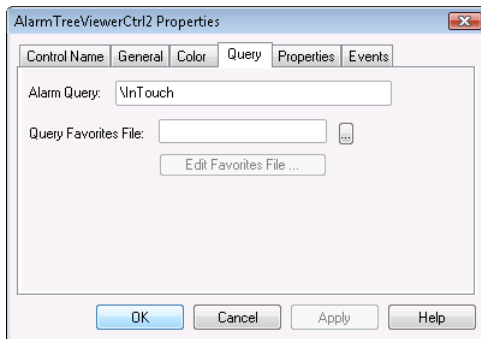
アラーム クエリーで **\Galaxy** を指定して **Galaxy** をクエリーする場合、**Galaxy** に配置されたすべての **InTouch** アラーム プロバイダが表示されます。次に例を示します。

\\Node\Galaxy!Area[名前]

同じ名前のグループが含まれる複数のアラーム プロバイダを持つノードから情報をクエリーする場合、ツリーの最後のアラーム プロバイダに対するレコードが表示されます。

アラーム クエリーを設定するには

1. **Alarm Tree Viewer** コントロールを右クリックして、**[プロパティ]** をクリックします。
[AlarmTreeViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. **[クエリー]** タブをクリックします。



3. **[アラーム クエリー]** ボックス、最初のアラーム クエリーへのパスを入力します。
4. **[適用]** をクリックします。

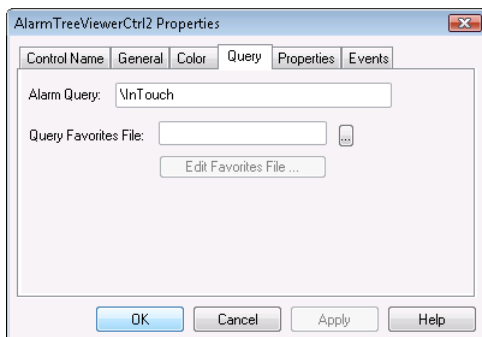
クエリー設定を使用したカスタム保存クエリーの作成

オペレータがショートカット メニューから選択できるクエリー設定のリストを設定できます。

クエリー ファイルは任意のフォルダに保存できます。**InTouch** アプリケーション フォルダに保存する必要はありません。アラーム クエリー ファイルは、**.xml** ファイルです。

クエリー設定ファイルを設定するには

1. Alarm Tree Viewer コントロールを右クリックして、[プロパティ] をクリックします。
[AlarmTreeViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [クエリー] タブをクリックします。



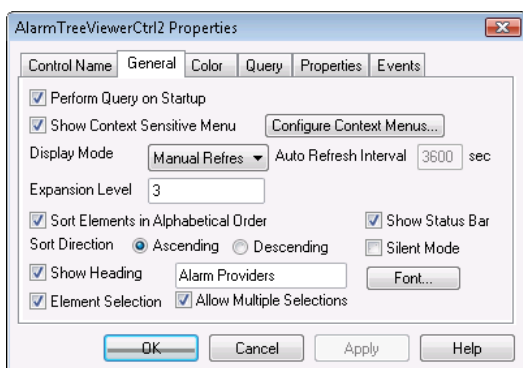
3. クエリー設定ファイルを設定します。
 - a. [クエリー設定ファイル] ボックスに、ネットワーク パスとファイル名を入力するか、省略記号 ボタンをクリックしてファイルを参照します。
 - b. クエリー設定ファイルを編集するには、[設定ファイルの編集] ボタンをクリックします。[アラーム クエリー] ウィンドウが開き、設定ファイルのフィルタを追加、変更、または削除できます。完了したら、[OK] をクリックして変更を保存し、ウィンドウを閉じます。
4. [OK] をクリックします。

アラーム グループのソート順の設定

Alarm Tree Viewer コントロールでは、ノードとアラーム グループをアルファベットの昇順または降順のいずれかで表示できます。

アラーム グループのソート順を設定するには

1. Alarm Tree Viewer コントロールを右クリックして、[プロパティ] をクリックします。
[AlarmTreeViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [一般] タブをクリックします。

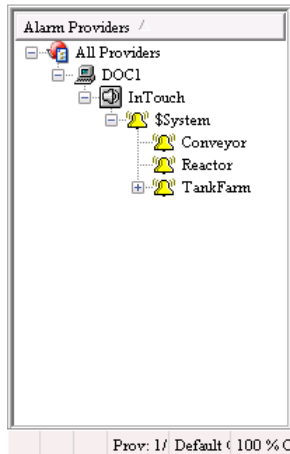


3. アラーム グループをアルファベット順で一覧表示するには、[アルファベット順に要素をソート] チェック ボックスをオンにします。
4. [昇順] または [降順] をクリックして、ソート順を指定します。

5. **[OK]** をクリックします。

ランタイムでの Alarm Tree Viewer コントロールの使用

Alarm Tree Viewer コントロールを使用して、アラーム プロバイダとアラーム グループの階層をナビゲートできます。



Alarm Tree Viewer コントロールには、複数のノードとアラーム プロバイダを表示できます。

- ノードはコンピュータ アイコンで示されます。
- アラーム プロバイダはスピーカー アイコンで示されます。
- アラーム グループはベル アイコンで示されます。

1 つまたは複数のアラーム グループを選択した状態で、Alarm Tree Viewer コントロールや Alarm DB View コントロールで使用するアラームのクエリを生成できます。複数のアラーム グループを選択するには、**Shift** キーを押したままグループをクリックします。すべてのグループの選択を解除するには、空白領域をクリックします。

コントロールが設定されている方法に基づいて、ランタイムでのショートカットメニューに以下の 1 つまたは複数のコマンドが表示されます。

- **リフレッシュ** – アラームの手動による更新を施行します。
- **停止** – アラームの更新を停止します。
- **クエリー設定** – **[アラーム クエリー]** ダイアログ ボックスが開き、以前定義されたアラーム クエリーのリストからアラーム クエリーを選択できます。
- **設定ファイルに追加** – 選択したグループ（存在する場合）に基づいてクエリー文字列が入力された状態の **[クエリーの追加]** ダイアログ ボックスが開きます。
- **[ソート]** – アラーム グループをアルファベットの昇順または降順でソートするオプションを持つ **[ソート]** ダイアログ ボックスが開きます。
- **[統計]** – 取得済みのアラーム プロバイダのパーセンテージを示す **[アラームの統計]** ダイアログ ボックスが開きます。

Alarm Tree Viewer コントロールのステータス バー情報の理解

Alarm Tree Viewer コントロールのステータス バーはウィンドウの一番下に表示され、以下の情報が表示されます。

- 現在のクエリーの名前
- 現在のクエリーの完了状況を示すパーセンテージ

クエリー設定の使用

Alarm Tree Viewer コントロールのショートカット メニューの [クエリー設定] コマンドを使用すると、定義済みのアラーム クエリーのリストから素早くアラーム クエリーを選択して実行できます。また、新しい名前付きクエリーを作成したり、既存のクエリーを編集したり、既存のクエリーを削除したりすることもできます。

アラーム クエリーを選択して実行するには

1. ランタイムで Alarm Tree Viewer コントロールを右クリックします。
2. [クエリー設定] をクリックします。[アラーム クエリー] ダイアログ ボックスが表示されます。
3. 現在定義されているクエリーのリストから表示する名前付きクエリーを選択します。
4. [OK] をクリックします。Alarm Tree Viewer コントロールに、選択したクエリーのアラーム グループ情報が表示されます。

Alarm Tree Viewer コントロールの ActiveX プロパティの使用

スクリプトを使用して Alarm Tree Viewer コントロールのプロパティの値を直接設定するか、InTouch タグまたは I/O 参照に割り当てることができます。プロパティの設定の詳細については、「[ActiveX コントロールのスクリプト](#)」を参照してください。

以下の表には、Alarm Tree Viewer コントロールのプロパティが一覧表示されています。色の値の設定の詳細については、「[ActiveX コントロールの色の設定](#)」を参照してください。

プロパティ名	説明
AddtoFavoritesMenu	[設定ファイルに追加] ショートカット メニュー コマンドを有効または無効にします。
AlarmQuery	最初のアラーム クエリーを表示し、クエリーを変更できるようにします。有効な構文は、\\<ノード>\<プロバイダ> または \<プロバイダ> です。
ElementSelection	ランタイム中にオペレータがツリーの要素を選択することができるかどうかを制御します。
ExpansionLevel	コントロールを手動で更新するときに、アラーム ツリーが開く分岐レベルを設定します。値 1 はプロバイダのみ表示し、値 2 はプロバイダの直接アラーム グループを表示します。
Font	コントロールに表示されるレコードと見出しのフォントを取得または設定します。
FreezeMenu	[停止] メニュー コマンドを有効または無効にします。
HeaderText	Alarm Tree Viewer コントロールの見出しに表示されるテキストを取得または設定します。

プロパティ名	説明
MultiSelection	Alarm Tree Viewer コントロールで複数の要素を選択できるようにします。
QueryFavoritesFile	クエリー設定ファイルの名前を取得または設定します。
QueryFavoritesMenu	[クエリー設定] メニュー コマンドを有効または無効にします。
QueryStartup	選択した場合、デフォルトのクエリー プロパティを使用して、Alarm Tree Viewer コントロールを自動的に更新します。選択しない場合、Alarm Tree Viewer コントロールを更新するには再クエリーする必要があります。
RefreshInterval	コントロールの自動更新間隔を秒単位で取得します。
RefreshMenu	ショートカット メニューに [リフレッシュ] コマンドを表示するかどうかを決定する値を取得または設定します。
SelTextBackColor	選択した要素の背景色を取得または設定します。
SelTextColor	選択した要素のテキスト色を取得または設定します。
ShowContextMenu	ショートカット メニューを有効または無効にします。
ShowHeading	Alarm Tree Viewer コントロールのタイトル バーを表示または非表示にします。
ShowStatusBar	ステータス バーが表示されるかどうか決定する値を取得するか、設定します。
SilentMode	コントロールがサイレント モードであるかどうかを決定する値を取得または設定します。
SortElements	Alarm Tree Viewer コントロールのソートを有効または無効にします。
SortMenu	[ソート] メニュー コマンドを有効または無効にします。
SortOrder	ソート順を取得または設定します。可能な値は「昇順」および「降順」です。それぞれ、0 および 1 で示されます。
StatsMenu	[統計] メニュー コマンドを有効または無効にします。
TextColor	Alarm Tree Viewer コントロールのテキスト色を取得または設定します。
TitleBackColor	タイトル バーの背景色を取得または設定します。ShowHeading プロパティが設定されている場合のみ使用できます。
TitleForeColor	タイトル バーの前景色を取得または設定します。ShowHeading プロパティが設定されている場合のみ使用できます。
WindowColor	Alarm Tree Viewer コントロールのウィンドウの背景色を取得または設定します。

Alarm Tree Viewer コントロールの ActiveX メソッドの使用

Alarm Tree Viewer コントロールのメソッドをスクリプトで使用すると、以下の操作を行えます。

- コントロールに関する情報を取得します。
- アラーム階層内の特定のエントリに関する情報を取得します。
- コントロールを停止します。
- クエリー文字列を作成します。
- クエリーを実行します。

メソッドの呼び出しの詳細については、「[ActiveX コントロールのスクリプト](#)」を参照してください。

コントロールに関する情報の取得

以下のメソッドを使用して、Alarm Tree Viewer コントロールに関する情報を取得できます。

- [AboutBox\(\) メソッド](#), [AboutBox\(\) メソッド](#), [AboutBox\(\) メソッド](#)
- [GetElementCount\(\) メソッド](#), [GetElementCount\(\) メソッド](#)

AboutBox() メソッド

Alarm Tree Viewer の [バージョン情報] ダイアログ ボックスを表示します。

GetElementCount() メソッド

ツリーの要素の合計数を取得します。

構文

```
Object.GetElementCount()
```

例

コントロールの名前は AlarmTreeViewCtrl1 であり、nTag1 は整数型または実数型のタグ変数です。

```
nTag1 = #AlarmTreeViewCtrl1.GetElementCount();
```

特定のエントリに関する情報の取得

一組のメソッドを使用して、Alarm Tree Viewer コントロールのウィンドウに表示されている要素に関する情報を取得できます。

- [CheckElementMembership\(\) メソッド](#)
- [GetElementCount\(\) メソッド](#), [GetElementCount\(\) メソッド](#)
- [GetElementName\(\) メソッド](#)
- [GetElementPath\(\) メソッド](#)
- [GetSelectedElementCount\(\) メソッド](#)
- [GetSelectedElementName\(\) メソッド](#)
- [GetSelectedElementPath\(\) メソッド](#)
- [GetSubElementCount\(\) メソッド](#)
- [GetSubElementName\(\) メソッド](#)
- [GetSubElementPath\(\) メソッド](#)

CheckElementMembership() メソッド

下位のツリー要素が上位のツリー要素の一部であるかどうかを確認します。

構文

```
Object.CheckElementMembership(PathName, DescendantElementName, AncestorElementName)
```

パラメータ

PathName

パスの名前です。たとえば、\InTouch または \\NodeName です。

DescendantElementName

下位要素の名前です。たとえば、GroupA です。

AncestorElementName

上位要素の名前です。たとえば、GroupB です。

GetElementCount() メソッド

ツリーの要素の合計数を取得します。

構文

```
Object.GetElementCount()
```

例

コントロールの名前は AlarmTreeViewCtrl1 であり、nTag1 は整数型または実数型のタグ変数です。

```
nTag1 = #AlarmTreeViewCtrl1.GetElementCount();
```

GetElementName() メソッド

インデックスに対応する要素名を取得します。

構文

```
Object.GetElementName(ElementIndex)
```

パラメータ

ElementIndex

要素のインデックスです。

例

コントロール名は AlarmTreeViewCtrl1 であり、StrTag はメッセージ型タグ変数です。

```
StrTag = #AlarmTreeViewCtrl1.GetElementName(3);
```

GetElementPath() メソッド

インデックスに対応する要素パスを、指定された拡張レベルまで取得します。

構文

```
Object.GetElementPath(ElementIndex, ExpansionLevel)
```

パラメータ

ElementIndex

要素のインデックスです。

ExpansionLevel

拡張レベルです。

例

コントロール名は AlarmTreeViewCtrl1 であり、StrTag はメッセージ型タグです。インデックス 17 の要素の 4 レベルまでのパスを返します。

```
StrTag = #AlarmTreeViewCtrl1.GetElementPath(17, 4);
```

GetSelectedElementCount() メソッド

ツリー内で選択した要素の数を取得します。

構文

```
Object.GetSelectedElementCount()
```

例

コントロールの名前は AlarmTreeViewCtrl1 であり、nTag1 は整数型または実数型のタグ変数です。

```
nTag1 = #AlarmTreeViewCtrl1.GetSelectedElementCount();
```

GetSelectedElementName() メソッド

Alarm Tree Viewr コントロールで選択した要素の名前を取得します。

構文

```
Object.GetSelectedElementName()
```

例

コントロール名は AlarmTreeViewCtrl1 であり、StrTag はメッセージ型タグです。

```
StrTag = #AlarmTreeViewCtrl1.GetSelectedElementName();
```

GetSelectedElementPath() メソッド

選択した要素のパスを、指定した拡張レベルまで取得します。

構文

```
Object.GetSelectedElementPath(ExpansionLevel)
```

パラメータ

ExpansionLevel

拡張レベルです。

例

コントロール名は AlarmTreeViewCtrl1 であり、StrTag はメッセージ型タグ変数です。

```
StrTag = #AlarmTreeViewCtrl1.GetSelectedElementPath(3);
```

GetSubElementCount() メソッド

指定した要素からのサブ要素の合計数を取得します。

構文

```
Object.GetSubElementCount(Path, ElementName)
```

パラメータ

Path

パスの名前です。次に例を示します。

```
\\NodeName\InTouch
```

Path パラメータが空白である場合、Alarm Tree Viewer コントロールは指定した要素名に一致するツリーの最初の要素を検索します。

ElementName

要素の名前です。たとえば、Group1 です。

例

コントロールの名前は AlarmTreeViewCtrl1 であり、nTag1 は整数型または実数型のタグ変数です。

```
nTag1 = #AlarmTreeViewCtrl1.GetSubElementCount("", "Group1" );  
nTag1 = #AlarmTreeViewCtrl1.GetSubElementCount( "\\NodeName", "Group1" );  
nTag1 = #AlarmTreeViewCtrl1.GetSubElementCount( "\\InTouch", "Group1" );  
nTag1 = #AlarmTreeViewCtrl1.GetSubElementCount( "\\NodeName\\InTouch", "Group1" );
```

GetSubElementName() メソッド

指定した要素の場合、対応するインデックスのサブ要素の名前を取得します。

構文

```
Object.GetSubElementName(Path, ElementName, ElementIndex)
```

パラメータ**Path**

パスの名前です。次に例を示します。

\\NodeName\\InTouch

Path パラメータが空白である場合、Alarm Tree Viewer コントロールは指定した要素名に一致するツリーの最初の要素を検索します。

ElementName

要素の名前です。たとえば、Group1 です。

ElementIndex

要素のインデックスです。

例

コントロール名は AlarmTreeViewCtrl1 であり、StrTag はメッセージ型タグ変数です。

```
StrTag = #AlarmTreeViewCtrl1.GetSubElementName("", "Group1", 1);  
StrTag = #AlarmTreeViewCtrl1.GetSubElementName("\\NodeName", "Group1", 1);  
StrTag = #AlarmTreeViewCtrl1.GetSubElementName("\\InTouch", "Group1", 1);  
StrTag = #AlarmTreeViewCtrl1.GetSubElementName("\\NodeName\\InTouch", "Group1", 1);
```

GetSubElementPath() メソッド

要素名のインデックスからのサブ要素のパスを、指定した拡張レベルまで取得します。

構文

```
Object.GetSubElementPath(Path, ElementName, ElementIndex, ExpansionLevel)
```

パラメータ**Path**

パスの名前です。次に例を示します。

\\NodeName\\InTouch

Path パラメータが空白である場合、Alarm Tree Viewer コントロールは指定した要素名に一致するツリーの最初の要素を検索します。

ElementName

要素の名前です。たとえば、Group1 です。

ElementIndex

要素のインデックスです。

ExpansionLevel

拡張レベルです。

例

コントロール名は AlarmTreeViewCtrl1 であり、StrTag はメッセージ型タグ変数です。

```
StrTag = #AlarmTreeViewCtrl1.GetSubElementPath("", "Group1", 1, 3);  
StrTag = #AlarmTreeViewCtrl1.GetSubElementPath("\\NodeName", "Group1", 1, 3);  
StrTag = #AlarmTreeViewCtrl1.GetSubElementPath("\\InTouch", "Group1", 1, 3);  
StrTag = #AlarmTreeViewCtrl1.GetSubElementPath("\\NodeName\\InTouch", "Group1", 1, 3);
```

ツリーの停止

Freeze() メソッドを使用すると、Alarm Tree Viewer コントロール ツリーを今後発生する変更内容で更新しないようにできます。

Freeze() メソッド

Alarm Tree Viewer コントロール ツリーを停止します。

構文

```
Object.Freeze(Frozen)
```

パラメータ

Frozen

ツリーが更新できるかどうかを制御します。

1 = ツリーを停止します。

0 = ツリーの停止を解除します。

例

Tag1 はメモリ論理型タグ変数として定義されており、コントロールの名前は AlarmTreeViewCtrl1 です。

```
Tag1 = 1;  
#AlarmTreeViewCtrl1.Freeze(Tag1);
```

選択によるクエリー文字列の作成

GetAlarmQueryFromSelection() メソッドを使用して、Alarm Tree Viewer コントロールで選択した要素からのクエリー文字列を取得できます。クエリー文字列は、Alarm Viewer コントロールで動的に使用できます。

GetAlarmQueryFromSelection() メソッド

Alarm Tree Viewer コントロールで選択した要素のクエリー文字列を返します。

構文

```
Object.GetAlarmQueryFromSelection()
```

例

コントロール名は AlarmTreeViewCtrl1 であり、StrTag はメッセージ型タグ変数です。次に例を示します。StrTag は \\NodeName\\InTouch\\GroupA に設定されます。

```
StrTag = #AlarmTreeViewCtrl1.GetAlarmQueryFromSelection();
```

クエリーの実行

メソッドを使用して **Alarm Tree Viewer** コントロールでクエリーを実行できます。クエリー設定ファイルに保存されている既存のクエリーを取得するかアラーム プロバイダの新しい収集を指定する文字列を設定するメソッドを使用して、**Alarm Tree Viewer** に対してクエリーを実行できます。

- [SetQueryByName\(\) メソッド](#), [SetQueryByName\(\) メソッド](#)
- [SetQueryByString\(\) メソッド](#)

SetQueryByName() メソッド

渡されるクエリー名によって指定されている通りに現在のクエリーを設定します。クエリーはクエリー設定ファイルに存在する必要があります。

構文

```
Object.SetQueryByName(QueryName)
```

パラメータ

QueryName

クエリー設定を使用して作成されたクエリーの名前です。たとえば、**Turbine Queries** です。

例

コントロールの名前は **AlarmTreeViewCtrl1** です。

```
#AlarmTreeViewCtrl1.SetQueryByName("Turbine Queries");
```

SetQueryByString() メソッド

アラーム プロバイダの新しい収集を指定する新しい文字列として現在のクエリーを設定します。

構文

```
Object.SetQueryByString(NewQuery)
```

パラメータ

NewQuery

アラーム クエリーが含まれる文字列です。次に例を示します。

```
\\MasterNode\InTouch
```

例

コントロールの名前は **AlarmTreeViewCtrl1** です。

```
#AlarmTreeViewCtrl1.SetQueryByString("\\MasterNode\InTouch");
```

メソッドとプロパティを使用するときのエラー処理

Alarm Tree Viewer コントロールのすべてのエラー メッセージは、**Logger** に送信されます。**Alarm Tree Viewer** コントロールをサイレント モードで実行するように設定すると、ランタイム エラーは表示されません。

Alarm Tree Viewer コントロール ActiveX イベントを使用したスクリプトのトリガ

マウス クリックやダブルクリックなどの **Alarm Tree Viewer** コントロール イベントに、**QuickScript** を割り当てることができます。イベントが発生すると、**QuickScript** が実行されます。

Alarm Tree Viewer コントロールは、以下のイベントをサポートしています。

- クリック
- DoubleClick
- ShutDown
- StartUp

Click イベントは ClicknElementID というパラメータを 1 つ持ち、これはランタイム時にクリックされたツリー内の要素を識別します。

DoubleClick イベントは DoubleClicknElementID というパラメータを 1 つ持ち、これはランタイム時にダブルクリックされたツリー内の要素を識別します。

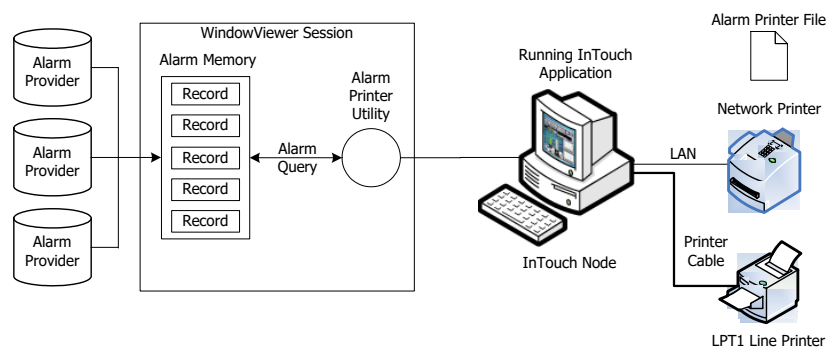
Click イベントおよび DoubleClick イベントの場合、-1 の ElementID が [すべてのプロバイダ] ノードに戻されます。

注: ユーザー インターフェイス メソッドが StartUp イベントから呼び出される場合、コントロールはまだ表示されていないため、Alarm Tree Viewer コントロールでは無視されます。このようなメソッドには、AboutBox()、CheckElementMembership()、Freeze()、GetAlarmQueryFromSelection()、GetElementCount()、GetElementName()、GetElementPath()、GetSelectedElementCount()、GetSelectedElementName()、GetSelectedElementPath()、GetSubElementCount()、GetSubElementName()、GetSubElementPath()、および Refresh() があります。

ActiveX イベントのスクリプトの詳細については、「[ActiveX コントロールのスクリプト](#)」を参照してください。

アラームの印刷

複数ノードからのアラームを印刷するには、InTouch Alarm Printer を使用します。専用のラインプリンタまたはネットワーク プリンタを使用して、イベントごとにアラーム メモリに保存されるアラーム レコードを印刷できます。また、Alarm Printer を使用して、アラーム レコードをファイルに保存することもできます。



分散アラーム システムは、特定のイベントが発生するごとにラインプリンタでそれらのイベントを印刷するように設定できます。通常、致命的なエラーが発生した場合にすぐにアラームを印刷して、情報を記録します。一般的に、シリアルポートまたはパラレルポート経由で InTouch アプリケーションを実行中のコンピュータに直接接続されているドットマトリックス プリンタを使用します。Windows のネットワーク プリンタとレーザー プリンタは、実際に 1 枚のページを印刷する前に全ページがメモリに保持されるため、破壊的なイベントのデータ収集には適切ではありません。

アラーム印刷とログの設定

Alarm Printer のインスタンスは複数実行できます。Alarm Printer の各インスタンスは別のプリンタに印刷するように設定され、個別のアラーム クエリーで設定される必要があります。

Alarm Printer の個別のインスタンスが、特定の優先度範囲でアラームを印刷するように設定することもできます。たとえば、1 つの Alarm Printer インスタンスで優先度の高いアラームだけを印刷し、別のインスタンスで優先度の低いアラームだけを印刷するように設定できます。同様に、Alarm Printer の 1 つのインスタンスを使用して、工場の 1 つの領域からのアラームを印刷し、別のインスタンスでは別の領域からのアラームを印刷するように設定できます。

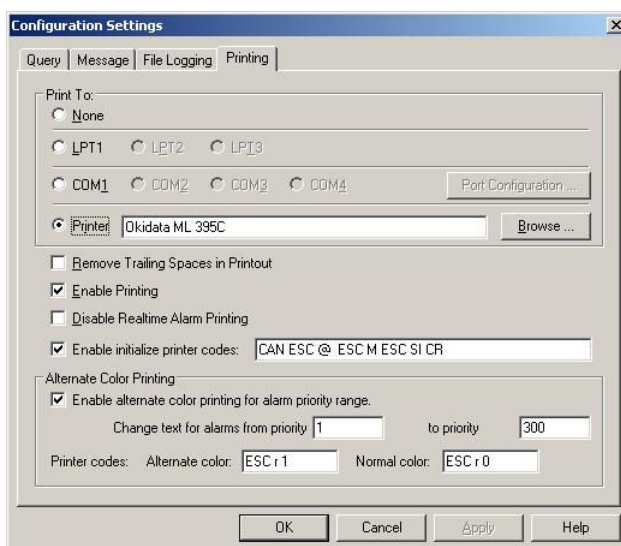
Alarm Printer 設定を拡張子 .alc を持つファイルに保存できます。必要な数の設定ファイルを作成できます。Alarm Printer では、実行中の Alarm Printer の各インスタンスに対して個別の設定ファイルが使用されます。

プリンタの設定

Alarm Printer で使用するプリンタを指定する必要があります。また、アラーム発生時にアラームを印刷するかどうかを選択する必要があります。アラームを印刷する場合は、専用プリンタを使用します。ローカルに接続されたプリンタまたはネットワークに接続されたプリンタを使用できます。任意の Windows プリンタを Alarm Printer の出力先として使用できます。

プリンタを設定するには

1. Alarm Printer ユーティリティを開きます。以下の手順を実行します。
 - a. WindowMaker の [ツール] ビューで、[アプリケーション] を展開します。
 - b. [Alarm Printer] をダブルクリックします。
2. メニューバーの [設定] をクリックします。[プロパティの設定] ダイアログ ボックスが表示されます。
3. [印刷] タブをクリックします。



4. [出力先] 領域で、アラーム プリンタへの接続を選択します。
 - プリンタを使用しない場合は、[なし] をクリックします。

- InTouch アプリケーションを実行中のコンピュータにパラレルポート経由で接続されたプリンタを使用するには、**[LPT1-3]** をクリックします。
- InTouch アプリケーションを実行中のコンピュータにシリアルポート経由で接続されたプリンタを使用するには、**[COM1-4]** をクリックします。**[ポート設定]** をクリックして、**[COMのプロパティ]** ダイアログボックスを表示し、選択した COM ポートに割り当てられたデフォルト値を変更します。
- InTouch アプリケーションを実行中のコンピュータにネットワーク経由で接続されたプリンタを使用するには、**[プリンタ]** をクリックします。ボックスにプリンタの名前を入力するか、**[参照]** をクリックして、使用可能なプリンタを選択します。

注: 必要なプリンタが表示されない場合は、Windows のプリンタの追加ウィザードを使用してプリンタを追加します。

5. プリンタで空白行または空白ページが印刷されないようにするには、**[後に続くスペースを削除]** チェックボックスをオンにします。
6. アラームを印刷するには、**[印刷可能]** チェックボックスをオンにします。
7. アラームの発生時に Alarm Printer でアラームが印刷されないようにするには、**[リアルタイムのアラーム印刷を無効にする]** チェックボックスをオンにします。
8. または、特定の設定でプリンタを初期化し、指定した優先度範囲に合わせて印刷特性を変更するよう印刷コマンドシーケンスを設定します。詳細については、「[アラーム印刷コマンドの設定](#)」を参照してください。
9. **[OK]** をクリックします。

アラーム印刷コマンドの設定

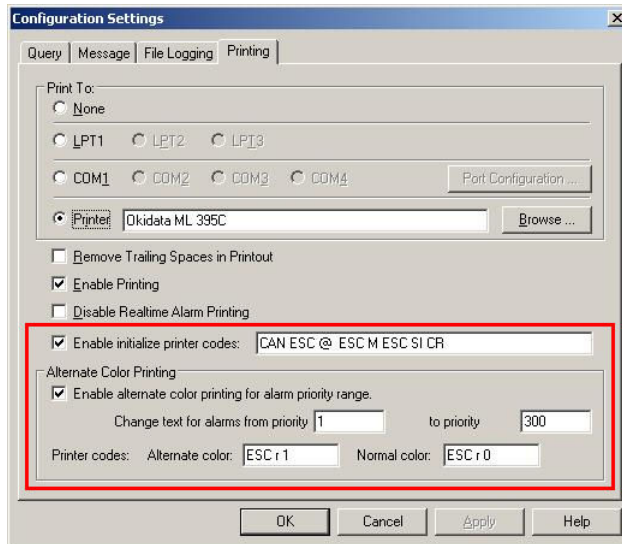
プリンタを最初に開いたとき（クエリーが開始および停止するたび）にプリンタ コマンド初期化シーケンスを送信するよう Alarm Printer を設定できます。プリンタ コマンド初期化シーケンスを使用してプリンタを設定できます（文字ピッチや左右の印刷マージンなど）。

また、印刷色でアラームがアラーム優先度の指定範囲内にあるかどうかを示すよう、代替の印刷色および標準の印刷色を設定できます。代替および標準の印刷コマンドでは、必ずしも色を指定する必要はありません。これらのコマンドを使用して、ダブルストライク、強調印刷、下線、斜体、またはその他の特殊印刷特性を有効/無効にし、特定のアラーム優先度をその他の優先度と視覚的に異なるように表示できます。

注: 白黒印刷の場合、ダブルストライク印刷の方が強調印刷よりも高速であることがあります。

印刷コマンド設定

[プロパティの設定] ダイアログボックスの **[印刷]** タブには、プリンタ コマンドシーケンスを指定する設定があります。



次の表は、これらの設定について説明しています。

設定	説明
プリンタ コード` の初期化を有効にする:	クエリーが開始および停止するたびに、該当するボックスに入力したコマンドシーケンスがプリンタに送信されます。
アラーム優先度範囲の代替色印刷を有効にする	このオプションを設定すると、指定したアラーム優先度範囲で使用する代替色および標準色のコマンドが渡されます。
アラーム優先度始点のテキストを変更する	少ない数値（優先度の高いアラーム）から大きい数値（優先度の低いアラーム）までのアラームの優先度範囲を入力します。
代替色	アラーム優先度が指定範囲内にあるとき、毎回 1 行のアラーム テキストをプリンタに送信するプリンタ コマンドシーケンスがプリンタに送信されます。
標準色	アラーム優先度が指定範囲内にあるとき、毎回 1 行のアラーム テキストをプリンタに送信するプリンタ コマンドシーケンスはプリンタに送信されません。

プリンタ コマンドシーケンスの例

Okidata Microline 395C カラー ドットマトリックス プリンタのプリンタ コマンドシーケンスの例を以下に示します。このコマンドシーケンスの詳細については、プリンタに付属するドキュメントを参照してください。

プリンタ初期化コマンド:

CAN ESC @ ESC M ESC SI CR

コマンドの要素の説明を以下に示します。

- **CAN** – プリンタの内部バッファに残っているデータをキャンセルします。
- **ESC @** – プリンタをメニュー デフォルトに設定します。
- **ESC M** – プリンタを **12 CPI (Elite)** に設定します。
- **ESC SI** – プリンタを圧縮に設定して **20 CPI** を取得します。
- **CR** – プリンタ ヘッドを左マージンに移動します。

印刷色を赤に設定する代替色コマンド：

```
ESC r 1
```

印刷色を黒に設定する標準色コマンド：

```
ESC r 0
```

プリンタ コマンドエスケープシーケンス構文

各プリンタ コマンドエスケープシーケンスは、**1** つ以上のバイトで構成されます。これらのバイトの詳細については、プリンタのドキュメントを参照してください。プリンタが **ESC/P or ESC/P2** 構文を解析できる場合、バイトの詳細を参照できます。これらの構文に関する情報は、インターネットで公開されています。

一般的に、プリンタ コマンドエスケープシーケンス テキストの長さに関する情報は提供されていません。パーサは、テキスト トークンのストリームを解釈し、連続するバイト値のシーケンスに変換します。その後、バイト値は、適切なとき（プリンタの初期化時やアラームのプリンタ設定を変更および復元する必要があるときなど）にプリンタにそのまま送信されます。

プリンタ コマンドエスケープシーケンスをテキストとして指定する場合、各バイト文字はスペース文字で区切る必要があります。スペース文字は、トークンが開始および終了する場所をパーサに示します。たとえば、プリンタのドラフトモードを **ESCx0** でアクティブにすることができる場合、以下のコマンドシーケンスを入力します。

```
ESC x 0
```

スペース文字をプリンタに印刷する場合は、そのバイトを表す略記 (**SP**) を使用します。

注: パーサでは大文字と小文字が区別されるので、すべての略記は大文字で入力する必要があります。たとえば、**esc** や **Esc** ではなく、**ESC** を使用します。

以下の種類のアイテムをコマンドシーケンスに含めることができます。

- 制御文字の略記名
- 数値、文字、または印刷可能なその他の文字
- **ASCII** 文字の **10** 進数
- **ASCII** 文字の **16** 進数

16 進数または **10** 進数の制御文字および印刷可能文字のリストについては、公開されている **ASCII** テーブルを参照してください。

制御文字の名前の入力

制御文字はキーボードから容易に入力できないので、制御文字の名前を入力できます。制御文字名のリストについては、公開されている **ASCII** テーブルの **Abbr** カラムを参照してください。一般的に **ASCII** テーブルに一覧表示されていない唯一の文字は特殊文字です。特殊文字は、**SP** として入力する必要があります。

注: パーサは、一般的な C 言語エスケープ文字 (\r、\n、\t など) を処理しません。

印刷可能文字の入力

ボックスに入力して印刷可能文字を入力します。唯一の例外はスペース文字です (16 進数値は 0x20、10 進数値は 32)。スペース文字は SP として入力する必要があります。文字の値は、ASCII テーブルで表現されている番号として解釈されます。この番号はプリンタに送信される値です。

印刷可能文字のリストについては、公開されている ASCII テーブルを参照してください。

文字の 10 進数値の入力

ボックスに入力して文字の 10 進数値を入力します。入力できる最大の 10 進数値は 255 です (16 進数値は 0xFF)。大きすぎる 10 進数値 (497 など) を入力しても警告は表示されず、255 が返されます。ネーティブ数値はサポートされません。

文字の 16 進数値の入力

ボックスに入力して文字の 16 進数値を入力します。有効な値の範囲は 0x00 ~ 0xFF です。

印刷するアラームの設定

Alarm Printer はアラーム メモリをクエリーし、印刷またはログ ファイルに保存するレコードを選択します。クエリーは、以下の内容に基づいて内部アラーム メモリからレコードを選択します。

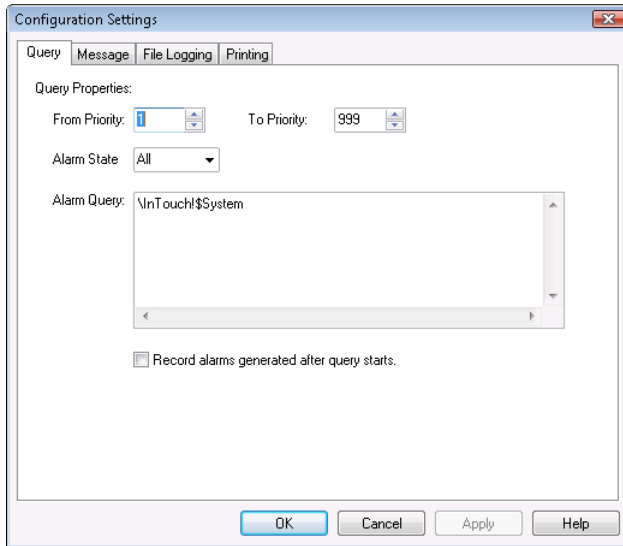
- アラーム優先度
- 現在のアラーム状況 (未確認/確認済み)
- アラーム グループ メンバーシップ

各アラームには、アラームの重要度を表す優先度番号が割り当てられています。アラーム優先度範囲は 1~999 です。最も重要度の高いアラームには 1 の優先度が割り当てられます。最も重要度の低いアラームには 999 の優先度が割り当てられます。

ネットワークまたはプリンタの接続に失敗すると、Alarm Printer では、すべてのアラームの再印刷は行われず、接続が失敗する前に印刷されていないアラームのみが印刷されます。

印刷するアラームを設定するには

1. Alarm Printer ユーティリティを開きます。以下の操作を行います。
 - a. WindowMaker の [ツール] ビューで、[アプリケーション] を展開します。
 - b. [Alarm Printer] をダブルクリックします。
2. メニュー バーで、[設定] をクリックします。[プロパティの設定] ダイアログ ボックスが表示されます。
3. [クエリー] タブをクリックします。



4. [優先度始点] ボックスに、優先度が最も高いアラーム値（1～999）を入力します。
5. [優先度終点] ボックスに、優先度が最も低いアラーム値（1 ～ 999）を入力します。
6. [アラーム状況] リストで、アラーム状況をクリックします。

クリック対象	表示する内容
All	すべてのアラーム
Ack	確認されたアラームのみ
Unack	未確認のアラームのみ

7. [アラーム クエリー] ボックスに、1つまたは複数のアラーム クエリーを入力します。1つまたは複数のアラーム プロバイダとグループを指定できます。複数のクエリーを区切るにはスペースを使用します。
8. クエリーが開始した後で発生したアラームのみを含めるには、[クエリー開始後アラームを記録する] チェック ボックスを選択します。アラーム メモリに存在し、Alarm Printer がクエリーを開始する前にトリガされたアラーム レコードは無視されます。
9. [OK] をクリックします。

印刷とファイル出力の形式の設定

選択した各オプションは、印刷出力で個別のフィールドとして表示されます。指定されたフィールド最大長を超えるデータを含むフィールドは、指定された制限に切り詰められます。

アラーム印刷とファイル出力の形式を設定するには

1. Alarm Printer ユーティリティを開きます。以下の手順を実行します。
 - a. WindowMaker の [ツール] ビューで、[アプリケーション] を展開します。
 - b. [Alarm Printer] をダブルクリックします。
2. メニュー バーの [設定] をクリックします。[プロパティの設定] ダイアログ ボックスが表示されます。

3. [メッセージ] タブをクリックします。

4. [日付/時間] 領域で、[日付] チェック ボックスをオンにし、リストから日付形式を選択します。一覧表示される日付形式には、以下のコンポーネントが含まれています。

オプション	説明
DD	日付を表す 2 桁の数 (01 ~ 31)
MM	月を表す 2 桁の数 (01 ~ 12)
YY	年を表す下 2 桁の数
YYYY	年を表す 4 桁の数
MMM	月を表す 3 文字の略語

5. [時間] チェック ボックスをオンにし、リストから時間形式を選択します。一覧表示された形式には、以下のコンポーネントが含まれます。

オプション	説明
AP	AM/PM 時間形式を選択します。たとえば、午後 3 時は 15:00 と表示されます。AP が指定されていない場合、時間はデフォルトで 24 時間制で表示されます。たとえば、午後 3 時は 15:00 と表示されます。

オプション 説明	
HH	アラーム/イベントが発生した時間を表す 2 桁の数 (00 ～ 23 または 01 ～ 12)
MM	アラーム/イベントが発生した分を表す 2 桁の数 (00 ～ 59)
SS	アラーム/イベントが発生した秒を表す 2 桁の数 (00 ～ 59)
SSS	アラーム/イベントが発生したミリ秒を表す 3 桁の数

6. アラームの開始時刻に従った、アラーム レコードにアラームが表示される順序を選択します。

オプション 説明	
OAT	(元のアラーム時刻) アラームが開始した日付/時刻スタンプです。
LCT	(最終変更時刻) アラームのインスタンスに対して発生した最新の状態変化 (アラームの開始、サブステートの変化、平常への復帰、または確認) の日付/時刻スタンプです。
LCT (確認時には OAT)	(最終変更時刻、ただし ACK 時には元のアラーム時刻) - アラームが未確認の間は LCT、確認された後は OAT が使用されます。

7. 印刷するアラーム情報を選択します。

オプション	説明
アラーム状態	アラーム状態を印刷します (未確認、確認など)。
アラーム クラス	アラーム クラスを印刷します (VALUE、DEV、ROC など)。
アラーム タイプ	アラーム タイプを印刷します (HIHI、LO、MAJDEV など)。
優先度	アラーム優先度 (1 ～ 999) を印刷します。
7.1 デフォルト (ボタン)	InTouch バージョン 7.1 用の Alarm Printer ユーティリティのデフォルト設定を選択します。
7.11 デフォルト (ボタン)	InTouch バージョン 7.11 用の Alarm Printer ユーティリティのデフォルト設定を選択します。
後に続く空白を削除	実際のフィールド値がそのフィールドに設定されている値より短い場合、印刷フィールドから末尾の余分なスペースを削除します。
最小カラム スペース	より多くのフィールドが印刷されたページに適合するように、カラム間のスペースを削減します。

オプション	説明
アラーム名	アラーム名（タグ）を印刷します。[長さ] ボックスに、アラーム名に使用できる文字数を入力します（半角 64 文字まで）。
グループ名	アラームのグループ名を印刷します。[長さ] ボックスに、アラーム グループ名に使用できる文字数を入力します（半角 64 文字まで）。
アラーム プロバイダ	アラーム プロバイダの名前を印刷します。[長さ] ボックスに、アラーム プロバイダ名に使用できる文字数を入力します（半角 64 文字まで）。
アラーム値	タグの値を印刷します。[長さ] ボックスに、アラーム値に使用できる文字数を入力します（半角 32 文字まで）。
しきい値	タグのアラーム制限を印刷します。[長さ] ボックスに、アラームしきい値に使用できる文字数を入力します（半角 32 文字まで）。目的の精度レベルを提供するために十分な文字数である必要があります。
オペレータ ノード	アラーム状況に関連付けられたオペレータ ノードを印刷します。[長さ] ボックスにオペレータのノードで許可する文字数（最大 64 文字）を入力します。 ターミナル サービス環境では、これは該当するオペレータがターミナル サービス セッションを確立したクライアント コンピュータの名前です。ノード名を取得できない場合、ノードの IP アドレスが代わりに使用されます。
オペレータ名	アラーム状況に関連付けられたオペレータ ノードを印刷します。[長さ] ボックスに、オペレータ名に使用できる文字数を入力します（半角 16 文字まで）。
オペレータ フルネーム	ログオンしたオペレータのフル ネームを印刷します。
オペレータ ドメイン	アラーム状況に関連付けられているログオンしたオペレータ ドメインを印刷します。
コメント	タグに関連付けられたアラーム コメントを印刷します。[長さ] ボックスに、コメントに使用できる文字数を入力します（半角 131 文字まで）。
タグ コメント	タグのコメントを印刷します。
ユーザー 1	アラームに対応するユーザー定義番号 1 の数値を印刷します。
ユーザー 2	アラームに対応するユーザー定義番号 2 の数値を印刷します。

オプション	説明
ユーザー 3	アラームに関連付けられたユーザー定義文字列プロパティの文字列値を印刷します。最大文字数は 131 です。

8. [適用] をクリックします。

アラームのログ ファイルの設定

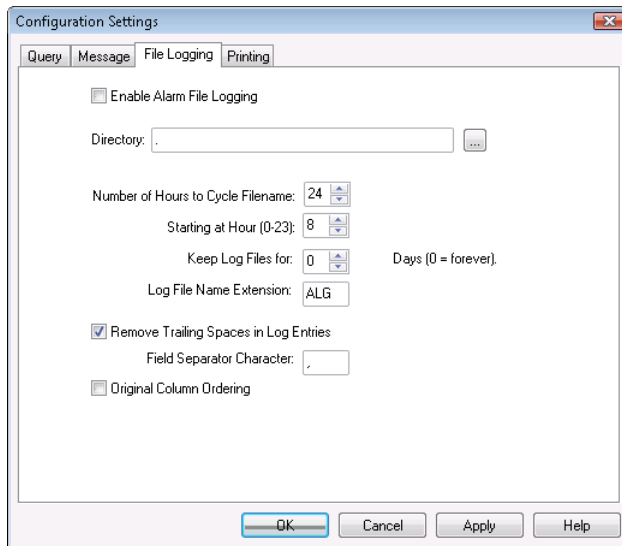
アラーム レコードは、ファイルにログ記録できます。デフォルトでは、Alarm Printer では以下の命名規則に基づいてログ ファイルに名前が割り当てられます。

YYMMDDHH.ALG

表記法	説明
YY	ログ ファイルが作成された年を表す下 2 桁
MM	ログ ファイルが作成された月を表す 2 桁の数 (01 ~ 12)
DD	ログ ファイルが作成された日を表す 2 桁の数 (01 ~ 31)
HH	ログ ファイルが作成された時間を表す 2 桁の数 (00 ~ 23)

アラーム レコード ログを設定するには

- Alarm Printer ユーティリティを開きます。以下の手順を実行します。
 - WindowMaker の [ツール] ビューで、[アプリケーション] を展開します。
 - [Alarm Printer] をダブルクリックします。
- メニューバーの [設定] をクリックします。[プロパティの設定] ダイアログ ボックスが表示されます。
- [ファイル ログ] タブをクリックします。



4. アラーム レコードをログ ファイルに保存するには、[アラーム ファイル ログを有効にする] チェック ボックスをオンにします。
5. [ディレクトリ] ボックスに、パスを入力するか、アラーム ログ ファイルを保存するフォルダの場所を参照します。
6. [アラーム ログ ファイル名作成サイクル] ボックスで、アラーム レコードを個別のログ ファイルに保存するために適切な時間数を入力します。

有効なエントリは 1 ～ 24 です。InTouch アプリケーションを継続的に実行する場合は、長さの等しい日次ログ ファイルを作成するファイル ログ間隔を選択します。たとえば、[アラーム ログ ファイル名作成サイクル] を 6 に設定すると、長さの等しい 4 つの日次ログ ファイルが作成されます。

7. [記録開始時間] ボックスに、毎日ファイルへのアラーム レコードのログ記録を開始する時間を入力します。

有効なエントリは 0 ～ 23 です。

たとえば、ある石油精製所では 1 日 3 交代で稼動しています。最初の交代は、6:00 に開始します。管理者は、交代ごとにアラーム レコードが記録されることを求めています。この場合、[アラーム ログ ファイル名作成サイクル] オプションに 8 を入力します。[記録開始時間 (0-23)] オプションに 6 を入力します。Alarm Printer では、06:00 ～ 14:00、14:00 ～ 22:00、22:00 ～ 06:00 からのログ ファイルが 3 つ作成されます。

8. [ログ ファイル保存期間] ボックスに、ログ ファイルを保持する日数を入力します。

Alarm Printer では、ログ ファイルは現在の日付から指定した日数だけ保持されます。保存期間を過ぎたログ ファイルは削除されます。ログ ファイルを無制限に保存するには、0 を入力します。

9. [ログ ファイル名拡張子] ボックスで、デフォルトの ALG ファイル名拡張子を受け入れるか、別の 3 文字の拡張子をログ ファイルに割り当てます。

.csv 拡張子を使用すると、ログ ファイルを Excel またはメモ帳に直接インポートできます。

10. ログ ファイル内のエントリの末尾のスペースを削除するには、[後続くスペースを削除] チェック ボックスをオンにします。

ログ ファイル内の各レコードの最後にフィールド区切り文字を指定することもできます。

11. アラーム表示と同じ順序を保持してファイル レコードをログ記録するには、**[オリジナル カラム順序]** チェック ボックスをオンにします。
12. **[適用]** をクリックします。

設定ファイルの保存とロード

WindowMaker から Alarm Printer を起動すると、**[Alarm Printer]** ダイアログ ボックスにデフォルト設定が表示されます。Alarm Printer の設定は、アラーム設定ファイル (.alc) に保存されます。

プリンタ設定は、アラームを印刷する前にロードできるファイルに保存できます。

特定のアラーム設定ファイルは、ランタイムでコマンドラインから開いた場合、または *.alc ファイル名をダブルクリックして開いた場合に表示できます。

新しい Alarm Printer 設定ファイルを作成するには

1. Alarm Printer ユーティリティを開きます。以下の手順を実行します。
 - a. WindowMaker の **[ツール]** ビューで、**[アプリケーション]** を展開します。
 - b. **[Alarm Printer]** をダブルクリックします。
2. **[ファイル]** メニューで、**[新規]** をクリックし、Alarm Printer をデフォルト値で表示します。
3. メニュー バーの **[設定]** をクリックします。**[プロパティの設定]** ダイアログ ボックスが表示されます。
4. アラームを設定します。
5. **[ファイル]** メニューで、**[保存]** を選択します。

既存の Alarm Printer 設定ファイルを編集するには

1. **[ファイル]** メニューで、**[開く]** を選択します。
2. 編集する Alarm Printer 設定ファイルを選択します。
3. ファイルを編集します。
4. **[ファイル]** メニューで、**[保存]** を選択します。既存ファイルを変更しないで新しいファイルに変更を保存するには、**[名前を付けて保存]** を選択します。

Alarm Printer でのアラームの印刷の概要

各クエリーは、現在開いている Alarm Printer 設定ファイル (.alc) に指定されているすべてのアラームをログ記録します。ファイルが指定されていない場合、Alarm Printer の設定中に現在選択されている設定が使用されます。

Alarm Printer では、複数のクエリーを実行できます。各クエリーは異なるパラメータを使用し、Alarm Printer の個別のインスタンスと関連付けられます。Alarm Printer の 2 つのインスタンスが同じクエリーを実行している場合、エントリは重複されます。

Alarm Printer の実行中は、クエリーを手動で開始または停止できます。印刷を有効にしていることを確認してください。

アラーム クエリーを開始するには



- **[クエリー]** メニューで、**[開始/停止]** をクリックします。

アラーム クエリーを停止するには



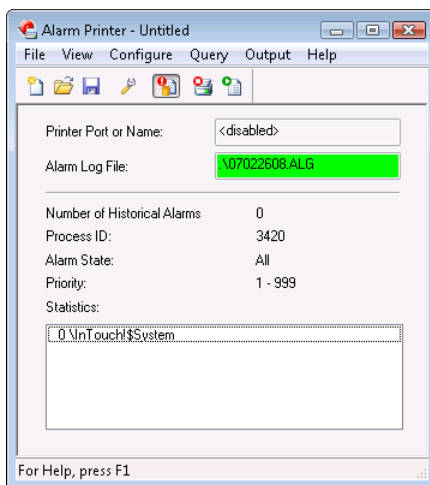
- [クエリー] メニューで、[開始/停止] をクリックします。

アラームのファイルへのログ

Alarm Printer を使用すると、アラーム レコードをファイルにログ記録できます。現段階で、[プロパティの設定] ダイアログ ボックスですでにロギングが設定されている状態となっています。

アラームをファイルにログ記録するには

1. Alarm Printer ユーティリティを開きます。
 - a. WindowMaker の [ツール] ビューで、[アプリケーション] を展開します。
 - b. [Alarm Printer] をダブルクリックします。
2. 必要に応じて、クエリーを設定し、ロギング用にアラーム レコードを収集します。
3. [ロギングの切り替え] ボタンをクリックします。
4. アラーム クエリーを実行します。



クエリーで収集されるアラーム レコードは、設定済みのログ ファイルに書き込まれます。

特定の設定での Alarm Printer の起動

システムの起動時に次のコマンドをバッチ ファイルで実行することによって、Alarm Printer を自動的に起動し、特定のファイルを開くことができます。

```
ALMPRT.EXE MYQUERY.ALC
```

ここで、MYQUERY.ALC は、開かれる Alarm Printer 設定ファイルの名前です。.exe ファイル名拡張子はオプションです。

バッチ ファイルが InTouch HMI がインストールされているフォルダに切り替わることを確認します。

システムの予期しないシャットダウンや再起動によるクエリー データの消失を防ぐために、次のコマンドをバッチ ファイルから実行し、Alarm Printer を自動的に起動して、特定のクエリーを自動的に実行することができます。

```
ALMPRT.EXE -q MYQUERY.ALC
```

コマンドで -q を使用すると、システムの起動時にクエリーが自動的に実行されます

スクリプトを使用した Alarm Printer の制御

スクリプトで Alarm Printer 関数を使用して、アラーム印刷を制御できます。

Alarm Printer 関数は、関数が正常に完了したかどうかを示す整数のエラー コードを返します。以下の表は、エラー コードを示しています。

エラー コード	エラー メッセージ
0	成功
1	インスタンスが見つからないか、実行中ではありません
2	インターフェイスが初期化されていません
3	仮想メモリにアクセスできませんでした
4	無効なエラー コードです
5	すでに実行中のインスタンスが多すぎます
6	結果文字列が長くなりすぎます
7	無効なインスタンス インデックスが関数に渡されました
8	Alarm Printer アプリケーションにメッセージをポストできませんでした
9	Alarm Printer アプリケーションからの応答を待機できませんでした
20	「優先度終点」は「優先度始点」に等しいか、大きい値である必要があります。
21	無効な優先度値です
22	無効なアラーム状況です
23	クエリーが実行しているためコマンドを実行できませんでした
24	クエリー文字列が有効ではありません
25	無効なクエリー処理状況です
26	無効な印刷状況セレクタです
27	Alarm Printer ウィンドウが受け取るコマンドが認識されません
28	クエリーが開始できませんでした

Alarm Printer インスタンスまたはクエリーの停止および開始

Alarm Printer インスタンスとアラーム クエリーを開始したり、停止したりするには、以下の関数を使用します。

- [APUStartInstance\(\) 関数](#)
- [APUStartQuery\(\) 関数](#)
- [APUStopInstance\(\) 関数](#)
- [APUStopQuery\(\) 関数](#)

APUStartInstance() 関数

設定ファイルに指定された値で、Alarm Printer のインスタンスを最小化状態で開始します。

カテゴリ

表示

構文

```
[Result=] APUStartInstance(sFilePath, iTagInstance);
```

引数

sFilePath

設定ファイルのフルパス（入力文字列）

iTagInstance

整数型タグ変数関数は、正常に実行されると、インスタンス番号を返します。

備考

Alarm Printer の最大 16 個のインスタンスを開始できます。この関数は、インスタンス番号（0 ～ 15）を iTagInstance パラメータに書き込みます。新しい Alarm Printer インスタンスが開始するとき、インスタンス番号は増加します。

このインスタンス番号は、Alarm Printer インスタンスを識別するために、他の Alarm Printer 関数によって使用されます。

この関数は 0 を返します。エラーの場合は、エラー コードを返します。

Alarm Printer プログラムは、アラームの処理をアラーム メモリから自動的に開始しません。

APUStartQuery() 関数を使用して、インスタンスに対するアラーム メモリのデータの処理を開始します。

例

```
Status = APUStartInstance("c:\MyAlarmCfg\Area1Alarms.alc", Inst);
```

参照項目

APUStartQuery()、**APUStopInstance()**、**APUStopQuery()**

APUStartQuery() 関数

アラーム メモリから処理されるレコードの日付および時間しきい値を設定し、クエリーを開始します。

カテゴリ

表示

構文

```
[Result=] APUSstartQuery(iInstance, iYear, iMonth, iDay, iHour, iMinute);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

iYear

年を表す数

iMonth

月を表す数

iDay

日を表す数

iHour

時間を表す数

iMinute

秒を表す数

備考

クエリーがすでに実行している際、別のクエリーを開始しようとするエラーが発生します。

すべての日付と時間の値を 0 に設定すると、すべてのアラームが印刷されます。これは、0 が 1900 年 1 月 1 日の 00:00 時と解釈されるためです。指定された時間と日付はローカル時間です。年を表す値が -1 の場合、日付はコマンドが処理される現在の時間に設定されます。

整数型のエラー コードを返します。

例

```
Status = APUSstartQuery(Inst,2007,4,16,22,12);
```

参照項目

APUSstartInstance()、**APUSstopInstance()**、**APUSstopQuery()**

APUSstopInstance() 関数

Alarm Printer の指定インスタンスを停止します。印刷されるレコードのさらなる追加は停止し、現在実行している印刷クエリーは停止し、プログラムのインスタンスは閉じられます。

カテゴリ

表示

構文

```
[Result=] APUSstopInstance(iInstance);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

備考

整数型のエラー コードを返します。

例

```
Status = APUStopInstance(5);
```

参照項目

APUStartInstance()、**APUStartQuery()**、**APUStopQuery()**

APUStopQuery() 関数

指定のインスタンスにクエリーの実行を停止するよう要求します。アプリケーションの実行は継続されますが、クエリーは処理されません。**APUStartQuery()** を呼び出すと、インスタンスがクエリーを開始できます。

カテゴリ

表示

構文

```
[Result=] APUStopQuery(iInstance);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

備考

整数型のエラー コードを返します。

例

```
Status = APUStopQuery(5);
```

参照項目

APUStartInstance()、**APUStartQuery()**、**APUStopInstance()**

アラーム クエリー情報のクエリー

Alarm Printer 設定ファイル内で設定されたタグ変数優先度と値に基づいてクエリー パラメータを取得するには、以下の関数を使用します。

- [APUGetAlarmGroupText\(\) 関数](#)
- [APUGetQueryFromPriority\(\) 関数](#)
- [APUGetQueryToPriority\(\) 関数](#)
- [APUGetConfigurationFilePath\(\) 関数](#)
- [APUGetPrinterJobCount\(\) 関数](#)
- [APUGetQueryAlarmState\(\) 関数](#)
- [APUGetQueryProcessingState\(\) 関数](#)

APUGetAlarmGroupText() 関数

アラーム クエリー アラーム グループ テキストを取得します。

カテゴリ

表示

構文

```
[Result=] APUGetAlarmGroupText(iInstance, sTagGroup);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

sTagGroup

テキスト - アラーム グループ

備考

初期アラーム グループ テキストは、指定したインスタンスを持つ Alarm Printer が使用している .alc 設定ファイルから読み取られます。アラーム グループ テキストは、sTagGroup パラメータの InTouch メッセージ型タグに渡されます。

整数型のエラー コードを返します。

例

TagGroup メッセージ型タグには、関数が実行された後で、以下の値が含まれる可能性があります。

```
\intouch!$system
```

```
Status = APUGetAlarmGroupText(Inst,TagGroup);
```

参照項目

APUGetConfigurationFilePath()、**APUGetPrinterJobCount()**、**APUGetQueryAlarmState()**、**APUGetQueryFromPriority()**、**APUGetQueryProcessingState()**、**APUGetQueryToPriority()**

APUGetQueryFromPriority() 関数

クエリーの「優先度始点」の値を取得します。

カテゴリ

表示

構文

```
[Result=] APUGetQueryFromPriority(iInstance, iTagPriority);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

iTagPriority

「優先度始点」の値を受け取る整数型タグ変数

備考

初期優先度は、指定インスタンスを持つ Alarm Printer が使用している .alc 設定ファイルから読み取られます。「優先度始点」の値は、iTagPriority パラメータの InTouch 整数型タグ変数に渡されます。

整数型のエラー コードを返します。

例

この例では、FromPri は、「優先度始点」の値を含む整数型タグ変数です。たとえば、1 です。

```
Status = APUGetQueryFromPriority(Inst, FromPri);
```

参照項目

APUGetAlarmGroupText()、**APUGetConfigurationFilePath()**、**APUGetPrinterJobCount()**、**APUGetQueryAlarmState()**、**APUGetQueryProcessingState()**、**APUGetQueryToPriority()**

APUGetQueryToPriority() 関数

クエリーの「優先度終点」を取得します。

カテゴリ

表示

構文

```
[Result=] APUGetQueryToPriority(iInstance,iPriority);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

iPriority

「優先度終点」の値を受け取る整数型タグ変数

備考

APUGetQueryToPriority() 関数を含むスクリプトと同時に、別のクエリーを実行することはできません。「優先度終点」の値は、関数の ***iPriority*** パラメータに書き込まれます。これは整数型タグ変数です。

整数型のエラー コードを返します。

例

整数型タグ変数 **ToPri** は、「優先度終点」の値を受け取ります。たとえば、999 です。

```
Status = APUGetQueryToPriority(Inst,ToPri);
```

参照項目

APUGetAlarmGroupText()、**APUGetConfigurationFilePath()**、**APUGetPrinterJobCount()**、**APUGetQueryAlarmState()**、**APUGetQueryFromPriority()**、**APUGetQueryProcessingState()**

APUGetConfigurationFilePath() 関数

クエリーに使用される .alc 設定ファイルのフル ファイル パスを返します。

カテゴリ

表示

構文

```
[Result=] APUGetConfigurationFilePath(iInstance, sTagFilePath);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

sTagFilePath

Alarm Printer インスタンスが使用している設定ファイルのファイルパスの名前を取得するメッセージ型タグ変数

備考

ファイルパス テキストは、この関数の `sTagFilePath` パラメータとして指定できるメッセージ型タグ変数に返されます。

整数型のエラー コードを返します。

例

`CfgFilePath` メッセージ型タグ変数は、`Inst` 整数型タグ変数に含まれる Alarm Printer インスタンスと関連付けられている設定ファイルのファイルパスを受け取ります。たとえば、`c:\MyAlarmCfg\Area1Alarms.alc` などです。

```
Status = APUGetConfigurationFilePath(Inst, CfgFilePath);
```

参照項目

APUGetAlarmGroupText()、**APUGetPrinterJobCount()**、**APUGetQueryAlarmState()**、**APUGetQueryFromPriority()**、**APUGetQueryProcessingState()**、**APUGetQueryToPriority()**

APUGetPrinterJobCount() 関数

このインスタンスによって使用されるプリンタの最新の Windows プリンタ状況ジョブ カウントを返します。

カテゴリ

表示

構文

```
[Result=] APUGetPrinterJobCount(iInstance, iTagCount);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

iTagCount

カウント値を受け取る整数型タグ変数

備考

この関数は、`iTagCount` パラメータでカウント値を返します。これは整数型タグ変数です。

クエリーが実行していない限り、結果は現在の内容ではありません。アラームが印刷されていない限り、結果は現在の内容ではありません。ジョブ カウントは通常、プリンタが最初に開かれたときに更新され、またアラーム行が印刷されるたびに更新されます。

返されるジョブ カウント値は Windows プリンタに対してのみ有効であり、パラレル ポートまたはシリアル ポートに関連付けられているプリンタに対してはあまり意味を持ちません。

整数型のエラー コードを返します。

例

`PJCount` は、指定インスタンスからのカウント値を受け取る整数型タグ変数です。

```
Status = APUGetPrinterJobCount(Inst, PJCount);
```

参照項目

APUGetAlarmGroupText()、**APUGetConfigurationFilePath()**、**APUGetQueryAlarmState()**、**APUGetQueryFromPriority()**、**APUGetQueryProcessingState()**、**APUGetQueryToPriority()**

APUGetQueryAlarmState() 関数

クエリーのアラーム状態を返します。

カテゴリ

表示

構文

```
[Result=] APUGetQueryAlarmState(iInstance, iTagState);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

iTagState

指定インスタンスに関連付けられたクエリーのアラーム状態を受け取る整数型タグ変数。値の意味は以下のとおりです。

0 = 全て

1 = 確認

2 = 未確認

備考

初期アラーム状態は .alc ファイルから読み取られます。この関数は、整数型タグである関数の *iTagState* パラメータでアラーム状態を返します。

整数型のエラー コードを返します。

例

AlmState は、0、1、または 2 を含む整数型タグ変数です。

```
Status = APUGetQueryAlarmState(Inst,AlmState);
```

参照項目

APUGetAlarmGroupText()、**APUGetConfigurationFilePath()**、**APUGetPrinterJobCount()**、**APUGetQueryFromPriority()**、**APUGetQueryProcessingState()**、**APUGetQueryToPriority()**

APUGetQueryProcessingState() 関数

アラーム クエリー処理の状況を返します。

カテゴリ

表示

構文

```
[Result=] APUGetQueryProcessingState(iInstance, iTagState);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

iTagState

関数から処理状況を受け取る整数型タグ変数。値の意味は以下のとおりです。

0 = 停止

1 = 開始

備考

この関数は、整数型タグ変数である **iTagState** パラメータに整数値を返します。

整数型のエラー コードを返します。

例

ProcState は整数型タグ変数であり、クエリーが実行していない場合は 0、クエリーが実行している場合は 1 に設定されます。

```
Status = APUGetQueryProcessingState(Inst, ProcState);
```

参照項目

APUGetAlarmGroupText()、**APUGetConfigurationFilePath()**、**APUGetPrinterJobCount()**、**APUGetQueryAlarmState()**、**APUGetQueryFromPriority()**、**APUGetQueryToPriority()**

インスタンス情報のクエリー

Alarm Printer インスタンスの現在の状況を返すには、以下の関数を使用します。

- [APUFindAlarmGroupInstance\(\) 関数](#)
- [APUFindFileInstance\(\) 関数](#)
- [APUFindPrinterInstance\(\) 関数](#)
- [APUGetInstanceCount\(\) 関数](#)
- [APUIsInstanceUsed\(\) 関数](#)

APUFindAlarmGroupInstance() 関数

指定アラーム グループ文字列を使用して、Alarm Printer の最初のインスタンスを返します。

カテゴリ

表示

構文

```
[Result=] APUFindAlarmGroupInstance(sGroup, iInstance);
```

引数

sGroup

インスタンス間で検出されるアラーム グループの名前です。

iInstance

指定グループ名を使用する検出されたインスタンスの値を受け取る整数型タグ変数

備考

この関数は、**iInstance** パラメータとして整数型タグ変数にインスタンス番号を返します。初期アラームグループ文字列は **alc** ファイルから読み取られます。インスタンスが見つからない場合、関数はエラーコード（使用可能なインスタンスなし）として 1 を返し、整数型タグ変数パラメータに 0 を書き込みます。

整数型のエラー コードを返します。

例

FoundInstance は、クエリーとして \$System を使用する検出された最初のインスタンスの番号を受け取る整数型タグ変数です。

```
Status = APUFindAlarmGroupInstance("$System", FoundInstance);
```

参照項目

APUFindFileInstance()、**APUFindPrinterInstance()**、**APUGetInstanceCount()**、**APUIsInstanceUsed()**

APUFindFileInstance() 関数

指定の .alc 設定ファイルを使用して、Alarm Printer の最初のインスタンスを見つけます。

カテゴリ

表示

構文

```
[Result=] APUFindFileInstance(sFilePath, iInstance);
```

引数

sFilePath

検出されるインスタンスを持つ .alc 設定ファイルのパス

iInstance

インスタンスの番号を受け取る整数型タグ変数

備考

この関数は、iInstance パラメータとして整数型タグ変数にインスタンス番号を返します。Alarm Printer の必要なインスタンスを初期に取得するには、この関数を使用します。ファイルパス文字列一致では、大文字と小文字は区別されません。

インスタンスが見つからない場合、関数はエラー コード（使用可能なインスタンスなし）として 1 を返し、整数型タグ変数パラメータに 0 を書き込みます。

整数型のエラー コードを返します。

例

InstFound は、設定ファイル c:\MyAlarmCfg\Area1Alarms.alc を使用する最初のインスタンスの番号を受け取る整数型タグ変数です。

```
Status = APUFindFileInstance("c:\MyAlarmCfg\Area1Alarms.alc", InstFound);
```

参照項目

APUFindAlarmGroupInstance()、**APUFindPrinterInstance()**、**APUGetInstanceCount()**、**APUIsInstanceUsed()**

APUFindPrinterInstance() 関数

指定のプリンタ名またはポートを使用して、Alarm Printer の最初のインスタンスを検出します。

カテゴリ

表示

構文

```
[Result=] APUFindPrinterInstance(sPrinter, iInstance);
```

引数

sPrinter

検出されるインスタンスを持つプリンタの名前です。

iInstance

インスタンスの番号を受け取る整数型タグ変数

備考

この関数は、***iInstance*** パラメータとして整数型タグ変数にインスタンス番号を返します。Alarm Printer の必要なインスタンスを初期に取得するには、この関数を使用します。プリンタ名は保存され、.alc ファイルから読み取られます。プリンタ名文字列一致では、大文字と小文字は区別されません。インスタンスが見つからない場合、関数はエラー コード（使用可能なインスタンスなし）として **1** を返し、整数型タグ変数パラメータに **0** を書き込みます。

整数型のエラー コードを返します。

例

FoundInst は関連付けられた .alc ファイルでプリンタ名として LPT1 を使用する最初のインスタンスの番号を受け取る整数型タグ変数です。

```
Status = APUFindPrinterInstance("LPT1", FoundInst);
```

参照項目

APUFindAlarmGroupInstance()、**APUFindFileInstance()**、**APUGetInstanceCount()**、**APUIsInstanceUsed()**

APUGetInstanceCount() 関数

Alarm Printer の実行中のインスタンスの数を返します。最大 16 個のインスタンスです。

カテゴリ

表示

構文

```
[Result=] APUGetInstanceCount(iCount);
```

引数

iCount

インスタンスの数を受け取る整数型タグ変数

備考

この関数は、パラメータとして整数型タグ変数にインスタンス数を返します。同時に実行中の最初の 16 個を超えるインスタンスはどれも動的に制御できません。また、ステータスを取得することもできません。

整数型のエラー コードを返します。

例

iCount は整数型のタグ変数です。ランタイム値が 7 は、実行している Alarm Printer インスタンスが現在 7 個あるという意味です。

```
Status = APUGetInstanceCount( iCount );
```

参照項目

APUFindAlarmGroupInstance()、**APUFindFileInstance()**、**APUFindPrinterInstance()**、**APUIsInstanceUsed()**

APUIsInstanceUsed() 関数

インスタンスが現在使用中であることを示す論理値を返します。

カテゴリ

表示

構文

```
[Result=] APUIsInstanceUsed(iInstance);
```

引数

iInstance

Alarm Printer のインスタンスの数 (0 ～ 15)

備考

結果は 0 または 1 です。

0 = インスタンスは使用されていません。

1 = インスタンスは使用されています。

例

InUse は論理型タグ変数です。インスタンス 5 が使用中の場合は true (1) に設定され、インスタンス 5 が使用中でない場合は false (0) に設定されます。

```
InUse = APUIsInstanceUsed(5);
```

参照項目

APUFindAlarmGroupInstance()、**APUFindFileInstance()**、**APUFindPrinterInstance()**、**APUGetInstanceCount()**

プリンタ情報のクエリー

Alarm Printer のステータスを返すには、以下の関数を使用します。

- [APUGetPrinterName\(\) 関数](#)
- [APUGetPrinterStatus\(\) 関数](#)

APUGetPrinterName() 関数

このインスタンスによって使用されるプリンタの Windows プリンタ名およびポート名を返します。

カテゴリ

表示

構文

```
[Result=] APUGetPrinterName(iInstance, sTagPrinter);
```

引数

iInstance

Alarm Printer のインスタンス番号 (0 ～ 15)

sTagPrinter

指定インスタンスに関連付けられた設定のプリンタ名またはポート名を受け取るメッセージ型タグ変数

備考

この関数は、インスタンスに対してプリンタが設定されていない場合、値 **NONE** を返します。プリンタ名は保存され、.alc ファイルから読み取られます。この関数は、sTagPrinter パラメータとして、プリンタ名またはポート名をメッセージ型タグ変数に返します。

整数型のエラー コードを返します。

例

PrtName は、Alarm Printer のインスタンス 3 に関連付けられた設定のプリンタ名またはポート名を受け取るメッセージ型タグ変数です。

```
Status = APUGetPrinterName(3,PrtName);
```

参照項目

APUGetPrinterStatus()

APUGetPrinterStatus() 関数

このインスタンスによって使用される Windows プリンタの最新のステータスを返します。

カテゴリ

表示

構文

```
[Result=] APUGetPrinterStatus(iInstance, iSelector, iTagStatus);
```

引数

iInstance

Alarm Printer のインスタンス (0 ~ 15)

iSelector

以下を指定する整数値:

0 = Alarm Printer エラーのステータスを取得します。

1 = Alarm Printer 用紙切れのステータスを取得します。

2 = Alarm Printer オフラインのステータスを取得します。

3 = Alarm Printer オーバーフローのステータスを取得します。

iTagStatus

指定のインスタンス番号および iSelector パラメータの選択タイプに関連付けられたプリンタのステータスを受け取る整数型または実数型のタグ変数

備考

この関数は、iTagStatus パラメータとして整数型または実数型のタグ変数にプリンタのステータスを返します。クエリが実行していて、アラームが印刷されていない場合、結果は現在の内容ではありません。ステータスは通常、プリンタが最初に開くとき、およびアラーム行が印刷されるときに更新されます。

このステータス情報は、Microsoft または Windows のドライバ標準に基づいて、プリンタに照会されています。すべてのプリンタ製造業者が以下の標準に従っているわけではありません。したがって、すべてのプリンタでステータス情報が返されるわけではありません。

整数型のエラー コードを返します。

例

PrtStat は、インスタンス 5 に関連付けられたプリンタから「プリンタ オフライン」ステータスを受け取る整数型タグ変数です。

```
Status = APUGetPrinterStatus(5, 2, PrtStat);
```

参照項目

APUGetPrinterName()

アラーム クエリー情報の設定

Alarm Printer クエリーで使用されるパラメータを設定するには、以下の関数を使用します。

- [APUSetAlarmGroupText\(\) 関数](#)
- [APUSetQueryAlarmState\(\) 関数](#)
- [APUSetQueryFromPriority\(\) 関数](#)
- [APUSetQueryToPriority\(\) 関数](#)
- [APUSetTimeoutValues\(\) 関数](#)

APUSetAlarmGroupText() 関数

アラーム クエリー アラーム グループ テキストを設定します。

カテゴリ

View

構文

```
[Result=] APUSetAlarmGroupText(iInstance, sGroup);
```

引数

iInstance

Alarm Printer のインスタンス (0 ~ 15)

sGroup

アラーム グループ テキスト

備考

クエリーが実行中の場合、この関数は正常に実行されません。

整数型のエラー コードを返します。

例

この例では、Alarm Printer インスタンス 1 のクエリーを \InTouch!GroupA に設定します。

```
Status = APUSetAlarmGroupText(1, "\InTouch!GroupA");
```

参照項目

[APUSetQueryAlarmState\(\)](#)、[APUSetQueryFromPriority\(\)](#)、[APUSetQueryToPriority\(\)](#)、[APUSetTimeoutValues\(\)](#)

APUSetQueryAlarmState() 関数

クエリーのアラーム状況を設定します。

カテゴリ

表示

構文

```
[Result=] APUSetQueryAlarmState(iInstance, iState);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

iState

以下の値を持つ整数 :

0 = 全て

1 = 確認

2 = 未確認

備考

APUSetQueryAlarmState() 関数を使用してスクリプトを実行している間、同時にクエリーを実行することはできません。

整数型のエラー コードを返します。

例

この例では、Alarm Printer インスタンス 3 のクエリーを設定して、確認アラームのみをクエリーします。
`Status = APUSetQueryAlarmState(3, 1);`

参照項目

APUSetAlarmGroupText()、**APUSetQueryFromPriority()**、**APUSetQueryToPriority()**、**APUSetTimeoutValues()**

APUSetQueryFromPriority() 関数

アラーム クエリーの下限または優先度始点を設定します。

カテゴリ

表示

構文

```
[Result=] APUSetQueryFromPriority(iInstance, iPriority);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

iPriority

優先度始点の整数値 (1 ～ 999)

備考

クエリーが実行中の場合、この関数は正常に実行されません。

整数型のエラー コードを返します。

例

この例では、Inst 整数型タグ変数のインスタンス値に関連付けられているクエリーの「優先度始点」の値を FromPri 整数型タグ変数の値に設定します。

```
Status = APUSetQueryFromPriority(Inst, FromPri);
```

参照項目

APUSetAlarmGroupText()、**APUSetQueryAlarmState()**、**APUSetQueryToPriority()**、**APUSetTimeoutValues()**

APUSetQueryToPriority() 関数

クエリーの「優先度終点」を設定します。

カテゴリ

表示

構文

```
[Result=] APUSetQueryToPriority(iInstance, iPriority);
```

引数

iInstance

Alarm Printer のインスタンス (0 ～ 15)

iPriority

「優先度終点」の整数値は 1 ～ 999 の範囲です。指定インスタンスの同じクエリーの「優先度始点」以上に設定する必要があります。

備考

有効なアラーム優先度範囲を設定するには、「優先度終点」は「優先度始点」以上である必要があります。**APUSetQueryToPriority()** 関数を含むスクリプトは、クエリーと同時に実行できません。

整数型のエラー コードを返します。

例

この例では、インスタンス 0 から 240 に関連付けられたクエリーの「優先度終点」の値を設定します。

```
Status = APUSetQueryToPriority(0,240);
```

参照項目

APUSetAlarmGroupText()、**APUSetQueryAlarmState()**、**APUSetQueryFromPriority()**、**APUSetTimeoutValues()**

APUSetTimeoutValues() 関数

APUSetTimeoutValues() 関数は、タイムアウト間隔 (秒単位) を設定します。タイムアウト間隔は、プログラム実行中に、メモリ アクセスによって発生するエラー数、または有効応答の取得失敗が確認される回数を制御します。

デフォルトのメモリ アクセス タイムアウトは 2 秒です。デフォルトの短い応答待機時間は 10 秒です。デフォルトの長い応答待機時間は 20 秒です。

カテゴリ

表示

構文

```
[Result=] APUSetTimeoutValues(iMemory, iShort, iLong);
```


引数

iMemory

整数 - アクセス タイムアウト

iShort

短い応答待機時間（整数）

iLong

長い応答待機時間（整数）

例

```
Status = APUSetTimeoutValues(iMemory,iShort,iLong);
```

参照項目

APUSetAlarmGroupText()、**APUSetQueryAlarmState()**、**APUSetQueryFromPriority()**、**APUSetQueryToPriority()**

Alarm Printer エラーの処理

Alarm Printer エラー コードをエラー メッセージに変換するには、**APUTranslateErrorCode()** 関数を使用します。

APUTranslateErrorCode() 関数

APU 関数の 1 つによって返されるエラー コードを、エラー コードを簡単に説明する英語の文字列に変換します。

カテゴリ

表示

構文

```
[Result=] APUTranslateErrorCode(iErrorCode, sTagMessage);
```

引数

iErrorCode

通常は、他のほとんどの APU 関数によって返される整数型のエラー コードです。

sTagMessage

エラー メッセージを受け取るメッセージ型タグ

備考

この関数は、**sTagMessage** パラメータとして、エラー メッセージをメッセージ型タグに返します。不明なエラー コードが渡された場合、この関数は失敗する可能性があります。

整数型のエラー コードを返します。

例

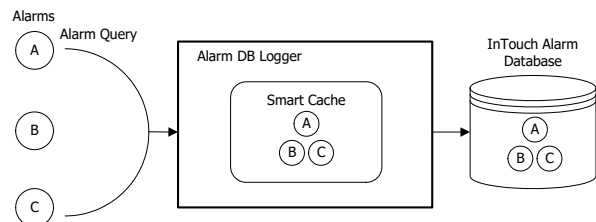
この例では、現在実行している Alarm Printer のインスタンス 15 が存在しない場合、メッセージ型タグ **errmsg** を「インスタンスがありません。」に設定します。

```
Status = APUTranslateErrorCode(APUSetAlarmGroupText(15,"$system"), ErrMsg);
```

アラーム データベースへのアラームの記録

InTouch 分散アラーム システムには、アラームやイベントをアラーム データベースにログする Alarm DB Logger ユーティリティが含まれています。

Alarm DB Logger はアラーム コンシューマです。1 つまたは複数のクエリーを使用して Alarm DB Logger を設定し、InTouch アラーム プロバイダからのアラームを選択します。クエリーで選択したアラームは、スマート キャッシュと呼ばれるメモリ キャッシュに一時的に格納されます。Alarm DB Logger は、一定の周期でスマート キャッシュの内容をアラームおよびイベント レコードとしてアラーム データベースに書き込みます。



Alarm DB Logger は自動再接続が可能です。データベースへの接続が失われた場合は、Logger は一定の周期でデータベースへの接続をチェックし、アラーム データベースとの接続が再確立されると、ログへの記録が再開されます。

サービスとして実行されているか標準のアプリケーションとして実行されているかに関係なく、Alarm DB Logger はすべてのエラーを Logger に報告します。

Alarm DB Logger Manager 用の SQL Server アカウント

Alarm DB Logger Manager は SQL Server データベースに以下のアカウントを作成して使用します。

アカウント	パスワード
wwAdmin	wwadmin
wwPower	wwpower
wwUser	wwuser

Alarm DB Logger Manager の使用

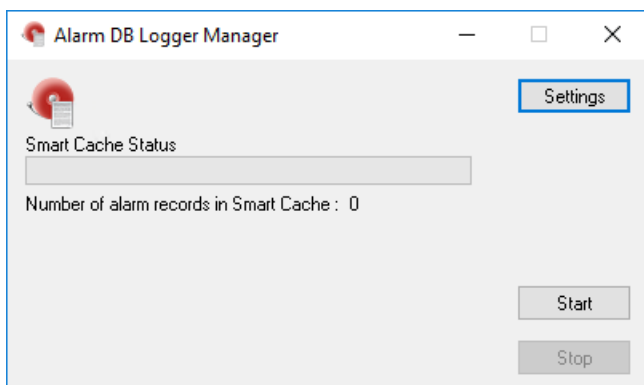
Alarm DB Logger Manager は、ログ処理を開始または停止する場合に使用します。Alarm DB Logger Manager は、サービスまたは標準のアプリケーションとして起動できます。

アラームのログは Alarm DB Logger 設定ウィザードを使用して設定します。このウィザードは Alarm DB Logger Manager から起動します。

Alarm DB Logger Manager には、スマート キャッシュに格納されたアラーム レコードの割合が表示されます。アラームがキャッシュされるのは、SQL Server との接続がダウンした場合や、Alarm DB Logger がアラーム データベースにログできる速度よりも速くアラームが発生した場合です。

Alarm DB Logger Manager を開くには

1. [ツール] ビューで [アプリケーション] を展開します。
2. [Alarm DB Logger Manager] をダブルクリックします。Alarm DB Logger Manager が表示されます。

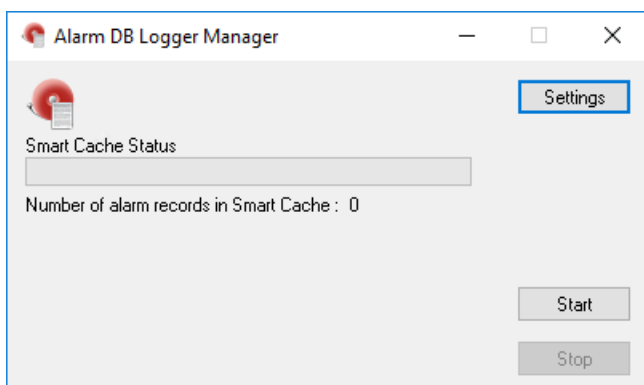


アラーム データベース ログの開始と停止

Alarm DB Logger はレコードをアラーム データベースに書き込みます。Alarm DB Logger Manager の設定ウィザードで設定した実行モードに応じて、サービスまたは標準のアプリケーションとして起動および動作します。

アラームのログを開始または停止するには

1. [ツール] ビューで [アプリケーション] を展開します。
2. [Alarm DB Logger Manager] をダブルクリックします。Alarm DB Logger Manager が表示されます。



[スマート キャッシュ ステータス] に、メモリ内キャッシュに格納されているアラーム レコードの割合が表示されます。

3. [開始] をクリックして、アラームのログを開始します。
4. [停止] をクリックして、アラームのログを終了します。

アラーム データベース ロギングの設定

InTouch のアラームやイベントのデータをアラーム データベースにログするには、Alarm DB Logger Manager から以下の操作を実行します。

- アラーム データベースとの接続の設定
- アラーム データベースにログするアラームの選択
- レコードをアラーム データベースにログする周期の設定
- Alarm DB Logger の実行方法の選択

データベース接続の確立

Alarm DB Logger とアラーム データベース間の接続を確立する必要があります。

Alarm DB Logger は、SQL Server および Windows 認証と連動します。SQL Server は混在モードに設定する必要があります。

データベース接続を設定するには

1. Alarm DB Logger Manager を開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Logger Manager] をダブルクリックします。
2. [設定] をクリックします。[Alarm DB Logger Manager - 設定] ウィザードが表示されます。

3. [SQL Server/MSDE] セクションで以下の手順を実行します。
 - a. [認証] リストで認証方法 (SQL Server 認証または Windows 認証) を選択します (デフォルトは Windows 認証)。

注記: Windows 認証は、SQL 認証よりも優れたアプリケーションセキュリティを提供できます。この理由から、Windows 認証から SQL 認証に切り替えると、Windows 認証を使用することを推奨するポップアップダイアログが表示されます。この警告を無視して SQL 認証を使用する場合は、[OK] をクリックします。同様のメッセージが OCMC Log Viewer に記録されます。

- b. [サーバー名] ボックスに、アラーム データベースがインストールされているコンピュータのノード名を入力します。

- c. [データベース] ボックスに、InTouch アラーム データベースの名前を入力します。
4. [資格情報] セクションの [資格情報] ドロップダウンから認証に使用する資格情報を選択します。

注記:

- [資格情報] ドロップダウンは、SQL Server 認証タイプが選択されている場合にのみ有効になります。Windows 認証では、現在ログインしているユーザーの資格情報が使用され、ユーザー名とパスワードフィールドが無効になります。
 - スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報 マネージャから取得されます。詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「資格情報マネージャの操作」を参照してください。
-

5. [ロギングモード] 領域で、レコードの保存方法を設定します。以下のいずれかを実行します。
- a. 各アラーム条件（アラーム、確認、および通常状態への復帰）を個別のレコードに保存する場合は、[詳細] をクリックします。
 - b. アラームのすべての状態（アラーム、確認、および通常状態への復帰）を各状態移行のタイムスタンプを付けて単一のレコードに保存する場合は、[統合] をクリックします。
6. 必要に応じて、[作成] をクリックして、データベースを作成します。
7. [接続テスト] をクリックして、アラーム データベースへの接続性を確認します。データベースに正常に接続できたかどうかを示すメッセージが表示されます。
8. [次へ] をクリックして、ログに記録するアラームを設定します。詳細な手順については、「[ログするアラームの設定](#)」を参照してください。

ログするアラームの設定

Alarm DB Logger Manager 設定ウィザードの 2 ページ目では、InTouch または Galaxy アラーム プロバイダからのアラームを選択するための 1 つまたは複数のクエリーを定義します。また、データベース クエリーの一部であるアラーム優先度の値の範囲も選択します。

リモート ノードには次のクエリー構文を使用します。

\\ノード名\Provider!AlarmGroup

ローカル ノードには次のクエリー構文を使用します。

\\Provider!AlarmGroup

例:

\\ProdSvr\InTouch!\$System

次に、Galaxy アラーム プロバイダを使用した例を示します。このクエリーは、特定の領域からのすべてのアラームを取得します。

\\Galaxy!Area

注記: Alarm DB Logger Manager では、最大 50 のクエリーおよび 3072 文字がサポートされます。

ログに記録するアラームを設定するには

1. Alarm DB Logger Manager を開いて、設定ウィザードを起動します。以下の操作を行います。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Logger Manager] をダブルクリックします。

- c. [設定] をクリックします。[Alarm DB Logger Manager - 設定] ウィザードが表示されます。
2. [次へ] をクリックします。[Alarm DB Logger Manager - クエリーの選択] ページが表示されます。

読み取り専用の [アラーム状況] ボックスに、ログ対象のアラーム状況が表示されます。読み取り専用の [クエリータイプ] ボックスには、クエリーのタイプが表示されます。

3. [優先度始点] ボックスに、アラームの優先度範囲の開始値を入力します。
4. [優先度終点] ボックスに、アラームの優先度範囲の終了値を入力します。
5. [アラームクエリー] ボックスに、アラームデータベースに対してデータを保存または取得するために使用するアラームクエリーを入力します。
6. [次へ] をクリックして、ログの間隔を設定します。「[ログの間隔の設定](#)」を参照してください。

ログの間隔の設定

Alarm DB Logger Manager 設定ウィザードの 3 ページ目では、クエリーの詳細設定を行います。イベントをアラームデータベースにログすることができます。また、アラームレコードを内部のアラームメモリからデータベースに書き込む頻度を設定をすることで、アラームログのパフォーマンスを調整することもできます。

ログの間隔は、SQL Server の再接続速度と同じではありません。ログの間隔はアラームを読み取る間隔です。一方、再接続速度は、SQL Server に関連付けられている接続試行のタイムアウトによって異なります。

ログの間隔の設定が低過ぎると、システムパフォーマンスに悪影響が生じます。

ログの間隔を設定するには

1. Alarm DB Logger Manager を開いて、設定ウィザードを起動します。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。

- b. **[Alarm DB Logger Manager]** をダブルクリックします。
- c. **[設定]** をクリックします。**[Alarm DB Logger Manager - 設定]** ウィザードが表示されます。
2. **[次へ]** をクリックします。**[Alarm DB Logger Manager - クエリーの選択]** ページが表示されます。
3. **[次へ]** をクリックします。**[Alarm DB Logger Manager - 詳細設定]** ページが表示されます。

Alarm DB Logger Manager - Advanced Setting

Running Logger As

☐ Normal application ☒ Windows Service

Log on as

☒ Virtual account

☐ This account: NT SERVICE\New_Alamlogger

Password:

Performance Tuning

Log Alarms Every 100 m sec

Miscellaneous

☒ Log Events

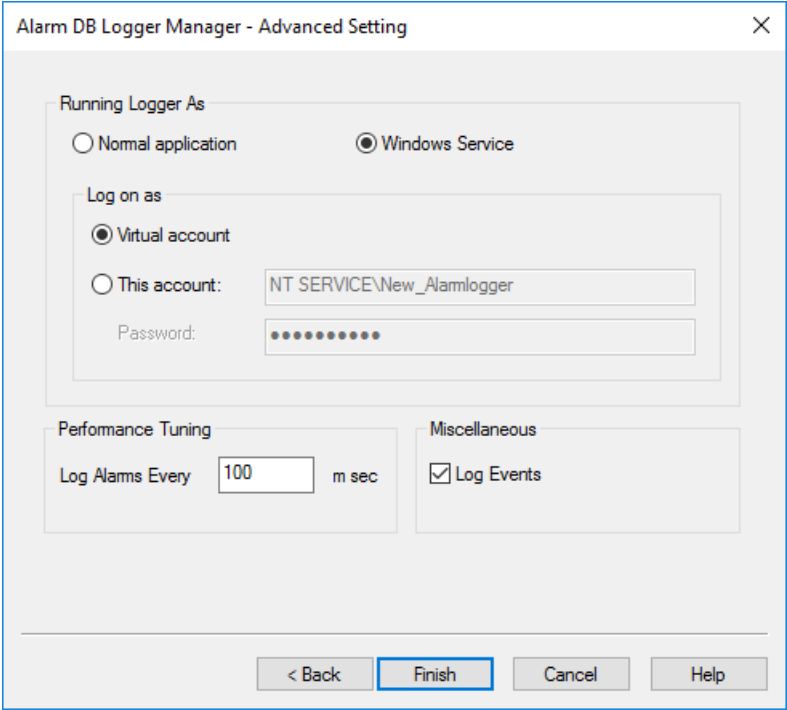
< Back Finish Cancel Help

4. InTouch イベント レコードをアラーム データベースに保存する場合は、**[イベント ログ]** チェックボックスをオンにします。Application Server Galaxy から取得されたイベントを保存することもできます。
5. **[パフォーマンスの調整]** 領域に、アラーム レコードをアラーム データベースに書き込む間隔をミリ秒単位で入力します。
6. **[完了]** をクリックします。

サービスとしての Alarm DB Logger の設定

Windows サービスとして実行するように Alarm DB Logger を設定できます。管理者権限で Alarm DB Logger を実行する場合にセキュリティの脅威にさらされる可能性を削減するために、Alarm DB Logger をサービスとして実行するユーザー アカウントの許可は非対話型に設定されています。

1. コンピュータに管理者としてログインします。
2. Alarm DB Logger Manager を開いて、設定ウィザードを起動します。以下の操作を行います。
 - a. **[ツール]** ビューで **[アプリケーション]** を展開します。
 - b. **[Alarm DB Logger Manager]** をダブルクリックします。
 - c. **[設定]** をクリックします。**[Alarm DB Logger Manager - 設定]** ウィザードが表示されます。
3. **[次へ]** をクリックします。**[Alarm DB Logger Manager - クエリーの選択]** ページが表示されます。
4. **[次へ]** をクリックします。**[Alarm DB Logger Manager - 詳細設定]** ページが表示されます。



5. [ロガーの実行] 領域で、[標準アプリケーション] または [Windows サービス] のいずれかをクリックします。
- 資格情報は [Alarm DB Logger Manager - 設定] ウィザードページで選択した認証方法に応じて異なります。
6. [ログオンに使用するアカウント] 領域で以下の条件に基づいてログインアカウントを選択し、[完了] をクリックします。

認証方法	ロガーの実行	ログオンに使用するアカウント	ユーザー名	パスワード
SQL Server 認証	Windows サービス	仮想アカウント	--	--
Windows 認証	Windows サービス	仮想アカウント	--	--
Windows 認証	Windows サービス	指定アカウント	<ユーザー名>	<パスワード>

注記: 仮想アカウントの詳細については、「[仮想アカウントの使用](#)」を参照してください。

デフォルトでは、[指定アカウント] を選択すると、現在ログインしているユーザーのユーザー名が表示されます。パスワードを入力する必要はありません。ユーザー名を変更した場合、パスワードを入力する必要があります。

設定を完了するには、適切な権限のあるユーザー アカウントを指定する必要があります。

次回 Alarm DB Logger を起動すると、設定の詳細が自動的に入力されます。

仮想アカウントの使用

仮想アカウントを選択するとセキュリティを強化できます。このオプションを選択すると、Alarm DB Logger が New_AlarmLogger という NT Service アカウントで起動します。

仮想アカウントは SQL 認証と Windows 認証の両方で有効になります。[LocalSystem] で設定された以前のシステムの場合、アカウントは次のように移行されます。

- 以前のシステムで [指定アカウント] または [ローカル システム アカウント] オプションが [LocalSystem] に設定されている場合、[指定アカウント] にはユーザーとして [LocalSystem] が設定されます。
- 以前のシステムで [指定アカウント] が [NT Service\New_AlarmLogger] として設定されている場合、[仮想アカウント] に移行されます。

アラーム データベースのビュー

SQL Server データベース ビューのセットを使用すると、過去および現在のアラームやイベントの発生状況について簡単にクエリーを実行できます。

データベース ビューとは、基礎となる複数のデータベース テーブルの情報を組み合わせた 1 つの論理的なテーブルのことです。データベース ビューは仮想データベース テーブルと呼ばれる場合もあります。これは、実際のテーブルと同じように標準の SQL ステートメントを使ってクエリーを実行できるからです。ビューがクエリーされると、レコード（または行）のセットが返されます。各行には、レコードのデータを含む情報のカラムがいくつかあります。

アラーム データベースのビューを使用すると、複雑なクエリーを実行できます。たとえば、工場の特定領域である一定の時間帯に起こった HiHi アラームすべてのアラーム レコードを取り出すことができます。または、あるアラーム プロバイダ ノードでログしたデータ変化イベントをすべて取り出すこともできます。

ビューの文字列はすべて Unicode です。

アラーム履歴ビュー

v_AlarmHistory ビューには、選択した時間範囲内に発生したすべての履歴アラームとアラーム変化のイベントが表示されます。クエリーで、(カラム EventStamp または EventStampUTC で) 開始と終了の日付と時刻を指定します。返されるレコードには、アラームの発生、アラームの確認、アラームの有効化、アラームの無効化、アラームから通常状態に戻ったイベントが含まれます。

以下の表は、ビューの列について説明しています。

列名	データ型	説明
EventStamp	Datetime	(データベースのローカル時間における) アラーム イベントの日付と時刻
AlarmState	nChar	アラーム状態 (UNACK、UNACK_RTN、ACK、ACK_RTN、LATCHED のいずれか)
TagName	nChar	TIC101 など、アラームを生成したオブジェクト名
Description	nVarchar	アラームの説明文字列。デフォルトはオブジェクトの説明 (または InTouch HMI のコメント)。または、確認レコードの確認コメント。
Area	nChar	アラームの領域、またはグループ名
Type	nChar	Hi、HiHi、ROC、PV.HiAlarm など、アラーム タイプ

列名	データ型	説明
Value	nChar	アラーム発生時に有効だったアラーム値
CheckValue	nChar	アラーム発生時のアラームしきい値
Priority	Integer	アラーム優先度
Category	nChar	アラーム クラス、またはアラーム カテゴリ。Value、Dev、ROC、Process、Batch、System など。
Provider	nChar	アラーム プロバイダ。node/InTouch や GalaxyName など。
Operator	nChar	ユーザーの名前
DomainName	nChar	ドメインの名前
UserFullName	nChar	オペレータであるユーザーのフル ネーム。 (Kenichi Suzuki など)
UNACKDuration	Float	アラームが最後に変化（アラームまたはサブステート）してから確認されるまでの時間（該当する場合）
User1	Float	ユーザー定義のフィールド数 1
User2	Float	ユーザー定義のフィールド数 2
User3	nChar	ユーザー定義のフィールド、文字列
EventStampUTC	DateTime	アラーム イベントの UTC 日付/時刻
Millisec	Small Int	0.1 ミリ秒単位のイベント スタンプ
OperatorNode	nvarchar(32)	ユーザーがアラームを確認したノードの名前

v_AlarmHistory2 ビューは v_AlarmHistory ビューの一種です。

列名	データ型	説明
EventStamp	Datetime	(データベースのローカル時間における) アラーム イベントの日付と時刻
AlarmState	nChar	アラーム状態 (UNACK、UNACK_RTN、ACK、ACK_RTN、ACK_ALM、UNACK_ALM、LATCHED のいずれか)
TagName	nChar	TIC101 など、アラームを生成したオブジェクト名
Description	nVarchar	アラームの説明文字列。デフォルトで、オブジェクトの説明か、InTouch のコメントに設定できます。または、確認レコードの確認コメント。
Area	nChar	アラームの領域、またはグループ名

列名	データ型	説明
Type	nChar	Hi、HiHi、ROC、PV.HiAlarm など、アラーム タイプ
Value	nChar	アラーム発生時に有効だったアラーム値
CheckValue	nChar	アラーム発生時のアラームしきい値
Priority	Integer	アラーム優先度
Category	nChar	アラーム クラス、またはアラーム カテゴリ。Value、Dev、ROC、Process、Batch、System など。
Provider	nChar	アラーム プロバイダ。node/InTouch や GalaxyName など。
Operator	nChar	ユーザーの名前。Suzuki など（該当する場合）。
DomainName	nChar	ドメインの名前
UserFullName	nChar	オペレータであるユーザーのフル ネーム。 (Kenichi Suzuki など)
AlarmDuration	Float	アラームが発生してから通常状態に戻るまでの時間
User1	Float	ユーザー定義のフィールド数 1
User2	Float	ユーザー定義のフィールド数 2
User3	nChar	ユーザー定義のフィールド、文字列
EventStampUTC	DateTime	アラーム イベントの UTC 日付/時刻
Millisec	Small Int	0.1 ミリ秒単位のイベント スタンプ

クエリーの例 - アラーム履歴ビュー

この例では、アラーム履歴ビューからすべてのレコードを選択します。

```
SELECT * FROM v_AlarmHistory
```

この例では、アラーム履歴ビューから、優先度が 100 より大きいすべてのレコードを選択します。

```
SELECT * FROM v_AlarmHistory WHERE Priority >100
```

イベント履歴ビュー

v_EventHistory データベース ビューには、選択した時間範囲内に発生したすべての履歴イベントのリストが表示されます。クエリークライアントは開始と終了の日付と時刻の範囲を指定します。返されるレコードには、アラーム以外のイベントもすべて含まれます。

カラム名	データ タイプ	説明
EventStamp	Datetime	イベントの日付と時刻
TagName	nChar	Pump1 など、イベントを生成したオブジェクト名

カラム名	データ タイプ	説明
Description	nVarChar	イベントを説明する文字列。デフォルトで、オブジェクトの説明か、InTouch のコメントに設定できます。
Area	nChar	イベントの領域、またはグループ名
Type	nChar	「ユーザー データの変更」や「開始」など、イベントのタイプ
Value	nChar	新規値（該当する場合）
CheckValue	nChar	古い値（該当する場合）
Category	nChar	Value、Process、Batch、System など、イベント カテゴリまたはクラス
Provider	nChar	イベントを生成したオブジェクト (node/InTouch など)、またはユーザー変更のビュー エンジン名
Operator	nChar	ユーザー名。Suzuki など（該当する場合）。
DomainName	nChar	ドメインの名前
UserFullName	nChar	オペレータであるユーザーのフル ネーム。 (Kenichi Suzuki など)
User1	Float	ユーザー定義のフィールド数 1
User2	Float	ユーザー定義のフィールド数 2
User 3	nChar	ユーザー定義のフィールド、文字列
EventStampUTC	DateTime	イベントの UTC 日付／時間
Millisec	Small Int	0.1 ミリ秒単位のイベント スタンプ

アラーム イベント履歴ビュー

v_AlarmEventHistory データベース ビューには、選択時間内に発生したすべてのイベントとアラームの履歴リストが表示されます。クエリー クライアントは開始と終了の日付と時刻を指定します。返される値には、すべてのアラームとイベントが含まれます。このビューでは、アラーム ビューとイベント ビューを 1 つにまとめて、両方のビューのレコードが合わせて表示されます。

列名	データ型	説明
EventStamp	Datetime	イベントの日付と時刻
AlarmState	nChar	アラーム状態 (UNACK、UNACK_RTN、ACK、ACK_RTN、LATCHED のいずれか) イベントには適用されません。
TagName	nChar	TIC101 など、アラームを生成したオブジェクト名

列名	データ型	説明
Description	nVarchar	アラーム/イベントの説明文字列。デフォルトで、オブジェクトの説明か、InTouch のコメントに設定できます。 または、確認レコードの確認コメント。
Area	nChar	アラームの領域、またはグループ名
Type	nChar	Hi、HiHi、ROC、PV.HiAlarm、ユーザー データの変更など、アラームやイベントのタイプ
Value	nChar	アラーム発生時に有効だったアラーム値
CheckValue	nChar	アラーム発生時のアラームのしきい値、またはイベントの古い値
Priority	Integer	アラーム優先度
Category	nChar	Value、Process、Batch、System など、アラームやイベントのクラス、またはアラーム カテゴリ
Provider	nChar	node/InTouch や GalaxyName など、アラーム プロバイダ
Operator	nChar	確認したユーザー、またはデータを変更したユーザー名
DomainName	nChar	ドメインの名前
UserFullName	nChar	オペレータであるユーザーのフル ネーム。 (Kenichi Suzuki など)
UNACKDuration	Float	アラーム変化が最後に発生してから確認までのミリ秒数
User1	Float	ユーザー定義のフィールド数 1
User2	Float	ユーザー定義のフィールド数 2
User3	nChar	ユーザー定義のフィールド、文字列
EventStampUTC	DateTime	イベントの UTC 日付/時間
Millisec	Small Int	0.1 ミリ秒単位のイベント スタンプ

v_AlarmEventHistory2 ビューは v_AlarmEventHistory ビューの一種です。

列名	データ型	説明
EventStamp	Datetime	イベントの日付と時刻
AlarmState	nChar	アラーム状態 (UNACK、UNACK_RTN、ACK、ACK_RTN、LATCHED のいずれか) イベントには適用されません。
TagName	nChar	TIC101 など、アラームを生成したオブジェクト名

列名	データ型	説明
Description	nVarchar	アラーム/イベントの説明文字列。デフォルトで、オブジェクトの説明か、InTouch のコメントに設定できます。または、確認レコードの確認コメント。
Area	nChar	アラームの領域、またはグループ名
Type	nChar	Hi、HiHi、ROC、PV.HiAlarm、ユーザー データの変更など、アラームやイベントのタイプ
Value	nChar	アラーム発生時に有効だったアラーム値
CheckValue	nChar	アラーム発生時のアラームのしきい値、またはイベントの古い値
Priority	Integer	アラーム優先度
Category	nChar	Value、Process、Batch、System など、アラームやイベントのクラス、またはアラーム カテゴリ
Provider	nChar	node/InTouch や GalaxyName など、アラーム プロバイダ
Operator	nChar	確認したユーザー、またはデータを変更したユーザー名
DomainName	nChar	ドメインの名前
UserFullName	nChar	オペレータであるユーザーのフル ネーム。 (Kenichi Suzuki など)
AlarmDuration	Float	アラームが発生してから通常状態に戻る (RTN) までのミリ秒数
User1	Float	ユーザー定義のフィールド数 1
User2	Float	ユーザー定義のフィールド数 2
User3	nChar	ユーザー定義のフィールド、文字列
EventStampUTC	DateTime	イベントの UTC 日付/時間
Millisec	Small Int	0.1 ミリ秒単位のイベント スタンプ

クエリーの例 - アラーム イベント履歴ビュー

次の例では、タグ変数 MyTag1 を持ち、アラーム状況が ACK_RTN または ACK のレコードすべてについて、TagName、Area、および Type の各カラムのデータを選択します。結果はアラーム プロバイダ順に並べます。

```
SELECT TagName, Area, Type FROM v_AlarmEventHistory
WHERE TagName='MyTag1'
AND (AlarmState='ACK_RTN' OR AlarmState='ACK')
ORDER BY Provider
```


アラーム データベースのストアードプロシージャ

SQL Server のストアードプロシージャのセットを使用すると、情報を簡単に検索してアラームとイベントの分析やレポートを実行できます。

ストアードプロシージャとは、SQL ステートメントの集まりで、特定の関数を実行し、データベースからデータを返すものです。ストアードプロシージャでは、入力パラメータを使用して、実行方法や、結果セットの形式でデータを返す方法を指定できます。

返される結果はレコードのセットで、SQL Server のデータベース ビューと類似した SQL レコードセットのように見えます。ストアードプロシージャを使用するとパフォーマンスを向上できます。これはストアードプロシージャが、組み込み SQL ステートメントと共にデータベース内に存在するため、クライアントへ戻す処理が必要なくなるからです。

ストアードプロシージャの詳細（ストアードプロシージャの定義の表示方法を含む）については、Microsoft SQL Server のマニュアルを参照してください。

ストアードプロシージャの呼び出し

ストアードプロシージャは、次のように EXECUTE ステートメントを使用して呼び出します。

```
EXECUTE sp_AlarmCounter @StartDate='2007-01-01', @EndDate='2007-03-31', @Tagname = 'tag1',
@Type = 'LO', @Provider = 'WW21353\InTouch', @Comment = 'SSAADD'
```

この例では、StartDate および EndDate パラメータが必須で、残りのパラメータはオプションです。特定のパラメータの値を指定しなかった場合は、結果セットをフィルタする際にそのパラメータは使用されません。

ストアードプロシージャに日付／時刻変数が含まれる場合は、SQL Server のマニュアルで指定されている有効な形式であればどれでも使用できます。

AlarmCounter データベース ストアードプロシージャ

このストアードプロシージャは、一定の周期内に固有のアラームが発生した回数を返します。固有のアラームを識別するには、Tagname、Provider、Alarm Type、Category という 4 つのパラメータを使用します。

発生数は、アラームの発生のみを計算するもので、確認や平常状態に戻った場合は含まれません。このため、アラームが発生して確認され、平常状態に戻った場合は、「3」ではなく「1」と認識されます。

クエリーでは、開始と終了の日付と時刻を指定します。これには、Tagname、Class、Type、Provider、および Comment という 5 つのオプションパラメータがあります。

次のような場合に、このビューで答えを見つけることができます。ある期間中に、プロバイダ Node1| InTouch (Provider) にあるオブジェクト TIC101 (TagName) の値がアラーム (Category) になり、そのアラーム タイプが HiHi (Type) であった回数を特定する場合。

注意： AlarmCounter ストアードプロシージャは詳細ロギングモードにのみ適用され、統合ロギングでは使用できません。

sp_AlarmCounter

カラム名	データ タイプ	説明
TagName	Nchar	TIC101 など、アラームを生成したオブジェクト名
GroupName	Nchar	アラームの領域、またはグループ名
AlarmType	Nchar	Hi、HiHi、ROC、PV.HiAlarm など、アラーム タイプ

カラム名	データ タイプ	説明
AlarmClass	Nchar	Value、Process、Batch などのアラーム クラス、またはアラーム カテゴリ
AlarmCount	整数	特定期間中のアラーム発生数。 開始日付および時刻より前にアラームが発生した場合、そのアラームは発生数に含まれません。
Priority	整数	アラーム優先度
Provider	Nchar	node/InTouch や GalaxyName など、アラーム プロバイダ
Comment	Nchar	アラーム コメント

次にクエリーの例を示します。

```
EXECUTE sp_AlarmCounter @StartDate='2007-01-01 23:23:23', @EndDate='2007-03-31 23:23:23',
@Tagname = '$NewAlarm'
```

EventCounter データベース ストアド プロシージャ

このデータベース ストアド プロシージャは、ある期間内で、特定タグ変数での特定タイプのイベント数を表示します。クエリー クライアントは、その期間の最初と最後を指定する必要があります。これには、Tagname、Provider、Comment の 3 つのオプションパラメータがあります。このカウンタは、アラーム以外のイベントのみに適用されます。このビューは、各イベントの頻度を示します。たとえば、ポンプが特定期間内に何回オンになったかを調べます。TagName、Provider、Category、および Type を使ってイベントを認識し、発生数を計算します。

sp_EventCounter

カラム名	データ タイプ	説明
TagName	NChar	TIC101 など、アラームを生成したオブジェクト名
Area	NChar	アラームの領域、またはグループ名
Type	NChar	イベントのタイプ
Category	NChar	Value、Process、Batch などのアラーム クラス、またはアラーム カテゴリ
EventCount	整数	ある期間中に発生した特定タグ変数に対するこのタイプのイベント数
Provider	NChar	アラーム プロバイダ。node/InTouch や GalaxyName など。
Comment	NChar	イベントのコメント

次にクエリーの例を示します。

```
EXECUTE sp_EventCounter @StartDate='2007-01-01 23:23:23', @EndDate='2007-03-31 23:23:23', @Tagname = '$NewAlarm'
```

レコードされたアラームの表示

Alarm DB View ActiveX コントロールを使用すると、アラーム データベースからのデータを視覚化できます。このコントロールを使用して、ランタイムに InTouch アプリケーションから生成されたすべてのアラームおよびイベント情報を表示できます。

このコントロールでは、以下の内容を設定できます。

- コンテキスト メニュー機能
- 表示モード
- リスト制御オプション
- 異なるプロパティの色
- フォント タイプ、スタイル、サイズ
- データベースの指定（サーバー名、ユーザー ID、パスワード）
- クエリー フィルタ
- カラムの管理
- ソート

ランタイム ユーザーは、アプリケーション実行中にオプションを変更し、表示中のデータを選択できます。以下の操作を行えます。

- カラム内の情報のソート
- 表示の更新
- クエリーの実行
- カラム幅のサイズ変更

ActiveX コントロールの詳細については、「[ActiveX コントロール](#)」を参照してください。

Alarm DB View コントロールの設定

Alarm DB View コントロールを設定する場合、以下の操作を行えます。

- アラーム データベースへの接続を設定します。
- グリッドの外観を設定する
- 表示フォントを選択する
- 表示外観を設定する
- アプリケーションが実行している間、ユーザーがアクセスできる機能を設定する
- 表示するアラームを設定する
- カスタム フィルタを作成する
- アラーム レコードのタイプの色を設定する
- 表示されるように時間形式を設定する
- 表示されるアラーム レコードのソート順を設定する

Alarm DB View と Alarm Database への接続

Alarm DB View コントロールとアラーム データベースの間の接続を設定する必要があります。

アラーム データベースへの適切な読み取り専用アクセスのあるアカウントを使用します。システム管理者アカウントではありません。

アラーム データベースへの接続を設定するには

1. Alarm DB View コントロールを右クリックし、[プロパティ] をクリックします。[AlmDbViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [データベース] タブをクリックします。

3. アラーム データベースへの接続を設定します。以下の手順を実行します。
 - a. [認証] リストで認証方法 (**SQL Server 認証**または **Windows 認証**を選択します (デフォルトは Windows 認証)。

注記: Windows 認証は、SQL 認証よりも優れたアプリケーションセキュリティを提供できます。この理由から、Windows 認証から SQL 認証に切り替えると、Windows 認証を使用することを推奨するポップアップ ダイアログが表示されます。この警告を無視して SQL 認証を使用する場合は、[OK] をクリックします。同様のメッセージが SMC Log Viewer に記録されます。

- b. [サーバー名] ボックスに、アラーム データベースがインストールされているコンピュータのノード名を入力します。

- c. [データベース名] ボックスにデータベース名を入力します。
- d. [資格情報] ボックスで認証に使用する資格情報を選択します。

注記:

- [資格情報] ドロップダウンは、SQL Server 認証タイプが選択されている場合にのみ有効になります。Windows 認証では、現在ログインしているユーザーの資格情報が使用され、ユーザー名とパスワードフィールドが無効になります。

- スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報 マネージャから取得されます。詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「資格情報マネージャの操作」を参照してください。

- 4. [自動接続] チェック ボックスをオンにし、InTouch アプリケーションが実行を開始したときに、Alarm DB View コントロールが自動的にアラーム データベースに接続されるようにします。

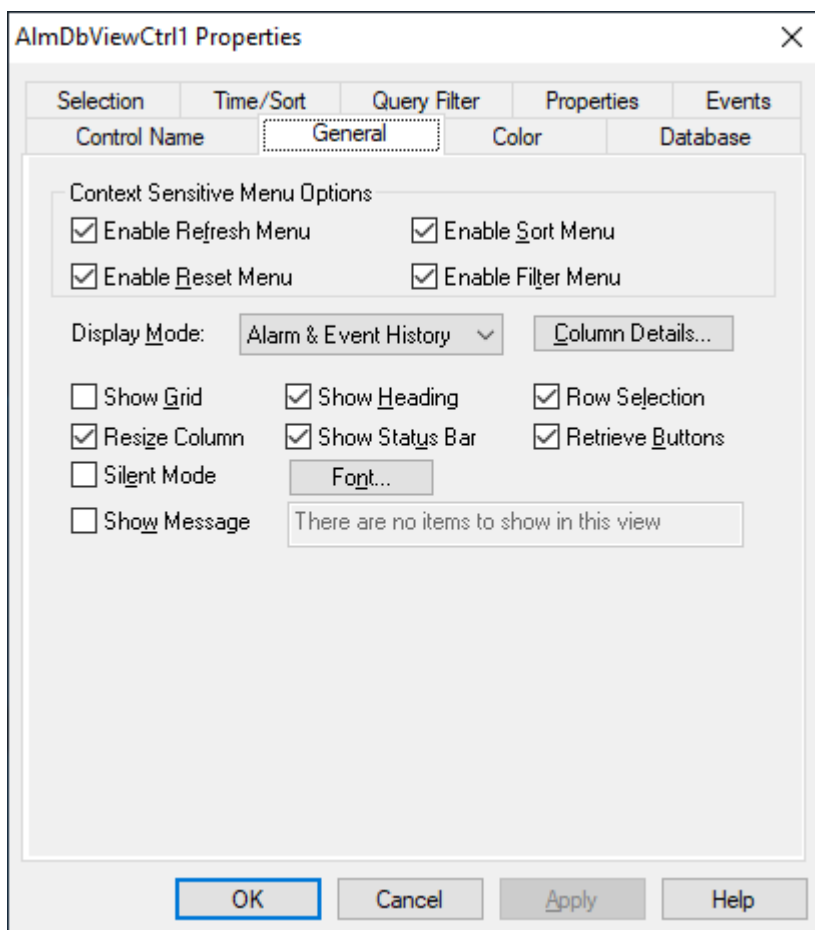
[自動接続] チェック ボックスをオンにしない場合は、Connect() メソッドを明示的に呼び出して、Alarm DB View コントロールがアラーム データベースに接続されるよう設定する必要があります。Connect メソッドの詳細については、「[Connect\(\)](#)」を参照してください。
- 5. [接続テスト] をクリックして、アラーム データベースへの接続性を確認します。接続が正常に確立されたことを示すメッセージが表示されます。
- 6. [適用] をクリックします。

グリッドの外観プロパティの設定

Alarm DB View コントロールの表示外観を決定するプロパティを設定できます。

グリッドの外観を設定するには

- 1. Alarm DB View コントロールを右クリックし、[プロパティ] をクリックします。[AlmDbViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
- 2. [全般] タブをクリックします。



3. 一般的な外観を設定します。以下の手順を実行します。

- [グリッドの表示] チェック ボックスをオンにし、グリッドを表示します。
- [サイレント モード] チェック ボックスをオンにして、ランタイムにコントロールでエラー メッセージが表示されないようにします。
- アラーム クエリーがレコードを返さない場合にメッセージを表示するには、[メッセージの表示] チェック ボックスをオンにします。ボックスに表示するメッセージを入力します。
- [見出しの表示] チェック ボックスをオンにして、コントロールの見出しを表示します。
- ステータス バーを表示するには、[ステータス バーの表示] チェック ボックスをオンにします。

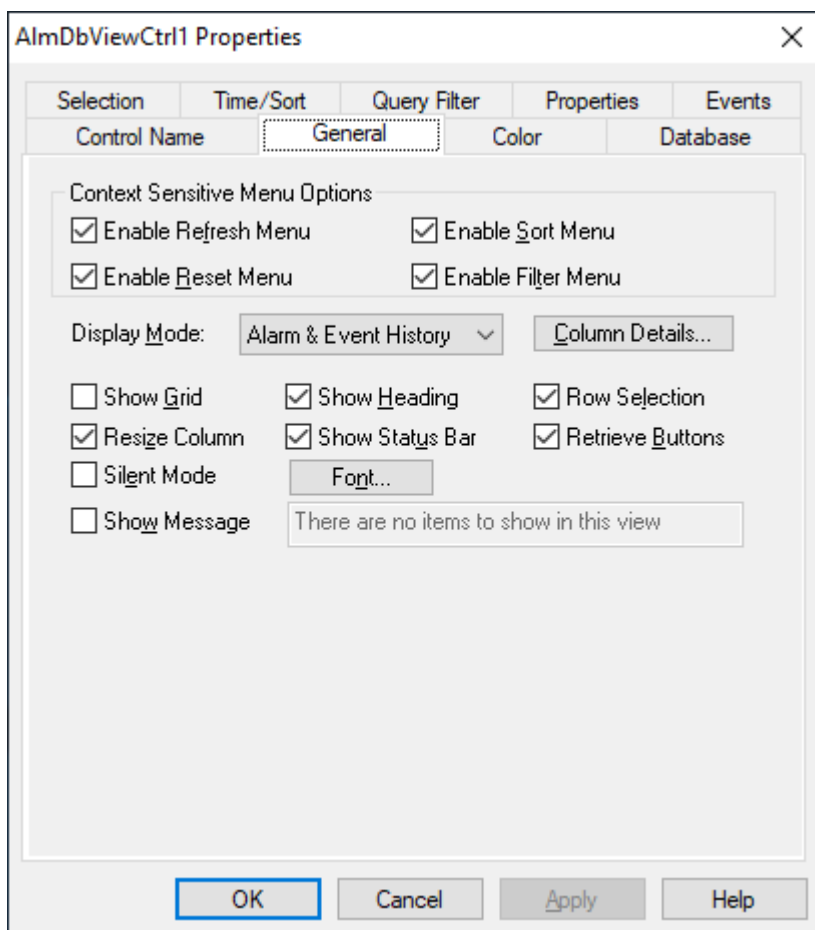
4. [適用] をクリックします。

表示フォントの設定

Alarm DB View コントロールに表示されるすべてのテキストのフォントを選択できます。

フォントのプロパティを設定するには

1. Alarm DB View コントロールを右クリックし、[プロパティ] をクリックします。[AlmdbViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [全般] タブをクリックします。



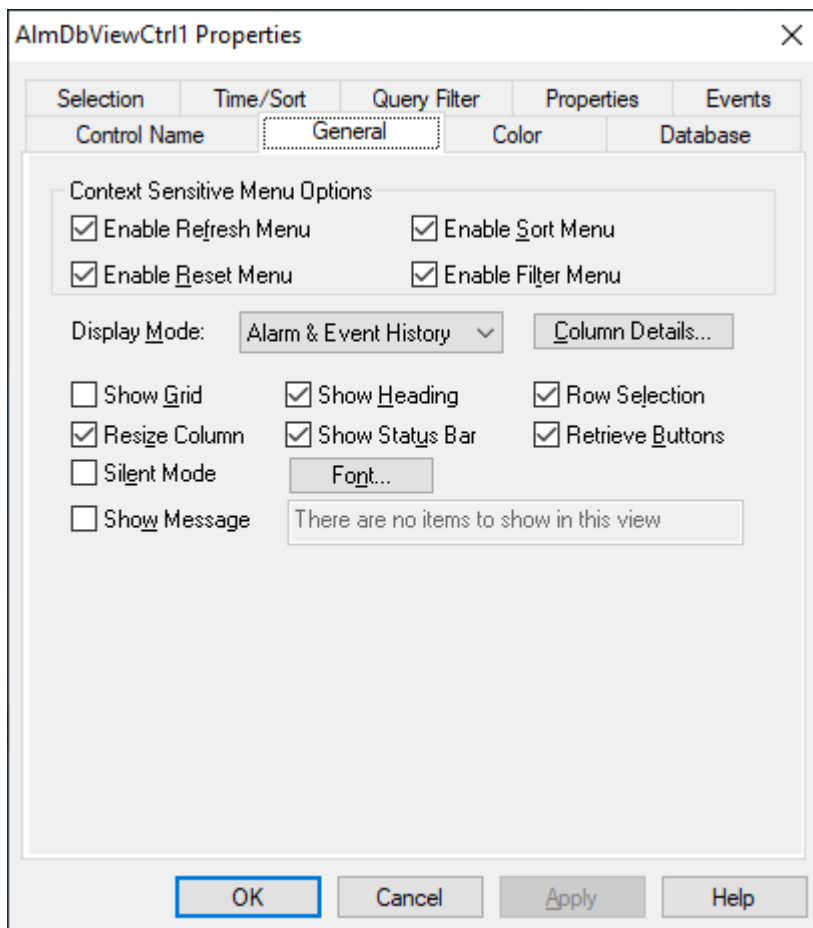
3. [フォント] をクリックします。Windows 標準の [フォント] ダイアログ ボックスが表示されます。フォントを設定して、[OK] をクリックします。
4. [適用] をクリックします。

アラーム データまたはイベント データの選択

アラーム レコード、イベント レコード、またはその両方を Alarm DB View コントロールに表示するかどうかを設定できます。

データ タイプを選択するには

1. Alarm DB View コントロールを右クリックし、[プロパティ] をクリックします。[AlmdbViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [全般] タブをクリックします。



3. [表示モード] リストで、リストの履歴レコードのタイプをクリックします。
 - a. [アラーム/イベント履歴] をクリックし、アラームおよびイベントの履歴データベース レコードを表示します。
 - b. [アラーム履歴] をクリックし、履歴アラーム レコードだけを表示します。
 - c. [イベント履歴] をクリックし、履歴イベント レコードだけを表示します。
4. [適用] をクリックします。

表示カラムの選択と設定

Alarm DB View コントロールの場合、以下の操作を行えます。

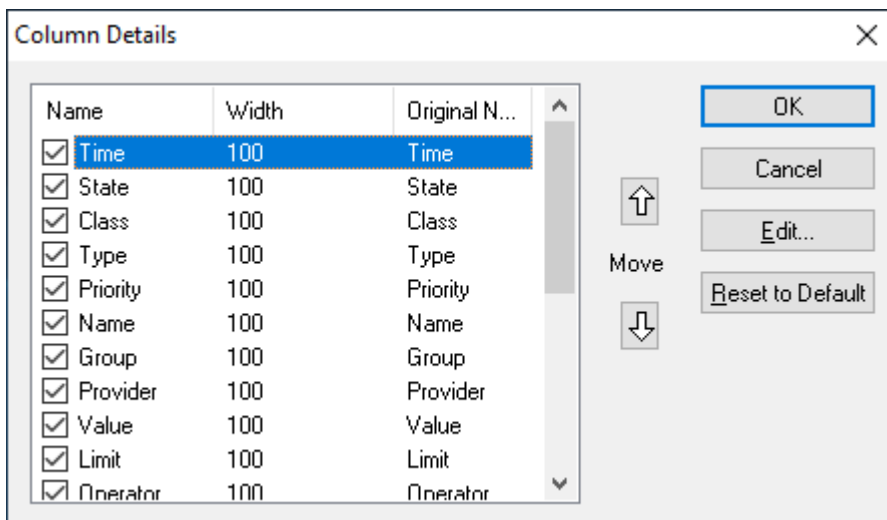
- カラムを選択して、順序を設定します。
- カラム幅をピクセル単位で設定します。
- カラムの名前を変更します。

少なくとも、カラムを 1 つ選択する必要があります。

重要: 表示モードが変更されると、カラム名はデフォルトのカラム名にリセットされます。カラム名を変更する前に、表示モードを最初に選択することをお勧めします。

表示カラムの詳細を設定するには

1. Alarm DB View コントロールを右クリックし、[プロパティ] をクリックします。[AlmDbViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [全般] タブをクリックします。
3. [カラムの詳細] をクリックします。[カラムの詳細] ダイアログ ボックスが表示されます。

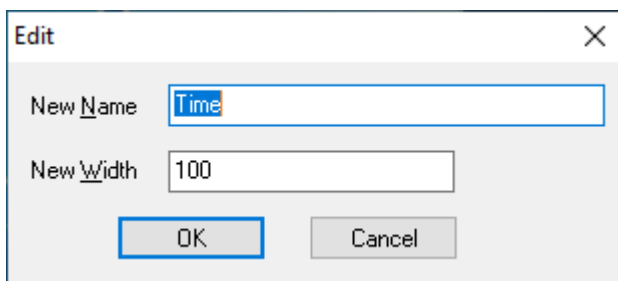


4. [アラーム名] カラムで、Alarm DB View コントロールに表示するカラム名の横のチェック ボックスをオンにします。リストから少なくとも 1 つ以上のカラムを選択する必要があります。

カラム名	説明
時刻	アラームが発生した時刻を表示します。
状態	アラームの状態を表示します。
クラス	アラーム カテゴリを表示します。
タイプ	アラーム タイプを表示します。
優先度	アラーム優先度を表示します。
名前	アラームまたはイベントの原因となったタグ変数またはソースを表示します。
グループ	アラーム グループ名を表示します。
プロバイダ	アラーム プロバイダ名を表示します。
値	アラームが発生したときのタグの値を表示します。
しきい値	タグ変数のアラームしきい値を表示します。
オペレータ	アラーム状況に関連付けられているログオンしたオペレータの ID を表示します。

カラム名	説明
オペレータ フル ネーム	ログオンしたオペレータのフル ネームを表示します。
オペレータ ノー ド	アラーム状況に関連付けられているログオンしたオペレータ ノード を表示します。 ターミナル サービス環境で、オペレータ ノードはオペレータがターミ ナル サービス セッションを確立したクライアント コンピュータの名 前です。ノード名を取得できない場合、ノードの IP アドレスが代わり に使用されます。
オペレータ ドメ イン	アラーム状況に関連付けられているログオンしたオペレータ ドメ インを表示します。
アラーム コメン ト	タグ変数のコメントを表示します。このコメントは、データベースで タグ変数のアラームが定義されたときに [アラーム コメント] ボック スに入力されたものです。 アラームに対して確認コメントが指定されると、新しいコメントはこ のコメント カラムで更新されます。
ユーザー 1	アラームに対応する [ユーザー定義番号 1] の数値を表示します。
ユーザー 2	アラームに対応する [ユーザー定義番号 2] の数値を表示します。
ユーザー 3	アラームに関連付けられたユーザー定義文字列プロパティの文字列値 を表示します。
期間	オペレータの選択に基づいて、未確認期間またはアラーム期間を表示 します。
UTC 時間	アラームの時刻を UTC で表示します。

- カラム名を選択し、上下の矢印を使用して、カラムを並べ替えます。[カラムの詳細] ダイアログ ボックスの一番上に表示されるカラム名は、アラーム コントロールの一番左のカラムの名前です。
- カラムの名前または幅を変更するには、カラムを選択して、[編集] をクリックします。[編集] ダイアログ ボックスが表示されます。



The image shows a dialog box titled "Edit" with a close button (X) in the top right corner. It contains two input fields: "New Name" with the text "Time" entered, and "New Width" with the value "100". At the bottom, there are two buttons: "OK" and "Cancel".

- [新しい名前] ボックスに、新しいカラム名を入力します。

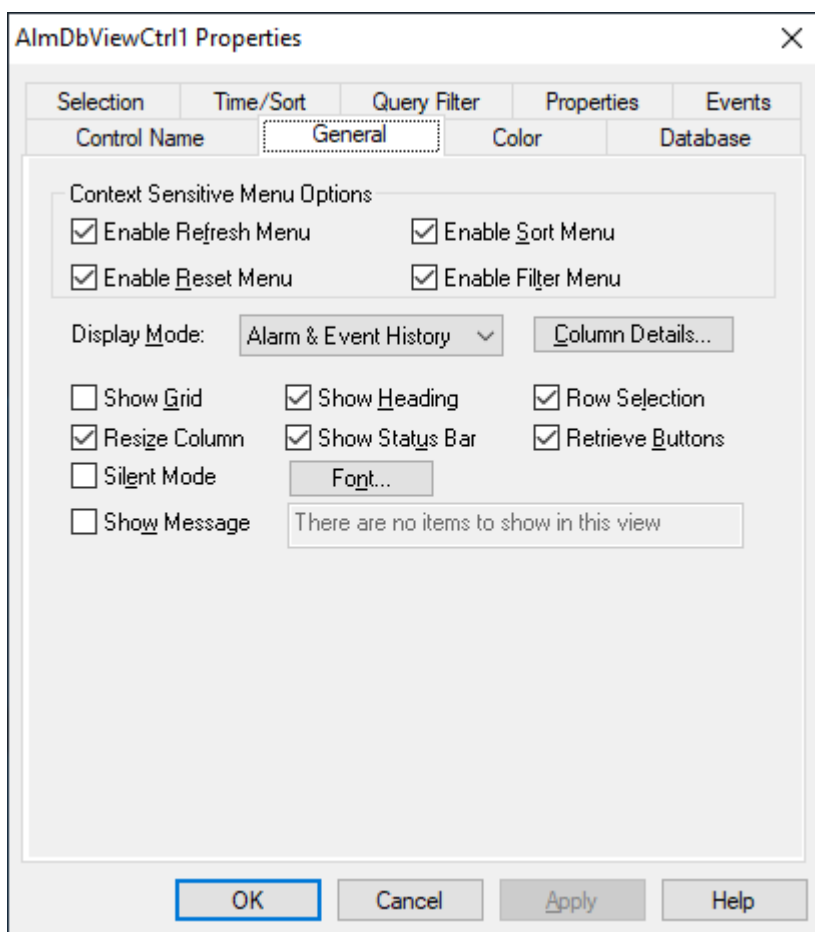
- b. [新しい幅] ボックスに、カラム幅を入力します。カラム幅は、1 ～ 999 ピクセルの範囲で設定できます。
 - c. [OK] をクリックします。
7. デフォルトのカラムの詳細設定に戻すには、[デフォルトにリセット] をクリックします。
 8. [カラムの詳細] ダイアログ ボックスの [OK] をクリックします。
 9. [適用] をクリックします。

ランタイム時にユーザーがアクセスできる機能の管理

このコントロールで利用できるショートカット メニュー機能を有効または無効にできます。

ランタイム機能を設定するには

1. Alarm DB View コントロールを右クリックし、[プロパティ] をクリックします。[AlmDbViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [全般] タブをクリックします。



3. [コンテキストメニューのオプション] 領域で、ランタイムで利用できるメニュー コマンドを設定します。
- ランタイム時にコントロールのショートカット メニューの [リフレッシュ] メニュー オプションを有効にするには、[リフレッシュ メニューを有効化] チェック ボックスをオンにします。[リ

フレッシュ] メニューは、データベースへのコントロールを更新します。正常に接続された場合は、1 から **MaxRecords** プロパティで定義された値までの範囲のレコードが表示されます。

- ランタイム時にコントロールのショートカットメニューの **[リセット]** メニュー オプションを有効にするには、**[リセット メニューを有効化]** チェック ボックスをオンにします。**[リセット]** メニューを使用すると、すべてのカラムがデザイン時に保存した設定に戻ります。
 - ランタイム時にコントロールのショートカットメニューの **[ソート]** メニュー オプションを有効にするには、**[ソート メニューを有効化]** チェック ボックスをオンにします。このメニューを使用すると、ユーザー定義のカラム ソートを設定するために使用される **[セカンダリ ソート]** メニューが表示されます。
 - ランタイム時にコントロールのショートカットメニューの **[フィルタ]** メニュー オプションを有効にするには、**[フィルタ メニューを有効化]** チェック ボックスをオンにします。このメニューを使用すると、ユーザー定義のフィルタ条件を設定するために使用される **[フィルタ]** メニューが表示されます。
- カラムのサイズを変更できるようにするには、**[カラムのサイズ変更]** チェック ボックスをオンにします。
 - アラーム レコードの選択を有効にするには、**[行の選択]** チェック ボックスをオンにします。
 - コントロールの右側の取得ボタンを表示するには、**[取得ボタン]** チェック ボックスをオンにします。
 - [適用]** をクリックします。

アラーム レコードに対して表示される時間形式とタイム ゾーンの設定

Alarm DB View コントロールに表示されるアラーム レコードの時間形式とタイム ゾーンを設定できます。

時間の形式を設定するには

- Alarm DB View コントロールを右クリックし、**[プロパティ]** をクリックします。**[AlmDBViewCtrl のプロパティ]** ダイアログ ボックスが表示されます。
- [時間/ソート]** タブをクリックします。

AlmDbViewCtrl1 Properties

Control Name General Color Database

Selection Time/Sort Query Filter Properties Events

Time Format: %m/%d/%Y %l:%M:%S %p

Displayed Time Zone : Local Time

Primary Sort Column: Time

Secondary Sort Column:

Sort Order

☒ Ascending

☐ Descending

OK Cancel Apply Help

3. [時間の形式] リストで、時間形式をクリックします。

文字列	説明	例
d	日付を表す 2 桁の数	31
b	月を表す 3 文字の略語	8 月
y	年を表す 4 桁の数	2007
m	月を表す 2 桁の数	11
/	日付を区切るための記号	06/2007
y	年を表す 2 桁の数	07
#x	日付と曜日	2002 年 8 月 9 日 金曜日
B	月の名前	9 月
-	日付を区切るための記号 (-)	06-07
.	日付を区切るための記号 (.)	06.07

文字列	説明	例
,	日付を区切るための記号 (,)	8 月 09, 2007
H	24 時間形式の時間	22:15
:	4:41 など時間を区切るための記号	
M	分	0:41
p	AM または PM の表示	
S	秒	16:41:07
s	秒の端数	16:41:07.390
l	AM/PM で表す 12 時間形式の時間	16:41

カスタム テキストと書式設定文字を使用して、リスト ボックスに独自の形式の文字列を手動で入力できます。以下に、時間形式の文字列のサンプルを示します。

時間形式の文字列	表示
%d %b	09 8 月
%m/%d/%Y	08/09/2002
%#x	2002 年 8 月 9 日 金曜日
%Y-%m-%d	08/09/2002
%m/%d/%Y %H:%M %p	08/09/2002 16:56 PM
%m/%d/%Y %H:%M:%s %p	08/09/2002 16:56:38.07
%l:%M %p	16:56

4. [表示されるタイムゾーン] リストで、必要なタイムゾーンをクリックします。

タイムゾーン	説明
GMT	アラーム時刻スタンプでは、グリニッジ標準時が使用されます。
ローカル時間	アラーム時刻スタンプは、Alarm DB View コントロールをホストするクライアントのローカル時間で表示されます。
標準時間	アラーム時刻スタンプは、アラーム プロバイダのローカル時間で表示されます。

5. [適用] をクリックします。

アラーム データベースからのデータ期間の選択

選択した時間に基づいてレコードを選択するようにクエリー値を設定できます。また、表示するレコードの最大数、アラーム クエリーの開始時間と終了時間、クエリー タイム ゾーンを設定することもできます。

アラーム データベース クエリーによって抽出されるレコードの数は、**Alarm DB View** コントロールにレコードが表示されるまでの時間に影響します。**Alarm DB View** コントロールでは、アラーム データベースから抽出された 50000 件のレコードを表示するのに約 30 秒かかります。

Alarm DB View コントロールのパフォーマンスを向上させるには、**[最大レコード数]** の値を減らして、レコードがすばやく表示されるようにします。また、**[期間の設定]** フィールドでレコードの取得間隔を短くして、**Alarm DB View** コントロールのパフォーマンスを向上させることもできます。

データの期間を選択するには

1. **Alarm DB View** コントロールを右クリックし、**[プロパティ]** をクリックします。**[AlmDBViewCtrl のプロパティ]** ダイアログボックスが表示されます。
2. **[選択]** タブをクリックします。

The screenshot shows the 'AlmDBViewCtrl Properties' dialog box with the 'Selection' tab selected. The 'Use Specific Time' checkbox is unchecked. The 'Duration' dropdown is set to 'Last Hour'. The 'Start Time' is '10/20/2020 11:59:20' and the 'End Time' is '10/20/2020 12:59:20'. The 'Duration Column' section has 'UnAck Duration' selected. The 'Query Time Zone' section has 'UTC' selected. The 'Maximum Records' field is set to '100'. At the bottom are 'OK', 'Cancel', 'Apply', and 'Help' buttons.

3. **UTC** タイム ゾーンを使用してデータを照会する事前定義された時間間隔を使用するには、**[期間の設定]** リストから間隔をクリックします。
4. 特定の開始時間と終了時間を使用するには、**[指定時間を使用]** をクリックして詳細を設定します。

- a. **[開始時間]** ボックスに、アラーム レコードを取得する開始時間を入力します。文字列は YYYY/MM/DD HH:MM:SS 形式にします。1970 年 1 月 1 日 0 時 00 分から 2038 年 1 月 18 日 19 時 14 分 07 秒の間なら、どのタイム ゾーンのものでも使用できます。
 - b. **[終了時間]** ボックスに、アラーム レコードの取得を停止する終了時間を入力します。文字列は YYYY/MM/DD HH:MM:SS 形式にします。1970 年 1 月 1 日 0 時 00 分から 2038 年 1 月 18 日 19 時 14 分 07 秒の間なら、どのタイム ゾーンのものでも使用できます。
 - c. **[クエリー タイム ゾーン]** 領域で、**[UTC]** または **[標準時間]** のどちらかをクリックします。UTC 時間はグリニッジ標準時 (Greenwich Mean Time) です。国際標準時または Zulu とも呼ばれます。標準時間は、オペレータのタイム ゾーンの現在の時間です。詳細については、「[クエリー タイム ゾーン](#)」トピックを参照してください。
5. **[間隔カラム]** 領域で、表示される間隔のタイプを設定します。間隔はミリ秒単位で測定されます。
- 最後のアラームの移行 (アラームまたはサブステート) から確認されるまでの時間を表示するには、**[未確認間隔]** をクリックします。
 - アラームの最初の発生から通常状態に戻るまでに経過した時間を表示するには、**[アラーム間隔]** をクリックします。
- LATCHED 状態の未確認間隔とアラーム間隔の詳細については、「[LATCHED アラームのアラーム間隔と未確認間隔](#)」トピックを参照してください。
6. **[最大レコード数]** ボックスに、1 つのインスタンスで、コントロールから表示するレコード数を入力します。また、最大レコードの有効な範囲は、1 ~ 1000 です。
7. **[適用]** をクリックします。

クエリー タイム ゾーン

[クエリー タイム ゾーン] オプションは、**[標準時間]** と **[UTC]** です。

- 標準時間は、ユーザーのタイム ゾーンのローカル時間です。
- UTC 時間はグリニッジ標準時 (Greenwich Mean Time) です。国際標準時または Zulu とも呼ばれます。

[指定時間を使用] が選択されている場合、2 つのクエリー タイム ゾーン オプションを選択できます。[指定時間を使用] が選択されていない場合、[期間の設定] で選択された事前定義された時間間隔によって UTC タイム ゾーンを使用してデータが常に照会されます。

夏時間切り替えの間の問題を避けるために、[クエリー タイム ゾーン] では常に UTC を使用することが推奨されます。代わりに「ローカル時間」を使用すると、夏時間から標準時間へ移行する間にアラーム レコードが失われる可能性があります。

異なるタイム ゾーンの設定を持つ複数の異なるコンピュータを実行していて、それらのコンピュータがすべて同じアラーム データベースにログを記録している場合、各レコードは UTC のタイムスタンプを取得することに加えて、そのタイムスタンプを対応する標準時間に変換するために必要なタイム ゾーン オフセットと夏時間の調整を取得します。その結果、データベースの各エントリは 2 つのタイムスタンプを持ちます。ログ記録を行ったコンピュータからの UTC 時間と標準時間です。これにより、取得が高速になります。テーブルエントリで、UTC 時間は「TransitionTime」として、また標準時間は「EventStamp」として識別されます。

LATCHED アラームのアラーム間隔と未確認間隔

このセクションでは、LATCHED 状態のアラーム間隔と未確認間隔の計算について説明します。アラーム間隔は、アクティブアラームと非アクティブアラームの間の期間です。未確認間隔は、UNACK アラー

ムと ACK アラームの間の期間です。LATCHED 状態のアラーム間隔と未確認間隔は、2 つの方法で計算できます。

オプション 1: アラームが確認されて通常状態に戻される

アラーム状態	名称	アラーム間隔	未確認間隔
UNACK	Tt	-	-
ACK	Ta	-	Tf-Tt
LATCHED	Tf	Tf-Tt	Tf-Tt
ACKRTN	Td	Tf-Tt	-

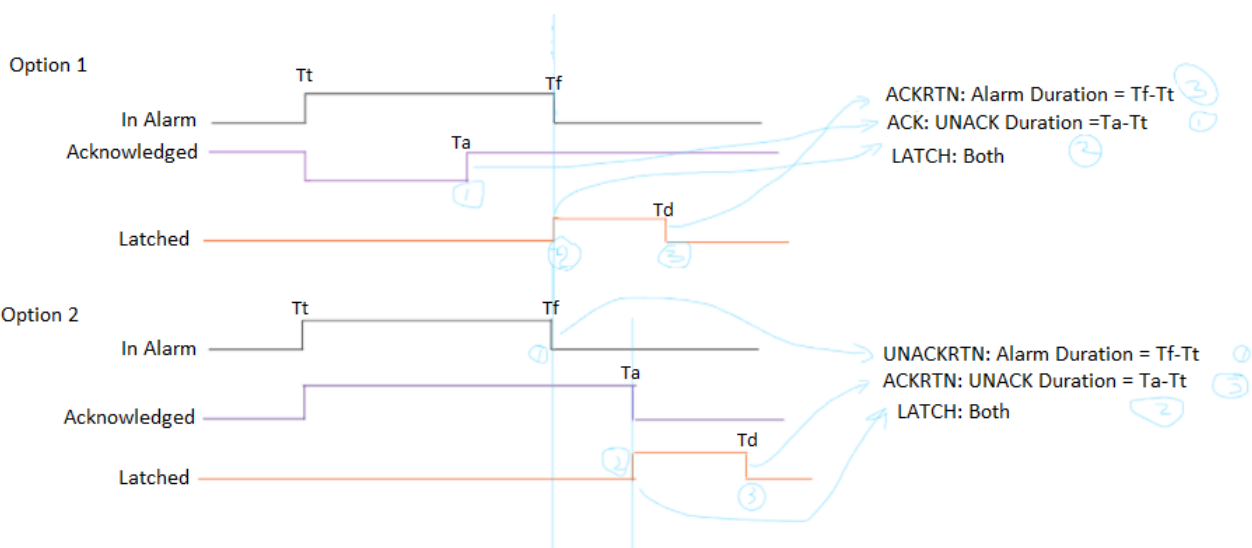
アラームが確認されると、未確認間隔は ACK 状態と LATCHED 状態の間の時間差 ($T_a - T_t$) になります。アラーム間隔はありません。アラーム状態が LATCHED に変化すると、アラーム間隔が LATCHED と UNACK の時間差 ($T_f - T_t$) で更新されます。未確認間隔は変化しません。LATCHED 状態が ACKRTN になると、アラーム間隔は同じ ($T_f - T_t$) で、未確認間隔はありません。

オプション 2: アラームが通常状態に戻されて確認される

アラーム状態	名称	アラーム間隔	未確認間隔
UNACK	Tt	-	-
UNACKRTN	Tf	Tf-Tt	-
LATCHED	Ta	Tf-Tt	Tf-Tt
ACKRTN	Td	-	Tf-Tt

アラームが返されると、アラーム間隔は UNACKRTN と UNACK の間の時間差 ($T_f - T_t$) になります。アラーム状態が LATCHED になると、アラーム間隔は同じです。未確認間隔は、LATCHED と UNACK の時間差 ($T_a - T_t$) になります。アラームが ACKRTN になると、未確認間隔は同じで、アラーム間隔はありません。

以下の図は、LATCHED アラームのアラーム間隔と未確認間隔の両方のオプションの間隔図を示します。



カスタム フィルタの作成とフィルタ設定の使用

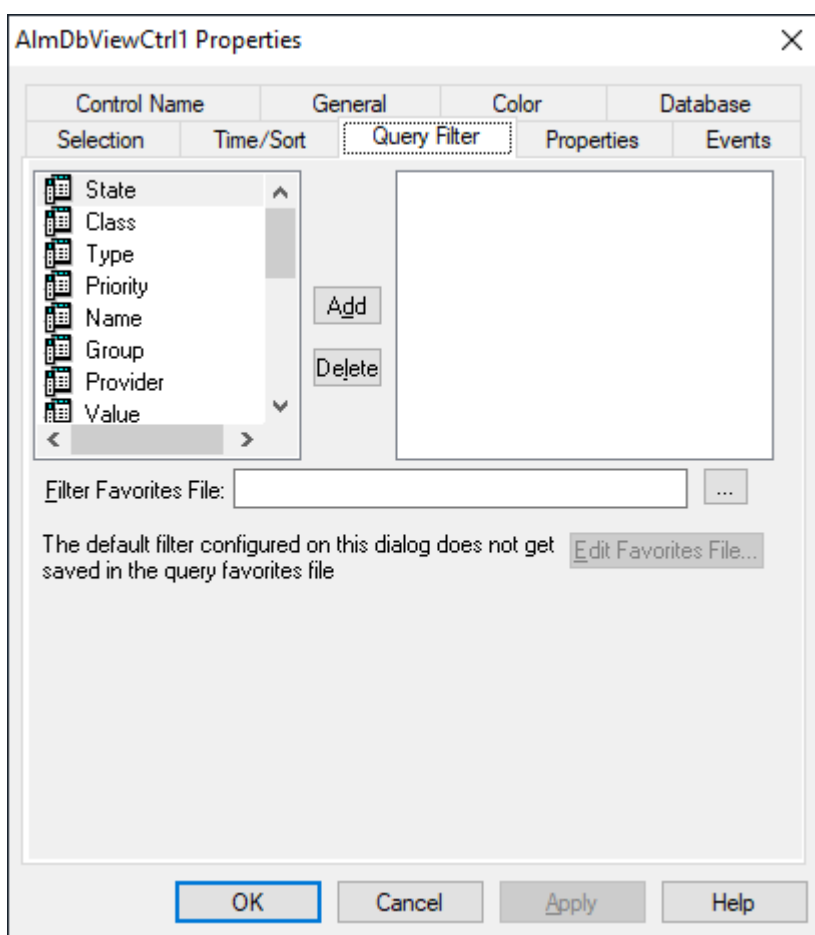
クエリー結果に含めるレコードを選択できます。たとえば、レコードの日付またはアラームの状態によってフィルタを選択できます。クエリー結果を制限または拡大するために複数のフィールドを選択できます。

カスタム フィルタを定義しないと、すべてのレコードが返されます。

言語を切り替えるために文字列を翻訳している場合、それらの文字列をフィルタ条件として使用することはできません。翻訳した文字列はアラーム データベースの外部に保存されます。

カスタム フィルタを作成するには

1. **[Alarm DB View]** コントロールを右クリックし、**[プロパティ]** をクリックします。**[AlmDBViewCtrlのプロパティ]** ダイアログ ボックスが表示されます。
2. **[クエリー フィルタ]** タブをクリックします。



3. 左ペインでフィルタ フィールドを選択し、**[追加]** をクリックして、これらフィールドをフィルタに含めます。フィルタは右ペインに表示されます。フィルタ フィールドの詳細については、次の表を参照してください。

フィールド名	フィルタ クエリー
状態	アラーム状態。詳細については、「 [状況] カラムの値 」を参照してください。
クラス	アラーム クラス
タイプ	アラーム タイプ
優先度	アラーム優先度
名前	アラーム名
グループ	アラーム グループ名
プロバイダ	アラーム プロバイダ
値	アラーム値。[値] カラムの値は英数字値として表示されます。クエリー フィルタのこれらの値の比較は、文字列の比較として行われます。
しきい値	アラームしきい値。値は英数字です。クエリー フィルタの値の比較は、文字列の比較として行われます。
オペレータ	オペレータ
オペレータ フル ネーム	オペレータ フル ネーム
オペレータ ノード	アラームと関連のあるオペレータ ノード名
オペレータ ドメイン	アラームと関連のあるオペレータ ドメイン名
コメント	アラーム コメント
ユーザー 1	アラーム ユーザー定義数値 1
ユーザー 2	アラーム ユーザー定義数値 2
ユーザー 3	アラーム ユーザー定義文字列値
期間	未確認とアラーム期間ゼロに等しい値で設定された間隔カラムでは、クエリーで null 値を持つレコードは生成されません。

4. フィルタ ペインからフィールドを削除するには、削除するフィールドをクリックし、**[削除]** をクリックします。フィルタの削除は元に戻せません。メッセージが表示されたら、**[はい]** をクリックします。
5. 各フィルタ フィールドの条件を設定します。詳細については、「[カラム フィルタ条件の定義](#)、[カラム フィルタ条件の定義](#)」を参照してください。
6. フィルタのオペレータとグループ化を設定します。詳細については、「[アラーム カラムのグループ化](#)、[アラーム カラムのグループ化](#)」を参照してください。
7. クエリー設定ファイルを設定します。以下の手順を実行します。

- a. [フィルタ設定ファイル] ボックスに、ネットワーク パスとファイル名を入力するか、省略記号ボタンをクリックしてファイルを参照します。
- b. [フィルタ設定] ファイルを編集するには、[設定ファイルの編集] ボタンをクリックします。[フィルタ設定] ウィンドウが開き、設定ファイルのフィルタを追加、変更、または削除できます。完了したら、[OK] をクリックして変更を保存し、ウィンドウを閉じます。

8. [適用] をクリックします。

カラム フィルタ条件の定義

クエリーに含める各カラム フィルタについて、フィルタ条件を設定する必要があります。たとえば、特定のオペレータについてのアラームだけを表示することもできます。

カラム フィルタを定義するには

1. フィールドを右クリックし、[フィルタの編集] をクリックします。[フィルタの定義] ダイアログボックスが表示されます。

2. [演算子] リストで、必要な演算子を選択します。
3. [値] ボックスに、一致する必要がある条件を入力します。[値] ボックスでは、選択したクエリーについて処理できない値は受け入れられません。「Like」および「Not Like」のフィルタ演算子が英数字のカラム名に使われている場合、[値] ボックスでは次のワイルドカード文字が使用できます。

文字	検索
%	ゼロ文字以上の任意の文字列
_	任意の 1 文字
[]	指定範囲 ([a-f] など)、またはセット([abcdef] など) 内の任意の 1 文字
[^]	指定範囲にない ([^a-f] など)、またはセット内にない([^abcdef] など) 任意の 1 文字

次の [値] ボックスしきい値がさまざまなフィールドに適用されます。

フィールド しきい値

すべてのフィールド	ユーザー 1、ユーザー 2、および優先度を除く、すべての英数字
-----------	---------------------------------

フィールド しきい値

優先度 1 ～ 999 の整数値が使用できます。

ユーザー 負の数、正の数、または小数のみを使用できます。

1、ユーザー

2

4. **[OK]** をクリックします。

アラーム カラムのグループ化

複数のフィールドが定義されると、カラムはブール演算子を使用して結合されます。

- **AND** 演算子は、選択したフィールドのすべての値に一致するレコードを返します。
- **OR** 演算子は、選択した任意のフィールドの値に一致するレコードを返します。

AND/OR 演算子を使用してフィルタ選択条件を使用するには、対応するフィールドをグループ化する必要があります。フィルタ ペインのアイテムに対しては 1 つのフィルタ式のみを作成できます。複数の式が必要な場合、そのアイテムをフィルタ ペインに再び追加する必要があります。

デフォルトで、グループ化されたフィールドには **AND** 演算子があります。

AND 演算子と **OR** 演算子は親ノードです。各親ノードの下にある選択されたフィールドは子ノードです。フィールド親ノードを子ノードにドラッグすることはできません。

アラーム カラムをグループ化するには

1. フィールドを右クリックし、**[グループ]** をクリックします。
2. フィールドを別のフィールドにドラッグします。

クエリー フィルタのコピーまたは移動

Alarm Pareto コントロールの複数のインスタンスがあり、複数のインスタンスについて同じフィルタを使用する場合は、定義済みフィルタをあるインスタンスからコピーまたは切り取りして、別のフィルタに貼り付けます。

フィルタをコピーするには

1. **Alarm Pareto** コントロールの最初のインスタンスで、フィルタを定義します。
2. フィルタを右クリックし、**[コピー]** をクリックします。フィルタを移動するには、**[切り取り]** をクリックします。
3. **Alarm Pareto** コントロールの最初のインスタンスを終了します。
4. **Alarm Pareto** コントロールの次のインスタンスを開き、**[クエリー フィルタ]** タブをクリックします。
5. 右ペインに矢印を位置付けします。選択したフィルタを右クリックし、**[貼り付け]** をクリックします。

[状況] カラムの値

フィルタ クエリーに **[状況]** カラムを追加する場合、**[フィルタの定義]** ダイアログ ボックスの **[値]** メニューから値を割り当てることができます。使用できる値を以下の表で説明します。

値	説明
ACK	すべてのシステム確認のクエリーを作成します。
ACK_ALM	すべての確認済みアラームに対するクエリーを作成します。
UNACK_ALM	すべての未確認アラームに対するクエリーを作成します。
ACK_RTN	通常状態に戻ったすべての確認済みアラームのクエリーを作成します。
UNACK_RTN	通常状態に戻った未確認アラームのクエリーを作成します。
All UNACK Records	すべての未確認レコードのクエリーを作成します。
All ACK Records	すべての確認レコードのクエリーを作成します。
All ALM Records	すべてのアラーム レコードのクエリーを作成します。
All RTN Records	通常状態に戻ったすべてのアラームのクエリーを作成します。

注記: 拡張サマリ アラーム モードのタグがアラームの作成に使用され、メインアラームが確認されたときにそのアラームも通常状態に戻ると、2つのレコードが作成されます。新しいアラームはすでに通常状態に戻っているため、最初のレコードは「確認され、通常状態に戻った」レコードです。2番目のレコードは確認され、メインアラームの確認に対応します。以前の ACK_ALM は ACK に変更されています。

さまざまなタイプのアラーム レコードの色の設定

アラーム レコードとイベント レコードの色を設定できます。

アラーム表示の色を設定するには

1. Alarm DB View コントロールを右クリックし、[プロパティ] をクリックします。[AlmDBViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [色] タブをクリックします。

3. [色] ボックスをクリックしてパレットを開き、以下のそれぞれの色を設定します。

オプション	説明
-------	----

アラーム復帰前景色	正常に復帰するアラームの記録前景色を設定します。
-----------	--------------------------

アラーム復帰背景色	正常に復帰するアラームの記録背景色を設定します。
-----------	--------------------------

イベントの前景色	イベント レコードの前景色を設定します。
----------	----------------------

イベント背景色	イベント レコードの背景色を設定します。
---------	----------------------

4. [アラーム優先度] ボックスに、アラーム表示の変化境界値を入力します。アラームの優先度に基づいて異なる色が表示されるように、境界値を割り当てることができます。デフォルトのアラーム優先度の最小値は 1、最大値は 999 です。

たとえば、優先度範囲 250 ～ 499 はオレンジ、優先度範囲 1 ～ 249 は赤になるように **Unack Alm Forecolor** を設定したとします。優先度 254 のアラームが発生した場合、そのアラームはオレンジ色のフォントで表示されます。優先度 12 のアラームが発生した場合、そのアラームは赤色のフォントで表示されます。

5. [色] ボックスをクリックしてパレットを開き、以下のそれぞれの色を設定します。

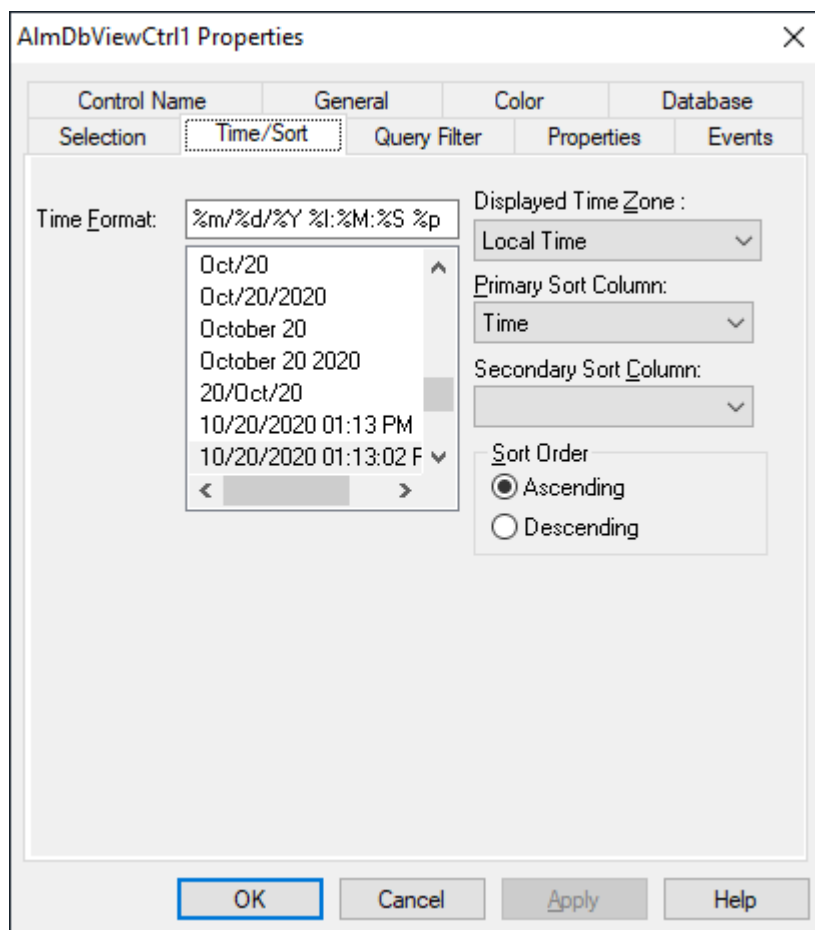
オプション	説明
未確認アラーム前景色	未確認アラームのそれぞれの色優先度範囲の前景色を設定します。
未確認アラーム背景色	未確認アラームのそれぞれの色優先度範囲の背景色を設定します。
確認アラーム前景色	確認アラームのそれぞれの色優先度範囲の前景色を設定します。
確認アラーム背景色	確認アラームのそれぞれの色優先度範囲の背景色を設定します。

6. [適用] をクリックします。

アラーム レコードのソート順の設定

アラーム レコードがコントロールでソートされる方法を制御できます。

1. Alarm DB View コントロールを右クリックし、[プロパティ] をクリックします。[AlmDBViewCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [時間/ソート] タブをクリックします。



3. [メインソートカラム] リストで、プライマリ ソート カラムの名前をクリックします。[カラムのソート] リストに表示されるのは、表示されるカラムのみです。目的のカラムが見つからない場合は、[全般] タブをクリックし、[カラムの詳細] からカラムを選択します。
4. [セカンダリ ソート カラム] リストで、セカンダリ ソート カラム名をクリックします。
5. [昇順] または [降順] を選択して、ソート順を設定します。
6. [OK] をクリックします。

ランタイムでの Alarm DB View コントロールの使用

コントロールが設定される方法に基づいて、以下の操作を行えます。

- データを取得およびリフレッシュする
- データをソートする
- データをフィルタする
- 行を選択する
- カラムをドラッグしてカラム順序を変更する
- すべてのカラムをデザイン時に保存した設定にリセットする

レコードのソート

表示でレコードをソートできます。すべての行をソートするには、見出しをクリックします。

コントロールを右クリックし、[ソート] をクリックして、[セカンダリ ソート] ダイアログ ボックスを表示します。このダイアログ ボックスで、昇順または降順で1つまたは複数のカラムをソートできます。

ソートするカラムを指定するには、カラム名の横にあるチェック ボックスをオンにします。[ソート順] 矢印ボタンを使用して、カラムを並べ替えることができます。

複数のアラーム イベントが同じタイムスタンプを持つ場合、これらのイベントは期待される順序で表示されない可能性があります。

たとえば、目的のソート順がアラーム状態を基準にした降順である場合:

1. [日付] と [状態] の両方のチェック ボックスをオンにします。
2. [状態] 行を選択します。
3. 上向きのソート順の矢印をクリックします。
4. [ソート タイプ] 領域で、[降順] をクリックします。
5. [OK] をクリックします。

ステータス バー情報の理解

ステータス バーには、コントロールの現在の状況が示されます。

- 左フレームには、接続されるサーバー名とデータベースが表示されます。
- 中央のフレームには、クエリーによって返されたレコード合計数のうち、表示されているレコードの数が表示されます。
- フレームの右側には、サーバーとの接続状況が表示されています。

Alarm DB View ActiveX プロパティの使用

スクリプトを使用して Alarm DB View コントロールのプロパティの値を直接設定するか、InTouch タグまたは I/O 参照に割り当てることができます。プロパティの設定の詳細については、「[ActiveX コントロールのスクリプト](#)」を参照してください。

色の値の設定の詳細については、「[ActiveX コントロールの色の設定](#)」を参照してください。

AckAlmBackColor プロパティ

確認されたアラーム背景色を取得または設定します。この設定は、確認済みアラームの個別の範囲の色設定を上書きします (AckAlmBackColorRange1～AckAlmBackColorRange4)。

タイプ

整数

デフォルト

白

構文

```
Object.AckAlmBackColor [= color]
```

値

color

背景色を決定する値または定数

AckAlmBackColorRange1 プロパティ

確認されたアラーム背景色を取得または設定します。優先度が 1～ColorPriorityRange1 の範囲で状況が「ACK_ALM」のレコードをこの色でコントロールに表示します。

タイプ

整数

デフォルト

白

構文

```
Object.AckAlmBackColorRange1 [= color]
```

値

color

背景色を決定する値または定数

AckAlmBackColorRange2 プロパティ

確認されたアラーム背景色を取得または設定します。優先度が ColorPriorityRange1～ColorPriorityRange2 の範囲で状況が「ACK_ALM」のレコードをこの色でコントロールに表示します。

タイプ

整数

デフォルト

白

構文

```
Object.AckAlmBackColorRange2 [= color]
```

値

color

背景色を決定する値または定数

AckAlmBackColorRange3 プロパティ

確認されたアラーム背景色を取得または設定します。優先度が ColorPriorityRange2～ColorPriorityRange3 の範囲で状況が「ACK_ALM」のレコードをこの色でコントロールに表示します。

タイプ

整数

デフォルト

白

構文

```
Object.AckAlmBackColorRange3 [= color]
```

値

color

背景色を決定する値または定数

AckAlmBackColorRange4 プロパティ

確認されたアラーム背景色を取得または設定します。優先度が ColorPriorityRange3～999 の範囲で状況が「ACK_ALM」のレコードをこの色でコントロールに表示します。

タイプ

整数

デフォルト

白

構文

```
Object.AckAlmBackColorRange4 [= color]
```

値

color

背景色を決定する値または定数

AckAlmForeColor プロパティ

確認されたアラーム前景色を取得または設定します。この設定は、確認済みアラームの個別の範囲の色設定を上書きします (AckAlmForeColorRange1～AckAlmForeColorRange4)。

タイプ

整数

デフォルト

黒

構文

```
Object.AckAlmForeColor [= color]
```

値

color

テキストの色を決定する値または定数

AckAlmForeColorRange1 プロパティ

確認されたアラーム前景色を取得または設定します。優先度が 1～ColorPriorityRange1 の範囲で状況が「ACK_ALM」のレコードをこの色でコントロールに表示します。

タイプ

整数

デフォルト

黒

構文

```
Object.AckAlmForeColorRange1 [= color]
```

値

color

テキストの色を決定する値または定数

AckAlmForeColorRange2 プロパティ

確認されたアラーム前景色を取得または設定します。優先度が ColorPriorityRange1～ColorPriorityRange2 の範囲で状況が「ACK_ALM」のレコードをこの色でコントロールに表示します。

タイプ

整数

デフォルト

黒

構文

```
Object.AckAlmForeColorRange2 [= color]
```

値

color

テキストの色を決定する値または定数

AckAlmForeColorRange3 プロパティ

確認されたアラーム前景色を取得または設定します。優先度が ColorPriorityRange2～ColorPriorityRange3 の範囲で状況が「ACK_ALM」のレコードをこの色でコントロールに表示します。

タイプ

整数

デフォルト

黒

構文

```
Object.AckAlmForeColorRange3 [= color]
```

値

color

テキストの色を決定する値または定数

AckAlmForeColorRange4 プロパティ

確認されたアラーム前景色を取得または設定します。優先度が ColorPriorityRange3～999 の範囲で状況が「ACK_ALM」のレコードをこの色でコントロールに表示します。

タイプ

整数

デフォルト

黒

構文

```
Object.AckAlmForeColorRange4 [= color]
```

値

color

テキストの色を決定する値または定数

AckRtnBackColor プロパティ

正常に復帰する確認済みアラームの背景色を取得または設定します（ACK_RTN）。

タイプ

整数

デフォルト

白

構文

```
Object.AckRtnBackColor [= color]
```

値

color

背景色を決定する値または定数

AckRtnForeColor プロパティ

正常に復帰する確認済みアラームのテキスト色を取得または設定します（ACK_RTN）。

タイプ

整数

デフォルト

青

構文

```
Object.AckRtnForeColor [= color]
```

値

color

テキストの色を決定する値または定数

AlmRtnBackColor プロパティ

返されたアラーム背景色を取得または設定します。この色は状況が「ALM_RTN」のレコードに適用されます。

タイプ

整数

デフォルト

白

構文

```
Object.AlmRtnBackColor [= color]
```

値

color

背景色を決定する値または定数

AlmRtnForeColor プロパティ

返されたアラーム前景色を取得または設定します。この色は状況が「ALM_RTN」のレコードに適用されます。

タイプ

整数

デフォルト

青

構文

```
Object.AlmRtnForeColor [= color]
```

値

color

テキストの色を決定する値または定数

Authentication プロパティ

[データベース] タブで選択された認証タイプを返します。

タイプ

メッセージ

構文

```
AType = AlarmDBCtrl1.Authentication;
```

値

AType

メッセージ値 ('SQL 認証' または 'Windows 認証')。

AuthenticationMode プロパティ

認証モードを設定します。

タイプ

整数型

構文

```
Object.AuthenticationMode = [Int]
```

値

Int

0 または 1 の整数値。0 は SQL 認証を示し、1 は Windows 認証を示します。

AutoConnect プロパティ

ランタイムでコントロールがデータベースに自動的に接続されるかどうかを決定する値を取得または設定します。

データ タイプ

整数型

デフォルト

False

構文

```
Object.AutoConnect [= Integer]
```

値

Integer

ランタイムでコントロールがデータベースに接続されるかどうかを指定する整数式

True = データベースに接続します。

False = (デフォルト) データベースに接続しません。

備考

データベースに接続するには、Connect() メソッドを明示的に呼び出す必要があります。

ColorPriorityRange1 プロパティ

アラームが表示される優先度範囲の境界を設定します。このプロパティの値は 1 より大きく、ColorPriorityRange2 の値よりも小さい値を指定する必要があります。

タイプ

整数

デフォルト

250

構文

```
Object.ColorPriorityRange1 [= integer または priority]
```

ColorPriorityRange2 プロパティ

アラームが表示される優先度範囲の境界を設定します。このプロパティには、ColorPriorityRange1 より大きく、ColorPriorityRange3 よりも小さい値を指定する必要があります。

タイプ

整数

デフォルト

500

構文

```
Object.ColorPriorityRange2 [= integer または priority]
```

ColorPriorityRange3 プロパティ

アラームが表示される優先度範囲の境界を設定します。このプロパティの値には、ColorPriorityRange2 より大きく、999 よりも小さい値を指定する必要があります。

タイプ

整数

デフォルト

750

構文

```
Object.ColorPriorityRange3 [= integer または priority]
```

ColumnResize プロパティ

カラムのサイズ変更ができるかどうかを決定する値を返すか、または設定します。

タイプ

論理型

デフォルト

True

構文

```
Object.ColumnResize [= Discrete]
```

値

Discrete

True = (デフォルト) ランタイムにカラムのサイズを変更できます。

False = カラムのサイズは変更できません。

ConnectStatus プロパティ

接続状況を返します。このプロパティは読み取り専用です。

データタイプ

メッセージ

構文

```
Object.ConnectStatus
```

値

接続中 = コントロールはデータベースに接続されています。

未接続 = コントロールはデータベースに接続されていません。

処理中 = コントロールをデータベースに接続しています。

例

コントロール名は AlmDbView1、タグ変数はメッセージ型タグ変数です。

```
TagName = #AlmDbView1.ConnectStatus;
```

CustomMessage プロパティ

アラーム レコードをアラーム データベースから取得できないとき、Alarm DB View コントロールが表示するメッセージを取得または設定します。

タイプ

メッセージ

デフォルト

表示するアイテムがありません。

構文

```
Object.CustomMessage [= string]
```

DatabaseName プロパティ

接続するデータベースを指定します。

タイプ

メッセージ

構文

```
Object.DatabaseName [= text]
```

DisplayMode プロパティ

コントロールの表示モードを返します。表示モードによって、アラームのみ、イベントのみ、またはアラームとイベントの両方が表示されます。このプロパティは読み取り専用です。

タイプ

文字列型

デフォルト

Alarms & Events History (アラーム／イベント履歴)

構文

```
Object.DisplayMode
```

備考

可能な値は以下のとおりです。

Alarms & Events History（アラーム／イベント履歴）

Alarms History（アラーム履歴）

Events History（イベント履歴）

例

コントロール名は `AlmDbView1`、タグ変数はメッセージ型タグ変数です。

```
tag = #AlmDbView1.DisplayMode;
```

DisplayedTimeZone プロパティ

表示されるタイムゾーンを取得または設定します。

タイプ

文字列型

デフォルト

ローカル時間

構文

```
Object.DisplayedTimeZone [= message]
```

備考

可能な値は以下のとおりです。

GMT - アラーム タイム スタンプでは、グリニッジ標準時が使用されます。

Local Time - アラーム時刻スタンプは、Alarm DB View コントロールをホストするクライアントのローカル時間で表示されます。

Origin Time - アラーム時刻スタンプは、アラーム プロバイダのローカル時間で表示されます

Duration プロパティ

開始時間と終了時間を設定する期間を取得または設定します。

タイプ

メッセージ

デフォルト

Last Hour

構文

```
Object.Duration [= text]
```

値

text

期間を含む文字列式。このプロパティには、以下の文字列の 1 つを持つ必要があります。

Last Minute（1 分前）

Last 5 Minutes（5 分前）

Last 15 Minutes（5 分前）

Last Half Hour（30 分前）

Last Hour (1 時間前)
Last 2 Hours (2 時間前)
Last 4 Hours (2 時間前)
Last 8 Hours (2 時間前)
Last 12 Hours (2 時間前)
Last Day (1 日前)
過去 2 日間
過去 3 日間
先週
過去 2 週間
Last 30 days (30 日前)
Last 90 days (30 日前)

EndTime プロパティ

終了時間を返すか、設定します。

タイプ

メッセージ

構文

```
Object.EndTime [= text]
```

値

text

終了時間を評価する文字列式。返される文字列は常に MM/DD/YYYY HH:MM:SS の形式です。同じ形式は、文字列の値を設定するためにも必要になります。このプロパティは、1970 年 1 月 1 日 0 時 00 分から 2038 年 1 月 18 日の 19 時 14 分 07 秒の間であれば、どのタイムゾーンでも対応できます。

EventBackColor プロパティ

イベントアラーム背景色を取得または設定します。この色は状況が「EVT_EVT」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

白

構文

```
Object.EventBackColor [= color]
```

値

color

背景色を決定する値または定数

EventForeColor プロパティ

イベントアラーム前景色を取得または設定します。この色は状況が「EVT_EVT」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

赤

構文

```
Object.EventForeColor [= color]
```

値

color

テキストの色を決定する値または定数

FilterFavoritesFile プロパティ

フィルタ設定ファイルを取得または設定します。このファイルは、**[フィルタ設定]** ダイアログ ボックスでフィルタ設定を読み取りまたは書き込みするために使用されます。

タイプ

文字列型

デフォルト

Null

構文

```
Object.FilterFavoritesFile [= String]
```

FilterMenu プロパティ

ショートカット メニューに **[フィルタ]** メニュー項目を表示するかどうかを決定する値を取得または設定します。

タイプ

論理型

デフォルト

True

構文

```
Object.FilterMenu [= Discrete]
```

値

True の場合、**[フィルタ]** メニュー項目が表示されます (デフォルト)。

False の場合、**[フィルタ]** メニュー項目は表示されません。

FilterName プロパティ

現在のフィルタ名を返します (フィルタ名が存在する場合)。

タイプ

文字列 (読み取り専用)

デフォルト

Null

構文

```
Object.FilterName [= String]
```

FromPriority プロパティ

コントロールの優先度始点の値を取得または設定します。

タイプ

整数

デフォルト

1

構文

```
Object.FromPriority [= integer]
```

備考

このプロパティを使用して、表示されるアラーム レコードをフィルタできます。たとえば、このプロパティを 760 に設定すると、プロパティが 760～ToPriority プロパティ値のアラームだけが表示されます。

GroupExactMatch プロパティ

GroupExactMatch プロパティが True の場合、GroupName プロパティ値と正確に一致するアラーム グループ名を持つアラームのみが表示されます。False の場合、グループ名で指定する必要があるのは、フィルタするアラーム グループ名の一部だけです。

タイプ

論理型

デフォルト

False

構文

```
Object.GroupExactMatch [= discrete]
```

備考

このプロパティを GroupName プロパティと共に使用して、Alarm DB View コントロールをフィルタします。

例

次に例を示します。

```
#AlarmDBViewCtrl1.GroupName = "Group"  
#AlarmDBViewCtrl1.GroupExactMatch = 0;  
#AlarmDBViewCtrl1.Refresh();
```

GroupName プロパティ

現在の Alarm DB View コントロール用のアラーム グループ名を取得または設定します。

タイプ

文字列型

デフォルト

(なし)

構文

```
Object.GroupName [= GroupName]
```

備考

このプロパティを「GroupA」に設定して、表示に対して再度クエリーを実行すると、GroupA アラームグループに属するタグ変数のみが表示されます。

MaxRecords プロパティ

取得する最大レコード数を返すか、設定します。

タイプ

整数

デフォルト

100

構文

```
Object.MaxRecords [=integer]
```

値

integer

指定の時間に取得されるレコード数を指定する整数式。最大レコード数は 1～1000 までの範囲です。最高のパフォーマンスを得るためには、この値をできるだけ小さくしてください。

Password プロパティ

データを取得するための SQL Server パスワードを返すか、設定します。

タイプ

メッセージ

構文

```
Object.Password [= text]
```

値

text

パスワードを評価する文字列式

PrimarySort プロパティ

アラーム表示をソートするために使用されるプライマリ カラム名を取得または設定します。

タイプ

メッセージ

デフォルト

(なし)

構文

```
Object.PrimarySort [= message]
```

ProviderExactMatch プロパティ

ProviderExactMatch プロパティが **True** の場合、ProviderName プロパティ値と正確に一致するアラーム プロバイダ名を持つアラームだけが表示されます。**False** の場合、プロバイダ名で指定する必要があるのは、フィルタするアラーム プロバイダ名の一部だけです。

タイプ

論理型

デフォルト

False

構文

```
Object.ProviderExactMatch [= discrete]
```

備考

このプロパティを ProviderName プロパティと共に使用して、Alarm DB View コントロールをフィルタします。

例

次に例を示します。

```
#AlarmDBViewCtrl1.ProviderName = "Provider"  
#AlarmDBViewCtrl1.ProviderExactMatch = 0;  
#AlarmDBViewCtrl1.Refresh();
```

ProviderName プロパティ

現在の Alarm DB View コントロール用のアラーム プロバイダ名フィルタを取得または設定します。

タイプ

文字列型

デフォルト

(なし)

構文

```
Object.ProviderName [= ProviderName]
```

備考

あるタグ変数が Provider1 アラーム プロバイダに属する場合、このプロパティを「Provider1」に設定し、表示のクエリを再実行すると、Provider1 アラーム プロバイダに属するタグ変数だけが表示されます。

QueryTimeZoneName プロパティ

特定の時間がクエリに使用されるとき、タイムゾーンを取得または設定します。

タイプ

論理型

デフォルト

False

構文

```
Object.QueryTimeZone [= Discrete]
```

値

True = GMT

False = 標準時間。アラーム プロバイダのローカル時間です。

RefreshMenu プロパティ

ショートカット メニューに [リフレッシュ] メニュー項目を表示するかどうかを決定する値を取得または設定します。

タイプ

論理型

デフォルト

True

構文

```
Object.RefreshMenu [= Discrete]
```

値

True の場合、[リフレッシュ] メニュー項目が表示されます (デフォルト)。

False の場合、[リフレッシュ] メニュー項目は表示されません。

ResetMenu プロパティ

ショートカット メニューに [リセット] メニュー項目を表示するかどうかを決定する値を取得または設定します。

タイプ

論理型

デフォルト

True

構文

```
Object.ResetMenu [= Discrete]
```

値

True の場合、[リセット] メニュー項目が表示されます (デフォルト)。

False の場合、[リセット] メニュー項目は表示されません。

RowCount プロパティ

コントロールで表示されているレコード数を返します。このプロパティは読み取り専用です。

タイプ

整数

構文

```
Object.RowCount
```

例

コントロール名は AlmPidView1、タグ変数は整数型タグ変数です。

```
tagname = #AlmPidView1.RowCount;
```

RowSelection プロパティ

ランタイムで行を選択できるかどうかを決定する値を返すか、設定します。

タイプ

論理型

デフォルト

True

構文

```
Object.RowSelection [= Discrete]
```

値

論理型

True = 行を選択できます (デフォルト)。

False = 行を選択できません。

備考

行を選択できない場合、Click イベントまたは DoubleClick イベントは生成されません。

SecondarySort プロパティ

アラーム表示をソートするために使用されるセカンダリ カラム名を取得または設定します。

タイプ

メッセージ

デフォルト

(なし)

構文

```
Object.SecondarySort [= text]
```

ServerName プロパティ

データを取得するためにコントロールが接続するサーバー名を返すか、設定します。

タイプ

メッセージ

構文

```
Object.ServerName [= text]
```

ShowFetch プロパティ

取得ボタンを表示するかどうかを決定する値を返すか、設定します。

タイプ

論理型

デフォルト

True

構文

```
Object.ShowFetch [= Discrete]
```

値

Discrete

True = 取得ボタンが表示されます（デフォルト）。

False = 取得ボタンが表示されません。

ShowGrid プロパティ

グリッド線を表示するかどうかを決定する値を返すか、設定します。

タイプ

論理型

デフォルト

False

構文

```
Object.ShowGrid [= Discrete]
```

値

Discrete

True = グリッド線が表示されます。

False = グリッド線が表示されません（デフォルト）。

ShowHeading プロパティ

カラム見出しを表示するかどうかを決定する値を返すか、設定します。

タイプ

論理型

デフォルト

True

構文

```
Object.ShowHeading [= Discrete]
```

値

Discrete

True = カラム見出しが表示されます（デフォルト）。

False = カラム見出しが表示されません。

ShowMessage プロパティ

アラーム データベースにレコードがないとき、「表示するアイテムがありません。」に対するカスタマイズしたメッセージを表示するかどうかを決定します。

タイプ

論理型

デフォルト

False

構文

```
Object.ShowMessage [= discrete]
```

ShowStatusBar プロパティ

ステータス バーを表示するかどうか決定する値を返すか、または設定します。

タイプ

論理型

デフォルト

True

構文

```
Object.ShowStatusBar [= Discrete]
```

値

Discrete

True = ステータス バーが表示されます（デフォルト）。

False = ステータス バーは表示されません。

SilentMode プロパティ

コントロールをサイレント モードにするかどうかを決定する値を取得または設定します。

タイプ

論理型

デフォルト

False

構文

```
Object.SilentMode [= Discrete]
```

値

True = サイレント モードをオンにします。

False = サイレント モードをオフにします（デフォルト）。

SortMenu プロパティ

ショートカット メニューに [ソート] メニュー項目を表示するかどうかを決定する値を返すか、設定します。

タイプ

論理型

デフォルト

True

構文

```
Object.SortMenu [= Discrete]
```

値

論理型式

True = [ソート] メニュー項目が表示されます（デフォルト）。

False = [ソート] メニュー項目は表示されません。

SortOrder プロパティ

ソートされるカラムに従って、アラームのソート順を取得または設定します（プライマリ ソート カラム）。

タイプ

論理型

デフォルト

True

構文

```
Object.SortOrder [= discrete]
```

値

論理値式

True = 昇順

False = 降順

SpecificTime プロパティ

コントロールが **StartTime** プロパティと **EndTime** プロパティを使用するかどうか、または **Duration** プロパティの値に基づいて開始時間と終了時間を計算するかどうかを決定する値を返すか、設定します。

タイプ

論理型

デフォルト

False

構文

```
Object.SpecificTime [= Discrete]
```

値

True = StartTime プロパティと EndTime プロパティを使用します。

False = "Duration" プロパティに基づいて StartTime と EndTime が計算されます（デフォルト）。

StartTime プロパティ

開始時間を返すか、設定します。

タイプ

メッセージ

構文

```
Object.StartTime [= text]
```

値

text

開始時間を評価する文字列式。返される文字列は常に MM/DD/YYYY HH:MM:SS の形式です。同じ形式は、文字列の値を設定するためにも必要になります。このプロパティは、1970 年 1 月 1 日 0 時 00 分から 2038 年 1 月 18 日の 19 時 14 分 07 秒の間ならば、どのタイムゾーンでも対応できます。

Time プロパティ

表示に使用される時間形式を取得および設定します。

タイプ

メッセージ

デフォルト

```
%m/%d/%Y %l:%M:%S %p
```

構文

```
Object.Time [= message]
```

備考

時間形式文字列の詳細については、「[アラーム レコードに対して表示される時間形式とタイムゾーンの設定](#)」を参照してください。

ToPriority プロパティ

コントロールの優先度終点の値を取得または設定します。

タイプ

整数

デフォルト

999

構文

```
Object.ToPriority [= integer]
```

備考

このプロパティを使用して、表示されるアラーム レコードをフィルタします。たとえば、このプロパティを 900 に設定すると、プロパティが **FromPriority** プロパティ値～ 900 のアラームだけが表示されます。

TotalRowCount プロパティ

現在のクエリーに対するレコードの合計数を返します。このプロパティは読み取り専用です。

タイプ

整数

構文

```
Object.TotalRowCount
```

備考

行カウントは現在のクエリーで返された行数であり、取得されたレコード数が **MaxRecords** プロパティよりも少ない場合を除いて、通常は **MaxRecords** プロパティと同じになります。たとえば、特定の条件下で 950 のレコードがあり、**MaxRecords** プロパティが 100 の場合、最終ページのレコード数は 50 になり、行数も 50 になります。同じ例で、**TotalRowCount** プロパティは必ず 950 になります。

例

コントロール名は **AlmDbView1**、タグ変数は整数型タグ変数です。

```
tagname = #AlmDbView1.TotalRowCount;
```

UnAckAlmBackColor プロパティ

未確認のアラーム背景色を取得または設定します。この色は状況が「**UNACK_ALM**」のコントロールに表示されるすべてのレコードに適用されます。**UnAckAlmBackColorRange1** プロパティ～**UnAckAlmBackColorRange4** プロパティの値による設定を上書きします。

タイプ

整数

デフォルト

白

構文

```
Object.UnAckAlmBackColor [= color]
```

値

color

特定オブジェクトの色を決定する値または定数

UnAckAlmBackColorRange1 プロパティ

未確認のアラーム背景色を取得または設定します。この色は優先度が 1～**ColorPriorityRange1** の範囲で状況が「**UNACK_ALM**」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

白

構文

```
Object.UnAckAlmBackColorRange1 [= color]
```

値

color

特定オブジェクトの色を決定する値または定数

UnAckAlmBackColorRange2 プロパティ

未確認のアラーム背景色を取得または設定します。この色は優先度が ColorPriorityRange1～ColorPriorityRange2 の範囲で状況が「UNACK_ALM」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

白

構文

```
Object.UnAckAlmBackColorRange2 [= color]
```

値

color

特定オブジェクトの色を決定する値または定数

UnAckAlmBackColorRange3 プロパティ

未確認のアラーム背景色を取得または設定します。この色は優先度が ColorPriorityRange2～ColorPriorityRange3 の範囲で状況が「UNACK_ALM」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

白

構文

```
Object.UnAckAlmBackColorRange3 [= color]
```

値

color

特定オブジェクトの色を決定する値または定数

UnAckAlmBackColorRange4 プロパティ

未確認のアラーム背景色を取得または設定します。この色は優先度が ColorPriorityRange3～999 の範囲で状況が「UNACK_ALM」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

白

構文

```
Object.UnAckAlmBackColorRange4 [= color]
```

値

color

特定オブジェクトの色を決定する値または定数

UnAckAlmForeColor プロパティ

未確認のアラーム テキスト色を取得または設定します。この色は状況が「UNACK_ALM」のコントロールに表示されるすべてのレコードに適用されます。UnAckAlmForeColorRange1 プロパティ～UnAckAlmForeColorRange4 プロパティの値による設定を上書きします。

タイプ

整数

デフォルト

赤

構文

```
Object.UnAckAlmBackColor [= color]
```

値

color

テキストの色を決定する値または定数

UnAckAlmForeColorRange1 プロパティ

未確認のアラーム前景色を取得または設定します。この色は優先度が 1～ColorPriorityRange1 の範囲で状況が「UNACK_ALM」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

赤

構文

```
Object.UnAckAlmForeColorRange1 [= color]
```

値

color

特定オブジェクトの色を決定する値または定数

UnAckAlmForeColorRange2 プロパティ

未確認のアラーム前景色を取得または設定します。この色は優先度が ColorPriorityRange1～ColorPriorityRange2 の範囲で状況が「UNACK_ALM」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

赤

構文

```
Object.UnAckAlmForeColorRange2 [= color]
```

値

color

特定オブジェクトの色を決定する値または定数

UnAckAlmForeColorRange3 プロパティ

未確認のアラーム前景色を取得または設定します。この色は優先度が ColorPriorityRange2～ColorPriorityRange3 の範囲で状況が「UNACK_ALM」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

赤

構文

```
Object.UnAckAlmForeColorRange3 [= color]
```

値

color

特定オブジェクトの色を決定する値または定数

UnAckAlmForeColorRange4 プロパティ

未確認のアラーム前景色を取得または設定します。この色は優先度が ColorPriorityRange3～999 の範囲で状況が「UNACK_ALM」のコントロールに表示されるレコードに適用されます。

タイプ

整数

デフォルト

赤

構文

```
Object.UnAckAlmForeColorRange4 [= color]
```


値

color

特定オブジェクトの色を決定する値または定数

UnAckOrAlarmDuration プロパティ

間隔カラムは「未確認間隔」または「アラーム間隔」のいずれかを表示します。FALSE (0) は未確認間隔、TRUE (1) はアラーム間隔です。

タイプ

整数

デフォルト

False

構文

```
Object.UnAckOrAlarmDuration [= integer]
```

UserID プロパティ

データを取得するためにコントロールが SQL Server に接続する際に使用するユーザー ID を返すか、設定します。

タイプ

メッセージ

構文

```
Object.UserID [= text]
```

値

text

ユーザー ID を評価する文字列式

Alarm DB View ActiveX メソッドの使用

Alarm DB View ActiveX メソッドを使用すると、以下の操作を行えます。

- データベース接続を制御します。
- アラーム データベースからレコードを取得します。
- アラーム情報を取得します。
- 表示の外観をリセットします。
- アラーム レコードをソートする
- [コンテキスト] メニューを表示します。
- フィルタ設定にアクセスします。

メソッドの呼び出しの詳細については、「[ActiveX コントロールのスク립ト](#)」を参照してください。

アラーム データベース接続の制御

アラーム データベースに接続するには、Connect() メソッドを使用します。接続を解除するには、Disconnect() メソッドを使用します。

Connect()

コントロールをデータベースに接続し、接続が成功すると 1～MaxRecords の範囲にあるレコードセットを表示します。

構文

```
Object.Connect();
```

例

コントロールの名前は AlmDbView1 です。

```
#AlmDbView1.Connect();
```

Disconnect()

コントロールをデータベースから接続解除します。

構文

```
Object.Disconnect();
```

例

コントロールの名前は AlmDbView1 です。

```
#AlmDbView1.Disconnect();
```

アラーム データベースからレコードを取得

以下のメソッドを使用して、データベースからレコードを取得し、表示を更新します。

- [SelectQuery\(\)](#)
- [GetPrevious\(\)](#)
- [GetNext\(\)](#)
- [Refresh\(\)](#)

SelectQuery()

現在の表示を .xml ファイルで指定されたクエリー名に設定します。

構文

```
Object.SelectQuery("QueryName");
```

パラメータ

QueryName

フィルタ設定ファイルで定義されたクエリー名

例

この例では、現在 AlmDbView1 コントロールに関連付けられているフィルタ設定ファイルの「HighPriority」と呼ばれるクエリーによって定義されるフィルタ条件を適用しています。

```
#AlmDbView1.SelectQuery("HighPriority");
```

GetPrevious()

データベースから以前のレコードセットを取得します（存在する場合）。

構文

```
Object.GetPrevious();
```

例

コントロールの名前は **AlmDbView1** です。

```
#AlmDbView1.GetPrevious();
```

GetNext()

データベースから次のレコードセットを取得します（存在する場合）。

構文

```
Object.GetNext()
```

例

コントロールの名前は **AlmDbView1** です。

```
#AlmDbView1.GetNext();
```

Refresh()

データベースからのコントロールを更新し、接続が成功すると **1**～**MaxRecords** の範囲にあるレコードセットを表示します。

構文

```
Object.Refresh
```

備考

Refresh() メソッドを呼び出すことによって **Alarm DB View** コントロールの表示の更新を初期化した後、更新が完了するまで、**RowCount** プロパティと **TotalRowCount** プロパティの値は **-1** に変化します（つまり、関連するすべてのレコードがデータベースから取得されます）。更新が完了すると、両方のプロパティは現在の正しい行数で更新されます。

Refresh() メソッドは非同期で機能します。すぐに呼び出しスクリプトにコントロールを返し、バックグラウンドで作業を続行します。つまり、**Refresh()** を呼び出した後ですぐに **RowCount** と **TotalRowCount** の値をクエリーすると、更新がまだ完了していない時点で値がクエリーされているため、**-1** が返される可能性が非常に高いです。正しい値を取得する方法の **1** つは、スクリプトを使用して、いずれかのプロパティの値が **-1** から異なる値にいつ変わるかを判断します。これで、正しい値が使用できるようになったことがわかります。

例

コントロールの名前は **AlmDbView1** です。

```
#AlmDbView1.Refresh();
```

アラーム情報の取得

次のメソッドを使用すると、データベースから特定のアラームに関するレコードを取得できます。

- [GetItem\(\) メソッド](#), [GetItem\(\) メソッド](#)
- [GetSelectedItem\(\) メソッド](#)

GetItem() メソッド

指定された行とカラムのデータを文字列として返します。

構文

```
Object.GetItem(Integer, Message)
```

パラメータ

Integer

コントロールの特定の行を評価する整数式です。

Message

コントロールのカラム名を評価する文字列式です。

例

コントロール名は `AlmDbView1` です。タグはメッセージ型タグとして定義されます。

```
tag = #AlmDbView1.GetItem(1, "Group");
```

GetSelectedItem() メソッド

選択した行、指定したカラムのデータを文字列で返します。

構文

```
Object.GetSelectedItem(Message)
```

パラメータ

Message

コントロールのカラム名を評価する文字列式

例

コントロール名は `AlmDbView1` です。タグはメッセージ型タグとして定義されます。

```
tag = #AlmDbView1.GetSelectedItem("State");
```

アラーム レコードのソート

以下のメソッドを使用すると、アラーム レコードをソートして、カラムのサイズ変更をリセットできます。

- [SortOnCol\(\) メソッド](#)
- [ShowSort\(\) メソッド](#), [ShowSort\(\) メソッド](#)
- [Reset\(\) メソッド](#)

SortOnCol() メソッド

表示されるアラーム レコードで、プライマリ ソートを実行します

構文

```
Object.SortOnCol(Message, Integer)
```

パラメータ

Message

コントロールのカラム名を評価する文字列式

整数型

使用するソート順。0 = 昇順、1 = 降順です。

例

コントロールの名前は `AlmDbView1` です。

```
tag = #AlmDbView1.SortOnCol("名前",1);
```

ShowSort() メソッド

SortMenu プロパティが有効である場合、[セカンダリ ソート] ダイアログ ボックスを表示します。

構文

```
Object.ShowSort()
```

例

コントロールの名前は AlmDbView1 です。

```
#AlmDbView1.ShowSort();
```

Reset() メソッド

すべてのカラムを、デザイン時に保存した設定に戻します。

構文

```
Object.Reset()
```

例

コントロールの名前は AlmDbView1 です。

```
#AlmDbView1.Reset();
```

コンテキスト メニューの表示

ショートカット メニューを表示するには、ShowContext() メソッドを使用します。

ShowContext() メソッド

RefreshMenu、ResetMenu、SortMenu の 3 つのプロパティのうち 1 つを有効にすると、ショートカット メニューが表示されます。

構文

```
Object.ShowContext()
```

例

コントロールの名前は AlmDbView1 です。

```
#AlmDbView1.ShowContext();
```

フィルタ設定へのアクセス

ShowFilter() メソッドを使用すると、[ファイル設定] ダイアログ ボックスを表示できます。

ShowFilter() メソッド

[フィルタ設定] ダイアログ ボックスを表示します。

構文

```
Object.ShowFilter()
```

例

コントロールの名前は AlmDbView1 です。

```
#AlmDbView1.ShowFilter();
```

その他の情報の表示

AboutBox() メソッドを使用して、[バージョン情報] ダイアログ ボックスを表示します。

AboutBox() メソッド

[バージョン情報] ダイアログ ボックスを表示します。

構文

```
Object.AboutBox()
```

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.AboutBox();
```

メソッドとプロパティを使用するときのエラー処理

SilentMode プロパティを使用すると、コントロールをサイレントモードにするかどうかを決定できます。コントロールがサイレントモードである場合、エラー メッセージは表示されません。エラーを確認するには、GetLastError() メソッドを呼び出し、エラー メッセージを返します。

GetLastError() メソッド

Alarm DB View コントロールがサイレントモードである場合、最後のエラー メッセージを返します。

構文

```
Object.GetLastError()
```

例

コントロール名は AlmDbView1 であり、タグ名は変数または文字列として定義されています。

```
Tagname = #AlmDbView1.GetLastError();
```

Alarm DB View ActiveX イベントを使用したスクリプトのトリガ

マウス クリックやダブルクリックなどの Alarm DB View コントロール イベントに、QuickScript を割り当てることができます。イベントが発生すると、QuickScript が実行されます。

Alarm DB View コントロールは、以下のイベントをサポートしています。

- クリック
- DoubleClick
- ShutDown
- StartUp

Click イベントは ClicknRow というパラメータを 1 つ持ち、これはランタイム時にクリックされた行を識別します。

DoubleClick イベントは DoubleClicknRow というパラメータを 1 つ持ち、これはランタイム時にダブルクリックされた行を識別します。

Click イベントおよび DoubleClick イベントはゼロを基準としています。Click イベントまたは DoubleClick イベント（あるいはその両方）が発行される時、表示されるバー カウントは 0 から始まります。

注: ユーザー インターフェイス メソッドが StartUp イベントから呼び出される場合、コントロールはまだ表示されていないため、Alarm DB View コントロールでは無視されます。このようなメソッドには、ShowSort()、ShowContext()、GetSelectedItem()、GetNext()、GetPrevious()、および AboutBox() があります。

ActiveX イベントのスクリプトの詳細については、「[ActiveX コントロールのスクリプト](#)」を参照してください。

複数のタグにわたるアラーム分散の分析

Alarm Pareto ActiveX コントロールを使用すると、指定の生産システムで最も頻繁に発生するアラームとイベントを分析できます。また、発生の中の期間で、アラームの頻度を分析できます。

Alarm Pareto コントロールには分析機能があるので、生産システムの最も大きな問題を明らかにできます。Alarm Pareto コントロールは、最も重大な改良を行うために、作業の重点をどこに置けばよいのか知るのに役立ちます。

Alarm Pareto コントロールでは、アラーム操作時間を表す棒グラフが表示されます。

ActiveX コントロールの詳細については、「[ActiveX コントロール](#)」を参照してください。

Alarm Pareto ActiveX コントロールの設定

Alarm Pareto コントロールを設定するとき、次の指定ができます。

- アラーム データベースへの接続
- パレート コントロールの外観（色とフォントを含む）
- ランタイムにユーザーがアクセスできる機能
- グラフに表示されるアラームおよび表示方法

アラーム データベースへの接続の設定

Alarm Pareto コントロールとアラーム データベースの間の接続を設定する必要があります。

管理者アカウントではなく、アラーム データベースへの適切な読み取り専用アクセスのあるアカウントを使用することが推奨されます。

アラーム データベースへの接続を設定するには

1. Alarm Pareto コントロールを右クリックし、**[プロパティ]** をクリックします。
 [AlarmPareto のプロパティ] ダイアログ ボックスが表示されます。
2. **[データベース]** タブをクリックします。

The screenshot shows the 'AlarmPareto2 Properties' dialog box with the 'Database' tab selected. The 'Authentication' dropdown is set to 'Windows Authentication'. The 'Server Name' dropdown is empty. The 'Database' text field contains 'WwAlmDb'. The 'Credentials' dropdown is set to 'Credential1'. There is an 'Auto Connect' checkbox which is unchecked, and a 'Test Connection' button. At the bottom are 'OK', 'Cancel', 'Apply', and 'Help' buttons.

3. 接続を設定します。以下の手順を実行します。

- a. [認証] リストで認証方法（**SQL Server 認証**または **Windows 認証**を選択します（デフォルトは Windows 認証）。

注記: Windows 認証は、SQL 認証よりも優れたアプリケーションセキュリティを提供できます。この理由から、Windows 認証から SQL 認証に切り替えると、Windows 認証を使用することを推奨するポップアップダイアログが表示されます。この警告を無視して SQL 認証を使用する場合は、[OK] をクリックします。同様のメッセージが OCMC Log Viewer に記録されます。

- b. [サーバー名] ボックスに、アラーム データベースがインストールされているコンピュータのノード名を入力します。
- c. [データベース名] ボックスにアラーム データベースの名前を入力します。
- d. [資格情報] ドロップダウンから認証に使用する資格情報を選択します。

注記:

- [資格情報] ドロップダウンは、SQL Server 認証タイプが選択されている場合にのみ有効になります。Windows 認証では、現在ログインしているユーザーの資格情報が使用され、ユーザー名とパスワードフィールドが無効になります。
- スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取

得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報マネージャから取得されます。詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[資格情報マネージャの操作](#)」を参照してください。

4. WindowViewer が起動したときに、Alarm Pareto コントロールをアラーム データベースに自動的に接続する場合は、[自動接続] チェック ボックスをオンにします。

[自動接続] チェック ボックスをオンにしない場合は、Connect() メソッドを明示的に呼び出して、Alarm Pareto コントロールがアラーム データベースに接続されるよう設定する必要があります。Connect メソッドの詳細については、「[Connect\(\) メソッド](#)」を参照してください。

5. [接続テスト] をクリックして、アラーム データベースへの接続性を確認します。接続が正常に確立されたことを示すメッセージが表示されます。
6. [適用] をクリックします。

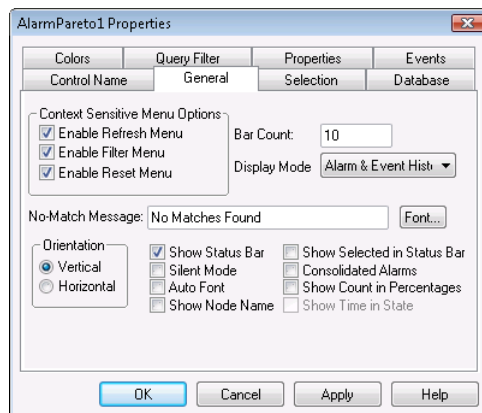
Alarm Pareto コントロールの外観と色を設定

Alarm Pareto コントロールの外観を設定できます。以下の操作を実行できます。

- ステータス バーを含めます。
- パレート グラフの棒の方向を設定します。
- グラフの棒の説明を含めます。
- Alarm Pareto グラフの色を選択します。

Alarm Pareto コントロールの外観を設定するには

1. Alarm Pareto コントロールを右クリックし、[プロパティ] をクリックします。[AlarmPareto のプロパティ] ダイアログ ボックスが表示されます。
2. [一般] タブをクリックします。



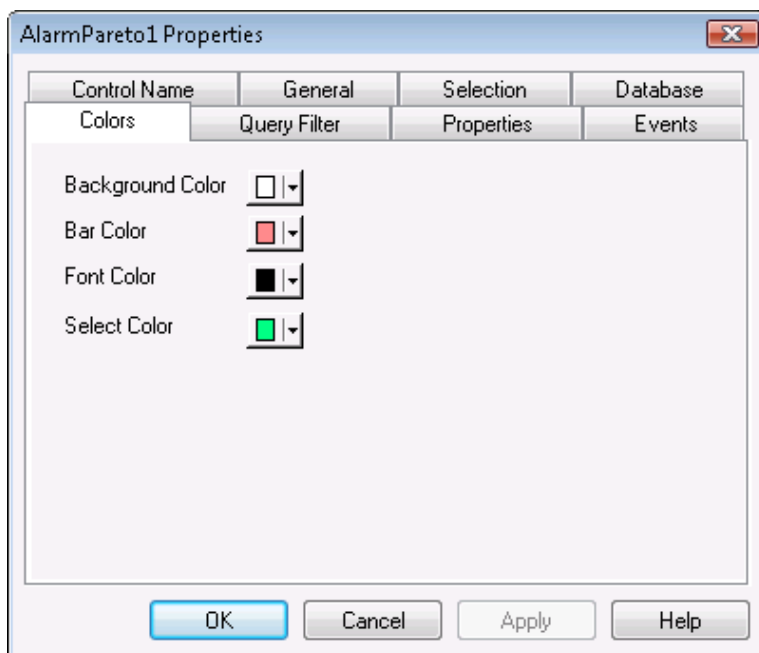
3. 一般的なオプションを設定します。以下の操作を行います。

プロパティ	説明
バー カウント	Alarm Pareto コントロールで表示されるバーの数を設定します。
表示モード	このリストには、利用できる表示オプションが示されます。オプションは、[アラーム／イベント履歴]、[アラーム履歴]、および [イベント履歴] です。

プロパティ	説明
一致しないメッセージ	Alarm Pareto コントロールで処理されるデータがないときを示すメッセージを設定します。
垂直方向	垂直スケールでバーを表示します。
水平方向	水平スケールでバーを表示します。
ステータス バーの表示	ステータス バーを有効化します。
サイレント モード	Alarm Pareto コントロールでは、ランタイム エラー メッセージは表示されません。チェックボックスをオフにすると、アラーム表示でエラー メッセージが表示されます。エラー メッセージは、常に Logger に送信されます。
自動フォント	利用できるスペースでは選択したバーのテキストを正しく表示できない場合、[自動フォント] によって、テキストが非表示になり、バーを選択したときのみテキストが表示されます。
ノード名を表示	棒グラフ上のノード名を表示します。
選択をステータス バーで表示	ステータス バーに選択したバーの説明を示します。
統合アラーム	アラーム状況を 2 つの状況に統合します。たとえば、アナログ タグ変数に 3 つの状況アラームがある場合、Hi、HiHi、および Normal、また Hi および HiHi アラーム状況は 1 つの状況と分類されます。
パーセント値でカウントを表示	合計カウントに対するカウントのパーセント値でバーを表示します。
状態を時間で表示	各タグ変数がアラーム状況にある時間に基づいて Alarm Pareto コントロールを表示します。 このオプションが有効なのは、表示モードが「アラーム履歴」に設定されている場合のみです。

4. [適用] をクリックします。

5. [色] タブをクリックします。



6. それぞれの色ボックスをクリックすると、カラーパレットが開きます。
7. 以下の各グラフプロパティに割り当てる色をクリックします。

プロパティ	説明
背景色	Alarm Pareto グラフの背景色を設定します。
バーの色	グラフのバーの色を設定します。
フォントの色	グラフに表示されるテキストの色を設定します。
選択の色	選択したバーの色を設定します。

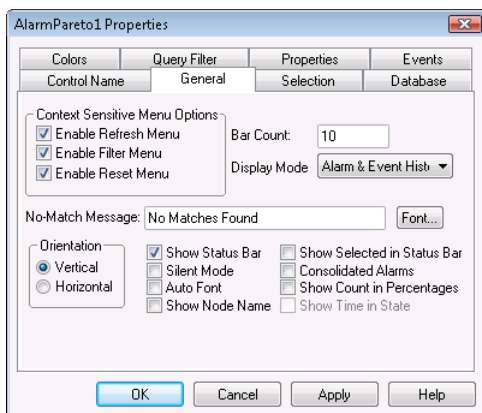
8. [適用] をクリックします。

フォントの表示の設定

Alarm Pareto グラフに表示されるテキストにフォントプロパティを割り当てできます。

フォントのプロパティを設定するには

1. Alarm Pareto コントロールを右クリックし、[プロパティ] をクリックします。[AlarmPareto のプロパティ] ダイアログボックスが表示されます。
2. [全般] タブをクリックします。

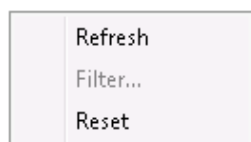


3. [フォント] をクリックします。Windows 標準の [フォント] ダイアログ ボックスが表示されます。フォントを設定して、[OK] をクリックします。

4. [OK] をクリックします。

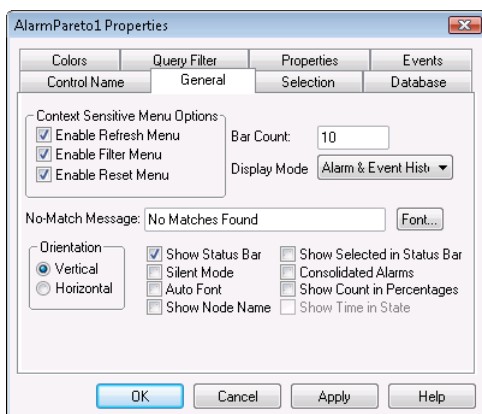
ランタイムにユーザーがアクセスできる機能の設定

Alarm Pareto グラフにはショートカット メニューが含まれています。ランタイムの間にユーザーがトレンドを右クリックすると、メニューが表示され、そのオプションで、トレンドに表示されたデータを動的に更新できます。



ランタイム機能を設定するには

1. Alarm Pareto コントロールを右クリックし、[プロパティ] をクリックします。[AlarmPareto のプロパティ] ダイアログ ボックスが表示されます。
2. [一般] タブをクリックします。



3. [コンテキスト メニューのオプション] 領域で、メニュー コマンドを設定します。

- a. [[リフレッシュ] メニューを有効化] チェック ボックスをオンにすると、Alarm Pareto トレンドに表示されたデータをランタイム ユーザーがリフレッシュして、範囲 1 から MaxRecords プロパティで定義された値までのレコードを表示できます。

- b. **[[フィルタ] メニューを有効化]** チェック ボックスをオンにすると、ユーザーが **[[フィルタ設定]** ダイアログ ボックスを表示して、Alarm Pareto トレンドのデータベース クエリー値を含むファイルを選択できます。
- c. **[[リセット] メニューを有効化]** チェック ボックスをオンにすると、ユーザーが、ランタイムの Alarm Pareto グラフを WindowMaker で指定された元の値に戻せます。ユーザーが行ったランタイム変更はすべて元のデザイン時の値に戻ります。

4. **[適用]** をクリックします。

分析するアラームの設定

Alarm Pareto グラフを使用して、分析するアラームを設定します。以下を指定します。

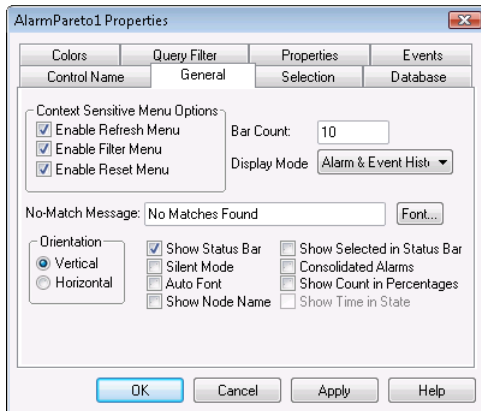
- データベース データのタイプ (アラーム データまたはイベント データ)
- レコードを選択する期間
- データをフィルタする条件

アラーム データまたはイベント データの選択

アラーム レコード、イベント レコード、またはその両方を Alarm Pareto グラフに表示するかどうかを設定できます。

データ タイプを選択するには

1. **Alarm Pareto** コントロールを右クリックし、**[プロパティ]** をクリックします。**[AlarmPareto のプロパティ]** ダイアログ ボックスが表示されます。
2. **[全般]** タブをクリックします。



3. **[表示モード]** リストで、レコードのタイプを設定します。以下のいずれかを実行します。

- **[アラーム/イベント履歴]** をクリックし、アラームおよびイベントの履歴データベース レコードを表示します。
- **[アラーム履歴]** をクリックし、履歴アラーム レコードだけを表示します。
- **[イベント履歴]** をクリックし、履歴イベント レコードだけを表示します。

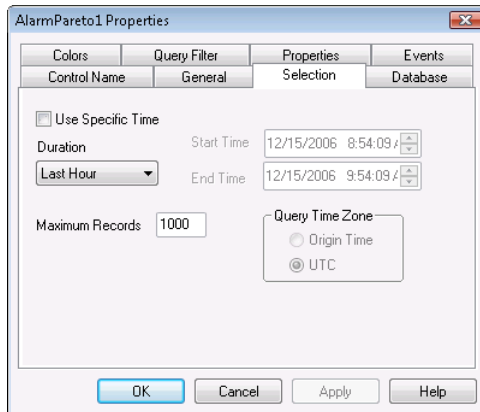
4. **[適用]** をクリックします。

期間の選択

選択した時間に基づいてレコードを選択するようにクエリー値を設定できます。また、表示するレコードの最大数、アラーム クエリーの開始時間と終了時間、クエリー タイム ゾーンを設定することもできます。

データの期間を選択するには

1. **Alarm Pareto** コントロールを右クリックし、**[プロパティ]** をクリックします。**[AlarmPareto のプロパティ]** ダイアログ ボックスが表示されます。
2. **[選択]** タブをクリックします。



3. UTC を使用してデータを問い合わせる定義済み間隔を使用するには、**[期間の設定]** リストで間隔をクリックします。
4. 特定の開始時間と終了時間を使用するには、**[指定時間を使用]** をクリックして詳細を設定します。
 - a. **[開始時間]** ボックスに、アラーム レコードを取得する開始時間を入力します。文字列は YYYY/MM/DD HH:MM:SS 形式にします。1970 年 1 月 1 日 0 時 00 分から 2038 年 1 月 18 日 19 時 14 分 07 秒の間なら、どのタイム ゾーンのどの日付でも使用できます。
 - b. **[終了時間]** ボックスに、アラーム レコードの取得を停止する終了時間を入力します。文字列は YYYY/MM/DD HH:MM:SS 形式にします。1970 年 1 月 1 日 0 時 00 分から 2038 年 1 月 18 日 19 時 14 分 07 秒の間なら、どのタイム ゾーンのどの日付でも使用できます。
 - c. **[クエリー タイム ゾーン]** 領域で、**[UTC]** または **[標準時間]** のどちらかをクリックします。UTC 時間はグリニッジ標準時 (Greenwich Mean Time) です。国際標準時または Zulu とも呼ばれます。標準時間は、オペレータのタイム ゾーンの現在の時間です。
5. **[最大レコード数]** ボックスに、1 つのインスタンスで、コントロールから表示するレコード数を入力します。また、最大レコードの有効な範囲は、0 ~ 1000000 です。
6. **[適用]** をクリックします。

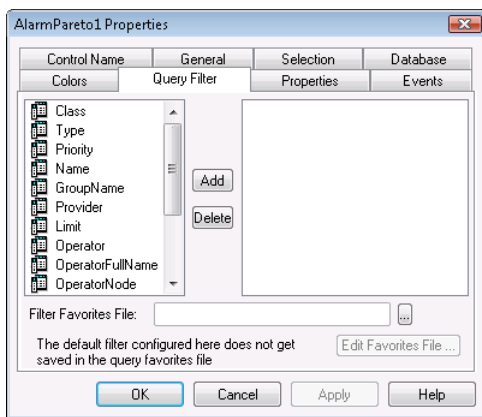
フィルタ設定を使用したカスタム フィルタの作成

クエリー結果に含めるレコードを選択できます。たとえば、レコードの日付またはアラームの状態によってフィルタを選択できます。クエリー結果を制限または拡大するために複数のフィールドを選択できます。

カスタム フィルタを定義しないと、すべてのレコードを問い合わせるデフォルトのフィルタが使用されます。

カスタム フィルタを作成するには

1. **Alarm Pareto** コントロールを右クリックし、[プロパティ] をクリックします。[AlarmPareto のプロパティ] ダイアログ ボックスが表示されます。
2. [クエリー フィルタ] タブをクリックします。



3. 左ペインでフィルタ フィールドを選択し、[追加] をクリックして、これらフィールドをフィルタに含めます。フィルタは右ペインに表示されます。フィルタ フィールドの詳細については、次の表を参照してください。

フィールド名	フィルタ クエリー
クラス	アラーム クラス
タイプ	アラーム タイプ
優先度	アラーム優先度
名前	アラーム名
GroupName	アラーム グループ名
プロバイダ	アラーム プロバイダ
しきい値	アラームしきい値。値は英数字です。 クエリー フィルタの値の比較は、文字列の比較として行われます。
オペレータ	オペレータ
オペレータ フルネーム	オペレータ フル ネーム
オペレータ ノード	アラームと関連のあるオペレータ ノード名
オペレータ ドメイン	アラームと関連のあるオペレータ ドメイン名
コメント	アラーム コメント

フィールド名	フィルタ クエリー
ユーザー 1	アラーム ユーザー定義数値 1
ユーザー 2	アラーム ユーザー定義数値 2
ユーザー 3	アラーム ユーザー定義文字列値
期間	未確認とアラーム期間ゼロに等しい値で設定された間隔カラムでは、クエリーで null 値を持つレコードは生成されません。

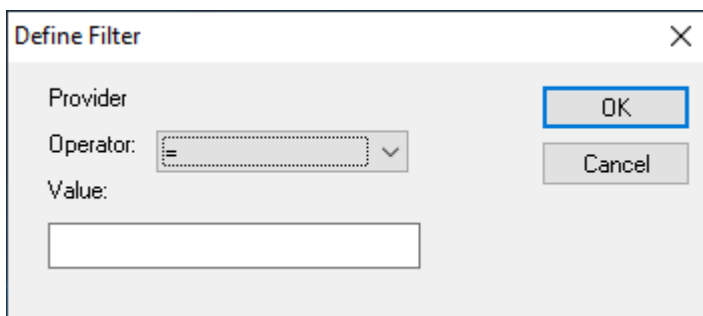
- フィルタ ペインからフィールドを削除するには、削除するフィールドをクリックし、**[削除]** をクリックします。フィルタの削除は元に戻せません。メッセージが表示されたら、**[はい]** をクリックします。
- 各フィルタ フィールドの条件を設定します。詳細については、「[カラム フィルタ条件の定義](#)、[カラム フィルタ条件の定義](#)」を参照してください。
- フィルタのオペレータとグループ化を設定します。詳細については、「[アラーム カラムのグループ化](#)、[アラーム カラムのグループ化](#)」を参照してください。
- フィルタ設定ファイルを設定します。
 - [フィルタ設定ファイル] ボックスに、ネットワーク パスとファイル名を入力するか、省略記号ボタンをクリックしてファイルを参照します。
 - [フィルタ設定] ファイルを編集するには、**[設定ファイルの編集]** ボタンをクリックします。[フィルタ設定] ウィンドウが開き、設定ファイルのフィルタを追加、変更、または削除できます。完了したら、**[OK]** をクリックして変更を保存し、ウィンドウを閉じます。
- [適用]** をクリックします。

カラム フィルタ条件の定義

クエリーに含める各カラム フィルタについて、フィルタ条件を設定する必要があります。たとえば、特定のオペレータについてのアラームだけを表示することもできます。

カラム フィルタを定義するには

- フィールドを右クリックし、**[フィルタの編集]** をクリックします。**[フィルタの定義]** ダイアログ ボックスが表示されます。



The image shows a 'Define Filter' dialog box with a close button (X) in the top right corner. It contains three input fields: 'Provider' (a text box), 'Operator:' (a dropdown menu showing '='), and 'Value:' (a text box). To the right of these fields are two buttons: 'OK' and 'Cancel'.

- [演算子]** リストで、必要な演算子を選択します。

3. [値] ボックスに、一致する必要がある条件を入力します。[値] ボックスでは、選択したクエリーについて処理できない値は受け入れられません。「Like」および「Not Like」のフィルタ演算子が英数字のカラム名に使われている場合、[値] ボックスでは次のワイルドカード文字が使用できます。

文字	検索
%	ゼロ文字以上の任意の文字列
_	任意の 1 文字
[]	指定範囲 ([a-f] など)、またはセット([abcdef] など) 内の任意の 1 文字
[^]	指定範囲にない ([^a-f] など)、またはセット内にない([^abcdef] など) 任意の 1 文字

次の [値] ボックスしきい値がさまざまなフィールドに適用されます。

フィールド	しきい値
すべてのフィールド	ユーザー 1、ユーザー 2、および優先度を除く、すべての英数字
優先度	1 ～ 999 の整数値が使用できます。
ユーザー 1、ユーザー 2	負の数、正の数、または小数のみを使用できます。

4. [OK] をクリックします。

アラーム カラムのグループ化

複数のフィールドが定義されると、カラムはブール演算子を使用して結合されます。

- AND 演算子は、選択したフィールドのすべての値に一致するレコードを返します。
- OR 演算子は、選択した任意のフィールドの値に一致するレコードを返します。

AND/OR 演算子を使用してフィルタ選択条件を使用するには、対応するフィールドをグループ化する必要があります。フィルタ ペインのアイテムに対しては 1 つのフィルタ式のみを作成できます。複数の式が必要な場合、そのアイテムをフィルタ ペインに再び追加する必要があります。

デフォルトで、グループ化されたフィールドには AND 演算子があります。

AND 演算子と OR 演算子は親ノードです。各親ノードの下にある選択されたフィールドは子ノードです。フィールド親ノードを子ノードにドラッグすることはできません。

アラーム カラムをグループ化するには

1. フィールドを右クリックし、[グループ] をクリックします。
2. フィールドを別のフィールドにドラッグします。

クエリー フィルタのコピーまたは移動

Alarm Pareto コントロールの複数のインスタンスがあり、複数のインスタンスについて同じフィルタを使用する場合は、定義済みフィルタをあるインスタンスからコピーまたは切り取りして、別のフィルタに貼り付けます。

フィルタをコピーするには

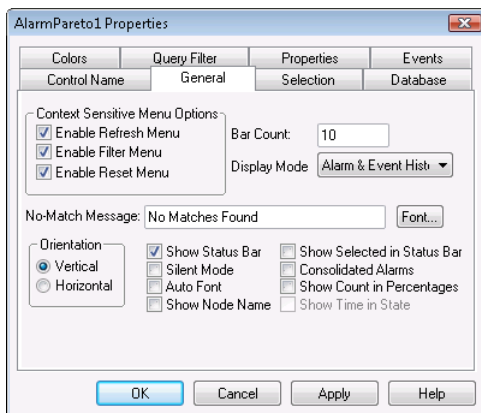
1. **Alarm Pareto** コントロールの最初のインスタンスで、フィルタを定義します。
2. フィルタを右クリックし、[コピー] をクリックします。フィルタを移動するには、[切り取り] をクリックします。
3. **Alarm Pareto** コントロールの最初のインスタンスを終了します。
4. **Alarm Pareto** コントロールの次のインスタンスを開き、[クエリー フィルタ] タブをクリックします。
5. 右ペインに矢印を位置付けします。選択したフィルタを右クリックし、[貼り付け] をクリックします。

分析結果の表示の設定

アラームの表示をクエリー結果に設定できます。

表示を設定するには

1. **Alarm Pareto** コントロールを右クリックし、[プロパティ] をクリックします。[AlarmPareto のプロパティ] ダイアログ ボックスが表示されます。
2. [全般] タブをクリックします。



3. [統合アラーム] チェック ボックスをオンにし、マルチ状態アラームから 1 つのタイムスタンプ レコードへとアラームを結合します。
4. [パーセント値でカウントを表示] チェック ボックスをオンにして、グラフに表示された各バーの合計のパーセント値を追加します。
5. [適用] をクリックします。

ランタイムでの Alarm Pareto コントロールの使用

ランタイムに **Alarm Pareto** コントロールを右クリックし、ショートカット メニューを表示します。次の表に、ショートカット メニューに表示される、可能性のあるすべてのオプションをリストします。

メニュー オプション	説明
再表示	表示をリフレッシュします。
フィルタ	フィルタを編集して、Pareto コントロールが受け取ったデータを変更できます。このメニュー項目は、フィルタ設定ファイルが設定されている場合のみ利用できます。
リセット	グラフをデフォルトのクエリーにリセットします。

ステータス バーに表示される情報について

Alarm Pareto コントロールのステータス バーに表示されるものは次のとおりです。

- コントロールとアラーム データベースの間のデータベース接続のステータス
- グラフに表示されたデータをリフレッシュする、グラフの更新ステータス

Alarm Pareto ActiveX プロパティの使用

スクリプトを使用して Alarm Pareto コントロールのプロパティの値を直接設定するか、InTouch タグまたは I/O 参照に割り当てることができます。プロパティの設定の詳細については、「[ActiveX コントロールのスクリプト](#)」を参照してください。

次の表に Alarm Pareto プロパティを示します一覧表示しています。

プロパティ名	説明
Authentication	[データベース] タブで選択された認証タイプを返します。
AuthenticationMode	認証タイプを決定する値を設定します (SQL 認証の場合は 0、Windows 認証の場合は 1)。
AutoConnect	コントロールが実行モードになるとすぐにコントロールがデータベースに接続されるかどうかを決定する値を返すか、設定します。
AutoFont	利用できるスペースでは選択したバーのテキストを正しく表示できない場合、[自動フォント] によってテキストが非表示になり、バーを選択したときのみ選択したバーのテキストが表示されます。
BackGndColor	Alarm Pareto グラフの背景色を設定します。
BarColor	Alarm Pareto コントロールのバーの色を設定します。
BarCount	Alarm Pareto コントロールで表示されるバーの数を設定します。
BarSelectColor	Alarm Pareto コントロールで選択したバーの色を設定します。

プロパティ名	説明
Connected	Alarm Pareto コントロールがデータベースに接続されているかどうか決定します。
ConsolidatedAlarms	アラーム状態を 2 つの状態に統合します。たとえば、アナログ型タグに 3 状態アラームがある場合、Hi、HiHi、および Normal、また Hi および HiHi アラーム状態は 1 つの状態と分類されます。
DatabaseName	Alarm Pareto コントロールが接続する必要があるデータベースの名前を設定します。
DisplayMode	表示モードを設定します。表示モードオプションは、[アラーム/イベント履歴]、[アラーム履歴]、または [イベント履歴] です。
Duration	コントロールが「開始時間」と「終了時間」を設定する間隔を返すか、または設定します。
EnableRefresh	Alarm Pareto コントロールをリフレッシュするコンテキストメニューを有効または無効にします。
EnableReset	Alarm Pareto コントロールをリセットするコンテキストメニューを有効または無効にします。
EnableSilentMode	サイレント モードを有効または無効にします。 サイレント モードが無効な場合、Alarm Pareto コントロールにはエラー メッセージボックスが表示されます。サイレント モードが有効な場合、エラー メッセージボックスは表示されません。 エラー情報がロガーに書き込まれます。
EndTime	終了日付および時間を返すか、または設定します。
FilterMenu	フィルタを編集して Pareto コントロールが受け取ったデータを変更できる、コンテキストメニューを有効または無効にします。このプロパティが有効なのは、フィルタメニューファイルが設定されている場合のみです。
FilterFavoritesFile	フィルタ設定ファイルを文字列として指定します。
Font	コントロール内のレコードと見出しのフォントを設定します。
FontColor	Alarm Pareto コントロールでレコード表示のフォントの色を設定します。
HorizontalChart	水平バーとしてグラフを表示します。 HorizontalChart が無効な場合、グラフには垂直バーが表示されます。

プロパティ名	説明
MaxRecords	指定された時間で取得できる最大レコード数を指定する値を返すか、または設定します。
NoMatchMessage	Alarm Pareto コントロールで処理されるデータがないときに表示されるメッセージを設定します。
QueryTimeZone	タイムゾーンを「UTC」または「標準時間」のどちらかに設定します。
ServerName	現在のサーバー名を返します。
ShowCountPercentage	選択した場合、合計カウントのパーセント値として各バーのカウントを表示します。選択しない場合、各バーの実際のカウントを表示します。
ShowNodeName	Alarm Pareto コントロールが、バーにノード名その他の情報を表示するよう設定します。
ShowSelectedInStatusBar	ステータス バーで選択したバーの情報の表示を有効または無効にします。
ShowStatusBar	ステータス バーを表示するかどうか決定する値を返すか、または設定します。
ShowTimeinState	タグがアラーム状態にとどまった時間を基礎にして、 Alarm Pareto コントロールがバーを表示するかどうかを決定する値を返すか、設定します。 無効な場合、アラームが発生した回数を基礎にして、コントロールはバーを表示します。
SpecificTime	コントロールが「開始時間」プロパティと「終了時間」プロパティを使用するかどうか、または「期間の設定」プロパティを基に開始時間と終了時間を計算するかどうかを決定する値を返すか、または設定します。
StartTime	開始日付および時間を返すか、または設定します。
User	SQL Server に接続するコントロールとして使用されるユーザーを返すか、設定します。

Alarm Pareto ActiveX メソッドの使用

Alarm Pareto ActiveX メソッドを使用して次のことができます。

- データベース接続を制御します。
- データベースからレコードを取得します。
- 特定のパレート バーについての情報を取得します。

メソッドの呼び出しの詳細については、「[ActiveX コントロールのスクリプト](#)」を参照してください。

データベース接続の制御

アラーム データベースに接続するには、**Connect()** メソッドを使用します。

Connect() メソッド

Alarm Pareto コントロール プロパティの **[データベース]** タブで設定されたデータベースに接続します。

構文

```
Object.Connect()
```

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.Connect();
```

データベースからレコードを取得

次の関数を使用して、データベースからレコードを取得します。

- [Refresh\(\) メソッド](#)
- [SelectQuery\(\) メソッド](#)

Refresh() メソッド

データベースのコントロールをリフレッシュします。接続が正常に行われた場合は、1 から MaxRecords プロパティで定義される値の範囲にあるレコードセットが表示されます。

構文

```
Object.Refresh()
```

例

```
#AlarmPareto1.Refresh();
```

SelectQuery() メソッド

クエリー設定ファイルとして設定されるフィルタを選択します。

構文

```
Object.SelectQuery(Filter)
```

パラメータ

フィルタ

クエリー フィルタの名前

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.SelectQuery("MyFilter");
```

特定のパレート バーの情報を取得

次の関数を使用して、特定のパレート バーの情報を取得します。

- [GetItemAlarmName\(\) メソッド](#)
- [GetItemAlarmType\(\) メソッド](#)
- [GetItemCount\(\) メソッド](#)
- [GetItemTotalTime\(\) メソッド](#)

- [GetItemEventType\(\) メソッド](#)
- [GetItemProviderName\(\) メソッド](#)

GetItemAlarmName() メソッド

特定のバーについてアラーム名を取得します。

構文

```
Object.GetItemAlarmName(BarIndex)
```

パラメータ

BarIndex

バーのインデックス

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.GetItemAlarmName(1);
```

GetItemAlarmType() メソッド

特定のバーについてアラームのタイプを取得します。

構文

```
Object.GetItemAlarmType(BarIndex)
```

パラメータ

BarIndex

バーのインデックス

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.GetItemAlarmType(1);
```

GetItemCount() メソッド

バーのアラーム数を取得します。

構文

```
Object.GetItemCount(BarIndex)
```

パラメータ

BarIndex

バーのインデックス

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.GetItemCount(1);
```

GetItemTotalTime() メソッド

アラーム状況のタグ変数の合計時間（秒単位）を取得します。

構文

```
Object.GetItemTotalTime(BarIndex)
```

パラメータ

BarIndex

バーのインデックス

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.GetItemTotalTime(1);
```

GetItemEventType() メソッド

特定のバーについてイベントのタイプを取得します。

構文

```
Object.GetItemEventType(BarIndex)
```

パラメータ

BarIndex

バーのインデックス

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.GetItemEventType(1);
```

GetItemProviderName() メソッド

特定のバーについて生成されたアラームからプロバイダ名を取得します。

構文

```
Object.GetItemProviderName(BarIndex)
```

パラメータ

BarIndex

バーのインデックス

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.GetItemProviderName(1);
```

その他の情報の表示

AboutBox() メソッドを使用して、[バージョン情報] ダイアログ ボックスを表示します。

AboutBox() メソッド

[バージョン情報] ダイアログ ボックスを表示します。

構文

```
Object.AboutBox()
```

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.AboutBox();
```


メソッドとプロパティを使用するときのエラー処理

Alarm Pareto コントロールでは、[サイレント モード] オプションに基づいてランタイム エラー メッセージが処理されます。詳細については、「[Alarm Pareto コントロールの外観と色を設定](#)」を参照してください。

[サイレント モード] を選択すると、Alarm Pareto コントロールで、ランタイム エラー メッセージが表示されません。チェックボックスをオフにすると、アラーム表示でエラー メッセージが表示されます。すべての Alarm Pareto エラー メッセージは、Logger に送信されます。

Alarm Pareto ActiveX イベントを使用したスクリプトのトリガ

マウス クリックやダブルクリックなどの Alarm Pareto コントロール イベントに、QuickScript を割り当てることができます。イベントが発生すると、QuickScript が実行されます。

Alarm Pareto コントロールは、以下のイベントをサポートしています。

- クリック
- DoubleClick
- ShutDown
- StartUp

Click イベントは ClicknBarIndex というパラメータを 1 つ持ち、これはランタイム時にクリックされたバーのインデックスを識別します。

DoubleClick イベントは DoubleClicknBarIndex というパラメータを 1 つ持ち、これはランタイム時にダブルクリックされたバーのインデックスを識別します。

Click イベントおよび DoubleClick イベントはゼロを基準としています。Click イベントまたは DoubleClick イベント（あるいはその両方）が発行されるとき、表示されるバー カウントは 0 から始まります。

注: ユーザー インターフェイス メソッドが StartUp イベントから呼び出される場合、コントロールはまだ表示されていないため、Alarm Pareto コントロールでは無視されます。このようなメソッドには、AboutBox() および Refresh() があります。

ActiveX イベントのスクリプトの詳細については、「[ActiveX コントロールのスクリプト](#)」を参照してください。

章 11 スクリプト

InTouch スクリプト言語 (QuickScript) を使用すると、より堅固なアプリケーションを構築できます。スクリプトには 8 種類あり、多くの組み込みスクリプト関数を使用できます。

スクリプトは、いつ実行されるか、および継続的に実行されている他のアプリケーションプロセスから独立して実行されているかどうか、などにより分類できます。スクリプトは通常、次の 2 つの異なる方法で実行されます。

- イベントが発生したときに、イベント ベースのスクリプトが 1 回実行されます。たとえば、オペレータがキーを押すか、またはタグ値が変更された後、イベント ベースのスクリプトを実行できます。
- 条件が満たされたときに、時間ベースのスクリプトが定期的に実行されます。たとえば、ウィンドウが開いているか、またはボタンを押し続けているときに、時間ベースのスクリプトを実行できます。

同じトリガで実行するように、複数のイベント ベースのスクリプトおよび時間ベースのスクリプトを設定できます。たとえば、キーが押されたときにあるスクリプトを実行し、同じキーを押し続けている間に他のスクリプトを 5 秒ごとに定期的に実行するように設定できます。

スクリプト言語で条件付きステートメント、ループ、およびローカル変数を使用すると、アプリケーションで複雑な効果を作成できます。

条件スクリプトでは、同期的または非同期的にスクリプトを実行できます。

- 同期スクリプトが実行されている場合、すべての InTouch アニメーションおよびタグ値の更新が停止します。スクリプトが停止すると、アニメーションおよびタグ値の更新が再開します。
- 非同期スクリプトが実行されると、スクリプトが実行されている期間、すべての InTouch アニメーションおよびタグ値の更新が続行されます。

組み込みスクリプト関数には、数学関数、三角関数、文字列関数、その他などがあります。これらの関数を使用すると、アプリケーション開発の時間を節約できます。

InTouch スクリプトには、OLE (Object Linking and Embedding) オブジェクトおよび ActiveX コントロールを含めることができます。

基本的なスクリプトの概念

スクリプトを開始する前に、以下の内容を理解する必要があります。

- スクリプトは、アプリケーションに何かを実行するように指示する命令のセットです。
- QuickScript は、InTouch HMI スクリプト言語です。
- 関数は、他のスクリプトから呼び出すことのできるスクリプトです。InTouch HMI には、すぐに使用できる事前定義済みの関数のセットが含まれています。
- クイック関数は、QuickScript で記述され、クイック関数ライブラリに保存される、再使用可能な関数です。クイック関数を作成するには、簡単に QuickScript を作成して、名前を付けます。クイック関数は、別のスクリプトにより呼び出すか、アニメーションリンク式から呼び出すことができます。

スクリプトのタイプ

InTouch では、スクリプトは、その実行条件に基づいて分類されます。たとえば、オペレータがキーボードの特定のキーを押したときにスクリプトを実行する場合は、「キー スクリプト」を作成します。

スクリプト タイプを選択した後、スクリプトを実行させる条件または状況を詳細に定義できます。たとえば、キーが押されたときでなく、キーが離されたときにスクリプトを実行することもできます。

スクリプトのタイプを以下に示します。

- **アプリケーション** スクリプトは、**WindowViewer** の実行中に継続して実行するか、**WindowViewer** が起動または終了したときに一度実行します。
- **ウィンドウ** スクリプトは、**InTouch** ウィンドウが開いているときに定期的に実行するか、**InTouch** ウィンドウが開くとき、または閉じるときに一度実行します。
- **キー** スクリプトは、特定のキーやキーの組み合わせが押されたとき、または離されたときに、一度または定期的に実行します。
- **条件** スクリプトは、特定の条件が満たされた場合、または満たされなかった場合に、一度または定期的に実行します。
- **データ変化** スクリプトは、特定のタグまたは式の値が変化したときに一度実行します。
- **アクション** スクリプトは、オペレータが **InTouch HMI** グラフィック オブジェクトをクリックしたときに、一度または定期的に実行します。アクション スクリプトは、プッシュボタンに使用されることがあります。
- **ActiveX イベント** スクリプトは、**ActiveX イベント** (**ActiveX** コントロールのクリックなど) が発生したときに一度実行します。

高度なスクリプトの概念

高度なスクリプト機能を使用すると、**InTouch HMI** の基本的なスクリプト機能よりも洗練された関数を作成できます。

OLE オブジェクトや **ActiveX** コントロールを使用すると、ネイティブ コンピュータ システム関数にアクセスして、**Manufacturing Engineering Module** などの他のプログラムと対話できます。

OLE オブジェクト

カスタム スクリプトでは、**OLE** オブジェクトを呼び出すことができます。**OLE** オブジェクトを使用すると、ネイティブ コンピュータ システム関数にアクセスして、**Manufacturing Engineering Module** などの他のプログラムと対話できます。

たとえば、**OLE** を使用すると、以下の操作を行えます。

- ランダム数を生成する
- ユーザー インターフェイス ダイアログ ボックスを作成する
- **Windows** の日時プロパティ パネルを開く
- レジストリへ読み込むまたは書き込む
- ウィンドウを最小化する

ActiveX コントロールを使用したスクリプト

InTouch HMI の [ウィザード] メニューには、いくつかの **ActiveX** コントロールが提供されています。
InTouch HMI は Windows オペレーティング環境に基づいているため、InTouch HMI ではほとんどすべての **ActiveX** コントロールを使用できます。

スクリプトの作成と編集

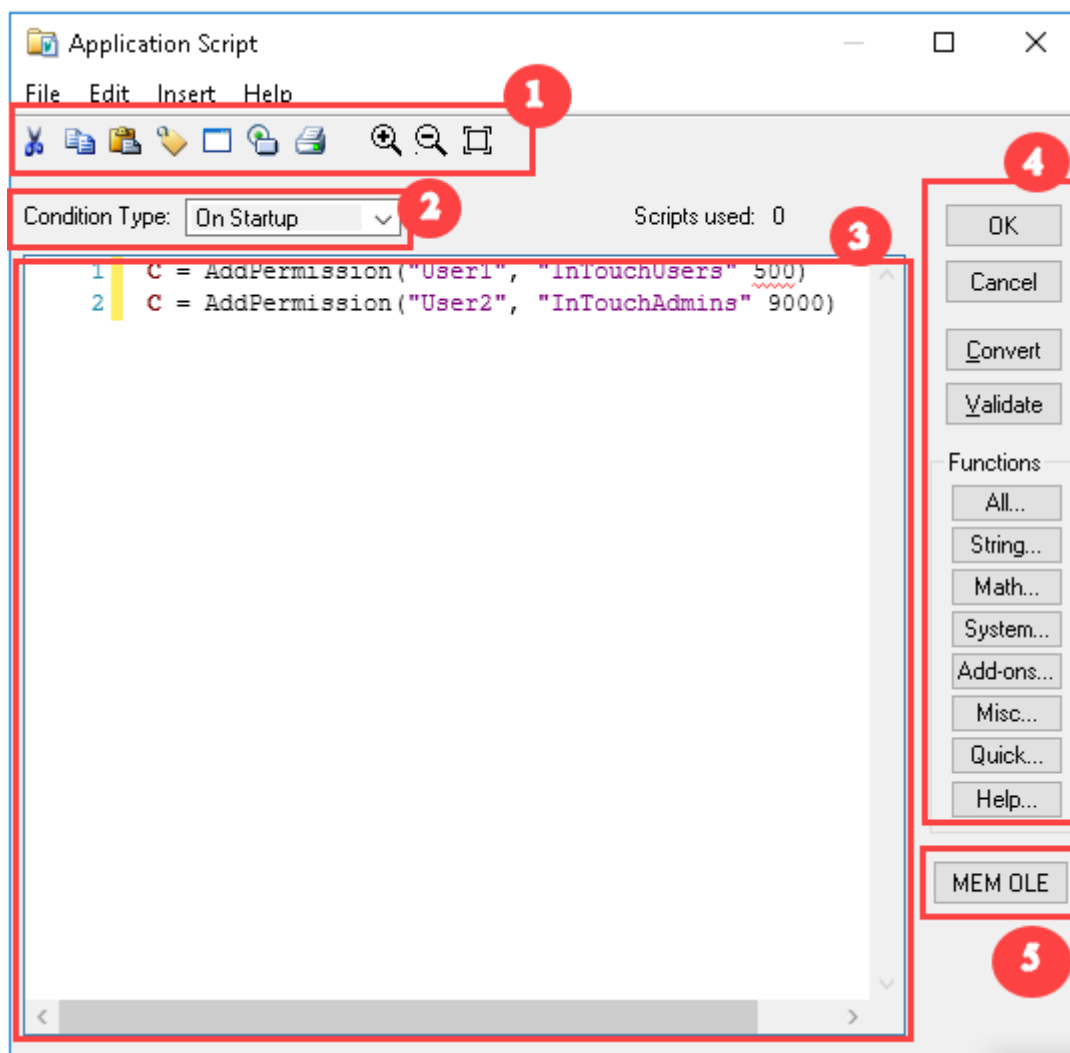
新しいスクリプトを作成する手順は、スクリプトのタイプに応じて異なります。通常は、スクリプトエディタを開いて条件タイプを選択し、ステートメントを入力して、スクリプトを保存します。

各タイプのスクリプトの作成の詳細については、以下のセクションを参照してください。

- [アプリケーションスクリプトの設定](#)
- [ウィンドウスクリプトの設定](#)
- [キースクリプトの設定](#)
- [条件スクリプトの設定](#)
- [データ変化スクリプトの設定](#)
- [アクションスクリプトの設定](#)
- [ActiveX イベントスクリプトの設定](#)

InTouch スクリプトエディタの操作

InTouch WindowMaker 内でスクリプトを作成および編集するには、InTouch HMI スクリプトエディタを使用します。



領域	説明
1	ツールバー
2	条件定義エリア [実行条件] ボックスは、記述しているスクリプトのタイプで利用できる実行条件を提供します。
3	スクリプト テキスト ウィンドウ
4	コマンド ボタン
5	組み込みスクリプト関数ボタン
6	MEM OLE ボタン 右下隅の [MEM OLE] ボタンは、MEM (Manufacturing Engineering Module) が InTouch HMI と共にインストールされている場合にのみ表示されます。このボタンをクリックすると、MEM を使用してスクリプトを作成できます。

これはアプリケーション スクリプトの例です。各タイプのスクリプトには固有のバージョンのスクリプト ダイアログ ボックスがあり、そのスクリプトのタイプに固有のオプションおよび選択肢があります。

エディタのタイトルバーは、使用しているスクリプトのタイプを示します。スクリプトのタイプの詳細については、「[スクリプトのタイプ](#)」を参照してください。

コンテキスト依存のポップアップ ウィンドウにはテキスト、等価、および数学演算のボタンがあります。これらのボタンをクリックすると、キーワード、関数、またはシンボルをスクリプトのカーソル位置に挿入できます。

InTouch スクリプトの色インジケータ

InTouch スクリプト エディタは、さまざまなテキストの色を使用してさまざまなスクリプト要素を示します。次の表は、スクリプト要素に関連付けられたテキストの色を示します。

要素	色
キーワード	青 入力中にハイライトされる構文
コメント（単一行と複数行）	緑 入力中にハイライトされる構文
文字列	紫 入力中にハイライトされる構文
関数名、定数、演算子、セミコロ ン、dim 変数、エイリアス変数など	黒 属性名と予約語の説明を参照してください。
属性、InTouch タグ、参照文字列	えび茶、太字
予約語	赤、非太字
.NET タイプ名	青緑、非太字

InTouch スクリプト エディタのオートコンプリート機能

これらの機能を使用すると、タグ、ドットフィールド、メソッド、プログラミングが可能なコンストラクト、およびその他のスクリプト要素の自動ワードコンプリートが有効になります。スクリプト エディタのオートコンプリート機能: InTouch スクリプト エディタのオートコンプリートには、InTouch スクリプトを記述する際に使用できる多くの機能が含まれています。これらの機能を使用すると、タグ、ドットフィールド、メソッド、プログラミングが可能なコンストラクト、およびその他のスクリプト要素の自動ワードコンプリートが有効になります。スクリプト エディタのオートコンプリート機能:

- 選択可能なリスト ボックスでオートコンプリート タグ 参照が提供されます。
- 定義、キーワード、スクリプト要素、オブジェクトと属性の名前、およびプログラミングが可能なコンストラクトを含むコンテキスト固有の推奨候補を始めとするメソッドパラメータがオートコンプリート リスト ボックスに表示されます。

これらの機能は、メソッドパラメータおよびスクリプト構文の便利なマニュアルとしてだけでなく、強力な入力方法としても機能します。

Ctrl + Space キーを押すと、スクリプト内で選択した場所で使用可能なすべてのオートコンプリート オプションおよび変数が表示されます。コンテキストは、オートコンプリート ポップアップ ウィンドウの下部に表示されるアイコンから識別することができます。

アイコン	意味
	メソッド
	キーワード
	演算子
	変数
	タグ
	ウィンドウ
	ActiveX インスタンス

InTouch スクリプト エディタのオートコンプリート推奨候補の受け入れ

オートコンプリート リスト ボックスから（端点やタブなしで）エディタのキャレットにアイテムを挿入するには、次のいずれかの操作を実行します。

- アイテムをダブルクリックします。
- アイテムをハイライト（選択）して、**Enter** キーまたは **Tab** キーを押します。

スペース、ピリオド、カンマ、開きカッコまたは閉じカッコ、またはプログラミング言語で使用するその他の句読点（:、;、[、]、=、<、>、-、+、/、*）を入力すると、オートコンプリート リスト ボックスでハイライトされたアイテムが追加文字と共にエディタ キャレットの位置に挿入されます。

スクリプトのオートコンプリートの無効化

スクリプトを記述する際にオートコンプリートの推奨候補を表示しない場合、スクリプトのオートコンプリート機能を無効にすることができます。

WindowMaker の設定画面からスクリプトのオートコンプリートを無効にするには

1. [ファイル] メニューの [設定] をクリックし、[WindowMaker] をクリックします。

WindowMaker の設定画面が表示されます。

2. [スクリプトのオートコンプリートを無効化] チェックボックスをオンにします。

スクリプト エディタにオートコンプリートの推奨候補が表示されなくなります。

スクリプト ウィンドウからスクリプトのオートコンプリートを無効にするには

1. スクリプト ウィンドウを起動します。
2. [編集] メニューで [スクリプトのオートコンプリートを無効化] をクリックします。
スクリプト エディタにオートコンプリートの推奨候補が表示されなくなります。

InTouch スクリプト記述での複数レベルの「元に戻す」および「やり直す」機能

スクリプトの変更のシーケンスを選択的に元に戻すことができます。元に戻すことのできる変更の数は、使用可能なメモリの量に応じて異なります。

- メインメニューの [編集] オプションの [元に戻す]、または **Ctrl + Z** キーの組み合わせを使用して編集を元に戻します。コンテキストメニュー オプションを使用して、変更を元に戻すことややり直すこともできます。
- メインメニューの [編集] オプションの [やり直す]、または **Ctrl + Y** キーの組み合わせを使用して編集をやり直します。コンテキストメニュー オプションを使用して、変更を元に戻すことややり直すこともできます。

元に戻した変更は、やり直すことができます。やり直す操作は、元に戻す操作の逆の操作です。

一般的に、1つの「元に戻す」操作は、一連の入力または削除操作で構成されます。これらの操作は、オートコンプリート リストでの操作やマウス カーソルの移動、またはスクリプト内の別の場所でのクリックによって中段されることがあります。

スクリプト エディタを閉じた場合やスクリプト エディタ内で別のスクリプトに切り替えた場合、保留中の「元に戻す」操作および「やり直す」操作は失われます。

InTouch スクリプト エラーの視覚的表示

InTouch スクリプト テキストのエラーは、赤い「波線」の下線でマークされます。

エラーの上にマウス カーソルを置くと、エラー メッセージがツールヒントとして表示されます。ツールヒントのエラー メッセージには、スクリプト検証ボタンをクリックしたときに表示されるものと同じ情報が表示されます。

場合によっては、複数のエラーに下線が表示されることがあります。エラーによっては、エラー箇所でのコンパイラの処理が中断されるので、複数のエラーに常に下線が表示されるわけではありません。

2つの文字列を連結してパスを形成するために「\」文字が使用されている場合、スクリプト エディタでは「\」の下に赤い下線が表示されます。これは例外で、無視できます。パスが実行時に正常に構築されます。

```
1 TagComment = InfoIntouchAppDir() + "\";
```

InTouch スクリプトの編集の管理

InTouch スクリプト エディタでは、編集を検索して管理するためのツールおよび視覚的リファレンスが提供されます。

スクリプト行番号

スクリプト エディタの左側のマージンには行番号が表示されます。

- スクリプト エディタがズームされていない場合、最大 4 桁の行番号が表示されます。

- スクリプト内の特定の行に移動するには、右クリックのコンテキストメニューの **［移動］** オプションを使用します。
- 現在の行のテキストを削除するには、**Ctrl + L** キーの組み合わせを押します。
- テキストを別の行に移動するには、ドラッグアンドドロップします。

変更バー

作業中のスクリプトの変更の視覚的な参照として、スクリプトテキストウィンドウの左側のマージンに表示される黄色い変更バーによって追加、行の挿入、および編集が示されます。

タグ定義

InTouch スクリプトエディタに新しいタグ名を入力して、**Ctrl + T** キーの組み合わせを押すか、メインメニューの **［編集］** をクリックしてタグを定義できます。**タグ名ディクショナリ**が表示されるので、タグ名定義を完了できます。

検索と置換

InTouch スクリプトエディタは、カスタマイズ可能な検索および置換機能を提供します。検索および置換機能の使用の詳細については、「[スクリプト内での検索](#)」を参照してください。

編集用にスクリプトを開く

既存のスクリプトを開く手順は、スクリプトタイプにより若干異なります。

アプリケーションスクリプトを開くには

1. **［スクリプト］** ペインで **［アプリケーション］** をダブルクリックします。
[アプリケーションスクリプト] ダイアログが表示されます。
2. **［実行条件］** リストで、編集するスクリプトのタイプをクリックします。

ウィンドウスクリプトを開くには

1. **［ウィンドウ］** ペインでウィンドウ名を右クリックし、**［ウィンドウスクリプト］** をクリックします。
または、スクリプトが関連付けられているウィンドウを開きます。ウィンドウの空白領域を右クリックして、**［ウィンドウスクリプト］** をクリックします。
2. **［実行条件］** リストで、スクリプトを実行させる条件をクリックします。

ActiveX イベントスクリプトを開くには

- 以下のいずれかを実行します。
 - **［スクリプト］** ペインで **［ActiveX イベント］** を展開してスクリプト名をダブルクリックします。
 - スクリプトが関連付けられている **ActiveX** コントロールインスタンスをダブルクリックします。
［イベント］ タブをクリックして、スクリプト名を含むセルをダブルクリックします。

アクションスクリプトを開くには

1. アクションスクリプトが関連付けられているグラフィック要素を含むウィンドウを開きます。アクションスクリプトの一般的な用法は、ボタンのアクションのスクリプト記述です。
2. アクションスクリプトが関連付けられているグラフィック要素をダブルクリックします。

3. [タッチ押しボタン] 領域で、[スクリプト ボタン] をクリックします。スクリプト エディタが表示されます。
4. [実行条件] リストで、スクリプトを実行させるアクションをクリックします。

キー スクリプト、条件スクリプト、またはデータ変化スクリプトを開くには

1. [スクリプト] ペインでスクリプト カテゴリを展開してスクリプト名をダブルクリックします。
 - スクリプト エディタが表示されます。[参照] ボタンをクリックして、スクリプト名をクリックします。
2. 必要に応じて、[実行条件] リストで、スクリプトを実行させる条件をクリックします。

スクリプトへの変更の保存または破棄

スクリプト エディタでの作業中または作業終了時に手動または自動でスクリプトを保存できます。すべてのスクリプトを破棄することもできます。

リスト オプションは、ウィンドウ スクリプトとアプリケーション スクリプトでは使用できません。

注意：変更の保存および破棄は、現在表示されている条件タイプだけではなく、1つのスクリプトタイプのすべての条件タイプに適用されます。

変更を保存して、スクリプトを開いたままにするには

- [スクリプト] メニューで、[保存] をクリックします。

変更を保存して、スクリプトを閉じるには

- [OK] をクリックします。

変更を破棄して、スクリプトを開いたままにするには

- [リストア] をクリックします。

変更を破棄して、スクリプトを閉じるには

- [キャンセル] をクリックします。

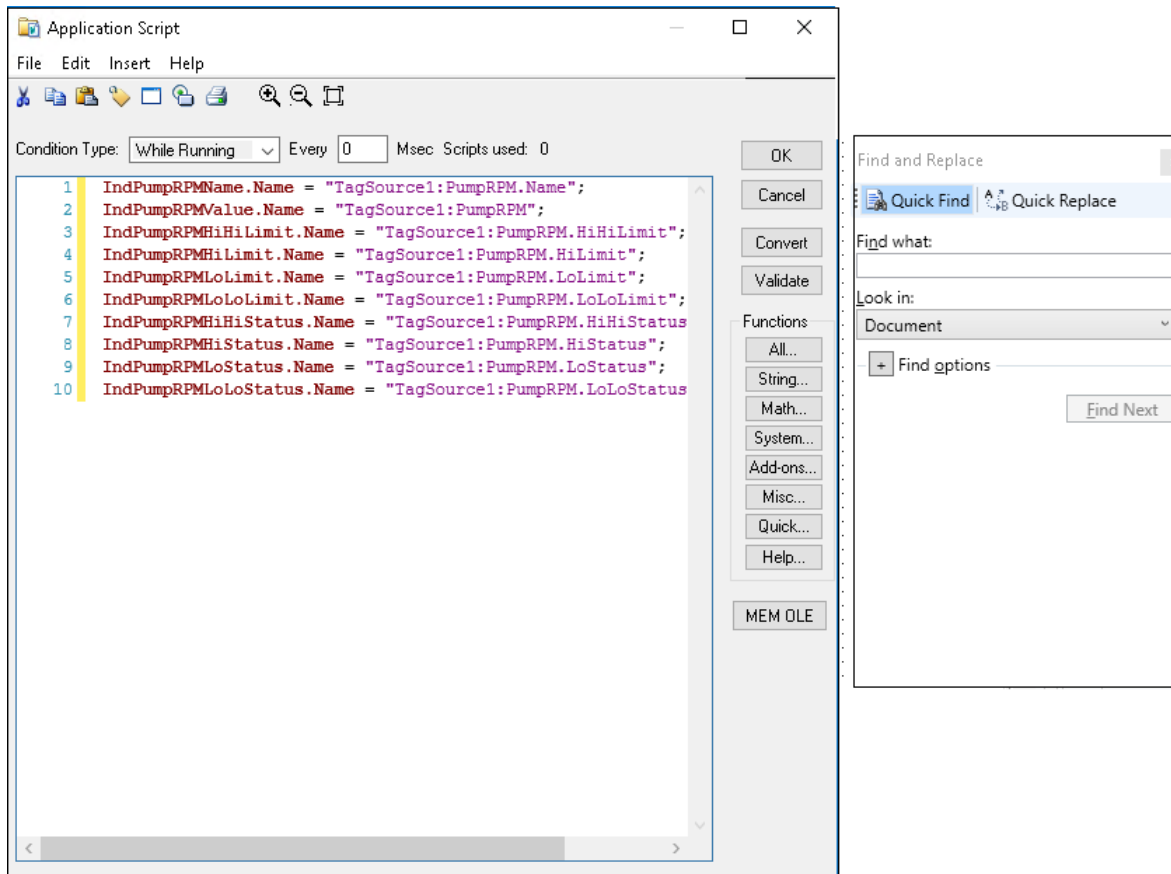
テキストのコピー、切り取り、貼り付け

スクリプト エディタでは、その他の Windows アプリケーションと同じ方法で、テキストをコピーしたり、切り取ったり、貼り付けたりすることができます。標準のキーボードショートカット Ctrl+C、Ctrl+X、および Ctrl+V、またはツールバー ボタンを使用します。

行を選択して、その行のテキストをスクリプト内の別の場所にドラッグすることもできます。

スクリプト内での検索

InTouch スクリプト エディタでは、開いているスクリプト内で検索文字列およびカスタマイズ可能な検索オプションのセットに基づいて検索と置換を行うことができます。[検索] と [置換] は、スクリプト エディタのメニュー バーの [編集] からアクセスできます。



また、**Ctrl + F**（クイック検索）および**Ctrl + H**（クイック置換）のキーボードショートカットを使用して、**[検索と置換]** ダイアログボックスを表示することもできます。

- デフォルトの検索オプションを使用したシンプルな検索では、**[クイック検索]** タブを選択して、**[検索する文字列]** フィールドに単語または語句を入力します。

[次を検索] を選択して検索を開始します。スクリプト内で検索文字列に一致する単語または語句の背景が黄色くハイライトされます。

- デフォルト オプションを使用したシンプルな置換操作では、**[クイック置換]** タブを選択し、**[検索する文字列]** フィールドに単語または語句を入力して、**[次の文字列に置換]** フィールドに置換する文字または語句を入力します。

[次を検索] を選択して置換操作を開始します。スクリプト内で、**[検索する文字列]** に入力した検索文字列に一致する単語または語句の背景が明るい青でハイライトされます。一致する単語または単語が見つかった後、3つの置換オプションを使用できます。

- [次を検索]** を選択すると、一致する現在の単語または語句を無視して、スクリプト内の次の一致が継続されます。
- [置換]** を選択すると、一致する現在の単語または語句が **[次の文字列に置換]** フィールドに入力した文字列で置き換えられます。
- [すべて置換]** を選択すると、一致するすべての単語または語句が **[次の文字列に置換]** フィールドに入力した文字列で置き換えられます。

検索または置換の設定

[検索と置換] ダイアログ ボックスには、検索オプションを設定する一連のオプションがあります。

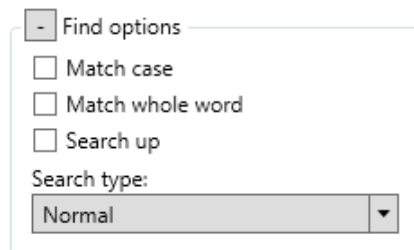
検索

[検索] フィールドには、スクリプト全体を検索するオプション ([Document] (ドキュメント)) またはスクリプトの選択範囲のみを検索するオプション (Selection (選択)) があります。デフォルト設定は [Document] (ドキュメント) です。

スクリプトの選択した部分内だけを検索する場合、検索を実行する前にマウスを使用してスクリプト内の検索領域を選択します。選択したスクリプト行の背景が青でハイライトされます。この領域が検索対象になります。検索結果には、選択したスクリプト領域内の一致項目のみが表示されます。

検索オプション

[検索オプション] セクションは展開と折り畳みが可能です。以下のオプションを選択または選択解除して検索結果を絞り込むことができます。



- 大文字と小文字を区別

このオプションを選択すると、[検索する文字列] に入力した文字列の内容と大文字／小文字が一致するインスタンスだけが検索結果に表示されます。たとえば、[大文字と小文字を区別] オプションを選択した状態で Triangle4 を検索した場合、Triangle4 が返されますが、triangle4 は返されません。

- 全単語一致のみ

このオプションを選択すると、[検索する文字列] に入力した文字列の語句に完全に一致するインスタンスだけが検索結果に表示されます。たとえば、LogicBit を検索する場合、LogicBit が返されますが、LogicBits は返されません。

- Search up (上方向に検索)

このオプションを選択すると、スクリプト内の現在の位置からスクリプトの最初に向かって検索が実行されます。デフォルトでは、スクリプト内の現在の位置からスクリプトの最後に向かって検索が実行されます。

Search type (検索のタイプ)

[Search type] (検索のタイプ) フィールドを使用すると、検索のタイプに基づいてスクリプト検索を実行できます。

- 通常

これは、デフォルトの検索タイプで、検索文字列の文字とスクリプト内のテキストが一致する必要があります。

- [正規表現による検索](#)

正規表現は、スクリプトを検索するときに 1 つまたは複数の文字列を表します。正規表現には、検索する文字列に一致する文字のテンプレートとして機能する通常の文字が含まれます。

- [ワイルドカード文字による検索](#)

ワイルドカード検索では、スクリプト内を検索するときに 1 つまたは複数の文字を表すためにアスタリスク (*) や疑問符 (?) などのキーボード文字を使用します。

- [頭文字による検索](#)

頭文字検索では、単語の最初で一致する 1 文字が検索され、その後に各大文字またはアンダースコアの後の文字列が検索されます。

- [簡易略記法による検索](#)

簡易略記法検索では、検索パターン文字列の間にホワイトスペース以外の文字を含めることによって「頭文字による検索」オプションが拡張されます。

構文に準拠する文字列、および選択した検索のタイプでサポートされる文字を入力する必要があります。

ワイルドカード文字による検索

ワイルドカード検索では、スクリプトの文字列を検索するときにいくつかのリテラル文字または空の文字列として解釈可能な 1 つのキーボード文字を使用します。

実際の文字がわからない場合や、完全な検索文字列の入力を省略する場合、1 つまたは複数の文字の代わりとしてワイルドカード文字を使用できます。

ワイルドカード文字	使用方法
アスタリスク (*)	<p>検索文字列内の 1 つのアスタリスクは文字の任意のシーケンスに一致します。アスタリスクは、ゼロ個以上の文字の代わりとして使用します。</p> <p>例</p> <ul style="list-style-type: none">• Logic* 「Logic1」および「LogicTest」が検索されますが、「ALogicTest」は検索されません。• *Test* 「LogicTest1」および「PumpTestABC」が検索されますが、「LogicTst1」は検索されません。
疑問符 (?)	<p>検索文字列内の 1 つの疑問符は、検索文字列内の特定の位置にある任意の文字を表します。</p> <p>例</p> <ul style="list-style-type: none">• LogicTest? 「LogicTest1」および「LogicTestA」が検索されますが、「ALogicTest1」は検索されません。• LogicTest?2

	「LogicTest12」が検索されますが、「LogicTest13」は検索されません。
--	--

正規表現による検索

正規表現は、英数字および特殊文字（メタ文字）を使用して、一致する 1 つまたは複数の文字列を表します。正規表現は、検索するスクリプトテキストと比較する文字パターンとして機能します。

正規表現は、算術表現と同様に構成されます。さまざまなメタ文字と演算子を使用して複数の小さい表現を組み合わせて大きな表現を作成できます。

正規表現の要素は、個々の文字、文字のセット、文字の範囲、または複数の文字の間の選択です。これらの要素の組み合わせも要素として使用できます。

スクリプトの正規表現

正規表現	説明	例
.	1 つの任意の文字に一致します（改行を除く）。	s.e 「step」の「ste」および「transfer」の「sfe」に一致しますが、「across」の「acro」には一致しません。
*	先行する式のゼロ以上の発生に一致します（可能な限り多くの文字に一致します）。	a*r 「rack」の「r」、「ark」の「ar」、および「aardvark」の「aar」に一致します。
.*	任意の文字列にゼロ回以上一致します（ワイルドカード*）。	「c.*e」は「racket」の「cke」、 「comment」の「comme」、および 「code」の「code」に一致します。
+,	先行する式の 1 つ以上の発生に一致します（可能な限り多くの文字に一致します）。	e.+e 「feeder」の「eede」に一致しますが、「ee」には一致しません。
.+	任意の文字列に 1 回以上一致します（ワイルドカード?）。	「e.+e」は「feeder」の「eede」に一致しますが、「ee」には一致しません。
?	先行する式のゼロ以上の発生に一致します（可能な限り少ない文字に一致します）。	e.?e 「feeder」の「ee」に一致しますが、「eede」には一致しません。
+?	先行する式の 1 つ以上の発生に一致します（可能な限り少ない文字に一致します）。	e.+?e 「enterprise」の「ente」および「erprise」に一致しますが、「enterprise」という単語には一致しません。
^	行または文字列の先頭にある一致文字列に一致します。	^car 行の最初にある「car」という単語にのみ一致します。
\r?\$	一致文字列を行の末尾にアンカーします。	End\r?\$ 行の末尾にある「end」という単語にのみ一致します。

[abc]	セット内の 1 つの任意の文字に一致します。	b[abc] 「ba」、「bb」、および「bc」に一致します。
[a-f]	文字の範囲に含まれる任意の文字に一致します。	be[n-t] 「between」の「bet」、「beneath」の「ben」、および「beside」の「bes」に一致しますが、「below」には一致しません。
()	括弧で囲まれた表現をキャプチャして暗示的に番号を設定します。	([a-z])X\1 「aXa」および「bXb」に一致しますが、「aXb」には一致しません。".「\1」は最初の式グループ「[a-z)」を参照します。
(?!abc)	一致を検証します。	real(?!ity) 「realty」および「really」の「real」に一致しますが、「reality」には一致しません。また、「realityreal」の（最初の「real」ではなく）2 番目の「real」も検索されます。
[^abc]	指定した文字のセットに含まれない任意の文字に一致します。	be[^n-t] 「before」の「bef」、「behind」の「beh」、および「below」の「bel」に一致しますが、「beneath」には一致しません。
	記号の前または後のいずれかの表現に一致します。	(sponge mud)bath 「spongebath」および「mudbath」に一致します。
\^	円記号の後の文字をエスケープします。	
{x},	先行する文字またはグループの発生数を指定します。	x(ab){2}x 「xababx」に一致します。「x(ab){2,3}x」は「xababx」と「xabababx」に一致しますが、「xababababx」には一致しません。
\p{X}	Unicode 文字クラスのテキストに一致します。ここで「X」は Unicode 番号を表します。	\p{Lu} 「Thomas Doe」の「T」と「D」に一致します。
\b	単語境界に一致します。	\bin 「inside」の「in」に一致しますが、「pinto」には一致しません。
\r?\n	改行（新しい行の前のキャリッジリターン）	End\r?\nBegin 「End」が行の最後の文字列で、次の行の最初の文字列が「Begin」の場合にのみ「End」と「Begin」に一致します。
\w	任意の英数字に一致します。	a\wd 「add」と「a1d」に一致しますが、「a d」には一致しません。

<code>(?^[^r\n])\s</code>	ホワイトスペース文字に一致します。	<code>Public\sInterface</code> 「Public Interface」という語句に一致します。
<code>\d</code>	任意の数字に一致します。	<code>\d</code> 「3456」の「3」、「23」の「2」、および「1」の「1」に一致します。
<code>\uXXXX</code> (「XXXX」は Unicode 文字値を指定します)	1 つの Unicode 文字に一致します。	<code>\u0065</code> 文字「e」に一致します。
<code>\b(\w+ [\w-[0-9\]]\w*)\b</code>	識別子に一致します。	「type1」に一致しますが、「&type1」および「#define」には一致しません。
<code>((\".+?\" '.'+?'))</code>	引用符内の文字列に一致します。	一重引用符または二重引用符に囲まれた文字列に一致します。
<code>\b0[xX]([0-9a-fA-F])\b</code>	16 進数文字に一致します。	「0xc67f」に一致しますが、「0xc67fc67f」には一致しません。
<code>\b[0-9]\.[0-9]+\b</code>	整数と「d」に一致します。	「1.333」に一致します。

優先順位

正規表現は左から右に評価され、優先順位に従います。

次の表に正規表現演算子の優先順位を高い順から示します。

演算子	説明
<code>\</code>	エスケープ
<code>()、(?:)、(?:=)、[]</code>	括弧と角括弧
<code>*, +, ?, {n}, {n,}, {n,m}</code>	修飾子
<code>^, \$, \任意のメタ文字</code>	アンカーとシーケンス
<code> </code>	代替

文字は代替演算子よりも優先順位が高いため、「`m|food`」は「m」または「food」に一致します。

頭文字による検索

頭文字検索では、単語の最初に一致する 1 文字が検索され、各大文字またはアンダースコアの後の文字列が検索されます。

例: "LB" では "LogicBits" のインスタンスが検索されます。

簡易略記法による検索

簡易略記法検索では、検索パターン文字列の間にホワイトスペース以外の任意の数の文字を含めることによって「頭文字による検索」オプションが拡張されます。

例: "ta" では "Triangle" のインスタンスが検索されます。

コード要素の挿入

リストからさまざまなコード要素を選択してスクリプトに自動的に挿入することができます。この機能を利用すると時間を節約し、入力エラーのリスクを軽減することができます。コード要素にアクセスするには、メインメニューの **〔挿入〕** オプションを使用するか、オートコンプリート ポップアップ ウィンドウのリストから目的の要素を選択します。

オートコンプリート ポップアップ ウィンドウとスクリプト エディタのメニュー項目の両方から挿入できるコード要素を以下に示します。

- 関数
- タグ名とドットフィールド（ドットフィールドの場合、タグ名の後にピリオドを入力してオートコンプリート ポップアップ ウィンドウを開きます）
- 変数
- ウィンドウ名
- **ActiveX** インスタンス
- キーワード
- 演算子

オートコンプリート ポップアップ ウィンドウの下部には、使用可能な要素に直接アクセスできる一連のアイコンが表示されます。

スクリプト関数のヘルプへのアクセス

特定のスクリプト関数のヘルプを検索する場合、スクリプト エディタから直接アクセスできます。

特定のスクリプト関数のヘルプを表示するには

1. スクリプト エディタの右下隅で、**〔ヘルプ〕** をクリックします。
関数のリストが表示されます。
2. 関数のヘルプを表示するには、リストに含まれる関数名をクリックします。
対応するヘルプ トピックが表示されます。

正確な構文を確認するためのスクリプトの検証

スクリプトを保存すると、その構文が正しいかどうかスクリプト エディタによって自動的に検証されます。エラーが検出された場合、詳細を示すメッセージが表示されます。スクリプトを保存する前に、すべての構文エラーを修正する必要があります。また、スクリプトを編集している間、手動で検証を開始することもできます。

スクリプト構文を手動で確認するには

- **〔確認〕** をクリックします。

スクリプトエラーのハイライト表示

InTouch スクリプト エディタには、スクリプト エラーをハイライトする機能もあります。詳細については、「[InTouch スクリプト エラーの視覚的表示](#)」を参照してください。

スクリプトの印刷

スクリプトエディタからスクリプトを個別に印刷したり、WindowMaker の印刷機能を使用して、特定のタイプのスクリプトをすべて印刷したりできます。

スクリプトエディタからスクリプトを個別に印刷したり、WindowMaker の印刷機能を使用して、特定のタイプのスクリプトをすべて印刷したりできます。

個々のスクリプトを印刷するには

1. スクリプトエディタでスクリプトを開きます。
2. ツールバーの **〔スクリプトの印刷〕** をクリックします。スクリプトは、Windows のデフォルトのプリンタに印刷されます。

特定のタイプのすべてのスクリプトを印刷するには

1. クイックアクセス ツールバーの **〔印刷〕** をクリックします。
〔アプリケーション情報の印刷〕 ダイアログ ボックスが表示されます。
2. ウィンドウ スクリプトを印刷するには、以下の手順を実行します。
 - a. **〔ウィンドウ〕** を選択します。
 - b. 印刷するウィンドウを選択します。
〔全て〕 を選択すると、アプリケーションにあるすべてのウィンドウの情報が印刷されます。
〔選択〕 を選ぶと、特定のウィンドウの情報のみが印刷されます。**〔ウィンドウの印刷〕** ダイアログ ボックスが開きます。アプリケーションで印刷するウィンドウを選択して **〔OK〕** をクリックします。
〔バッチ〕 を選択すると、.csv ファイルで指定したウィンドウの情報のみが印刷されます。
 - c. ウィンドウに関連付けられているスクリプトを印刷するには、**〔ウィンドウ スクリプト〕** を選択します。
3. その他のタイプのスクリプトを印刷するには、適切なチェック ボックスをオンにします。すべてのスクリプトを印刷するには、**〔すべてのスクリプト〕** をクリックします。
4. **〔次へ〕** をクリックします。**〔出力の送信先の選択〕** ダイアログ ボックスが表示されます。
5. 以下のいずれかを実行します。
 - **〔プリンタに出力を送信する〕** をクリックします。
 - **〔テキスト ファイルに出力を送信する〕** をクリックします。
6. **〔参照〕** をクリックして、プリンタを選択するか、ファイルを検索します。
7. **〔印刷〕** をクリックします。

すべてのスクリプトを印刷するには

1. **〔すべてのスクリプト〕** を選択すると、アプリケーションで使用されているすべてのスクリプトが印刷されます。
〔すべてのスクリプト〕 チェック ボックスをオフにすると、印刷対象を選択したタイプのスクリプトだけに制限できます。その場合は、印刷するスクリプト タイプの横のチェック ボックスをオンにします。

2. [次へ] をクリックします。[出力の送信先の選択] ダイアログ ボックスが表示されます。
3. タグ名ディクショナリの内容を印刷するか、または出力結果をテキスト ファイルや .html ファイルに送信するかのオプションを選択します。
4. [印刷] をクリックします。

スクリプトの削除

スクリプトを削除する手順は、スクリプト タイプにより異なります。以下のセクションを参照してください。

- [アプリケーションスクリプトの設定](#).
- [ウィンドウ スクリプトの設定](#).
- [キー スクリプトの設定](#).
- [条件スクリプトの設定](#).
- [データ変化スクリプトの設定](#).
- [アクション スクリプトの設定](#).
- [ActiveX イベント スクリプトの設定](#).

スクリプトを表示するズーム オプションの調整

スクリプトを表示するズーム オプション（ズーム イン、ズーム アウト、ズーム解除など）を調整できます。

ズーム インするには

1. スクリプト エディタを開きます。
2. スクリプト エディタ ツールバーで [ズーム イン] をクリックします。
または、[編集] をクリックして [ズーム イン] をクリックします。

ズーム アウトするには

1. スクリプト エディタを開きます。
2. スクリプト エディタ ツールバーで [ズーム アウト] をクリックします。
または、[編集] をクリックして [ズーム アウト] をクリックします。

ズーム解除するには

1. スクリプト エディタを開きます。
2. スクリプト エディタ ツールバーで [ズーム解除] をクリックします。
または、[編集] をクリックして [ズーム解除] をクリックします。

スクリプト トリガ

すべての InTouch HMI スクリプトは、スクリプト トリガにより実行されます。各スクリプト タイプには、そのスクリプトを起動する 1 つまたは複数のトリガがあります。

スクリプトエディタでは、スクリプトを実行するために使用するスクリプトトリガを選択できます。スクリプトトリガは、スクリプトがいつ、またどのように実行されるかに基づいて選択します。

ユーザーアクション、内部状態、およびタグ名の値の変更に基づいてさまざまなトリガを設定できます。ユーザーアクションには、キーを押す、グラフィック要素をクリックするなどがあります。内部状態トリガには、**WindowViewer** を含むことができます。

スクリプトは、以下のようなアクションにより起動されます。

- **WindowViewer** の起動およびシャットダウン。「[アプリケーションスクリプトの設定](#)」を参照してください。
- ウィンドウを開く、閉じる。「[ウィンドウスクリプトの設定](#)」を参照してください。
- キーおよびキーの組み合わせを押す。「[キースクリプトの設定](#)」を参照してください。
- タグ名や式の値など、特定の条件を満たす。「[条件スクリプトの設定](#)」を参照してください。
- タグ名の値またはタグ名フィールド値の変更。「[データ変化スクリプトの設定](#)」を参照してください。
- グラフィックオブジェクトのクリック。「[アクションスクリプトの設定](#)」を参照してください。
- コントロールのクリックなど、**ActiveX** コントロール内で発生するイベント。「[ActiveX イベントスクリプトの設定](#)」を参照してください。

また、スクリプトの実行を一時停止することもできます。デフォルトでは、**WindowViewer** が起動されると、ロジックが実行され、スクリプトが実行されます。スクリプトの実行は、ロジックを中断することによってランタイムで一時停止できます。一時停止した後、スクリプト実行は再開できます。詳細については、「[ランタイムでのスクリプト実行の一時停止](#)」を参照してください。

スクリプトトリガのタイプ

InTouch HMI では、スクリプトは7つのタイプに分割されています。スクリプトの各タイプには、スクリプトを起動するために選択できるトリガが1つまたは複数あります。

- アプリケーションスクリプトには、3つのトリガ（起動時、終了時、および実行中）があります。各トリガは、異なるスクリプトを実行できます。
- ウィンドウスクリプトには、3つのトリガ（開く時、閉じる時、表示中）があります。各トリガは、異なるスクリプトを実行できます。
- キースクリプトには、3つのトリガ（離れた時、押した時、押している間）があります。各トリガは、異なるスクリプトを実行できます。
- 条件スクリプトには4つのトリガ（**True** の時、**True** の間、**False** の時、および **False** の間）があります。各トリガは、異なるスクリプトを実行できます。
- データ変化スクリプトは、特定のタグ変数または式の値が変化したときに実行します。
- アクションスクリプトは、オペレータが InTouch HMI グラフィックオブジェクトをクリックしたときに、一度または定期的に実行します。
- **ActiveX** イベントスクリプトは、**ActiveX** コントロールのクリックなど、**ActiveX** イベントが発生したときに一度実行します。

複数のトリガの使用

ほとんどのスクリプトタイプでは、複数のトリガを使用して、さまざまなスクリプトを各トリガに関連付けることができます。

たとえば、**WindowViewer** が起動したときにスクリプトを実行して、**WindowViewer** の実行中に別のスクリプトを定期的に実行するように、アプリケーション スクリプトを設定できます。

[実行条件] リストでトリガを選択して、トリガの既存のスクリプトを表示します。

定期的なスクリプトの実行

定期的に行うスクリプトは、トリガ後すぐに実行するのではなく、指定期間後に実行します。

たとえば、特定のキーが押されている間 **5000** ミリ秒ごとに実行するようにキー スクリプトを設定すると、キーが押されて、押された状態のままで **5** 秒経過してからそのスクリプトは実行され、その後は **5** 秒ごとに実行されます。

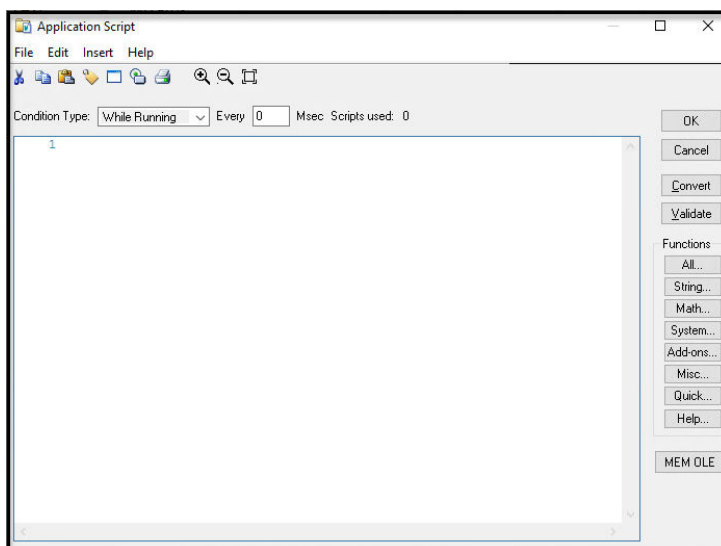
アプリケーション スクリプトの設定

アプリケーション スクリプトは、**InTouch HMI** アプリケーション全体にリンクされます。アプリケーション スクリプトを使用すると、以下の操作を実行できます。

- **WindowViewer** が起動されたときにスクリプトを一度実行する
- **WindowViewer** の実行中にスクリプトを定期的に実行する
- **WindowViewer** がシャットダウンしたときにスクリプトを一度実行する。

アプリケーション スクリプトを設定するには

1. [スクリプト] ペインで [アプリケーション] を右クリックし、[開く] をクリックします。
[アプリケーション スクリプト] ダイアログ ボックスが表示されます。



2. [実行条件] リストで、スクリプト実行の条件をクリックします。
 - **WindowViewer** が起動されたときに一度実行するようにスクリプトを設定するには、[起動時] をクリックします。

- WindowViewer の実行中に定期的に実行するようにスクリプトを設定するには、**[実行中]** をクリックします。
 - WindowViewer がシャットダウンされたときに一度実行するようにスクリプトを設定するには、**[終了時]** をクリックします。
3. 前の手順で **[実行中]** を選択した場合、**[実行周期]** ボックスに 1 ～ 360000 ミリ秒の間隔を入力します。この間隔は、スクリプトが実行される頻度を指定します。
 4. スクリプトをウィンドウに入力します。
 5. **[OK]** をクリックします。

アプリケーション スクリプトを削除するには

1. **[スクリプト]** ペインで **[アプリケーション]** を右クリックし、**[開く]** をクリックします。
[アプリケーション スクリプト] ダイアログ ボックスが表示されます。
2. **[実行条件]** リストで、削除するスクリプトの条件をクリックします。
スクリプトが、**[アプリケーション スクリプト]** ダイアログ ボックスのメインセクションに表示されます。
3. **[編集]** メニューで、**[クリア]** をクリックします。
スクリプトがメインセクションからクリアされ、関連付けられたスクリプトが削除されます。

アプリケーション スクリプトの制限事項

WindowViewer の起動時または終了時に実行されるアプリケーション スクリプトには、他のオブジェクトとの対話に関する制限事項があります。

起動時のアプリケーション スクリプトは、以下の操作を行うためには使用できません。

- ActiveX メソッド、プロパティ、またはイベントの参照
- コントロールおよび I/O タグ変数またはリモート リファレンスに対する読み取りまたは書き込み
- データ変化スクリプトおよび条件スクリプトの実行

終了時のアプリケーション スクリプトは、以下の操作を行うためには使用できません。

- コントロールおよび I/O タグ変数またはリモート リファレンスに対する読み取りまたは書き込み
- その他のアプリケーションの起動

ウィンドウ スクリプトの設定

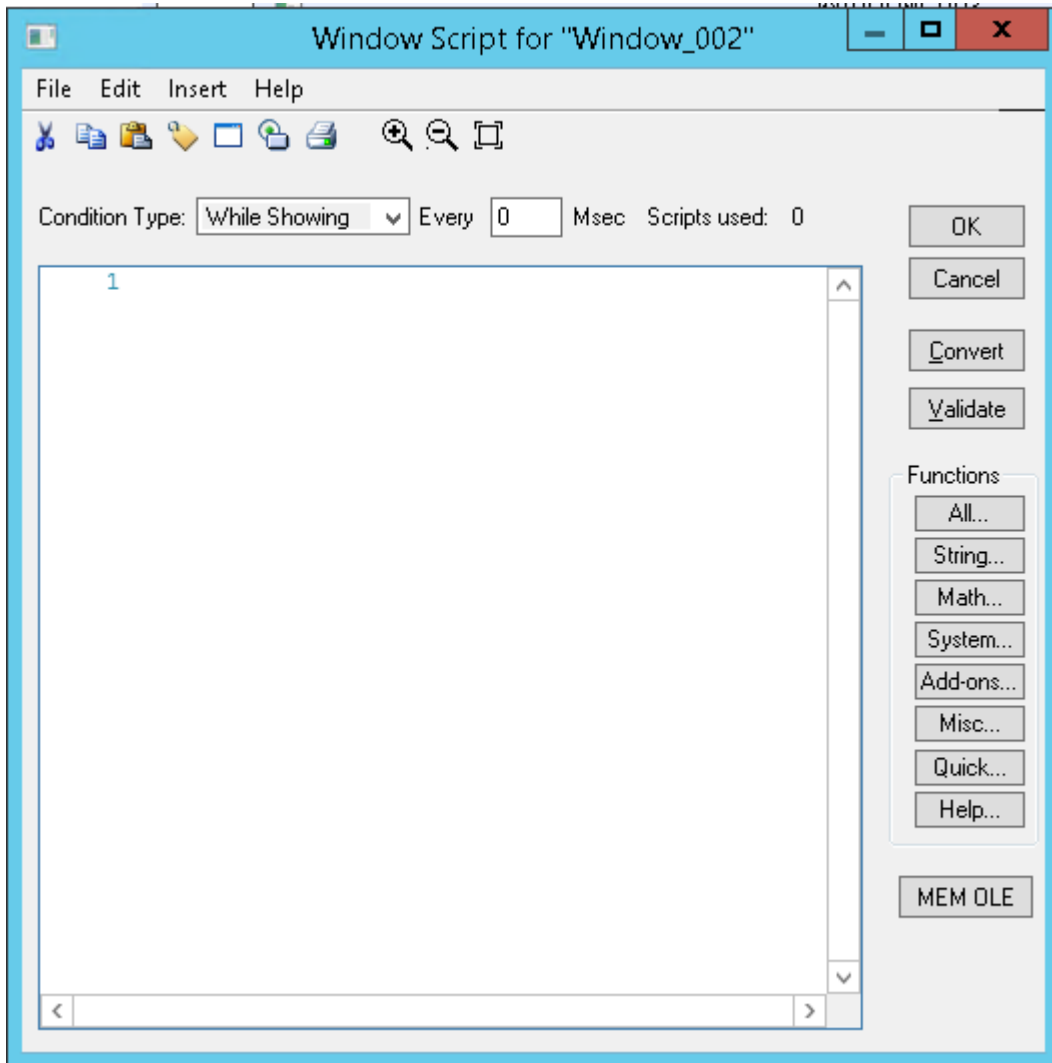
ウィンドウ スクリプトは、特定のウィンドウにリンクされるスクリプトです。**GetWindowName** スクリプト関数を使用して、ランタイム環境でのウィンドウの読み込みに必要なスクリプト作成を減らすことができます。ウィンドウ スクリプトを使用すると、以下の操作を実行できます。

- InTouch ウィンドウが開く時にスクリプトを一度実行する。
- InTouch ウィンドウが表示中にスクリプトを定期的に実行する。
- InTouch ウィンドウが閉じる時にスクリプトを一度実行する。

注記: InTouch ウィンドウを開くことは、「InTouch ウィンドウを表示する」とも言います。また、InTouch ウィンドウを閉じることは、「InTouch ウィンドウを非表示にする」とも言います。

ウィンドウ スクリプトを設定するには

1. [ウィンドウ] ペインでウィンドウを右クリックし、[ウィンドウ スクリプト] をクリックします。
[ウィンドウ スクリプト - ウィンドウ名] ダイアログ ボックスが表示されます。



2. [実行条件] リストで、以下のいずれかを実行します。
 - 関連付けられたウィンドウが起動したときに一度実行するようにスクリプトを設定するには、[開く時] をクリックします。
 - 関連付けられたウィンドウが開いている間に定期的に行うようにスクリプトを設定するには、[表示中] をクリックします。
 - 関連付けられたウィンドウが閉じられたときに一度実行するようにスクリプトを設定するには、[閉じる時] をクリックします。
3. 前の手順で [表示中] を選択した場合、[実行周期] ボックスに 1 ～ 360000 ミリ秒の間隔を入力します。
4. スクリプトをウィンドウに入力します。

5. **[OK]** をクリックします。

ウィンドウ スクリプトを削除するには

1. **[ウィンドウ]** ペインでウィンドウを右クリックし、**[ウィンドウ スクリプト]** をクリックします。
[ウィンドウ スクリプト - ウィンドウ名] ダイアログ ボックスが表示されます。
2. **[実行条件]** リストで、削除するスクリプトのスクリプト トリガをクリックします。スクリプトが、
[ウィンドウ スクリプト - ウィンドウ名] ダイアログ ボックスのメインセクションに表示されます。
3. **[編集]** メニューで、**[クリア]** をクリックします。

重要: **[閉じる時]** スクリプトを使用して、**I/O** タグ名に対する読み取りおよび書き込みを行わないでください。**I/O** 値の更新は、ウィンドウが非表示になる前に必ずしも完了するわけではありません。ウィンドウが閉じるときに **I/O** タグ名に対する読み取りまたは書き込みを行うには、データ変更スクリプトを設定して、**[閉じる時]** スクリプトからスクリプトをアクティブ化します。

キー スクリプトの設定

キー スクリプトは、特定のキーまたはキーの組み合わせを押す操作にリンクされるスクリプトです。キー スクリプトを使用すると、以下の操作を実行できます。

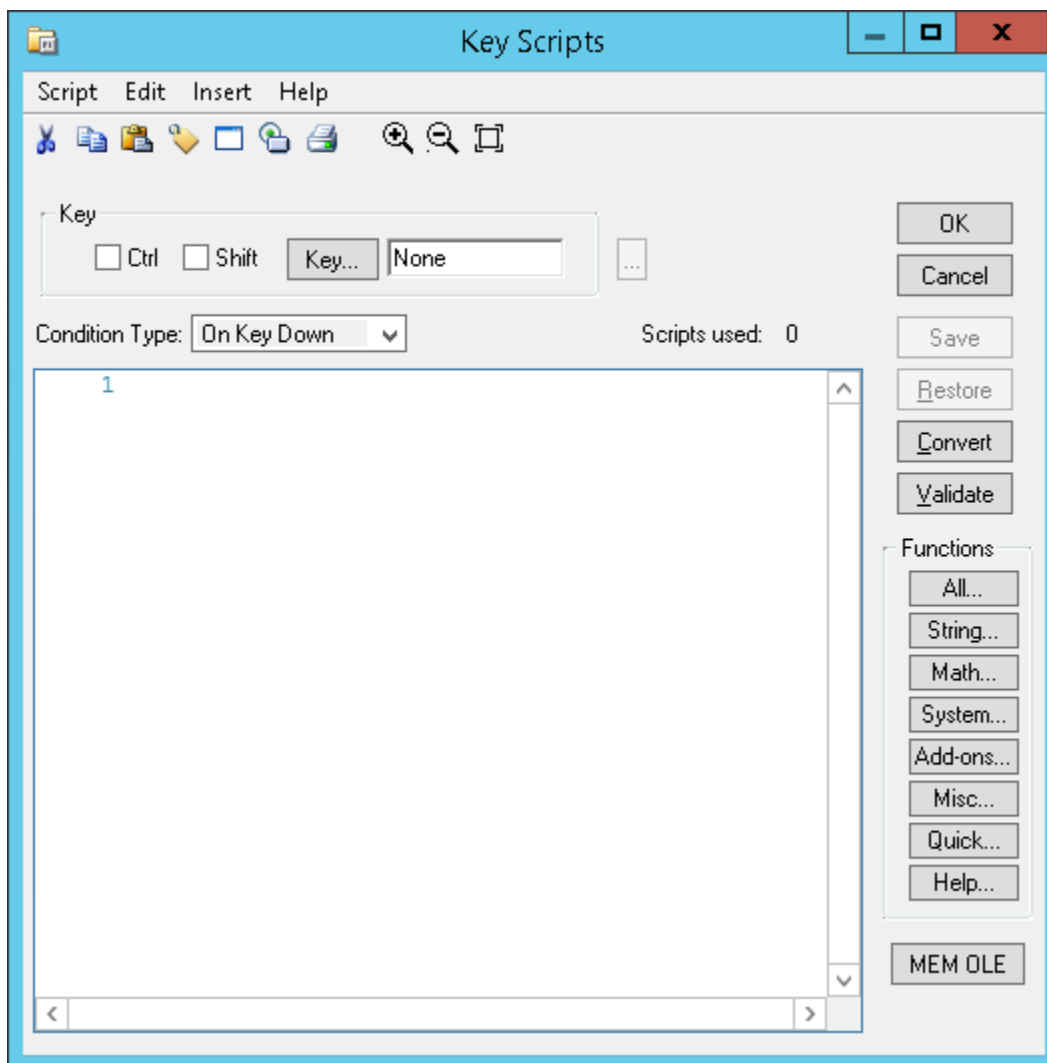
- キーまたはキーの組み合わせが押されたときにスクリプトを一度実行します。
- キーまたはキーの組み合わせが押されて、離されていないときにスクリプトを定期的に実行します。
- キーまたはキーの組み合わせが離されたときにスクリプトを一度実行します。

キー スクリプトは、スクリプトを開始するキーの名前で識別されます (**Ctrl+q** など)。

注記: トリガするために同じキーまたはキーの組み合わせを使用するアクション スクリプトを設定している場合、キー スクリプトは無視され、アクション スクリプトが実行されます。

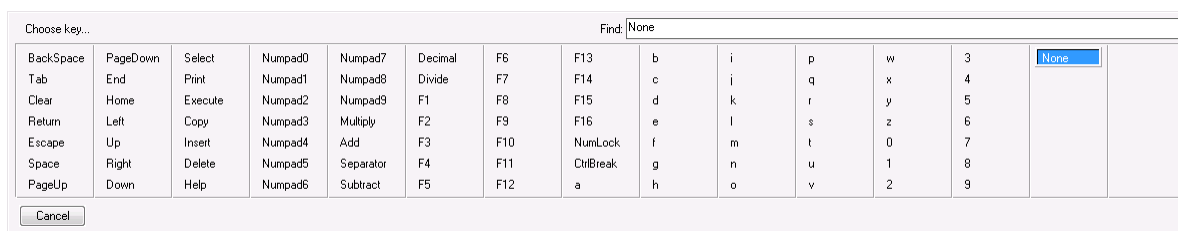
キー スクリプトを設定するには

1. **[スクリプト]** ペインで以下のいずれかを実行します。
 - 新しいキー スクリプトを設定するには、**[キー]** を右クリックして **[新規]** をクリックします。
[キー スクリプト] ダイアログ ボックスが表示されます。



- 既存のキー スクリプトを設定するには、[キー] を展開し、スクリプト名を右クリックして [編集] をクリックします。[キー スクリプトの編集] ダイアログ ボックスが表示されます。

2. [キー] をクリックして、[キーの選択] ダイアログ ボックスからキーを選択します。



3. [Ctrl] または [Shift]、あるいはこれら両方のチェック ボックスをオンにして、選択したキーに Ctrl キーや Shift キーの組み合わせを割り当てます。
4. [実行条件] リストで、以下のいずれかを実行します。
 - 関連付けられたキーまたはキーの組み合わせが押されたときに一度実行するようにスクリプトを設定するには、[押した時] をクリックします。

- 関連付けられたキーまたはキーの組み合わせが押されている間に定期的に行うようにスクリプトを設定するには、**[押している間]** をクリックします。
 - 関連付けられたキーまたはキーの組み合わせが離れたときに一度行うようにスクリプトを設定するには、**[離れた時]** をクリックします。
5. 前の手順で **[押している間]** を選択した場合、**[実行周期]** ボックスに 1 ～ 360000 ミリ秒の間隔を入力します。
 6. スクリプトをウィンドウに入力します。
 7. **[OK]** をクリックします。

キーに関連付けられているすべてのキー スクリプトを削除するには

- **[スクリプト]** ペインで **[キー]** を展開し、キー スクリプト名を右クリックして、**[削除]** をクリックします。メッセージが表示されたら、**[はい]** をクリックします。

キーに関連付けられているキー スクリプトを削除するには

1. **[クラシック ビュー]** を使用して、**[スクリプト]** ペインで、**[キー]** を展開し、キー スクリプト名を右クリックして、**[編集]** をクリックします。**[キー スクリプトの編集]** ダイアログ ボックスが表示されます。
2. **[実行条件]** リストで、削除するスクリプトのスクリプト トリガをクリックします。スクリプトが、**[キー スクリプトの編集]** ダイアログ ボックスのメイン セクションに表示されます。
3. **[編集]** メニューで、**[クリア]** をクリックします。スクリプトがメイン セクションからクリアされ、関連付けられたスクリプトが削除されます。

条件スクリプトの設定

条件スクリプトは、特定の論理条件が満たされたときにトリガされます。条件スクリプトを使用すると、以下のようにスクリプトを実行できます。

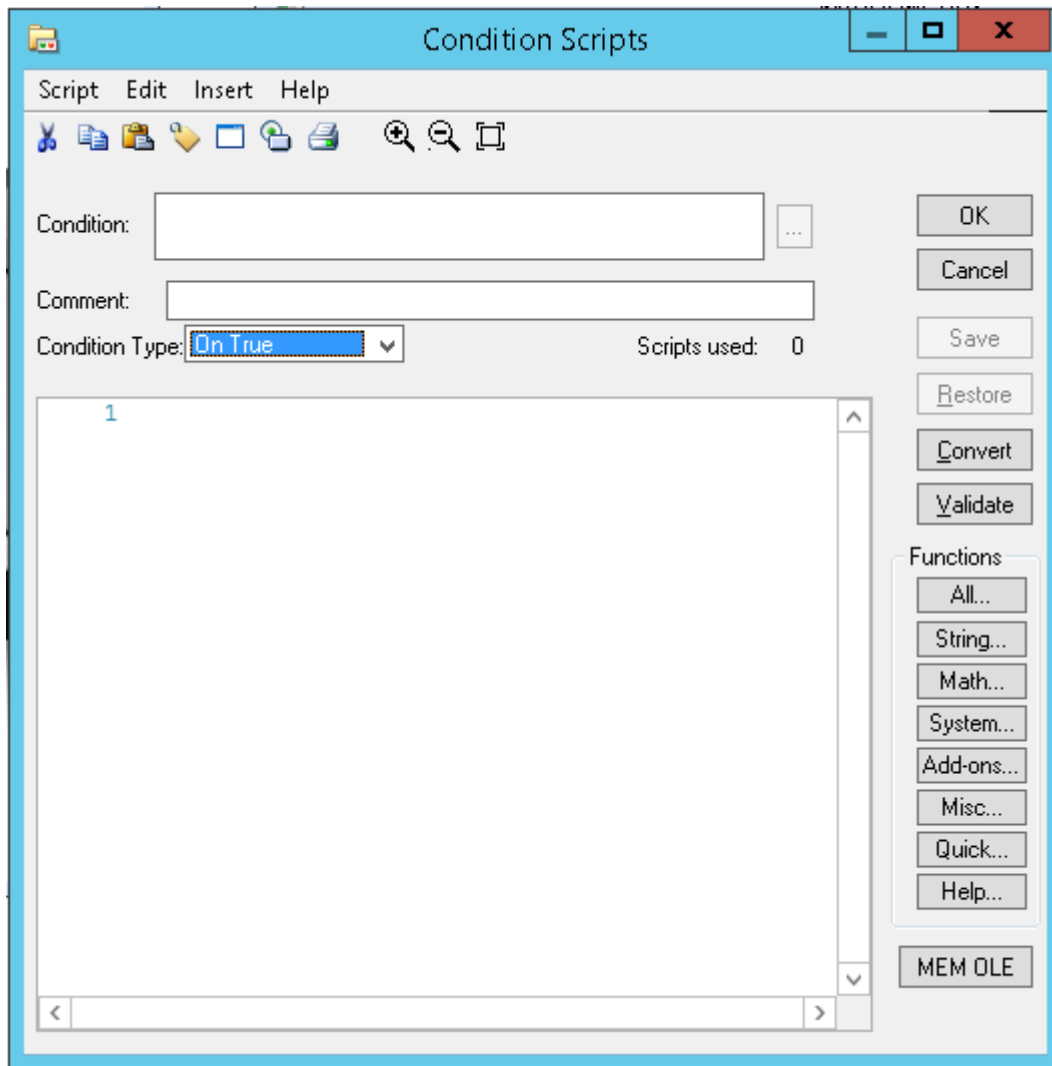
- 条件が満たされたときに一度実行する
- 条件が満たされなかったときに一度実行する
- 特定の条件が満たされている間定期的に実行する
- 特定の条件が満たされていない間定期的に実行する

条件スクリプトは、スクリプトを開始する条件構文により識別されます (tag1>=13 など)。

注記: True の時の条件タイプが割り当てられているスクリプトは、条件が False から True に変わったときにだけ実行されます。False の時の条件タイプが割り当てられているスクリプトは、条件が True から False に変わったときにだけ実行されます。

条件スクリプトを設定するには

1. **[スクリプト]** ペインで、**[条件]** を右クリックして、**[新規]** をクリックします。**[実行条件]** ダイアログ ボックスが表示されます。



- 既存の条件スクリプトを編集するには、[条件] の横にあるプラス記号をクリックし、条件スクリプト名を右クリックして、[編集] をクリックします。[条件スクリプトの編集] ダイアログボックスが表示されます。
2. [条件] ボックスに、条件として使用する式を入力します。
入力できる式の最大の長さは **1024** 文字です。
 3. [コメント] ボックスにコメントを入力することもできます。
 4. [実行条件] リストで、以下のいずれかを実行します。
 - 条件が **False** になったときに一度実行するようにスクリプトを設定するには、[False の時] をクリックします。
 - 条件が **False** の間に定期的に実行するようにスクリプトを設定するには、[False の間] をクリックします。
 - 条件が **True** になったときに一度実行するようにスクリプトを設定するには、[True の時] をクリックします。

- 条件が **True** の間に定期的に行うようにスクリプトを設定するには、**[True の間]** をクリックします。

5. 前の手順で **[False の間]** または **[True の間]** を選択した場合、**[実行周期]** ボックスに 1 ～ 360000 ミリ秒の間隔を入力します。

注記: 条件が **True** でなくなった場合、条件付き **WindowViewer** タイマーは停止します。たとえば、マウス ボタンが離された場合、**While Mouse Button Down** イベントはトリガされません。キーが離された場合、キー スクリプトは停止します。

1. スクリプトを入力するか、ウィンドウで既存のスクリプトを変更します。
2. **[OK]** をクリックします。

条件が関連付けられているすべての条件スクリプトを削除するには

- **[スクリプト]** ペインで **[条件]** を展開し、条件スクリプト名を右クリックして、**[削除]** をクリックします。メッセージが表示されたら、**[はい]** をクリックします。

条件が関連付けられている個々の条件スクリプトを削除するには

1. **[スクリプト]** ペインで **[条件]** を展開し、キー スクリプト名を右クリックして、**[編集]** をクリックします。**[条件スクリプトの編集]** ダイアログ ボックスが表示されます。
2. **[実行条件]** リストで、削除するスクリプトのスクリプト トリガをクリックします。スクリプトが、**[条件スクリプトの編集]** ダイアログ ボックスのメイン セクションに表示されます。
3. **[編集]** メニューで、**[クリア]** をクリックします。スクリプトがメイン セクションからクリアされ、関連付けられたスクリプトが削除されます。

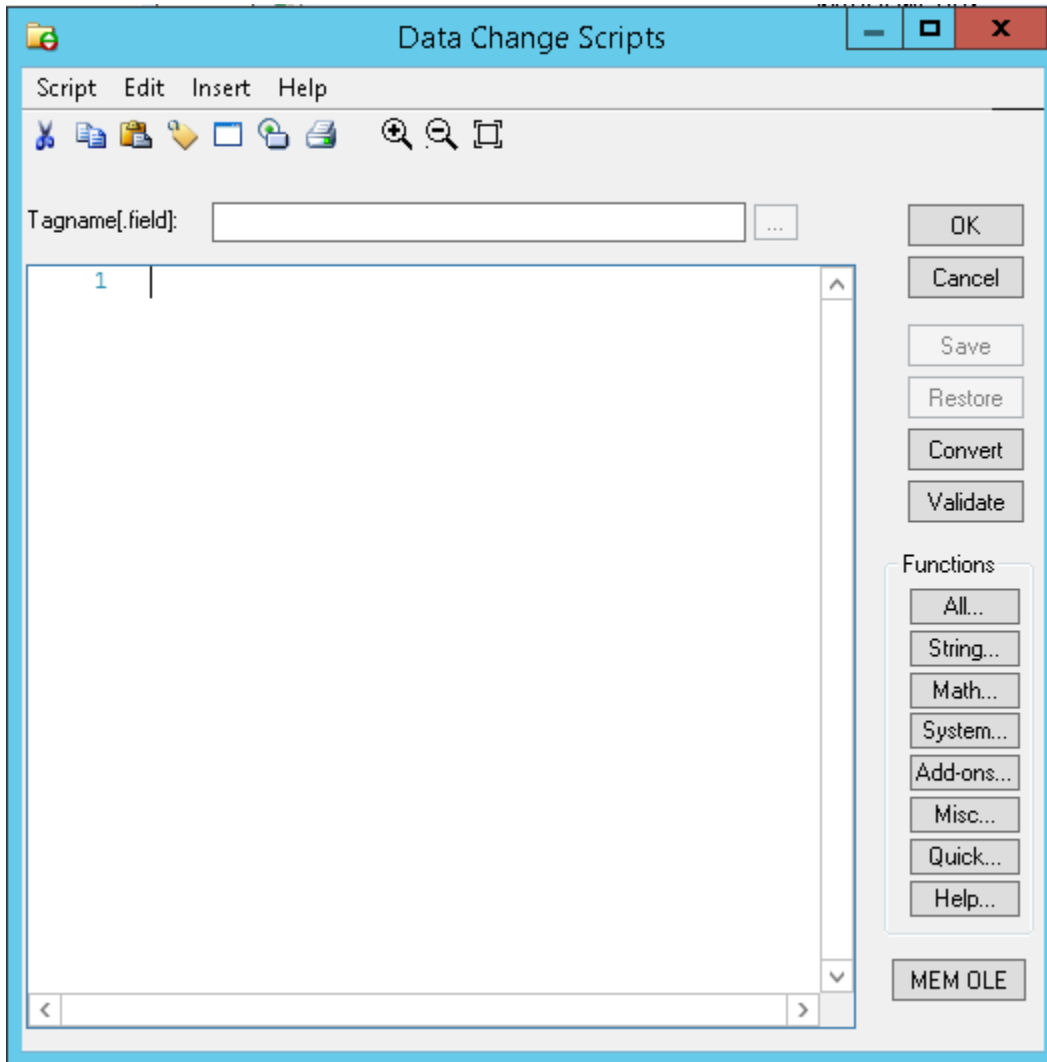
データ変化スクリプトの設定

データ変化スクリプトを使用すると、特定のタグ名またはドット フィールドの変化が定義済みのデッド バンドを超えた場合にスクリプトを一度実行できます。

データ変化スクリプトは、スクリプトを開始するタグ名またはタグ名フィールドで識別されます (Tag1 または Tag2.HiHiLimit など)。

データ変化スクリプトを設定するには

1. **[スクリプト]** ペインで、**[データ変化]** を右クリックして、**[新規]** をクリックします。
[データ変化スクリプト] ダイアログ ボックスが表示されます。



2. 新しいスクリプトを作成するには、[タグ名 [.フィールド]] ボックスに、タグ名またはタグ名フィールドを入力します。

既存のスクリプトを編集するには、[タグ名 [.フィールド]] ボックスの右側にある参照ボタンをクリックして、表示されるリストからスクリプトを選択します。

3. スクリプトをウィンドウに入力します。
4. [OK] をクリックします。

データ変化スクリプトを削除するには

- [スクリプト] ペインで [データ変化] を展開し、データ変化スクリプト名を右クリックして、[削除] をクリックします。メッセージが表示されたら、[はい] をクリックします。

アクション スクリプトの設定

アクションスクリプトを使用すると、オペレータアクションとグラフィックオブジェクトを関連付けることができます。以下の1つまたは複数のイベントをグラフィックオブジェクトで設定できます。

- 左、中央、右ボタンをクリックする

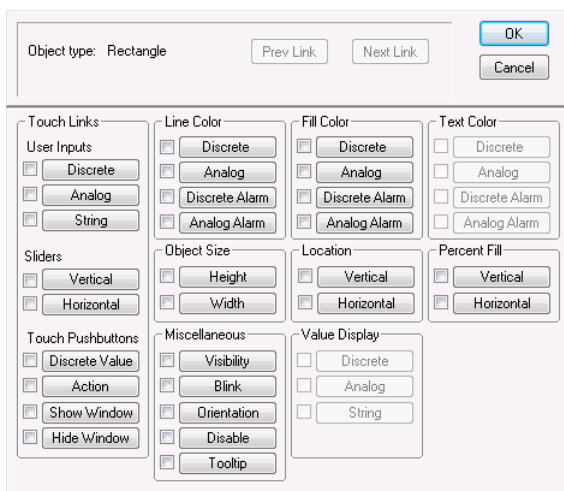
- 左、中央、右ボタンをクリックしたままの状態にする
- 左、中央、右ボタンを離す
- 左、中央、右ボタンをダブルクリックする
- キーおよびキーの組み合わせを押す
- キーおよびキーの組み合わせを押したままの状態にする
- キーおよびキーの組み合わせを離す
- マウス ポインタをオブジェクト上に移動させる

アクション スクリプトを設定できるのは、オブジェクト自体の [アニメーション リンクの選択] パネルだけです。

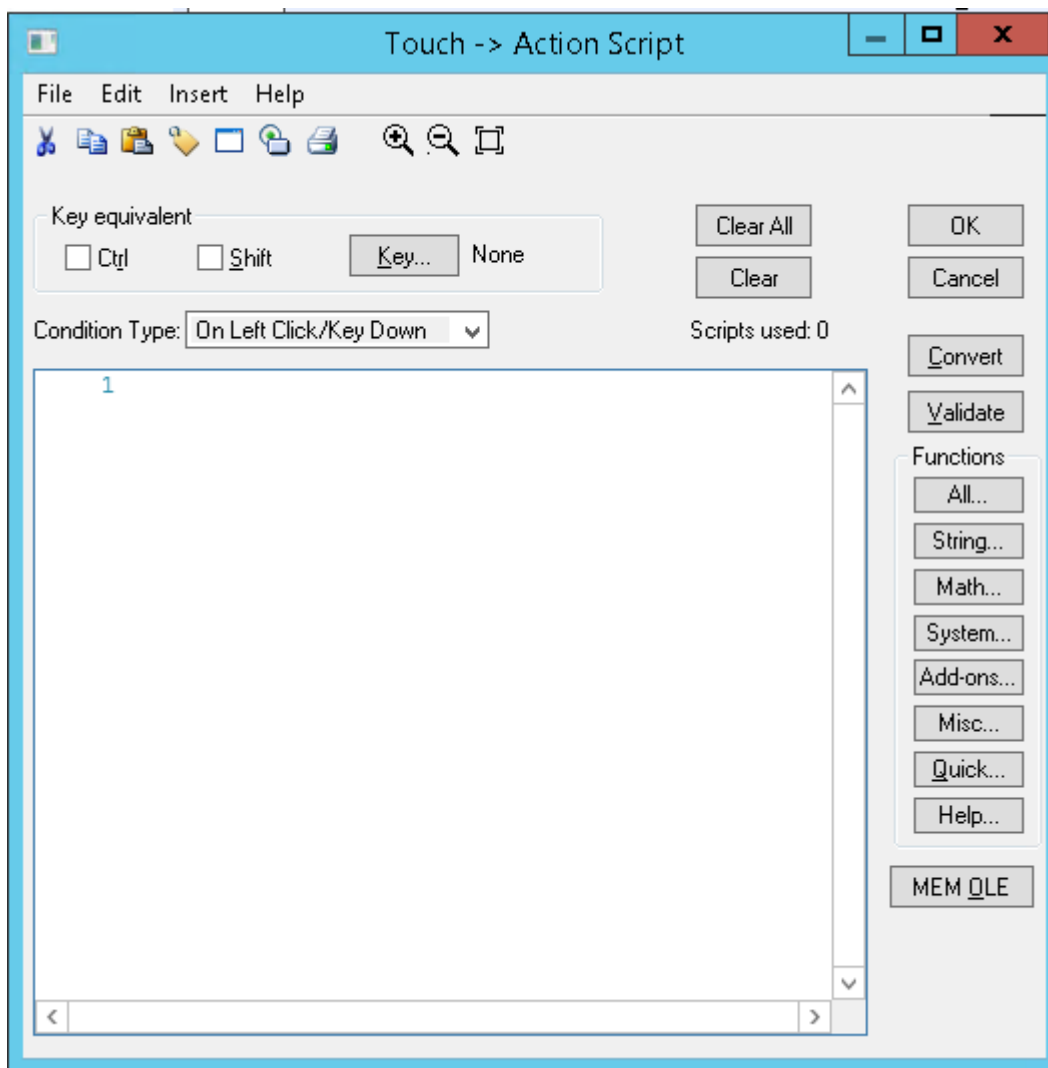
重要: アクション スクリプトと同じキーまたはキーの組み合わせによりトリガされるキー スクリプトが存在する場合、アクション スクリプトが実行され、キー スクリプトは無視されます。

アクション スクリプトを設定するには

1. グラフィック オブジェクトをダブルクリックします。アニメーション リンクの選択パネルが表示されます。



2. [スクリプト ボタン] をクリックします。[タッチアクションスクリプト] ダイアログ ボックスが表示されます。



3. [実行条件] リストで、以下のいずれかをクリックします。

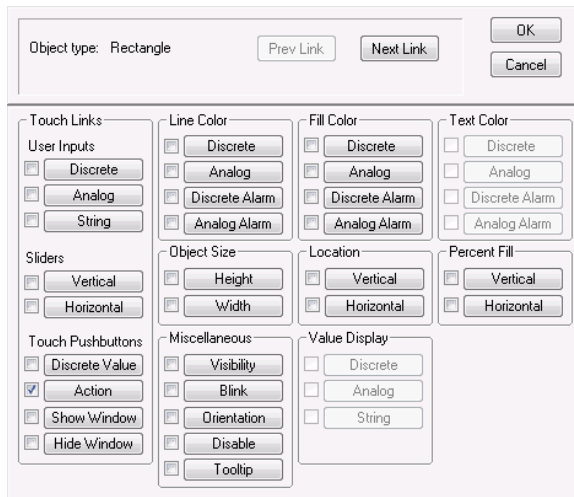
設定するスクリプトの実行条件	クリック
左マウス ボタンまたは特定のキーやキーの組み合わせが押されたときに一度実行する	左クリック／ホットキーを押したとき
左マウス ボタンまたは特定のキーやキーの組み合わせが押されたときに定期的に実行する	左クリック／ホットキーを押している間
左マウス ボタンまたは特定のキーやキーの組み合わせが離されたときに一度実行する	左クリック／ホットキーを離したとき
左マウス ボタンがダブルクリックされたときに一度実行する	左ダブルクリック時
右マウス ボタンが押されたときに一度実行する	右クリック時
右マウス ボタンが押されている間に定期的に実行する	右クリックを押している間

設定するスクリプトの実行条件	クリック
右マウス ボタンが離されたときに一度実行する	右クリックを離した時
右マウス ボタンがダブルクリックされたときに一度実行する	右ダブルクリック時
中央マウス ボタンが押されたときに一度実行する	中クリック時
中央マウス ボタンが押されている間に定期的に実行する	中クリックを押している間
中央マウス ボタンが離されたときに一度実行する	中クリックを離した時
中央マウス ボタンがダブルクリックされたときに一度実行する	中ダブルクリック時
マウスがオブジェクト上を移動したときに一度実行する	マウス ホバー時

4. [左クリック時／ホットキーを押したとき]、[左クリック／ホットキーを押している間] または [左クリック／ホットキーを離したとき] を選択した場合、以下を実行します。
 - a. [キー] をクリックします。[キーの選択] ダイアログ ボックスが表示されます。
 - b. キーをクリックします。
 - c. [Ctrl] または [Shift]、あるいはこれら両方のチェック ボックスをオンにして、コントロール キーやシフト キーの組み合わせを選択したキーに割り当てます。
5. [左クリック／ホットキーを押している間] または [右クリックを押している間] を選択した場合、[実行周期] ボックスに 1 ～ 360000 ミリ秒の間隔を入力します。
6. [マウス ホバー時] を選択した場合、[後] ボックスに、1 ～ 360000 ミリ秒の間隔を入力します。これは、マウスがオブジェクトを通過してから、スクリプトが実行されるまでの時間です。
7. スクリプトをウィンドウに入力します。
8. [OK] をクリックします。

InTouch グラフィック オブジェクトに関連付けられているすべてのアクション スクリプトを削除するには

1. グラフィック オブジェクトをダブルクリックします。オブジェクト プロパティ パネルが表示されます。



2. [スクリプト ボタン] チェック ボックスをオフにします。アクション スクリプトは、ランタイムでは実行されなくなります。[スクリプト ボタン] ボタンをクリックすると、エディタが開き、前回オブジェクトに対して保存したアクション スクリプトが表示されます。

個々のアクション スクリプトを削除するには

1. 削除するアクション スクリプトを持つグラフィック オブジェクトをダブルクリックします。オブジェクト プロパティ パネルが表示されます。
2. [スクリプト ボタン] ボタンをクリックします。[タッチアクションスクリプト] ダイアログ ボックスが表示されます。
3. [実行条件] リストで、スクリプト トリガをクリックします。
4. [編集] メニューで、[クリア] をクリックします。スクリプトがメインセクションからクリアされ、関連付けられたスクリプトが削除されます。

ActiveX イベント スクリプトの設定

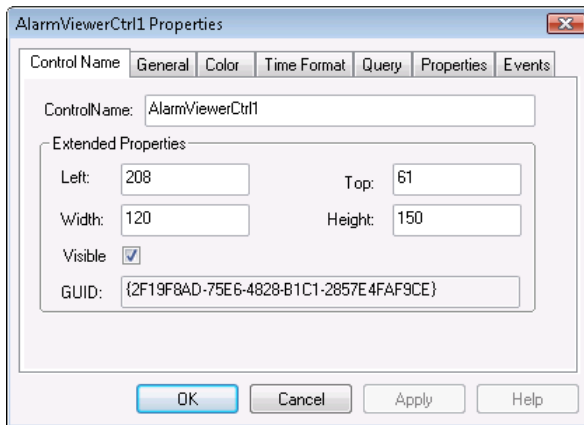
ActiveX イベント スクリプトを使用すると、ActiveX イベントが発生したときにスクリプトを実行できます。ActiveX コントロールにより、以下のイベントが含まれます。

- ActiveX コントロールが起動した：起動
- ActiveX コントロールが閉じられた：終了
- ユーザーが ActiveX コントロールをクリックした：クリック
- ユーザーが ActiveX コントロールをダブルクリックした：ダブルクリック

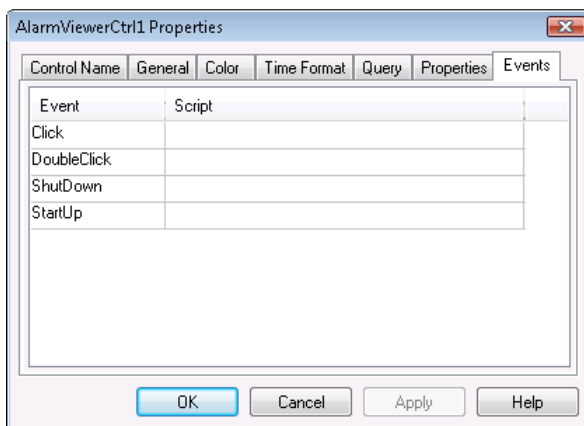
ActiveX イベント スクリプトは名前で識別されます。デフォルトでは、InTouch HMI は、コントロール名と、スクリプトが関連付けられているイベントを自動的に追加します (MyActiveXScript (AlarmViewerCtrl1::Click) など)。

新しい ActiveX イベント スクリプトを設定するには

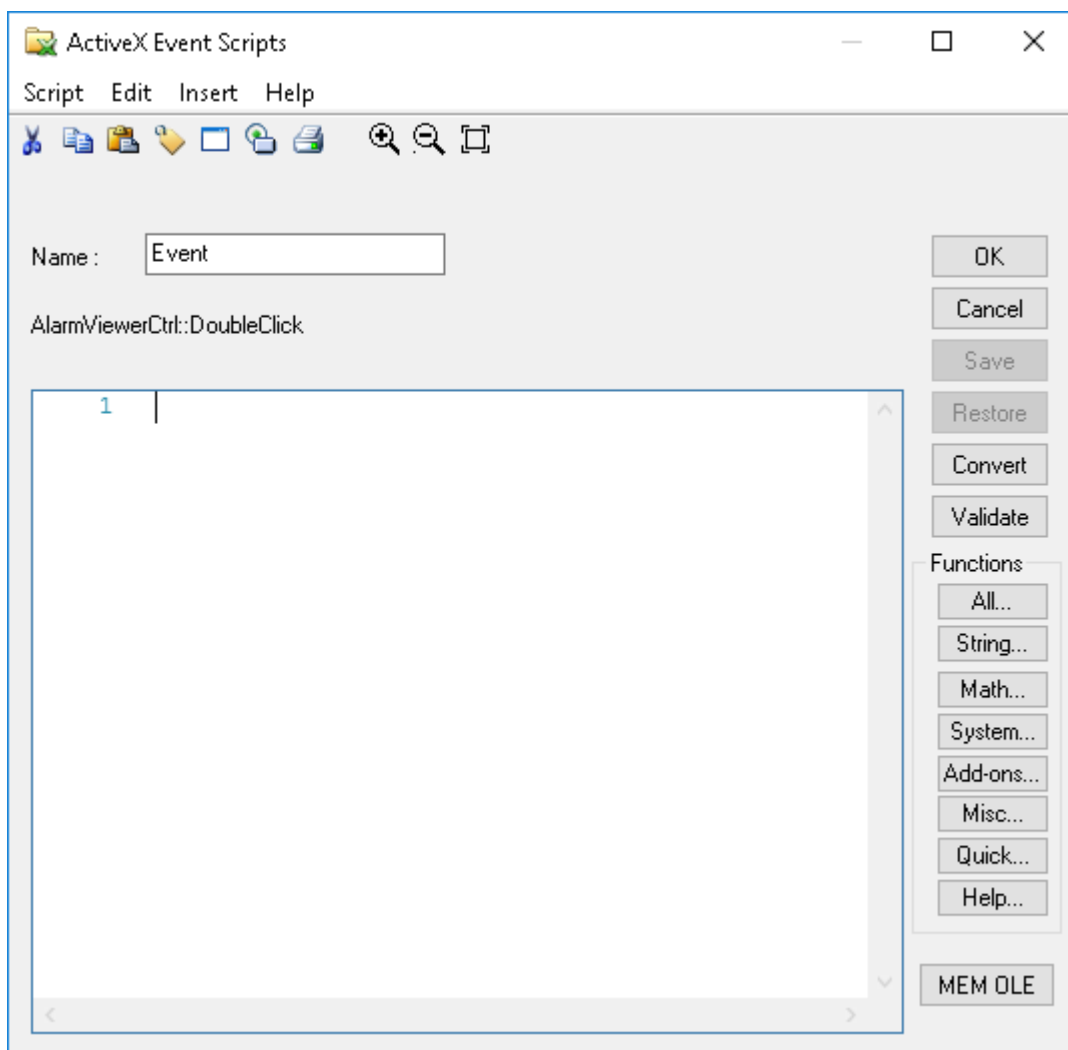
1. 設定する ActiveX コントロールをダブルクリックします。ActiveX コントロール プロパティ ダイアログ ボックスが表示されます。



2. [イベント] タブをクリックします。



3. クリック、ダブルクリック、終了、または起動などのイベントを選択します。
 4. イベントの [スクリプト] セルをクリックします。角括弧が表示されます。
 5. イベント スクリプトの新しい名前を入力して、[OK] をクリックします。メッセージが表示されたら、[OK] をクリックして、新しいスクリプトを作成します。
- [ActiveX イベント スクリプト] ダイアログ ボックスが表示されます。



1. [スクリプト名] ボックスで、ActiveX イベント スクリプト名を変更できます。
2. スクリプトをウィンドウに入力します。
3. [OK] をクリックします。

既存の ActiveX イベント スクリプトを編集するには

1. [スクリプト] ペインで [ActiveX イベント] を展開し、ActiveX スクリプト名を右クリックして、[編集] をクリックします。[ActiveX イベント スクリプト] ダイアログ ボックスが表示されます。
2. スクリプトを変更したら、[OK] をクリックします。

既存の ActiveX イベント スクリプトを削除するには

1. ActiveX コントロールが、削除する ActiveX イベント スクリプトを使用していないことを確認します。スクリプトを使用している ActiveX コントロールが存在する場合、まず以下の手順を実行します。
 - a. スクリプトを使用しているすべての ActiveX コントロールの [イベント] パネルで、ActiveX イベント スクリプト参照を削除します。
 - b. すべてのウィンドウを閉じ、使用タグの検索を更新します。

2. [スクリプト] ペインで [ActiveX イベント] を展開し、ActiveX スクリプト名を右クリックして、[削除] をクリックします。メッセージが表示されたら、[はい] をクリックします。ActiveX イベントスクリプトが削除されます。

ランタイムでのスクリプト実行の一時停止

デフォルトでは、WindowViewer が起動されると、ロジックが実行され、同期スクリプトが実行されます。スクリプトの実行は、ロジックを中断することによってランタイムで一時停止できます。一時停止した後、スクリプト実行は再開できます。

メニューからランタイムでスクリプト実行を一時停止するには

- [ロジック] メニューで、[ロジック ストップ] をクリックします。同期スクリプトは、実行を停止します。非同期スクリプトは、引き続き実行されますが、新しい非同期スクリプトは起動されません。

スクリプトでランタイムにスクリプト実行を一時停止するには

- 値 0 を論理型システム タグ変数 \$LogicRunning に書き込みます。同期スクリプトは、実行を停止します。非同期スクリプトは、引き続き実行されますが、新しい非同期スクリプトは起動されません。

ランタイムでスクリプト実行を再開する

- [ロジック] メニューで、[ロジック スタート] をクリックします。スクリプト実行が再開されます。

スクリプトを使用してランタイムでスクリプト実行を再開するには

- 値 1 を論理型システム タグ変数 \$LogicRunning に書き込みます。\$LogicRunning システム タグ変数は、ロジックが一時停止されたときに実行している非同期スクリプトに含める必要があります。

\$LogicRunning システム タグ変数

このシステム タグ変数は、スクリプト実行の監視や制御を行います。

使用法

\$LogicRunning

備考

値を 1 に設定すると、スクリプト実行が開始します。値を 0 に設定すると、スクリプト実行が停止します。

このシステム タグ変数は、WindowViewer の [ロジック] メニューで、[ロジック スタート] または [ロジック ストップ] を選択する操作と同じ機能を果たします。

現在実行している非同期スクリプトを停止することはできません。ただし、新しいスクリプトが実行されることは回避できます。

データタイプ

論理型（読み取り／書き込み）

スクリプト言語

InTouch HMI スクリプト言語を使用したスクリプトの作成には、以下の概念、技術、および構文規則を使用します。

- 基本構文規則。「[基本構文規則](#)」を参照してください。
- 定義済み関数またはカスタム関数の呼び出し。「[標準関数の呼び出し](#)」および「[カスタム関数（クイック関数）の呼び出し](#)」を参照してください。
- 値の割り当ておよびさまざまな演算子の使用。「[値の割り当てと演算子](#)」を参照してください。
- 条件付きステートメントの使用。「[条件付きプログラム分岐構造の使用](#)」を参照してください。
- ループの使用。「[プログラムループの使用](#)」を参照してください。
- ローカル変数の使用。「[ローカル変数の使用](#)」を参照してください。

スクリプトエディタの一般的な操作の詳細については、「[スクリプトの作成と編集](#)」を参照してください。

スクリプトトリガのさまざまなタイプの詳細については、「[スクリプトトリガ](#)」を参照してください。
標準スクリプト関数の参照については、「[組み込み関数](#)」を参照してください。

基本構文規則

基本構文規則では、InTouch HMI スクリプト言語に関する以下の側面について示しています。

- サブルーチン
- ステートメント
- インデント
- コメント
- タグリファレンス
- リテラルデータ値
- 値式
- 構文確認

サブルーチン

Visual Basic の「Sub」プロシージャなど、同じスクリプト内で個別のサブルーチンを使用するというコンセプトはありません。スクリプトを複数のサブルーチンで構成するには、各サブルーチンにカスタムクイック関数を作成する必要があります。「[カスタムスクリプト関数](#)」を参照してください。

ステートメント

- ステートメントには、値の割り当て、関数呼び出し、または制御構造を使用できます。
- スクリプトの各ステートメントは、セミコロン (;) で終了する必要があります。
- 各ステートメントがセミコロンで終了している限り、同じ行で複数のステートメントを使用できます。
- 改行 (Enter キーを押す) を使用することで、1つのステートメントを複数の行に分割できます。

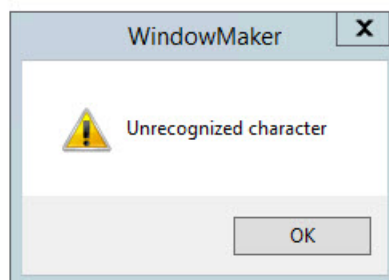
インデント

スクリプトコードは、任意の方法でインデントできます。インデントには、機能的な関連性はありません。

コメント

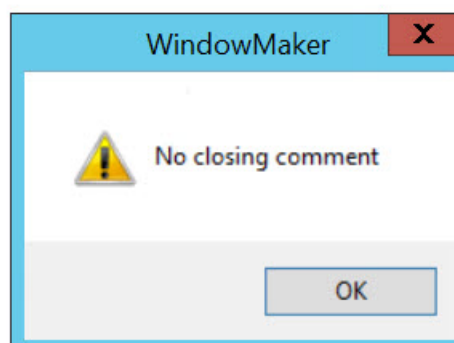
テキストをコメントにするには、括弧 { } で囲みます。コメントが複数の行に及んでも構いません。ネストされたコメントはサポートされません。たとえば、次の例では、最初の } はコメントを閉じ、2 番目の } に対してエラーが表示されます。

```
1 DIM msg AS Message;  
2 {  
3  
4 Comments across multiple lines  
5  
6 {  
7 Nested comments  
8 }  
9 }  
10 msg = $DateString + ":" + $Operator;
```



} がない場合、検証中にエラーが表示されます。

```
1 DIM msg AS Message;  
2 {  
3  
4 Comments across multiple lines  
5  
6  
7 Nested comments  
8  
9 msg = $DateString + ":" + $Operator;
```



タグ リファレンス

タグ リファレンスはいくつかの方法で作成できます。

- ローカル タグ変数ディクショナリで定義されているタグ変数を参照するには、簡単にタグ変数名を使用します。
- 特定のドット フィールドを参照するには、通常の参照形式 (Tagname.Dotfield) を使用します。
- リモート ノードのデータ アイテムを参照するには、通常のリモート タグ リファレンス (AccessName:Item) を使用します。
- また、範囲が現在のスクリプトに制限されるローカル変数を定義することもできます。「[ローカル変数の使用](#)」を参照してください。

リテラルデータ値

- 整数値を 10 進数または 16 進数で指定できます。たとえば、255 または 0xFF です。
- 浮動小数点値を 10 進数または 16 進数で指定できます。たとえば、0.001 または 1E-3 です。

- ブール値を指定するには、FALSE の場合は数値 0、TRUE の場合は数値 1 を使用します。
- 文字列値を指定するには、二重引用符で囲みます。次に例を示します。"This is a string."

値式

値式には、リテラル値、タグリファレンス、および関数呼び出し、または適切な演算子によりリンクされたこれらのすべての組み合わせを含めることができます。「[値の割り当てと演算子](#)」を参照してください。

構文確認

スクリプトを保存する場合、スクリプトエディタにより、その構文が正しいかどうか自動的に確認されます。[確認] ボタンをクリックすると、手動でこの確認を開始できます。「[正確な構文を確認するためのスクリプトの検証](#)」を参照してください。

標準関数の呼び出し

標準関数は、InTouch HMI で事前に定義されています。「[カスタム関数（クイック関数）の呼び出し](#)」を参照してください。

標準関数の呼び出しの構文

事前に定義されたスクリプト関数を呼び出す構文は、関数が結果を返すかどうか、およびどのように返すかにより異なります。

関数には、結果を返さない関数や、タグ変数に割り当てたり、式で使用するオプションの結果を返す関数、またタグ変数に割り当てたり、式で使用する必要がある結果を返す関数があります。

関数タイプを判断するには、関数の説明を参照してください。それぞれの関数の説明には、関数が結果を返すかどうか、結果がオプションかどうかを示す構文リストがあります。

結果を返さない関数を呼び出すには

- 関数名のみ（および存在する場合はパラメータ）をステートメントで使用します。次に例を示します。

```
FunctionName(Parameters);
```

結果を割り当てる必要がある関数を呼び出すには

- リテラル値または関連するデータ タイプのタグ変数を使用できるスクリプト内のいずれかの場所で、関数名（および存在する場合はパラメータ）を使用します。たとえば、値割り当てでは、以下ようになります。

```
ResultsTagname = FunctionName(Parameters);
```

または、別の標準関数のパラメータとして使用する、入れ子の関数呼び出しでは以下ようになります。

```
OtherStandardFunction(FunctionName(Parameters));
```

オプションの結果を返す関数を呼び出すには

- 前の手順のいずれかを使用します。

パラメータを関数に渡す

標準の事前定義関数へのパラメータは通常、値により渡されます。つまり、式がパラメータに対して必要なデータ タイプを評価する限り、有効な式をパラメータとして渡すことができます。このような式には、リテラル値、タグリファレンス、関数呼び出し、または適切な演算子によりリンクされたこれらの

すべての組み合わせを含めることができます。式および演算子の詳細については、「[値の割り当てと演算子](#)」を参照してください。

スクリプトが関数を呼び出すと、式が評価され、結果の値が関数に渡されます。

ただし、関数によっては、タグ リファレンスをパラメータとして必要とする場合があります。次に例を示します。

```
RecipeSelectRecipe(Filename, RecipeName, Number);
```

この例では、RecipeName パラメータはタグ リファレンスでなければなりません（つまり、RecipeName パラメータのリテラルパラメータを使用する必要があります）。式が有効なタグ変数を評価する場合でも、文字列式を代わりに渡すことはできません。

注意：パラメータが 1 つだけの従来の事前定義関数の一部は（たとえば、Ack() 関数）、括弧内にパラメータを渡すという標準構文には従いません。その代わり、パラメータは、スペースにより関数名と区別されます。特定の関数に関して不明な場合は、関数のマニュアル内の構文説明を確認してください。

カスタム関数（クイック関数）の呼び出し

カスタム クイック関数の呼び出しは、事前定義された標準関数の呼び出しとは若干異なります。

- キーワード **CALL** をクイック関数名の前に付ける必要があります。
- クイック関数により返される結果は、常にオプションです。これらを使用することはできますが、使用しなくても構いません。

結果を返さないクイック関数を呼び出すには

- キーワード **CALL** を前に付けた関数名（存在する場合はパラメータ）をステートメントで使用します。次に例を示します。

```
CALL QuickFunctionName(Parameters);
```

結果を返すクイック関数を呼び出すには

- 以下のいずれかを実行します。
 - 結果を返さなかったものとしてクイック関数を呼び出す（前述の手順を参照してください）
 - リテラル値または関連するデータ タイプのタグ変数を使用できるスクリプトのいずれかの場所で、キーワード **CALL** を前に付けた関数名（存在する場合はパラメータ）を使用します。たとえば、値割り当てでは、以下のようになります。

```
ResultsTagname = CALL QuickFunctionName(Parameters);
```

または、標準関数のパラメータとして使用する、入れ子の関数呼び出しでは以下のようになります。

```
OtherStandardFunction(CALL FunctionName(Parameters));
```

注意：クイック関数が別のクイック関数のパラメータとして使用されるように、クイック関数呼び出しを入れ子にすることはできません。たとえば、Call QF1(Call QF2()); は有効なステートメントではありません。

パラメータをクイック関数に渡す

クイック関数へのパラメータは常に、値によって渡されます。リファレンスによって、パラメータをクイック関数に渡すことはできません。

式がパラメータに対して必要なデータ タイプを評価する限り、有効な式をパラメータとして渡すことができます。このような式には、リテラル値、タグ リファレンス、関数呼び出し、または適切な演算子に

よりリンクされたこれらのすべての組み合わせを含めることができます。式および演算子の詳細については、「[値の割り当てと演算子](#)」を参照してください。スクリプトが関数を呼び出すと、式が評価され、結果の値が関数に渡されます。

注意：クイック関数が別のクイック関数のパラメータとして使用されるように、クイック関数呼び出しを入れ子にすることはできません。たとえば、`CALL QF1(CALL QF2());` は有効なステートメントではありません。

値の割り当てと演算子

スクリプトでは、値の割り当てを使用して、値をタグ変数に書き込むことができます。値の割り当ての構文は以下のとおりです。

```
Tagname = ValueExpression;
```

このステートメントが実行されると、**ValueExpression** が、**Tagname** により参照されるタグ変数に書き込まれます。**ValueExpression** には、データタイプがタグ変数データタイプに一致する任意の有効な式を使用できます。値式には、リテラル値、タグリファレンス、および関数呼び出し、または適切な演算子によりリンクされたこれらのすべての組み合わせを含めることができます。

「[サポートされている演算子](#)」を参照してください。

「[演算子の評価順の設定](#)」を参照してください。

「[式の例](#)」を参照してください。

サポートされている演算子

以下の表では、サポートされているすべての演算子を一覧表示しています。特定の演算子の使用の詳細については、関連するセクションを参照してください。

演算子	詳細
+	加算または連鎖：+
-	減算：-
*	乗算：*
/	除算：/
**	累乗：**
MOD	剰余：MOD
~	補数：~
SHL	左方向桁移動：SHL、および右方向桁移動：SHR
SHR	左方向桁移動：SHL、および右方向桁移動：SHR
&	ビット単位の論理積：&
	ビット単位の論理和：
^	ビット単位の排他的論理和：^
AND	論理積：AND

演算子	詳細
OR	論理和 : OR
NOT	否定 : NOT
<	比較 : <、>、<=、>=、==、<>
>	比較 : <、>、<=、>=、==、<>
<=	比較 : <、>、<=、>=、==、<>
>=	比較 : <、>、<=、>=、==、<>
==	比較 : <、>、<=、>=、==、<>
<>	比較 : <、>、<=、>=、==、<>

注意：数値計算では、計算結果が実数値の範囲内となるように、常に演算数を選択してください。このようにしないと、結果が正しいものとなりません。

加算または連鎖 : +

2 つの数値演算数を加算するか、または 2 つの文字列演算数を連結します。

有効な演算数

加算の場合：任意の整数値または実数値

連結の場合：メッセージ値

戻り値のデータ タイプ

加算の場合：整数型または実数型

連結の場合：メッセージ

例

```
MessageTag = "Setpoint value:" + Text(SetpointTag, "#.##");
```

減算 : -

2 つの演算数と共に使用すると、通常の数値減算が実行されます。

有効な演算数

任意の整数値または実数値

戻り値のデータ タイプ

整数型または実数型

例

TemperatureSetpoint が 70 の場合、この例のスクリプトを実行した後の TemperatureSetpoint は 65 になります。

```
TemperatureSetpoint = TemperatureSetpoint - 5;
```

乗算 : *

通常の数値乗算。

有効な演算数

任意の整数値または実数値

戻り値のデータ タイプ

整数型または実数型

除算 : /

通常の数値除算。ランタイムに 0 で除算しようとする、結果として 0 が返されます。

有効な演算数

任意の整数値または実数値

戻り値のデータ タイプ

整数型または実数型

累乗 : **

左側の演算数（基数）を右側の演算数（べき数）で累乗します。

有効な演算数

整数値または実数値。基数 0 と負のべき数、または負の奇数と分数のべき数を組み合わせて使用することはできません。このような場合、結果として 0 が返されます。

戻り値のデータ タイプ

整数型または実数型

例

8 ** (1/3) は 2 を返します（8 の立方根）

剰余 : MOD

2 つの整数値による除算の余りを返します。

有効な演算数

任意の整数値

戻り値のデータ タイプ

整数

例

37 MOD 4 は 1 を返します

補数 : ~

整数値の 1 の補数を返します。つまり、各 0 ビットと 1 ビット（またはその逆）を変換します。

有効な演算数

任意の整数値

戻り値のデータ タイプ

整数

左方向桁移動：SHL、および右方向桁移動：SHR

整数値のバイナリ表現を指定ビット位置の数まで右または左に移動します。左側の演算数は移動される値で、右側の演算数はビット位置の数です。桁あふれしたビットは切り捨てます。桁移動によって空いたビット位置は、ゼロに設定されます。

有効な演算数

任意の整数値

戻り値のデータ タイプ

整数

例

`IntTag = IntTag SHL 1;` の結果は、初期タグ変数値 5 に対して繰り返し実行された場合、以下のようになります。

反復回数	バイナリ パターン	タグ変数値
初期値	0[...]00000101	5
実行 1	0[...]00001010	10
実行 2	0[...]00010100	20

ビット単位の論理積：&

2 つの整数のバイナリ表現をビット単位で比較し、以下の表に示すように結果を返します。

最初の演算数のビット	2 つ目の演算数のビット	結果のビット
0	0	0
0	1	0
1	0	0
1	1	1

この演算子を使用すると、ビット パターンの特定の部分を素早く「マスク」(0 に設定) できます。たとえば、以下のステートメントは、`IntTag` タグ変数の上位 24 ビットをマスクします。

```
IntTag = IntTag & 255;
```

表に示すように、演算数ビットのいずれかが 0 の場合、結果のビットは常に 0 となります。255 のバイナリ表現では、下位 8 ビットだけが 1 なので、残りの 24 の 0 ビットにより、結果の対応するすべてのビットが 0 に設定されます。

有効な演算数

任意の整数値

戻り値のデータ タイプ

整数

ビット単位の論理和：|

2つの整数のバイナリ表現をビット単位で比較し、以下の表に示すように結果を返します。

最初の演算数のビット	2つ目の演算数のビット	結果のビット
0	0	0
0	1	1
1	0	1
1	1	1

この演算は、「包含的論理和」とも呼ばれます。

有効な演算数

任意の整数値

戻り値のデータ タイプ

整数

ビット単位の排他的論理和：^

2つの整数のバイナリ表現をビット単位で比較し、以下の表に示すように結果を返します。

最初の演算数のビット	2つ目の演算数のビット	結果のビット
0	0	0
0	1	1
1	0	1
1	1	0

この演算は、「排他的論理和」とも呼ばれます。

有効な演算数

任意の整数値

戻り値のデータ タイプ

整数

論理積：AND

両方の論理演算数が TRUE の場合 TRUE を返し、そうでない場合 FALSE を返します。この演算子の真理値表を以下に示します。

p	q	p AND q
F	F	F
F	T	F

p	q	p AND q
T	F	F
T	T	T

有効な演算数

任意の論理値

戻り値のデータ タイプ

論理型

論理和 : OR

少なくとも一方の論理演算数が TRUE の場合 TRUE を返し、そうでない場合 FALSE を返します。この演算子の真理値表を以下に示します。

p	q	p OR q
F	F	F
F	T	T
T	F	T
T	T	T

有効な演算数

任意の論理値

戻り値のデータ タイプ

論理型

否定 : NOT

論理値が FALSE の場合 TRUE を返し、TRUE の場合 FALSE を返します。この演算子の真理値表を以下に示します。

p	NOT p
F	T
T	F

有効な演算数

任意の論理値

戻り値のデータ タイプ

論理型

比較 : <, >, <=, >=, ==, <>

これらの演算子は、2つの値を比較して、演算子により指定された条件が満たされる場合 TRUE を返します。演算数には任意のデータ タイプを使用できます。文字列演算数の場合、比較は大文字小文字の区

別がないアルファベット順で行われます。たとえば、**b** は **a** より大きく、**c** は **b** より大きいなどです。論理演算数の場合、**TRUE** は **FALSE** より大きいとみなされます。以下の表は、すべての比較演算子およびそれらの条件を示しています。

演算	例	条件
より小さい	$a < b$	a は b より小さい
より大きい	$a > b$	a は b より大きい
以下	$a \leq b$	a は b 以下
以上	$a \geq b$	a は b 以上
等しい	$a == b$	a は b と等しい
等しくない	$a \neq b$	a は b と等しくない

有効な演算数

任意のデータ タイプの値（両方の値のデータ タイプは同じでなければなりません）。

戻り値のデータ タイプ

論理型

演算子の評価順の設定

任意の式で、括弧を使用することで、特定の順序で演算子を評価させることができます。これは、任意の数学計算と同じように機能します。括弧を使用しない場合、式は演算子のデフォルトの優先度規則に基づいて評価されます。優先度が最も高いレベルの演算が最初に実行され、その後で 2 番目、3 番目と続きます。

以下の表は、各演算子の優先度レベルを示しています。同じ行の演算子の優先度レベルは同じです。

-、NOT、~	最高優先度
**	
＊、/、MOD	
＋、－	
SHL、SHR	
<、>、<=、>=	
==、<>	
&	
^	
AND	
OR	
=	最低優先度

暗黙的なデータ タイプ変換

InTouch HMI スクリプト言語は、特定のデータ タイプ間の割り当てで、暗黙的な値変換の機能を提供します。ただし、これにより予測しない結果が生じることがあるので、この機能を使用する場合は十分に注意してください。

以下の表は、特定のタイプの値を異なるタイプのタグ変数に割り当てた場合の結果を示しています。

予測されるデータ タイプ	使用されるデータ タイプ	備考
論理型	整数型	値 0 は FALSE と解釈されます。その他の値は TRUE と解釈されます。
論理型	実数型	値 0 は FALSE と解釈されます。その他の値は TRUE と解釈されます。
整数型	論理型	FALSE の値は 0 に変換されます。TRUE の値は 1 に変換されます。
整数型	実数型	小数点より前の値のみが使用されます。小数点以下はすべて破棄されます。
実数型	論理型	FALSE の値は 0 に変換されます。TRUE の値は 1 に変換されます。
実数型	整数型	値は変更されずに保持されます。

他のデータ タイプ間で変換するスクリプト関数の使用については、「[データ タイプの変換](#)」を参照してください。

式の例

以下の表は、いくつかの有効な式およびその式の結果と結果のデータ タイプを示しています。

式	結果のデータ タイプ	Result
37 MOD 4	整数	1
37 MOD 4 == 1	論理型	TRUE
NOT (37 MOD 4 == 1)	論理型	FALSE
InfoAppActive(InfoAppTitle("xyz")) == 1	論理型	プロセス「xyz」が実行している場合 TRUE
"Batch " + Text(IntTag, "000")	メッセージ	IntTag の値が 10 の場合「Batch 010」

以下の表は、いくつかの無効な式とそれらの式が無効である理由を示しています。

式	問題
NOT (37 MOD 4)	NOT では論理演算数が必要です。

式	問題
<code>NOT 37 MOD 4 == 1</code>	NOT は他の演算子より優先されるため、InTouch HMI は比較の論理結果ではなく、整数値 37 に NOT を適用しようとします。
<code>"Batch " + IntTag</code>	+ 演算子を使用して文字列を連結する場合、両方の演算数が文字列でなければなりません。

条件付きプログラム分岐構造の使用

特定の条件の一致に基づいて、スクリプトの実行パスを動的に制御できます。InTouch HMI は、IF-THEN-ELSE 制御構造をこの目的でサポートしています。

IF-THEN-ELSE 制御構造の基本構文を以下に示します。

構文

```
IF Condition THEN
    ... ステートメントおよび/または別の IF-THEN-ELSE 構造
[ELSE
    ... ステートメントおよび/または別の IF-THEN-ELSE 構造]
ENDIF;
```

IF-THEN-ELSE 構造を使用する場合、以下の規則に注意してください。

- IF-THEN-ELSE 構造は、THEN セクションおよび ELSE セクションの両方で入れ子にすることができます。
- すべての IF ステートメントには、終了を示す ENENDIF ステートメントが必要です。ENDIF ステートメントは、同じ入れ子レベルで直前に入れ子となっている IF ステートメントに適用されます。
- Condition は、有効な論理式でなければなりません。THEN セクションは、Condition が TRUE の場合に実行されます。ELSE セクションは、Condition が FALSE の場合に実行されます。
- ELSE セクションはオプションです。
- 他のいくつかのプログラミング言語では、IF-THEN-ELSE 構造の同じ階層レベルの複数条件を確認でき、すべての条件が FALSE に評価された場合に実行される 1 つの汎用 ELSE セクションを使用できます (Visual Basic の If-ElseIf-Else 構造がこの例です)。ただし、InTouch HMI では行えません。確認するすべての条件に対して、新しい IF-THEN-ELSE 構造を開く必要があります。したがって、コードの 1 つのセクションをすべての条件の ELSE コードとして使用するには、最後の入れ子レベルの IF-THEN-ELSE 構造の ELSE セクションにコードを置く必要があります。

簡単な条件付き構造

以下のスクリプトは、簡単な条件付き構造を示しています。SuccessTag が TRUE の場合、「Success」ウィンドウが開き、FALSE の場合、「Failure」ウィンドウが開きます。

```
IF SuccessTag == 1 THEN
    Show "Success";
ELSE
    Show "Failure";
ENDIF;
```

入れ子の条件付き構造

以下のスクリプトは、複数の条件を確認する方法、およびいずれの条件も満たされない場合に実行されるコードがある 1 つの汎用 ELSE セクションを使用する方法を示しています。

```
IF ChoiceTag == 1 THEN
    Show "Procedure 1";
ELSE
    IF ChoiceTag == 2 THEN
        Show "Procedure 2";
    ELSE
        IF ChoiceTag == 3 THEN
            Show "Procedure 3";
        ELSE
            Show "Default Procedure";
        ENDIF;
    ENDIF;
ENDIF;
```

無効なスクリプトの例（ENDIF が見つからない）

Visual Basic を使い慣れている場合、以下のように簡単な IF ステートメントを作成できます。

```
IF OpenThisWindow == 1 THEN Show "This Window";
```

これは、InTouch HMI では機能しません。すべての IF ステートメントには、終了を示す ENDIF ステートメントが必要です。

無効なスクリプトの例（間違った入れ子）

Visual Basic などの言語を使い慣れている場合、以下のように、複数の条件とデフォルトの条件を持つ条件付きステートメントを作成できます。

```
IF ChoiceTag == 1 THEN
    Show "Procedure 1";
ELSE IF ChoiceTag == 2 THEN
    Show "Procedure 2";
ELSE IF ChoiceTag == 3 THEN
    Show "Procedure 3";
ELSE
    Show "Default Procedure";
ENDIF;
```

これは、InTouch HMI では機能しません。それぞれの IF は新しい入れ子レベルを開いているため、対応する ENDIF ステートメントが必要です。この例の正しいバージョンについては、「[入れ子の条件付き構造](#)」を参照してください。

プログラム ループの使用

ループを使用すると、コードのセクションを繰り返し実行できます。InTouch HMI は、FOR ループのみをサポートしています。FOR ループは、ループの繰り返しごとに増加または減少する数値ループ変数の値を監視することによって機能します。ループは、ループ変数の値が固定制限に達するまで実行されます。

構文

```
FOR LoopTag = StartExpression TO EndExpression [STEP ChangeExpression]
... ステートメントまたは別の FOR ループ ...
NEXT;
```

- **StartExpression**、**EndExpression**、および **ChangeExpression** は、合わせて繰り返しの回数を定義します。
- **StartExpression** は、ループ範囲の開始値を設定します。**EndExpression** は、ループ範囲の終了値を設定します。
- **STEP ChangeExpression** はオプションで、各ループ繰り返し中にループ タグ変数を増加または減少させる値を設定します。この値を指定しない場合、デフォルト値 **1** が使用されます。

FOR ループを実行すると、InTouch HMI は、以下の処理を実行します。

1. **LoopTag** を **StartExpression** の値に設定します。
2. **LoopTag** が **EndExpression** より大きいかどうかをテストします。大きい場合、InTouch HMI は、ループを終了します (**ChangeExpression** が負の場合、InTouch HMI は、**LoopTag** が **EndExpression** より小さいかどうかをテストします)。
3. ループ内で、ステートメントを実行します。
4. **LoopTag** を **ChangeExpression** の値 (値が指定されていない場合は **1**) だけ増加します。
5. 手順 2 ~ 4 を繰り返します。

FOR ループを使用する場合、以下の規則に注意してください。

- FOR ループは入れ子にすることができます。入れ子レベルの最大数は、使用できるメモリおよびシステム リソースにより異なります。
- すべての FOR ステートメントには、終了を示す **NEXT** ステートメントが必要です。**NEXT** ステートメントは、同じ入れ子レベルで直前に入れ子となっている FOR ステートメントに適用されます。
- **LoopTag** は、数値タグ変数 (またはローカル変数) でなければなりません。
- **StartExpression**、**EndExpression**、および **ChangeExpression** は、数値の結果として評価される有効な式でなければなりません。
- **ChangeExpression** が正の場合、**EndExpression** は、**StartExpression** より大きくなければなりません。**ChangeExpression** が負の場合、**StartExpression** は、**EndExpression** より大きくなければなりません。そうでない場合、ループは開始されません。
- ループを終了するには、**EXIT FOR** ステートメントを使用します。詳細については、「[ループの強制終了](#)」を参照してください。
- ループには時間制限があります。「[ループ実行の時間制限](#)」を参照してください。

注意：ループ実行は、他のランタイムプロセスに影響を与えます。詳細については、「[他のランタイムプロセスへのループの効果](#)」を参照してください。

ループの強制終了

ループは、以下のステートメントを呼び出すことによって、いつでも終了できます。

EXIT FOR;

このステートメントを使用すると、ループ **NEXT** ステートメントの直後のステートメントでスクリプト実行が続行されます。

例

以下のコード部分では、ループを使用して多くのダミーレコードをデータベーステーブルに挿入します。レコード挿入時にエラーが発生した場合、さらにエラーが発生しないように、ループが中断されます。

```
FOR Counter = 1 TO 1000
    ResultCode = SQLInsert(ConnectionID, "BatchDetails", "BindList1");
    IF ResultCode <> 0 THEN
        LogMessage("Error creating records!Aborting...");
        EXIT FOR;
    ENDIF;
NEXT;
```

他のランタイムプロセスへのループの効果

FOR ループの実行中、WindowViewer の他のすべてのランタイムプロセスは一時停止されます。これには以下のプロセスがあります。

- 画像の更新（アニメーションリンク、値表示、トレンドなど）。つまり、ループが完了するまでオブジェクトの動きは発生しないので、FOR ループをオブジェクトアニメーションに使用することはできません。
- I/O 通信。たとえば、FOR ループの I/O タグ変数の値を変更する場合、最後の実行後の値だけが、I/O デバイスに書き込まれます。
- 非同期クイック関数など、他のスクリプト。

非同期クイック関数で FOR ループを使用することで、他のランタイムプロセスの一時停止を回避することができます。

ループ実行の時間制限

無限ループを回避するため、FOR ループが実行を完了する必要がある時間制限があります。この時間が経過してもループ実行が完了しない場合、WindowViewer は自動的にこのループを終了して、Log Viewer に終了に関するメッセージを書き込みます。

デフォルトの時間制限は 5 秒です。この値は、以下の行をアプリケーションディレクトリの intouch.ini ファイルに追加することで、カスタマイズできます。

```
LoopTimeout=x
```

x は秒単位での時間制限で置き換えます。

注意：時間制限は、ループの NEXT ステートメントでだけ確認されます。したがって、制限時間を超える場合でも、ループの最初の繰り返しは必ず実行されます。

ループの例

以下のスクリプトは、簡単なループおよび間接タグ変数を使用して、100 のタグ変数（Tag001 ～ Tag100）を値 0 で再初期化します。

```
DIM Counter AS INTEGER;
FOR Counter = 1 TO 100
    IndirectInteger.Name = "Tag" + Text(Counter, "000");
    IndirectInteger.Value = 0;
NEXT;
```

以下のスクリプトは、2 つの入れ子ループおよび間接タグ変数を使用して、1000 のタグ変数（Line01_Tag001 ～ Line10_Tag100）を値 0 で再初期化します。

```
DIM LineCounter AS INTEGER;
```

```
DIM TagCounter AS INTEGER;  
FOR LineCounter = 1 TO 10  
  FOR TagCounter = 1 TO 100  
    IndirectInteger.Name = "Line" + Text(LineCounter, "00") + "_Tag" + Text(TagCounter,  
    "000");  
    IndirectInteger.Value = 0;  
  NEXT;  
NEXT;
```

ローカル変数の使用

ローカル変数をスクリプトで宣言して、一時的な結果または中間結果を保存できます。これにより、パフォーマンスが向上し、タグ変数の数を少なく保つことができます。ローカル変数は、スクリプトのタグ変数のように使用できます。ただし、以下のようにいくつかの違いがあります。

- ローカル変数は、宣言されたスクリプトの範囲内でのみ存在します。スクリプト実行が終了すると、その値は失われます。アプリケーションの他のスクリプトから参照することはできません。
- ローカル変数にはドット フィールドがありません。
- ローカル変数は、タグ変数の数には含まれません。

ローカル変数をスクリプトで使用するには、最初に宣言する必要があります。宣言しないと、リファレンスがタグ変数とみなされます。「[ローカル変数の宣言](#)」を参照してください。

タグ変数と同じ名前を持つローカル変数を宣言できます。「[ローカル変数とタグ変数の名前の競合](#)」を参照してください。

ローカル変数の宣言

ローカル変数は、最初の使用の前に宣言している限り、スクリプトの任意の場所で宣言できます。ローカル変数を宣言するには、以下のステートメントを使用します。

```
DIM LocVarName [AS DataType];
```

LocVarName は、ローカル変数の名前です。この名前は、タグ変数の命名規則に従う必要があります。詳細については、「[タグ名規則](#)」を参照してください。

DataType は、ローカル変数のデータ タイプです。有効な値は、Discrete、Integer、Real、および Message です。このオプションを指定しない場合、Integer がデフォルトとして使用されます。

宣言する各ローカル変数に対して、個別の DIM ステートメントを使用する必要があります。

任意の数のローカル変数を宣言できます。宣言できる数は、使用できるメモリにより制限されます。

例

Integer 変数を宣言する場合:

```
DIM MyLocalIntVar AS Integer;
```

複数の Real 変数を宣言する場合:

```
DIM MyLocalRealVar1 AS Real;  
DIM MyLocalRealVar2 AS Real;
```

以下のステートメントは有効ではありません。

```
DIM MyLocalRealVar1, MyLocalRealVar2 AS Real;
```

ローカル変数とタグ変数の名前の競合

既存のタグ変数と名前が同じローカル変数を宣言できます。ただし、スクリプトでその名前を参照する場合、タグ変数よりローカル変数が優先されます。たとえば、「iTag」という名前の Integer タグ変数がすでに存在する場合に、以下のスクリプトを実行するとします。

```
DIM iTag as Integer;  
iTag = 20;
```

この場合、値はローカル変数にのみ割り当てられます。同じ名前のタグ変数の値は変わりません。

カスタム スクリプト関数

InTouch HMI クイック関数は、他の環境ではマクロ、サブルーチン、またはプロシージャと呼ばれているスクリプトです。

クイック関数について

クイック関数は、他のスクリプトやアニメーションリンクから呼び出すことのできるスクリプトです。クイック関数の主な利点は、コードの重複が減少することです。

値はクイック関数に渡すことができ、その後クイック関数はその値を使用して、結果を返すことができます。

クイック関数は、非同期で実行できます。他のスクリプトとは異なり、メインプログラムフローに影響を与えることなく、バックグラウンドで実行できます。非同期で実行するクイック関数は、SQL データベース呼び出しなど、時間を費やす操作に対して使用できます。

注意：クイック関数で引数を変更する場合、クイック関数を使用するすべてのスクリプトからそのクイック関数へのすべての呼び出しを削除する必要があるため、クイック関数とその引数は十分注意して計画する必要があります。変更が行われた後で、クイック関数呼び出しをスクリプトに追加する必要があります。「[クイック関数の設定](#)」の注記を参照してください。

クイック関数には 3 つの基本部分があります。

- 名前
- 引数（オプション）
- スクリプト本文（およびオプションの戻り値）

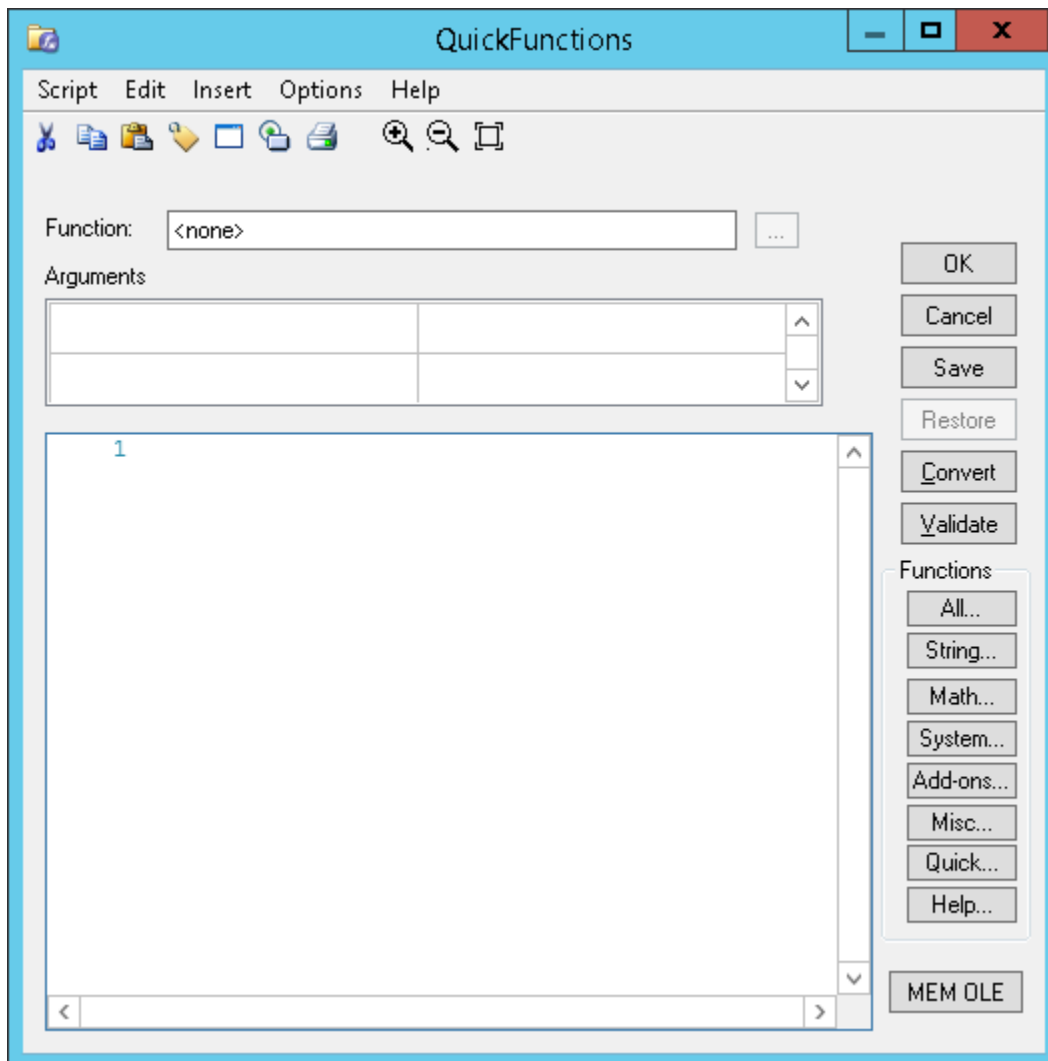
クイック関数は、アニメーションリンクまたは別のスクリプトで CALL 関数を使用することによって実行されます。「[クイック関数の呼び出し](#)」を参照してください。

クイック関数の設定

クイック関数は、作成、変更、または削除できます。

クイック関数を作成するには

1. [スクリプト] ペインで、[クイック関数] を右クリックして、[新規] をクリックします。
[クイック関数] ダイアログ ボックスが表示されます。



2. [関数名] ボックスで、クイック関数の名前を入力します。

3. [引数] 領域で、各引数に対して左側に名前を、右側にデータ タイプを入力します。

引数は、それらが定義されているクイック関数内にもみ存在するローカル変数です。各クイック関数に対して、最大 16 個までの引数を持つことができます。引数名は 31 文字まで持つことができますが、スペースは使用できません。引数名は、アルファベット文字で開始する必要があります。引数名は固有である必要があります。

4. スクリプトをウィンドウに入力します。

5. クイック関数で結果を返すようにするには、スクリプトに **RETURN value** を追加します。

Value には、リテラル値、ローカル変数、またはグローバル タグ名や計算式を使用できます。スクリプトは **RETURN** コマンドで終了し、呼び出し関数で続行します。

6. [OK] をクリックします。

クイック関数を変更するには

1. [スクリプト] ペインで、[クイック関数] を展開し、変更するクイック関数を右クリックして、[開く] をクリックします。[クイック関数の編集] ダイアログボックスが表示されます。

2. スクリプト本文を変更して、**[OK]** をクリックします。

注記: InTouch アプリケーションのクイック関数への呼び出しがある場合、引数は変更できません。この場合、最初にこれらの呼び出しを削除し、すべての InTouch ウィンドウを閉じて、使用タグの検索を更新する必要があります。

クイック関数を削除するには

1. クイック関数へのすべての呼び出しを削除し、すべての InTouch ウィンドウを閉じて、使用タグの検索を更新します。
2. **[スクリプト]** ペインで、**[クイック関数]** を展開し、削除するクイック関数を右クリックして、**[削除]** をクリックします。メッセージが表示されたら、**[はい]** をクリックします。

クイック関数の呼び出し

スクリプトおよびアニメーションリンクを設定して、クイック関数を呼び出し、可能な戻り値を処理または表示することができます。

クイック関数は、パラメータ値が変更されない場合、呼び出されません。**\$second** をパラメータとして使用すると、クイック関数が少なくとも毎秒実行されます。

詳細については、「[カスタム関数（クイック関数）の呼び出し](#)」を参照してください。

非同期クイック関数の作成

クイック関数をメインプログラムフローとは非同期に（つまり並行して）実行するように定義できます。

非同期クイック関数を作成するには：

1. スクリプトエディタで、クイック関数を作成します。
2. **[オプション]** メニューで、**[非同期]** をクリックします。

非同期クイック関数の制限事項

以下の内容は実行できません。

- 非同期クイック関数から値を返す
- 同じクイック関数の複数のインスタンスを同時に実行する
- 実行を開始した後で、非同期クイック関数を停止する

以下の操作を実行することは避けてください。

- 4 種類以上のクイック関数を同時に実行する。4 種類以上のクイック関数を同時に実行すると、システムパフォーマンスが著しく低下します。
- 非同期関数をアニメーションリンク（ツールヒントなど）の式の一部として使用してください。

非同期クイック関数が実行しているかどうかの確認

IsAnyAsyncFunctionBusy() 関数を使用すると、非同期クイック関数が実行しているかどうかを確認できます。この関数は、非同期クイック関数を呼び出す **QuickScript** に、その他の非同期クイック関数の処理がすべて完了するまで待機させる場合に使用できます。

IsAnyAsyncFunctionBusy() 関数

非同期クイック関数が実行しているかどうかを示す論理値を返します。

構文

```
result = IsAnyAsyncFunctionBusy(timeout)
```

引数

result

非同期クイック関数が実行しているかどうかを示す論理値を返します。値の意味は以下のとおりです。

- 0 = 非同期クイック関数は実行していません。
- 1 = 非同期クイック関数は実行しています。

timeout

非同期クイック関数が実行しているかどうかを確認するまでに待機する秒単位の時間です。リテラル整数値、整数型タグ名、または整数式です。

例

非同期クイック関数を使用して複数の SQL データベースに接続する場合、それらの接続を行うために 2 分必要であることを把握していると仮定します。

最初に、非同期クイック関数を実行して SQL データベースに接続します。

次に、QuickScript で IsAnyAsyncFunctionBusy(120) 関数を使用して、クイック関数が完了する前に SQL が接続を確立するために必要な時間を十分与えます。

2 分経過しても接続できず、非同期クイック関数がまだ接続を確立しようとしている場合は、IsAnyAsyncFunctionBusy() 関数によって値 1 (True) が返されます。

これで、オペレータに対して SQL の接続が正常に行われなかったことを知らせるエラー メッセージを表示できます。

以下のスクリプトは、上記の例を実装しています。

```
IF IsAnyAsyncFunctionBusy(120) == 1 THEN  
    SHOW "SQL Connection Error Dialog";  
ENDIF;
```

非同期クイック関数の実行の停止

開始している非同期クイック関数を停止することはできませんが、その後で実行される非同期クイック関数は、スクリプト ロジックを停止することで実行の開始を停止できます。これは、InTouch アプリケーションのすべての QuickScript に影響を与えます。

スクリプト実行の停止の詳細については、「[ランタイムでのスクリプト実行の一時停止](#)」を参照してください。

組み込み関数

InTouch QuickScript の関数を使用すると、指定した条件が満たされた場合に、コマンド操作やロジック操作を実行できます。QuickScript の関数は、それ自体で使用できます。また、特定の条件が満たされた場合に実行したり、アニメーション表示リンクで使用したりできます。事前定義関数は、機能に応じてグループに分類されています。グループを選択した後、スクリプトに挿入する事前定義関数を選択します。事前定義された数学関数のグループを選択した後に表示される、[関数の選択] ダイアログ ボック

スを使用します。関数を選択すると、スクリプト内でポインタが現在ある位置に事前定義関数が配置されます。

重要: この章には、32 ビットバージョンの Windows オペレーティングシステムでのみ機能するよう設計された従来の InTouch QuickScript 機能が含まれます。これらの機能は、64 ビットバージョンの Windows で実行するよう設計された InTouch QuickScript には含めないでください。この章では、これらの従来の 32 ビットバージョンの機能が識別されます。

アニメーション表示リンクでの強制更新

QuickScript をアニメーションリンクで使用する場合、このアニメーションリンクは、タグ変数が関連付けられている場合だけ更新されます。このタグ変数は、値が変更されたときにトリガとして機能します。アニメーションリンクの更新には、\$Second システムタグ変数または \$Minute システムタグ変数を使用することが好ましい選択です。

アニメーション表示リンクで強制更新を行うには

1. アニメーションリンクをオブジェクトプロパティウィンドウで開きます。
2. トリガタグ変数（たとえば、\$Second）を計算に追加します。次に例を示します。
 - アニメーションリンクが実数型または整数型の場合、\$Second/\$Second で式を乗算できます。
 - アニメーションリンクが文字列である場合、StringMid(\$TimeString, 0, 0) を式に追加できます。
 - アニメーションリンクが論理型の場合、(\$second.00 - \$second.00) を式に追加できます。

数学計算

InTouch HMI は、以下の処理を行う関数など、スクリプトやアニメーションリンクで利用できる基本的な数学関数をサポートしています。

- 数値の切り上げ／切り捨ておよび切り捨て
- サインおよびコサインの計算
- 対数および指数の計算
- 平方根の計算

切り上げ/切り捨て、切り捨て、符号の決定

スクリプトでは、以下の関数を使用して、数値の丸め処理、数値の切り捨て、および数値の符号の決定を行うことができます。

使用場所	目的
Abs()	値または式の絶対値を計算します。
Int()	値または式の整数値を計算します。
Round()	値または式を丸めます。
Sgn()	値または式の符号（マイナス、プラス、ゼロ）を決定します。
Trunc()	小数点の前の値または式を返します。

Abs() 関数

指定した数の絶対値を返します。負の数を正の数に変換するために使用できます。

構文

```
result = Abs (number)
```

パラメータ

number

リテラル数、アナログ型タグ変数、または数式です。

例

Abs(14) は 14 を返します。

Abs(-7.5) は 7.5 を返します。

Int() 関数

指定した数値以下の整数値を返します。

構文

```
result = Int (number)
```

パラメータ

number

リテラル数、アナログ型タグ変数、または数式です。

例

Int(4.7) は 4 を返します。

Int(-4.7) は -5 を返します。

注意：負の実数値の場合、この関数は指定した数値より小さい整数値を返します。たとえば、Int(-4.7) は -4 ではなく、-5 を返します。整数部分を返すには、Trunc() 関数を使用します。「[Trunc\(\) 関数](#)」を参照してください。

Round() 関数

数値を指定した精度に丸めます。結果は実数値です。

構文

```
result = Round (number, precision)
```

パラメータ

number

リテラル数、アナログ型タグ変数、または数式です。

precision

数値が切り上げ／切り捨てられる精度です。リテラル数、アナログ型タグ変数、または数式を使用できます。

例

Round(4.3, 1) は 4 を返します。

Round(4.3, 0.01) は 4.30 を返します。

Round(4.5, 1) は 5 を返します。

Round(-4.5, 1) は -4 を返します。

Round(106, 5) は 105 を返します。

Round(43.7, 0.5) は 43.5 を返します。

Sgn() 関数

数値の符号を返します。数値、タグ変数、または式が、負、正、またはゼロであるかを決定するときに使用できます。

構文

```
result = Sgn (number)
```

パラメータ

number

リテラル数、アナログ型タグ変数、または数式です。

例

Sgn(425) は 1 を返します。

Sgn(0) は 0 を返します。

Sgn(-37.3) は -1 を返します。

Trunc() 関数

数値が切り捨てられた値を返します。切り捨てられた値は、小数点の前の部分です。実数値の整数部分を操作するときに使用できます。

構文

```
result = Trunc (number)
```

パラメータ

number

リテラル数、アナログ型タグ変数、または数式です。

例

Trunc(4.3) は 4 を返します。

Trunc(-4.3) は -4 を返します。

注意：数値の小数点以下の部分を操作するときにも、この関数を使用できます。指定した数値の少数部分を返すには、Trunc() 関数を以下のように使用します。

```
result = number - trunc(number);
```

三角関数の使用

スクリプトでは、以下の関数を使用して、三角関数計算を実行できます。

使用	目的
Sin()	角度のサインを計算します。
ArcSin()	値または式のアークサインを計算します。

使用	目的
Cos()	角度のコサインを計算します。
ArcCos()	値または式のアークコサインを計算します。
Tan()	角度のタンジェントを計算します。
ArcTan()	値または式のアークタンジェントを計算します。

注意：InTouch HMI の三角 QuickScript 関数は、角度（0 ～ 360）を使用します。ラジアンを使用するには、パラメータを関数に渡す前に、または結果を関数から受け取った後に、対応する計算を実行する必要があります。

Sin() 関数

数値のサインを返します。三角関数では、数値は角度です。

構文

```
result = Sin (number)
```

パラメータ

number

リテラル数、アナログ型タグ変数、または数式です。

例

Sin(90) は 1 を返します。

Sin(0) は 0 を返します。

Sin(30) は 0.5 を返します。

100 * Sin (6 * \$second) は振幅 100、周期 1 分のサイン波を返します。

ArcSin() 関数

数値のアークサインを返します。Sin() 関数の逆関数です。ArcSin() 関数を使用すると、サインがその数値に等しい -90 ～ 90 の角度を計算できます。

構文

```
result = ArcSin (number)
```

パラメータ

number

-1 ～ 1 のリテラル数、アナログ型タグ変数、または数式です。

例

ArcSin(1) は 90 を返します。

ArcSin(0) は 0 を返します。

ArcSin(0.5) は 30 を返します。

Cos() 関数

数値のコサインを返します。三角関数では、数値は角度です。

構文

```
result = Cos (number)
```

パラメータ**数値型**

リテラル数、アナログ型タグ名、または数式です。

例

Cos(90) は 0 を返します。

Cos(0) は 1 を返します。

Cos(60) は 0.5 を返します。

20 + 50 * Cos(6 * \$second) は切片 20、振幅 50、周期 1 分のサイン波を生成します。

ArcCos() 関数

数値のアークコサインを返します。Cos() 関数の逆関数です。ArcCos() 関数を使用すると、コサインがその数値に等しい 0 ～ 180 の角度を計算できます。

構文

```
result = ArcCos (number)
```

パラメータ**number**

-1 ～ 1 のリテラル数、アナログ型タグ変数、または数式です。

例

ArcCos(1) は 0 を返します。

ArcCos(-0.5) は 120 を返します。

Tan() 関数

指定した数値のタンジェントを返します。三角関数では、数値は角度です。

構文

```
result = Tan (number)
```

パラメータ**number**

リテラル数、アナログ型タグ変数、または数式です。

例

Tan(45) は 1 を返します。

Tan(0) は 0 を返します。

ArcTan() 関数

数値のアークタンジェントを返します。Tan() 関数の逆関数です。ArcTan() 関数を使用すると、タンジェントがその数値に等しい角度を計算できます。

構文

```
result = ArcTan (number)
```

パラメータ

number

リテラル数、アナログ型タグ変数、または数式です。

例

ArcTan(1) は 45 を返します。

ArcTan(0) は 0 を返します。

π（パイ）の値を返す

スクリプトでは、Pi() 関数を使用して、定数 π を数学計算で使用できます。Pi() 関数は、小数点以下 7 桁と等しいものです。

構文

```
result = Pi ( )
```

例

Pi() は 3.1415927 を返します。

対数の計算

スクリプトでは、以下の関数を使用して、対数および指数関数で計算を実行できます。

使用場所	目的
Log()	値または式の自然対数を計算します。
Exp()	値または式の指数を計算します。
LogN()	別の値または式の底に対する値または式の対数を計算します。

Log() 関数

指定した正の数値の自然対数を返します。Exp() 関数の逆関数です。

注意：0 および負の数値の自然対数は使用できません。0 または負の数値を Log() 関数に渡すと、-99.0000000 が返されます。

構文

```
result = Log (number)
```

パラメータ

number

正のリテラル数、アナログ型タグ変数、または数式です。

例

Log(100) は 4.6051702 を返します。

Log(1) は 0 を返します。

Exp() 関数

指定した数値の指数を返します。これは、Log() 関数の逆関数で、べき乗した e と等価です。

注意：-88.72 ～ 88.72 の範囲外の数値を **Exp()** 関数に渡すと、-99.0000000 が返されます。

構文

```
result = Exp (number)
```

パラメータ

number

-88.72 ～ 88.72 のリテラル数、アナログ型タグ変数、または数式です。

例

Exp(1) は 2.7182818 を返します。

Exp(0) は 1 を返します。

LogN() 関数

指定した底に対する正の数値の自然対数を返します。これは、対数のべき乗の底の逆関数です。

例

構文

```
result = LogN (number, base)
```

パラメータ

number

正のリテラル数、アナログ型タグ変数、または数式です。

base

1 ではない正のリテラル数、アナログ型タグ変数、または数式です。

例

LogN(8,2) は 3 を返します。

LogN(num,btag) は底 btag に対する num の対数を返します。

注意：無効な数値を **LogN()** 関数に渡すと、-99.0000000 が返されます。

平方根の計算

スクリプトでは、**Sqrt()** 関数を使用して、負ではない数値の平方根を計算できます。

注意：負の数値を **Sqrt()** 関数に渡すと、-99.0000000 が返されます。

構文

```
result = Sqrt (number)
```

パラメータ

number

負ではないリテラル数、アナログ型タグ変数、または数式です。

例

Sqrt(36) は 6 を返します。

Sqrt(perftag) はタグ変数 perftga に保持されている値の平方根を返します。

文字列操作

スクリプトおよびアニメーションリンクでは、多くの基本的な文字列関数を使用できます。これらの関数を使用すると、以下の処理を実行できます。

- 文字列の一部を返す
- 文字列の大文字、小文字を変更する
- 文字列のスペースを削除したり、追加したりする
- 文字列で ASCII 値を処理する
- 文字列で検索置換を行う
- 文字列をそれぞれ比較する
- 長さなど文字列のその他の情報を返す

文字列の一部を返す

スクリプトで、`StringLeft()` 関数、`StringMid()` 関数、および `StringRight()` 関数を使用すると、文字列の一部を返すことができます。

StringLeft() 関数

文字列の先頭から指定した文字数を返します。

構文

```
result = StringLeft (string, Length)
```

パラメータ

string

リテラルテキスト、メッセージ型タグ変数、または文字列式です。

length

返す文字数です。リテラル数、アナログ型タグ変数、または数式です。

例

`StringLeft("Hello World",5)` は "Hello" を返します。

`StringLeft("Hello World",20)` は "Hello World" を返します。

`StringLeft("Hello World",0)` は "Hello World" を返します。

注意：長さ 0 を `StringLeft()` 関数に渡すと、文字列全体が返されます。

StringRight() 関数

文字列の末尾から指定した文字数を返します。

構文

```
result = StringRight (string, Length)
```

パラメータ

string

リテラルテキスト、メッセージ型タグ変数、または文字列式です。

length

返す文字数です。リテラル数、アナログ型タグ変数、または数式です。

例

`StringRight("Hello World",5)` は "World" を返します。

`StringRight("Hello World",20)` は "Hello World" を返します。

`StringRight("Hello World",0)` は "Hello World" を返します。

注意：長さ 0 を `StringRight()` 関数に渡すと、文字列全体が返されます。

StringMid() 関数

文字列の一部を返します。開始点および返す文字数を指定できます。

構文

```
result = StringMid (string, startpos, length)
```

パラメータ

string

リテラルテキスト、メッセージ型タグ変数、または文字列式です。

startpos

文字列の開始位置です。リテラル数、アナログ型タグ変数、または数式です。

length

返す文字数です。リテラル数、アナログ型タグ変数、または数式です。

例

`StringMid("Hello World",5,4)` は "o Wo" を返します。

`StringMid("Hello World",7,50)` は "World" を返します。

`StringMid("Hello World",4,0)` は "lo World" を返します。

注意：長さ 0 を `StringMid()` 関数に渡すと、開始位置からの文字列全体が返されます。

文字列の大文字小文字の変更

スクリプトで、`StringLower()` 関数および `StringUpper()` 関数を使用すると、指定した文字列を小文字および大文字で返します。指定した文字列に結果を割り当て、大文字と小文字を変換します。

StringLower() 関数

文字列を小文字にして返します。

構文

```
result = StringLower (string)
```

パラメータ

string

リテラルテキスト、メッセージ型タグ変数、または文字列式です。

例

`StringLower("TURBINE")` は "turbine" を返します。

`StringLower("The Value Is 22.2")` は "the value is 22.2" を返します。

`mtag = StringLower(mtag)` は mtag のメッセージ値を小文字に変換します。

StringUpper() 関数

文字列を大文字にして返します。

構文

```
result = StringUpper (string)
```

パラメータ

string

リテラルテキスト、メッセージ型タグ変数、または文字列式です。

例

StringUpper("abcd") は "ABCD" を返します。

StringUpper("The Value Is 22.2") は "THE VALUE IS 22.2" を返します。

mtag = StringUpper(mtag) は mtag のメッセージ値を大文字に変換します。

文字列からのスペースの削除

スクリプトで、StringTrim() 関数を使用すると、先頭および後に続くスペース（空白）を文字列から削除できます。この関数は、たとえば、ユーザー入力後に文字列から不要なスペースを削除するときに使用できます。

StringTrim() 関数

先頭および後に続くスペース（空白）を文字列から削除します。この関数は、たとえば、ユーザー入力後に文字列から不要なスペースを削除するときに使用できます。

構文

```
result = StringTrim (string, trimtype)
```

パラメータ

string

リテラルテキスト、メッセージ型タグ変数、または文字列式です。

trimtype

削除するスペースを決めるリテラル値、アナログ型タグ変数、または数式です。

- 1 = 先頭のスペース
- 2 = 後に続くスペース
- 3 = 先頭および後に続くスペース

備考

この関数は、先頭および後に続く空白を文字列からすべて削除します。空白はスペース（ASCII 0x20）および ASCII 0x09 ～ 0x0D の制御文字です。

例

アクションスクリプトで、メッセージ型タグ変数 mtag のすべてのスペースを削除するには、以下のスクリプトを使用します。

```
DIM i AS INTEGER;  
DIM tmp AS MESSAGE;  
mtag = StringTrim(mtag,3);      {mtag は調整されます}  
FOR i = 1 TO StringLen(mtag)    {mtag の文字を変数 i で実行します}
```

```

    IF StringMid(mtag, i, 1) " " THEN      {i-th 文字はスペースではありません}      tmp = tmp +
StringMid(mtag, i, 1);      {
文字を tmp へ追加します}
    ENDIF;
NEXT;
mtag = tmp;      {tmp を mtag へ返します}.

```

その他の例：

StringTrim(" Joe ",1) は "Joe ".を返します。

StringTrim(" Joe ",2) は "Joe" を返します。

このスクリプトは、mtag 値の左および右側のスペースをすべて削除します。

```
mtag = StringTrim(mtag,3)
```

スペースによる文字列の形式設定

スクリプトで、StringSpace() 関数を使用すると、スペース（空白）を文字列に追加できます。

構文

```
result = StringSpace (number)
```

パラメータ

number

リテラル数、数値タグ変数、または数式です。

例

StringSpace(4) は 4 つの空白で構成される文字列を返します。

"Pump"+StringSpace(1)+"Station" は "Pump Station" を返します。

文字と ASCII コードの変換

スクリプトで、StringChar() 関数および StringASCII() 関数を使用すると、文字列を ASCII コードに、ASCII コードを文字列に変換できます。

これらの関数は、複数バイトの文字セットをサポートしていません。0 ～ 255 の文字だけがサポートされています。

いくつかの数値計算を文字列で実行する場合（たとえば、文字列のコード化など）、ASCII コードを使用すると便利です。

StringChar() 関数

指定した ASCII コードに対応する 1 つの文字を返します。

構文

```
result = StringChar (ASCIICode)
```

パラメータ

ASCIICode

0 ～ 255 のリテラル数、数値タグ変数、または数式です。

備考

この関数は、制御文字を外部装置（ポインタやモデムなど）に渡す、または二重引用符を SQL クエリに渡すときに非常に便利です。

例

`StringChar(65)` は "A" を返します。

このスクリプトは、二重引用符で囲まれた "Hello World" を返します。

```
StringChar(34)+"Hello World"+StringChar(34)
```

以下の文字列は、改行およびラインフィードで分割された "Hello World" を返します。

```
"Hello"+StringChar(13)+StringChar(10)+"World"
```

StringASCII() 関数

文字列の最初の文字の ASCII コードを返します。

構文

```
result = StringASCII (string)
```

パラメータ**string**

リテラル文字列、メッセージ型タグ変数、または文字列式です。

例

`StringASCII("A")` は 65 を返します。

`StringASCII("hello world")` は 104 を返します。

文字列でのテキストの検索および置換

シングルバイト文字列セットを使用する言語（英語など）では、スクリプトで `StringInString()` および `StringReplace()` 関数を使用して、制限付き検索および置換機能をメッセージ型タグ変数で実行できます。

使用	目的
<code>StringInString()</code>	別の文字列で文字列を検索し、その位置を結果として返します。
<code>StringReplace()</code>	指定文字列の特定の文字または単語を他の文字または単語に置換して、新しい文字列を結果として返します。

StringInString() 関数

別の文字列での指定文字列の最初の位置を返します。

構文

```
result = StringInString (string, searchfor, startpos, casesens)
```

パラメータ**string**

検索される文字列です。リテラル文字列、メッセージ型タグ変数、または文字列式です。

searchfor

検索対象の文字列です。リテラル文字列、メッセージ型タグ変数、または文字列式です。

startpos

検索の文字列の開始位置です。リテラル値、数値タグ変数、または数式です。

casesens

検索で大文字と小文字を区別するかどうかを決定します。0 または 1、論理型タグ変数、またはブール式を指定できます。

0 - 大文字と小文字は区別されません（大文字と小文字は同じとみなされます）。

1 - 大文字と小文字は区別されます（大文字と小文字は別のものとみなされます）。

備考

この機能を使用すると、特定の文字列がメッセージ型タグ変数に含まれているか判別できます。検索の開始位置、および小文字を認識するかどうかを指定できます。

例

このスクリプトは 5 を返します。これは、"MTX" の最初の "M" が文字列の 5 番目にあるからです。

```
StringInString("DBO MTX-010","MTX",1,0)
```

このスクリプトは 3 を返します。これは、"MTX" の最初の "M" が文字列の 3 番目にあるからです。

```
StringInString("T-MTX 010 MTX","MTX",1,0)
```

このスクリプトは 11 を返します。これは、8 番目より後における "MTX" の最初の "M" が文字列の 11 番目にあるからです。

```
StringInString("T-MTX 010 MTX","MTX",8,0)
```

このスクリプトは 11 を返します。これは、大文字と小文字をした "MTX" と一致する最初の文字列が 11 番目にあるからです。

```
StringInString("t-mtx 030 MTX", "MTX",1,1)
```

このスクリプトは 0 を返します。これは、"Mty" が文字列に含まれていないからです。

```
StringInString("t-mtx 030 MTY-Mtx","Mty",1,1)
```

StringReplace() 関数

別の文字列で文字列を検索し、その文字列があった場合、別の文字列で置換します。以下の内容を指定できます。

- 大文字小文字の区別 - 大文字と小文字が同じ文字として扱われるかどうかを指定します。
- 置換するオカレンスの数 - 検索文字列のオカレンスが複数ある場合に役立ちます。
- 単語全体の一致 - 検索文字列が単語全体の場合に使用します。

注意：この関数は、ダブルバイトの文字セットをサポートしていません。

構文

```
result = StringReplace (string, searchfor, replacewith, casesens, numtoreplace, matchwholewords)
```

パラメータ

string

検索が実行される文字列です。リテラル文字列、メッセージ型タグ変数、または文字列式です。

searchfor

検索対象の文字列です。リテラル文字列、メッセージ型タグ変数、または文字列式です。

replacewith

置換後の文字列です。リテラル文字列、メッセージ型タグ変数、または文字列式です。

casesens

検索で大文字と小文字を区別するかどうかを決定します。0 または 1、論理型タグ変数、またはブール式を指定できます。

0 - 大文字と小文字は区別されません（大文字と小文字は同じとみなされます）。

1 - 大文字と小文字は区別されます（大文字と小文字は別のものとみなされます）。

numtoreplace

実行する置換数です。-1 を設定すると、一致した検索文字列のすべてのオカレンスが置換されます。リテラル整数値、整数型タグ変数、または整数式です。

matchwholewords

完全に一致する単語だけを対象とするかどうかを決めます。0 または 1、論理型タグ変数、またはブール式を指定できます。

0 - 文字列の任意の場所で、検索文字列が検索されます。

1 - 完全に一致する単語だけが対象となります。

例

このステートメントは、最初のオカレンスのみを置換して、“MTY 030 MTX” を返します。

```
StringReplace("MTX 030 MTX","MTX","MTY",0,1,0)
```

このステートメントは、すべてのオカレンスを置換して、“MTY 030 MTY” を返します。

```
StringReplace("MTX 030 MTX","MTX","MTY",0,-1,0)
```

このステートメントは、大文字と小文字が一致するすべてのオカレンスを置換して、“MTY 030 mty” を返します。

```
StringReplace("MTX 030 mty","MTX","MTY",1,-1,0)
```

このステートメントは、完全に一致するすべてのオカレンスを置換して、“MTY 030 QMTX” を返します。

```
StringReplace("MTX 030 QMTX","MTX","MTY",0,-1,1)
```

文字列の情報を返す

スクリプトで、StringLen() 関数および StringTest() 関数を使用すると、指定した文字列の長さを返し、文字が特定の文字グループに含まれるかテストできます。

StringLen() 関数

非表示文字を含む、指定された文字列の長さを返します。

構文

```
result = StringLen (string)
```

パラメータ

string

リテラル文字列、メッセージ型タグ変数、または文字列式です。

例

StringLen("Twelve percent") は 14 を返します。

StringLen("12%") は 3 を返します。

StringLen("The end."+ StringChar(13)) は 9 を返します。

StringTest() 関数

文字列の最初の文字が、特定の文字グループに含まれるかどうかをテストします。

構文

```
result = StringTest (string, group)
```

パラメータ

string

リテラル文字列、メッセージ型タグ変数、または文字列式です。

group

文字をテストするグループの番号です。1 ～ 11 のリテラル数、整数型タグ変数、または整数式です。

- 1 - 英数字 (A～Z、a～z、0～9)
- 2 - 数字 (0～9)
- 3 - 英字 (A～Z、a～z)
- 4 - 大文字の英字 (A～Z)
- 5 - 小文字の英字 (a～z)
- 6 - !、@、#、\$、%、^、&、* などの記号 (ASCII 0x21 ～ 0x2F)
- 7 - ASCII 文字 (ASCII 0x00 ～ 0x7F)
- 8 - 16 進数文字 (0～9、A～F、a～f)
- 9 - 印刷可能文字 (ASCII 0x20 ～ 0x7E)
- 10 - 制御文字 (ASCII 0x00 ～ 0x1F および 0x7F)
- 11 - 空白文字 (ASCII 0x09 ～ 0x0D および 0x20)

例

次の文字列は 1 を返します。これは、"A" が英数字だからです。

```
StringTest("ACB123",1)
```

次の文字列は 0 を返します。これは、"A" が小文字ではないからです。

```
StringTest("ABC123",5)
```

文字列の比較

スクリプトで、StringCompare() 関数、StringCompareNoCase() 関数、および StringCompareEncrypted() 関数を使用すると、2 つの文字列を比較できます。

使用	目的
StringCompare()	大文字と小文字を区別して比較します。
StringCompareNoCase()	大文字と小文字を区別しないで比較します。
StringCompareEncrypted()	暗号化された文字列を暗号化されていない文字列と比較します。

StringCompare() 関数

2 つの文字列をそれぞれ比較して、ブール値の結果を返します (0 = 文字列は同じです)。各文字の大文字と小文字は区別されるので、たとえば "A" と "a" は別の文字とみなされます。

構文

```
result = StringCompare (string1, string2)
```

パラメータ

string1

リテラル文字列、メッセージ型タグ変数、または文字列式です。

string2

リテラル文字列、メッセージ型タグ変数、または文字列式です。

例

StringCompare ("Apple","Apple") は 0 を返します。

StringCompare ("Apple","apple") は 1 を返します。

この文字列は、2つのメッセージ型タグ変数を比較して、論理値結果（0 または 1）を返します。

```
StringCompare (mtag1, mtag2)
```

StringCompareNoCase() 関数

2つの文字列をそれぞれ比較して、整数値結果を返します。各文字の大文字と小文字は区別されないの
で、たとえば、“A”と“a”は同じ文字とみなされます。

以下の整数値結果を返します。

- 両方の文字列が同じ（大文字と小文字は区別されません）場合は 0 です。
- 同じ文字列でない場合は 0 以外です。結果は、異なる文字の ASCII 値の差です（大文字と小文字は区別されません）。

注意：すべての 0 以外の値は InTouch スクリプトでは TRUE とみなされるので、StringCompareNoCase() 関数の結果は、論理値結果として使用できます。

構文

```
result = StringCompareNoCase (string1, string2)
```

パラメータ

string1

リテラル文字列、メッセージ型タグ変数、または文字列式です。

string2

リテラル文字列、メッセージ型タグ変数、または文字列式です。

例

以下の文字列は、0 を返します。これは、2つの文字列が同じとみなされるからです。

```
StringCompareNoCase("Apple","apple")
```

以下の文字列は、6 を返します。これは、2つの文字列が同じとみなされず、最初の異なる文字 "p" の ASCII 値から対応する文字 "v" の ASCII 値を引いた値が -6 だからです。

```
StringCompareNoCase("Apple","Avocado")
```

StringCompareEncrypted() 関数

暗号化された文字列を暗号化されていない文字列と比較し、ブール値結果を返します。この関数は、パスワードの確認に使用できます。パスワード暗号化の詳細については、「[オブジェクトのアニメーション化](#)」を参照してください。

構文

```
result = StringCompareEncrypted (plain, encrypted)
```

パラメータ

plain

リテラル文字列、メッセージ型タグ名、または文字列式です。

encrypted

暗号化されたメッセージ型タグ名です。

例

このスクリプトは、プレーンテキストと暗号化されたテキストが同じ場合 **1** を返し、そうでない場合 **0** を返します。パスワードは、暗号化されたユーザー入力の値を含むメッセージ型タグです。PlainTxt は、ユーザー入力と比較されるメッセージ型タグです。

```
StringCompareEncrypted(PlainTxt, Passwd)
```

データ タイプの変換

スクリプトで、変換 **QuickScript** を使用して、タグ名に含まれる値を他のデータ タイプに変換できます。これにより、文字列データを数学関数で操作、またはデバッグのために **Log Viewer** に値をログ記録できます。

- [Text\(\) 関数](#)
- [StringFromIntg\(\) 関数](#)
- [StringFromReal\(\) 関数](#)
- [StringToIntg\(\) 関数](#)
- [StringToReal\(\) 関数](#)
- [DText\(\) 関数](#)

Text() 関数

Text() 関数は、指定形式に従って、数値を文字列として返します。これは、特定の 방법으로値の形式を設定、またはその後の処理のために結果を他の文字列値と組み合わせるときに使用できます。

構文

```
result = Text (number, format)
```

パラメータ

number

リテラル数値、アナログ型タグ変数、または数式です。

format

"#"、"0"、"."、または","を使用します。

桁を表す場合は"#"、小数点区切りを表す場合は"."、先頭に0を配置する場合は"0"、カンマを挿入する場合は","を使用します。

書式設定でゼロを使用する場合、ゼロの後の文字はゼロを指定する必要があります。また、小数点の右側の文字も常にゼロにする必要があります。たとえば、000.00 は正しいですが、#0#0.0# は間違っています。

この関数は、必要に応じて値を切り上げ／切り捨てます。リテラル文字列、メッセージ型タグ変数、または文字列式です。

例

Text(66,"#.00") は "66.00" を返します。

Text (1234,"#") は "1234" を返します。

Text (123.4, "#,##0.0") は "123.4" を返します。

Text (12.3, "0,000.0") は "0,012.3" を返します。

Text(3.57,"#. #") は "3.6" を返します。

このスクリプトは、アナログ型タグ変数 "pressure" に値 1690.2743 が含まれている場合、文字列 "Reactor Pressure is 1690.3 mbar" を返します。

```
"Reactor Pressure is "+Text(pressure,"#. #")+ " mbar"
```

StringFromIntg() 関数

スクリプトで、StringFromIntg() 関数を使用すると、整数値を文字列値に変換できます。

この関数は、整数値の文字列値を返し、同時に進数変換を実行します。これは、たとえばテキストと整数値を同時に表示する場合、または整数値を 16 進数値に変換する場合に使用できます。

構文

```
result = StringFromIntg (number, base)
```

パラメータ

number

リテラル整数値、整数型タグ変数、または整数式です。

base

変換する進数です。これは、2 進 (2)、10 進 (10)、または 16 進 (16) など、値を別の進数に変換するときに使用されます。リテラル整数値、整数型タグ変数、または整数式です。

例

StringFromIntg(26,2) は "11010" (2 進数) を返します。

StringFromIntg(26,8) は "32" を返します。

(8 進数 : $26 = 3 \times 8 + 2$)

StringFromIntg(26,10) は "26" (10 進数) を返します。

StringFromIntg(26,16) は "1A" (16 進数) を返します。

StringFromReal() 関数

スクリプトで、StringFromReal() 関数を使用すると、実数値を文字列値に変換できます。

以下の内容を実行するように指定することもできます。

- 値を指定した精度に切り上げ／切り捨てる
- 値を指数表記で渡す

これは、たとえばテキストと実数値を同時に表示する場合、または実数値を指数表記で表示する場合に使用できます。

構文

```
result = StringFromReal (number, precision, type)
```

パラメータ

number

リテラル値、アナログ型タグ変数、または数式です。

precision

小数点以下の桁数を指定します。リテラル整数値、整数型タグ変数、または整数式です。

type

指数表記を使用するかどうかを指定します。リテラル文字列、メッセージ型タグ変数、または文字列式です。

"f" - 浮動小数点表記を使用します。

"e" - 小文字 "e" による指数表記を使用します。

"E" - 大文字 "E" による指数表記を使用します。

例

StringFromReal(263.355, 2, "f") は "263.36" を返します。

StringFromReal(263.355, 2, "e") は "2.63e2" を返します。

StringFromReal(263.55, 3, "E") は "2.636E2" を返します。

StringFromReal(0.5723, 2, "E") は "5.72E-1" を返します。

StringToIntg() 関数

スクリプトで、StringToIntg() 関数を使用すると、文字列に含まれる値を整数値に変換できます。

これは、文字列の先頭に含まれる値をその後の数学演算のために整数型タグ変数に渡すときに使用できます。

構文

```
result = StringToIntg (string)
```

パラメータ

string

リテラル文字列、メッセージ型タグ変数、または文字列式です。

備考

この関数は、文字列の最初の文字を確認します。最初の文字が数値の場合、この数値およびその後の文字を、数値以外の文字が検出されるまで整数値として読み取ります。この関数は、文字列の先頭のスペースを無視します。

例

StringToIntg("ABCD") は 0 を返します。

StringToIntg("13.4 mbar") は 13 を返します。

StringToIntg("Pressure is 13.4") は 0 を返します。

文字列 (mtag) から先頭でない最初の整数値を抽出し、その値を整数型タグ変数 itag に保存するには、以下のアクション スクリプトを使用します。

```
DIM i AS INTEGER;  
DIM tmp AS INTEGER;
```

```
FOR i = 1 TO StringLen(mtag) {mtag の文字に対して変数 i を実行}
  tmp = StringASCII(StringMid(mtag, i, 1)) - 48; {ASCII 値を検出}
  IF (tmp>=0 AND tmp<10) THEN {ASCII 値が "0" ~ "9" である場合}
    itag = StringToIntg(StringMid(mtag, i, 0)); {その位置の値を itag に設定してループを終了}
  EXIT FOR;
ENDIF;
NEXT;
```

StringToReal() 関数

スクリプトで、StringToReal() 関数を使用すると、文字列に含まれる値を実数値に変換できます。

これは、文字列の先頭に含まれる値をその後の数学演算のために実数型タグ変数に渡すときに使用できます。

注記: また、この関数は指数表記をサポートし、文字列式 **1e+6** を正確に **1000000** に変換します。

構文

```
result = StringToReal (string)
```

パラメータ

string

リテラル文字列、メッセージ型タグ変数、または文字列式です。

備考

この関数は、文字列の最初の文字を確認します。最初の文字が数値の場合、この数値およびその後の文字を、数値以外の文字が検出されるまで実数値として読み取ります。この関数は、文字列の先頭のスペースを無視します。

文字列（メッセージ型タグ変数 **mtag**）から先頭でない最初の実数値を抽出し、その値を実数型タグ変数 **rtag1** に保存するには、以下のスクリプトを使用します。

```
DIM i AS INTEGER;
DIM tmp AS INTEGER;
FOR i = 1 TO StringLen(mtag) {mtag の文字に対して変数 i を実行}
  tmp = StringASCII(StringMid(mtag, i, 1)) - 48; {ASCII 値を検出}
  IF (tmp>=0 AND tmp<10) THEN {ASCII 値が "0" ~ "9" である場合}
    rtag = StringToReal(StringMid(mtag, i, 0)); {その位置の値を rtag に設定してループを終了}
  EXIT FOR;
ENDIF;
NEXT;
```

例

StringToReal("ABCD") は 0 を返します。

StringToReal("13.4 mbar") は 13.4 を返します。

StringToReal("Pressure is 13.4") は 0 を返します。

DText() 関数

スクリプトで、DText() 関数を使用すると、ブール値を文字列値に変換できます。この関数を使用すると、カスタマイズしたメッセージ表示アニメーションリンクを使用できます。

この関数は、ブール値の値によって、異なる文字列値を返します。

構文

```
result = Dtext (Boolean, stringtrue, stringfalse)
```

パラメータ

Boolean

リテラルブール値、論理型タグ変数、またはブール式です。

stringtrue

Boolean が true の場合に返される文字列です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

stringfalse

Boolean が false の場合に返される文字列です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

例

このスクリプトは、論理型タグ変数 **switch** が TRUE の場合 "Running" を返し、それ以外の場合 "Stopped" を返します。

```
DText(switch,"Running","Stopped")
```

このスクリプトは、論理型タグ変数 **switch1** の値に応じて、別の論理型タグ変数 **switch2** のオン／オフメッセージを返します。

```
DText(switch1,switch2.OnMsg,switch2.OffMsg)
```

ランタイムでの InTouch ウィンドウの操作

スクリプトで、InTouch ウィンドウの動作および外観を制御できます。また、QuickScript を使用して、個々の InTouch ウィンドウまたは画面全体を印刷するスクリプトを作成することもできます。

ウィンドウ名のプロパティの公開

GetWindowName スクリプト関数を使用して、ランタイム環境でのウィンドウの読み込みに必要なスクリプト作成を減らすことができます。関数を呼び出したウィンドウの名前を取得することができます。

GetWindowName() 関数

関数が呼び出されたウィンドウの名前を取得します。

構文

このスクリプト関数の構文は、以下のとおりです。

```
resultcode = GetWindowName(tagname);
```

Resultcode は、スクリプト関数の成功または失敗を示します。**Resultcode** は論理型/整数型/実数型のデータタイプになります。**Resultcode** はスクリプト関数の成功または失敗によって、1 または 0 のいずれかになります。

- スクリプト関数がウィンドウ コンテキストから呼び出された場合、**Resultcode** は 1 になります。
- スクリプト関数がウィンドウ コンテキスト以外から呼び出された場合、**Resultcode** は 0 になります。

注: スクリプト関数の戻り値の設定はオプションです。これは InTouch の既存のスクリプト関数と似ています。

パラメータ

TagName

tagname はこの関数のパラメータです。これはウィンドウ名を取得するためのメッセージ型タグになります。

この関数のパラメータは、以下のどれでも構いません。

- InTouch タグ
- ドット フィールド
- スクリプトのローカル変数
- リモート タグ リファレンス
- Galaxy 参照

スクリプトブラウザによって読み込まれるデフォルトのテキストは、
GetWindowName(TagName) です。

注: "TagName" はメッセージ型タグまたはメッセージタイプであるリモート タグ リファレンス (RTR) です。

戻り値

GetWindowName スクリプト関数は、次のシナリオの場合、戻り値 1 とウィンドウの名前を返します。

- ウィンドウ スクリプト (すべての条件タイプ)
- 押しボタンの動作スクリプト (すべての条件タイプ)

GetWindowName スクリプト関数は、次のシナリオの場合、戻り値 0 と空の文字列を返します。

- アプリケーション スクリプト
- キー スクリプト
- 条件スクリプト
- データ変化スクリプト
- ActiveX イベント スクリプト
- クイック スクリプト関数 (同期および非同期)

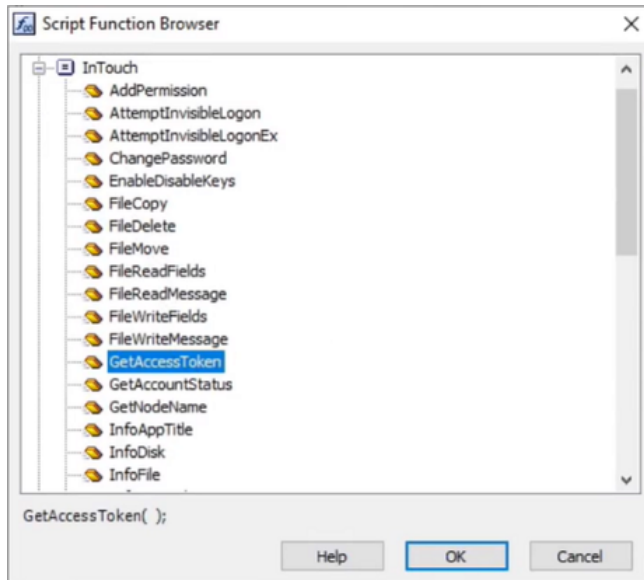
最新のアクセス トークンの公開

GetAccessToken スクリプトと **GetSecureAccessToken** スクリプトを使用することによって、AVEVA Connect アクセス トークンを InTouch に公開し、ランタイムの接続エクスペリエンスでのコントロールとウィジェットのシングル サインオンに使用できます。

GetAccessToken スクリプト関数を使用すると最新のトークン値を返すことができます。

GetSecureAccessToken スクリプトを使用すると最新のトークン値を安全な方法または暗号化された方法で返すことができます。アクセス トークンは時間にバインドされ、値は定期的に変更されるので、このスクリプト関数を使うと最新のアクセス トークン値を取得できます。

スクリプト関数ブラウザで、組み込みスクリプト **GetAccessToken** および **GetSecureAccessToken** を表示できます。これはアクセス トークンの値をロガーに記録します。



GetAccessToken() 関数

GetAccessToken () は、認証トークンを提供します。この認証トークンは、シングルサインオン機能をサポートするために、トレンドコントロール、アラーム クライアント コントロール、およびその他の InTouch コントロールのようなコントロールによって受け入れられます。

構文

このスクリプト関数の構文は、以下のとおりです。

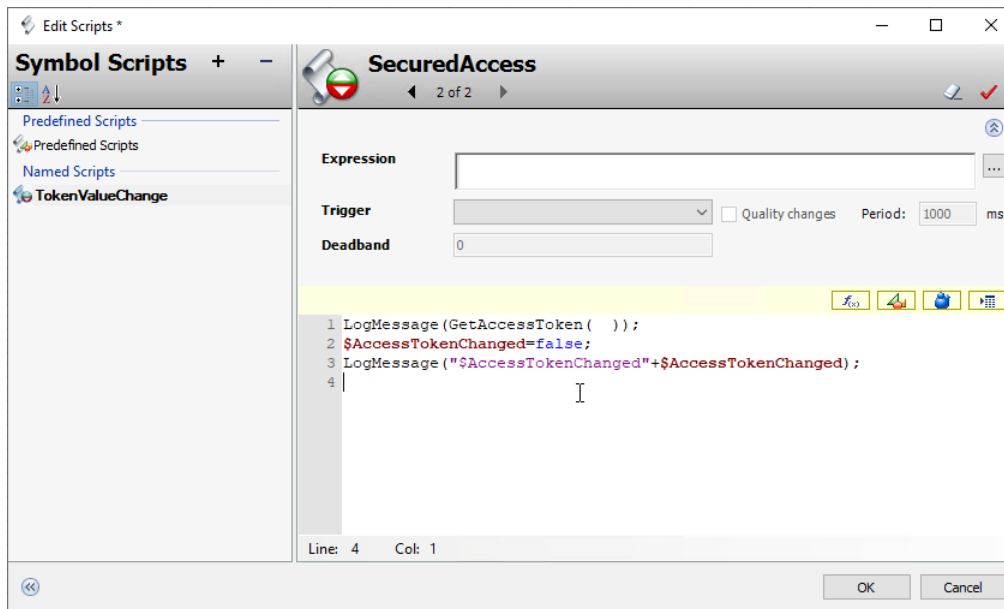
```
resultCode = GetAccessToken();
```

resultCode は、最新のアクセス トークンを示します。

例:

有効期限が切れるたびにトークン値を更新するには、スクリプトを次のように使用できます。

1. [シンボル スクリプト] ウィンドウで [スクリプトを追加] をクリックします。
2. スクリプトに名前を付けます。
3. [スクリプト関数ブラウザを表示] アイコンをクリックします。
4. [スクリプトブラウザ] 画面の [InTouch] で [GetAccessToken] を選択してスクリプト関数を挿入します。手動で入力することもできます。
5. [OK] をクリックします。



戻り値

ユーザーが AVEVA Connect を使用して認証されている場合、GetAccessToken スクリプト関数は、トークン値を含むアクセス トークンを返します。

ユーザーが AVEVA Connect を使用して認証されていない場合、GetAccessToken スクリプト関数は、空の文字列を返します。

GetSecureAccessToken() 関数

GetSecureAccessToken () は、セキュアな認証トークンを提供します。この認証トークンは、シングルサインオン機能をサポートするために、トレンドコントロール、アラーム クライアント コントロール、およびその他の InTouch コントロールのようなコントロールによって受け入れられます。

構文

このスクリプト関数の構文は、以下のとおりです。

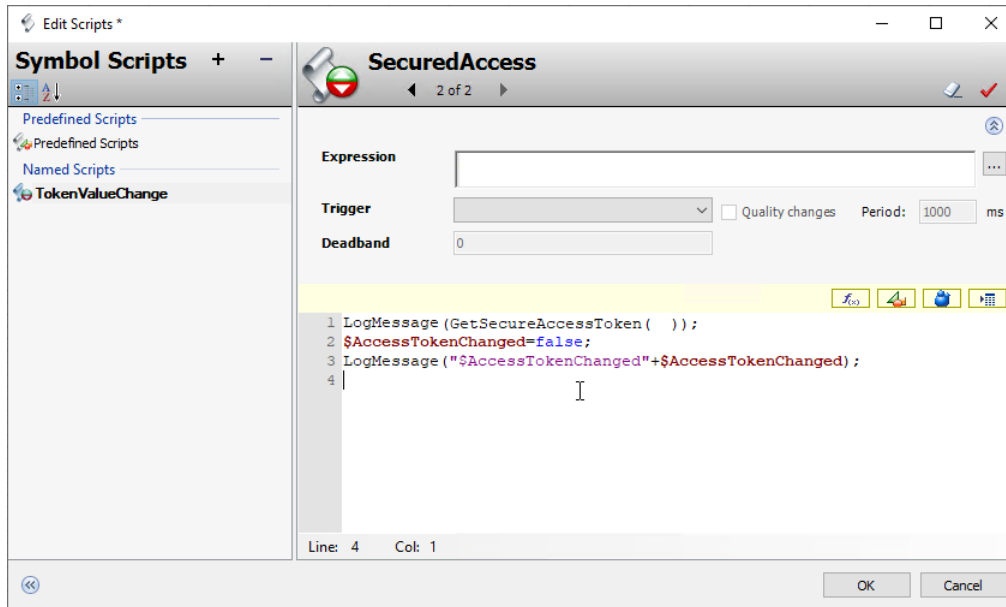
```
resultCode = GetSecureAccessToken();
```

resultCode は、最新のアクセス トークンを示します。

例:

有効期限が切れるたびにセキュアなトークン値を更新するには、スクリプトを次のように使用できます。

1. [シンボル スクリプト] ウィンドウで [スクリプトを追加] をクリックします。
2. スクリプトに名前を付けます。
3. [スクリプト関数ブラウザを表示] アイコンをクリックします。
4. [スクリプトブラウザ] 画面の [InTouch] で [GetSecureAccessToken] を選択してスクリプト関数を挿入します。手動で入力することもできます。
5. [OK] をクリックします。



戻り値

ユーザーが AVEVA Connect を使用して認証されている場合、GetSecureAccessToken スクリプト関数は、トークン値を含むアクセス トークンを返します。

ユーザーが AVEVA Connect を使用して認証されていない場合、GetSecureAccessToken スクリプト関数は、空の文字列を返します。

接続ステータスの取得

GetTokenConnectionStatus スクリプト関数を使用して、AVEVA Operations Control 接続エクスペリエンスの接続ステータスを取得できます。

GetTokenConnectionStatus() function

Retrieves status of the connection to the AVEVA Identity Manager and CONNECT in AVEVA Operations Control connected experience.

Syntax

The syntax of the script function is as follows:

```
resultcode = GetTokenConnectionStatus();
```

Resultcode indicates the token for connection status.

Return value

- Resultcode is 0 when the connection status is Fully Connected. That is the application is connected to both AVEVA Identity Manager and CONNECT.
- Resultcode is 1 when the connection status is Partially Connected. That is the application is connected to AVEVA Identity Manager and disconnected from CONNECT.
- Resultcode is 2 when the connection status is Fully Disconnected. That is the application is disconnected from both AVEVA Identity Manager and CONNECT.

Example:

```
int AccessTokenStatus=GetTokenConnectionStatus()
```

開いているウィンドウのリストの表示

スクリプトで、OpenWindowsList() 関数を使用すると、現在開いている InTouch ウィンドウのリストを示すダイアログ ボックスを表示できます。

OpenWindowList() 関数

現在開いている InTouch ウィンドウのリストを示すダイアログ ボックスを表示します。

この関数はアニメーション リンクでは使用できません。

構文

```
[result = ]OpenWindowsList();
```

例

このスクリプトは、[開かれているウィンドウのリスト] ダイアログ ボックスを開き、現在開いているすべての InTouch ウィンドウを示します。

```
OpenWindowsList()
```

注意：[メモリ内ウィンドウのキャッシュを使用する] WindowViewer オプションが有効な場合、閉じたウィンドウが OpenWindowList() 関数によって作成されたリストに表示される場合があります。

ウィンドウが開いているか、閉じているか、存在するかを確認する

スクリプトで、WindowState() 関数を使用すると、InTouch ウィンドウが開いているか、閉じているか、または存在しないかを確認できます。

WindowState() 関数

InTouch ウィンドウが開いているか、閉じているか、または存在しないかを確認します。

構文

```
result = WindowState (windowname)
```

パラメータ

windowname

ウィンドウの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

戻り値

整数値であり、値の意味は以下のとおりです。

0 - InTouch ウィンドウが存在し、現在閉じています。

1 - InTouch ウィンドウが存在し、現在開いています。

2 - InTouch ウィンドウは存在しません。

例

このスクリプトは、InTouch ウィンドウ Main が存在するが、開いていない場合、0 を返します。

```
WindowState("Main")
```

InTouch ウィンドウを開く

スクリプトで、以下の QuickScript 関数のいずれかを使用して、InTouch ウィンドウを開くことができます。

使用場所	目的
Show	位置設定で定義された場所に InTouch ウィンドウを開きます。
ShowAt()	指定した場所に InTouch ウィンドウを開きます。開かれたウィンドウは、その場所の中央に配置されます。この関数は、開いているウィンドウを移動するときにも使用できます。
ShowHome	[WindowViewer のプロパティ] ダイアログ ボックスの [起動時ウィンドウ] タブで指定した InTouch ウィンドウを開いて、他のウィンドウを閉じます。
ShowTopLeftAt()	指定した場所に InTouch ウィンドウを開きます。開かれたウィンドウは、その場所の左上隅に合わせて配置されます。この関数は、開いているウィンドウを移動するときにも使用できます。

Show() 関数

デフォルトの場所に InTouch ウィンドウを開きます。

構文

```
Show windowname
```

パラメータ

windowname

開かれるウィンドウの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

例

このスクリプトは、ウィンドウ **Main** を開きます。

```
Show "Main";
```

このスクリプトは、**wname** メッセージ型タグ変数に保存されている名前でウィンドウを表示します。

```
Show wname;
```

ShowAt() 関数

指定した場所に InTouch ウィンドウを開きます。また、すでに開いている InTouch ウィンドウを指定した場所に移動することもできます。場所は、ウィンドウの中央です。

注意：ウィンドウのいずれかの端が画面上にない場合、中央には配置されません。

構文

```
ShowAt (windowname, xpos, ypos)
```

パラメータ

windowname

開かれる、または移動されるウィンドウの名前です。

xpos

ウィンドウの中央が移動されるピクセル単位での水平位置です。リテラル値、アナログ型タグ変数、または数式です。

ypos

ウィンドウの中央が移動されるピクセル単位での垂直位置です。リテラル値、アナログ型タグ変数、または数式です。

例

このスクリプトは、x:450、y:130 の位置を中央として配置されるようにウィンドウ **Main** を開きます。
`ShowAt("Main",450,130);`

このスクリプトは、**UserDialog** というウィンドウを開いて、この関数を呼び出したオブジェクト（ボタンなど）の中央位置に合わせてウィンドウを配置します。
`ShowAt("UserDialog",$ObjHor,$ObjVer);`

ShowHome() 関数

〔**WindowViewer** のプロパティ〕ダイアログ ボックスの〔**起動時ウィンドウ**〕タブで指定した InTouch ウィンドウを開いて、他のウィンドウを閉じます。

構文

```
ShowHome;
```

ShowTopLeftAt() 関数

指定した場所に InTouch ウィンドウを開きます。開いているウィンドウを移動するときにも使用できます。

構文

```
ShowTopLeftAt (windowname, xpos, ypos)
```

パラメータ

開かれる、または移動されるウィンドウの名前です。

xpos

ウィンドウの左端が移動されるピクセル単位での水平位置です。リテラル値、アナログ型タグ変数、または数式です。

ypos

ウィンドウの上端が移動されるピクセル単位での垂直位置です。リテラル値、アナログ型タグ変数、または数式です。

例

このスクリプトは、x:450、y:130 の位置にウィンドウ左上隅が配置されるようにウィンドウ **Main** を開きます。
`ShowTopLeftAt("Main",450,130);`

ウィンドウの移動とサイズ変更

スクリプトで、**WWMoveWindow()** 関数を使用すると、開いている InTouch ウィンドウを移動したりサイズを変更したりできます。新しい位置および新しいサイズは、指定ウィンドウが開いている間、一時的に適用されます。

WWMoveWindow() 関数

開いている InTouch ウィンドウを指定位置に移動して、指定サイズに変更します。新しい位置および新しいサイズは、指定ウィンドウが開いている間、一時的に適用されます。

構文

```
WWMoveWindow (windowname, xpos, ypos, xsize, ysize)
```

パラメータ

windowname

開かれる、または移動されるウィンドウの名前です。

xpos

ウィンドウの左端が移動されるピクセル単位での水平位置です。リテラル値、アナログ型タグ変数、または数式です。

ypos

ウィンドウの上端が移動されるピクセル単位での垂直位置です。リテラル値、アナログ型タグ変数、または数式です。

xsize

指定したウィンドウのピクセル単位での水平サイズです。リテラル値、アナログ型タグ変数、または数式です。

ysize

指定したウィンドウのピクセル単位での垂直サイズです。リテラル値、アナログ型タグ変数、または数式です。

InTouch ウィンドウを非表示にする

スクリプトで、以下の関数のいずれかを使用して、InTouch ウィンドウを非表示にすることができます。

使用	目的
Hide	指定したウィンドウを非表示にします。
HideSelf	現在アクティブなウィンドウを非表示にします。

Hide() 関数

InTouch ウィンドウを非表示にします（閉じます）。

構文

```
Hide windowname;
```

パラメータ

windowname

非表示にされるウィンドウの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

例

このスクリプトは、ウィンドウ UserConfirmation を非表示にします。

```
Hide "UserConfirmation";
```

HideSelf() 関数

現在アクティブな InTouch ウィンドウを非表示にします（閉じます）。

注意：この関数は、アクション QuickScript でのみ使用できます。

構文

```
HideSelf;
```

例

```
HideSelf;
```

ウィンドウの色の変更

スクリプトで、`ChangeWindowColor()` 関数を使用すると、開いている InTouch ウィンドウの色を変更できます。

ChangeWindowColor() 関数

開いている InTouch ウィンドウの色を変更して、結果コードを返します。

構文

```
Result = ChangeWindowColor (windowname, rValue, gValue, bValue)
```

パラメータ

windowname

色を変更するウィンドウの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

rValue

赤色の輝度です。0 ～ 255 のリテラル整数値、整数型タグ変数、または整数式です。

gValue

緑色の輝度です。0 ～ 255 のリテラル整数値、整数型タグ変数、または整数式です。

bValue

青色の輝度です。0 ～ 255 のリテラル整数値、整数型タグ変数、または整数式です。

戻り値

値の意味は以下のとおりです。

0 - 失敗。ウィンドウが定義されていないか、RGB 値が範囲外です。

1 - 成功

2 - 失敗。ウィンドウは存在しますが、開いていません。

ランタイムでのウィンドウの印刷

スクリプトで、`PrintWindow()` 関数または `PrintScreen()` 関数を使用すると、個々の InTouch ウィンドウまたは `WindowViewer` 画面全体を印刷できます。また、`SetWindowPrinter()` 関数と組み合わせて使用するプリンタを設定できます。

SetWindowPrinter() 関数

ランタイムで、`SetWindowPrinter()` 関数と組み合わせて使用するプリンタを設定できます。

注意：この関数で設定されたプリンタは、`PrintHT()` で使用されるプリンタでもあります。

構文

```
SetWindowPrinter (printername)
```

パラメータ

printername

プリンタの名前です。これは、プロパティ ウィンドウに表示されるネットワーク共有またはプリンタ名のいずれかです。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

例

この例では、PRTSRV1 はノード名であり、PRT22SW1 はプリンタに指定された共有名です。

```
SetWindowPrinter("\\PRTSRV1\PRT22SW1");
```

この例では、Epson LX-300 はプリンタの [プロパティ] ウィンドウに表示されるプリンタの名前です。

```
SetWindowPrinter("Epson LX-300");
```

この例では、MyPrinter はインストールされている Windows プリンタの名前または共有ネットワーク プリンタのパスを含むメッセージ型タグ変数です。

```
SetWindowPrinter(MyPrinter);
```

印刷に関する推奨事項

以下のリストは、印刷時に考慮する必要のあるいくつかの問題を示しています。これらの問題は、1 つのウィンドウまたは WindowViewer 画面を印刷するときに該当します。

- 印刷するウィンドウを印刷前に開きます。このようにしないと、Windows コントロールおよび ActiveX コントロールでの印刷が正しく機能しないことがあります。
- 現在アラームを印刷しているプリンタと同じプリンタには印刷できません。
- 印刷する場合、ウィンドウおよびウィンドウのオブジェクトが重ならないようにします。
- 可能な限り、True Type フォントを使用します。デフォルトの InTouch フォント (システム) は True Type フォントではありません。
- 印刷処理を速くするには、背景色を白くする、オブジェクト数を減らす、グラフィックの代わりにテキストを使用するなどの処理を考慮してください。
- WindowViewer は、ウィンドウがプリンタ キューに送信される前に、一定時間待機します。この待機中、WindowViewer は、そのウィンドウの任意の I/O 値をバックグラウンドで更新します。この待機時間を変更するには、intouch.ini ファイルを開いて、以下の行を変更または追加します (ミリ秒単位)。PrintWindowWait=10000

PrintWindow() 関数

スクリプトで、PrintWindow() 関数を使用すると、InTouch ウィンドウを印刷できます。

注記: PrintWindow() 関数を含むスクリプトでは、InTouch ウィンドウ内の ListBox、DateTimePicker、CalendarControl、EditBox、CheckBox、RadioButtonGroup、ComboBox、AlarmClient、TrendClient の産業用シンボルのコントロールは印刷できません。

構文

```
[result = ] PrintWindow (windowname, leftmargin, topmargin, width, height, options);
```

パラメータ

windowname

印刷されるウィンドウの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

leftmargin

左マージン オフセット (インチ単位)。リテラル数値、アナログ型タグ変数、または数式です。

topmargin

上マージン オフセット (インチ単位)。リテラル数値、アナログ型タグ変数、または数式です。

width

印刷幅 (インチ単位)。縦横比を最大にする場合、この値を 0 に設定します。リテラル数値、アナログ型タグ変数、または数式です。

height

印刷高さ（インチ単位）。縦横比を最大にする場合、この値を **0** に設定します。リテラル数値、アナログ型タグ変数、または数式です。

options

論理値 **0** または **1** です。これは、**width** および **height** が **0** の場合だけ使用されます。リテラルブール値、論理型タグ変数、またはブール式です。値は以下のように設定されます。

1 - ウィンドウ サイズの整数の倍数である最大縦横比でウィンドウが印刷されます。

0 - ページに合わせた最大縦横比でウィンドウが印刷されます。

注記: ウィンドウにビットマップが含まれる場合、**options** を **1** に設定して、ビットマップが拡大しないようにします。

戻り値

0 - 印刷ジョブが正常にキューに送信されなかったか、ウィンドウが存在しません。

1 - 印刷ジョブが正常にキューに送信されました。

PrintScreen() 関数

PrintScreen() 関数を使用すると、WindowViewer 画面全体を印刷するスクリプトを記述できます。

構文

PrintScreen (ScreenOption, PrintOption)

パラメータ**ScreenOption**

印刷する WindowViewer 画面の領域を決定します。リテラル整数値、整数型タグ変数、または整数式です。

1 - クライアント領域を印刷します。メニューは印刷しません（デフォルト）。

2 - メニューを含むウィンドウの全体の領域を印刷します。

PrintOption

印刷される画像が印刷に合わせてどのように拡大されるかを決めます。

- 1 - 最適：
画像は、縦横比を変更せずに水平または垂直のいずれかに合わせて拡大されます（デフォルト）。
- 2 - 垂直：
画像は、縦横比を変更せずに垂直に合わせて拡大されます。画像は水平に切り取られることがあります。
- 3 - 水平：
画像は、縦横比を変更せずに水平に合わせて拡大されます。画像は垂直に切り取られることがあります。
- 4 - ページに合わせる：
画像は、水平および垂直に合わせて拡大されます。縦横比が変わることがありますが、画像は切り取られません。
- 0 など無効なオプションを指定すると、デフォルトで最適が適用されます。

注意： WindowViewer 画面領域上に拡張するポップアップウィンドウは切り取られます。

例

このスクリプトは、メニューを除く、現在の WindowViewer 画面の全領域の出力をプリンタ キューに送信します。出力には、出力寸法に合わせて拡大された画面領域が含まれます。

```
PrintScreen(1,4);
```

PrintHT() 関数

スクリプトで、ボタンを作成して、PrintHT QuickScript 関数を実行するアクション QuickScript にリンクすると、このボタンを使って履歴トレンドグラフを印刷できます。

トレンドグラフだけではなく、ウィンドウ全体を印刷する場合は、PrintHT() 関数の代わりに PrintWindow() 関数を使用します。

注意：[印刷] オプションまたは PrintHT() 関数を使用して履歴トレンドを印刷する場合、X 値および Y 値は印刷されません。X および Y 値を印刷するには、PrintWindow() または PrintScreen() を使用します。

構文

```
PrintHT(HistTrendTagname);
```

パラメータ

HistTrendTagname

印刷する履歴トレンドの履歴トレンドタグ変数名です。

Tag Viewer の起動

Tag Viewer はランタイム時にタグを監視し、タグの値を変更できるランタイム アプリケーションです。Tag Viewer およびその使用方法の詳細については、『AVEVA™ InTouch HMI アプリケーション ランタイム ガイド』を参照してください。

LaunchTagViewer() 関数

Tag Viewer は WindowViewer が実行中で、デザイン時に有効に設定されているときにのみ起動できます。

Tag Viewer の有効化に関する詳細は、『AVEVA™ InTouch HMI コンポーネントの標準の作成』ユーザー ガイドの「[一般的な WindowViewer プロパティの設定](#)」参照してください。

構文

```
LaunchTagViewer()
```

備考

LaunchTagViewer() 関数は OnStartup と OnShutdown 以外の、どのスクリプト タイプからでも実行することができます。

Tag Viewer が WindowMaker で有効化されていない場合、関数を呼び出しても Tag Viewer は起動せず、警告メッセージがロガーに表示されます。

Tag Viewer を起動するには、適切なセキュリティ権限が必要です。

日付と時刻情報の操作

スクリプトでシステム タグ変数および QuickScript 関数を使用して、計算でシステムの時間と日付の設定を使用できます。InTouch スクリプトは、複数のタイムゾーンと夏時間を使用する計算もサポートします。

数値の日付と時刻情報の取得

スクリプトで、さまざまな数値システム タグ変数、および 1 つのスクリプト関数を使用すると、システム時刻および日付に関する情報を取得できます。これらのタグ変数およびスクリプト関数は、他の数学演算で使用できます。以下のシステム タグ変数およびスクリプト関数を使用できます。

使用場所	目的
\$Year	現在の年度を返します。
\$Month	現在の月を返します。
\$Day	現在の日付を返します。
\$Hour	現在の時間を返します。
\$Minute	現在の分を返します。
\$Second	現在の秒を返します。
\$Msec	現在のミリ秒を返します。
\$Time	ローカル タイム ゾーンで真夜中以降に経過した時間をミリ秒単位で返します。
\$Date	ローカル タイム ゾーンで 1970 年 1 月 1 日以降に経過した日数を返します。
\$DateTime	ローカル タイム ゾーンで 1970 年 1 月 1 日以降に経過した日数（端数を含む）を返します。
DateTimeGMT()	世界協定時刻（UTC）で 1970 年 1 月 1 日以降に経過した日数（端数を含む）を返します。

\$Year システム タグ変数

現在の年を返します。

構文

\$Year

データタイプ

整数型（読み取り専用）

例

このスクリプトは、文字列 "Welcome to xxxx" を文字列 Welcome に割り当てます。ここで、xxxx は現在の年です。

```
Welcome = "Welcome to " + StringFromIntg($Year,10)
```

\$Month システム タグ変数

現在の月を返します。

構文

\$Month

データタイプ

整数型（読み取り専用）

例

このスクリプトは、現在の月が 10 の場合、文字列 "October" を文字列 MonthName に割り当てます。

```
IF $Month==10 THEN
    MonthName="October";
ENDIF;
```

\$Day システム タグ変数

現在の日付を返します。

構文

\$Day

データタイプ

整数型（読み取り専用）

例

このスクリプトは、現在の日付が 2 月 29 日の場合、文字列「今年はうるう年です!」を文字列 Msg2User に割り当てます。

```
IF $Day==29 AND $Month==2 THEN
    Msg2Usr="今年はうるう年です!";
ENDIF;
```

\$Hour システム タグ変数

現在の時間を返します。

構文

\$Hour

データタイプ

整数型（読み取り専用）

例

このスクリプトは、現在の時刻が 8 PM で、バックアップが実行されていないかどうか（論理型タグ変数 BackupAlreadyRun を使用）を確認し、そうである場合、RunBackup() という QuickFunction スクリプトを呼び出して、BackupAlreadyRun フラグを TRUE に設定します。

```
IF $Hour==20 AND BackupAlreadyRun==0 THEN
    CALL RunBackup();
    BackupAlreadyRun=1;
ENDIF;
```

\$Minute システム タグ変数

現在の分を返します。

構文

\$Minute

データタイプ

整数型（読み取り専用）

例

このスクリプトは、現在の時刻が 4:50 PM であるかを確認し、その場合、Shift End という名前のウィンドウを表示します。

```
IF $Minute==50 AND $Hour==16 THEN  
    Show "Shift End";  
ENDIF;
```

\$Second システム タグ変数

現在の秒を返します。

構文

\$Second

データタイプ

整数型（読み取り専用）

例

このスクリプトは、振幅 100 および周期 1 分のサイン波を生成します。

```
100*Sin(6*$Second)
```

このスクリプトは、毎秒ごとに変化する一連の 0 と 1 を生成します。

```
$second.00
```

\$Msec システム タグ変数

現在のミリ秒を返します。

注意：デフォルトでは、InTouch は、1000 ミリ秒ごとにすべてのタグ変数を更新します。このため、\$Msec システム タグ変数は変化していないように見えます。WindowViewer プロパティで更新速度を増加すると、\$Msec タグ変数が更新していることを確認できます。

構文

\$Msec

データタイプ

整数型（読み取り専用）

\$Time システム タグ変数

ローカル時刻で真夜中以降に経過した時間をミリ秒単位で返します。

構文

\$Time

データタイプ

整数型（読み取り専用）

例

このスクリプトは、真夜中以降に経過した秒数を返します。

```
$Time/1000
```

\$Date システム タグ変数

ローカル時刻で 1970 年 1 月 1 日以降に経過した日数を返します。

構文

\$Date

データ タイプ

整数型（読み取り専用）

例

このスクリプトは、現在の時刻を返します。

```
StringFromTime(($Date*86400)+($Time/1000),3);
```

\$DateTime システム タグ変数

ローカル時刻で 1970 年 1 月 1 日以降に経過した日数（端数を含む）を返します。

構文

\$DateTime

データ タイプ

実数型（読み取り専用）

例

このスクリプトは、現在の時刻を返します。

```
StringFromTime($DateTime*86400,3);
```

DateTimeGMT() 関数

世界協定時刻（UTC）で 1970 年 1 月 1 日以降に経過した日数（端数を含む）を返します。

注: この関数は、アニメーション表示リンクでは使用できません。

構文

```
result = DateTimeGMT();
```

戻り値

UTC で 1970 年 1 月 1 日以降の日数。リテラル実数値です。

例

このスクリプトは、UTC での現在の日付および時刻を返します。

```
StringFromTime(DateTimeGMT() * 86400.0, 3);
```

文字列の日付と時刻情報の取得

スクリプトで、日付および時刻情報を文字列として取得できます。これは、画面に日付または時刻を表示、または時刻／日付の文字列全体の計算が必要な場合に役立ちます。

以下のシステム タグ変数およびスクリプト関数を使用できます。

使用	目的
\$DateString	システム日付を短い形式で返します。
\$TimeString	システム時刻を返します。
UTCDateTime	ローカル コンピュータの UTC 時刻およびタイム ゾーン、あるいはその両方を返します。

\$DateString システム タグ変数

ローカル オペレーティング システムの [地域と言語のオプション] の定義に従い、システム日付を短い形式で返します。

構文

```
$DateString
```

データタイプ

文字列（読み取り専用）

例

このスクリプトは、オペレーティングシステムの［地域と言語のオプション］の短い日付形式設定に従い 2006/4/28 を返します。

```
$DateString
```

\$TimeString システム タグ変数

ローカル オペレーティングシステムの［地域と言語のオプション］の定義に従い、システム時刻を返します。

構文

```
$TimeString
```

データタイプ

文字列（読み取り専用）

例

このスクリプトは、オペレーティングシステムの［地域と言語のオプション］の時刻形式設定に従い 02:40:37 PM を返します。

```
$TimeString
```

UTCDateTime() 関数

UTC 時刻、UTC 日付および時刻、またはローカル タイム ゾーンを返します。

構文

```
result = UTCDateTime (format)
```

パラメータ

format

返される内容を決めます。以下のいずれかの値のリテラル文字列値、メッセージ型タグ変数、または文字列式です。

UTC_SHORT - UTC 時刻を返します。

UTC_LONG - UTC 日付および時刻を返します。

UTC_LOCAL - ローカル オペレーティングシステムのタイム ゾーン設定の設定に従い、タイム ゾーンの名前を返します。

他の値を指定すると、UTC 日付および時刻がデフォルト形式 (ddd mm dd hh:mm:ss yyyy) で返されます。

例

太平洋タイムゾーンで 2003 年 1 月 6 日、月曜 09:24 AM の場合、UTCDateTime() 関数は、以下のように情報を返します。

このスクリプトは、17:24:05 を返します。

```
UTCDateTime("UTC_SHORT")
```

このスクリプトは、01/06/2003 17:24:05 を返します。

```
UTCDateTime("UTC_LONG")
```

このスクリプトは、太平洋標準時 -8:0: 1 を返します。1

```
UTCDateTime("UTC_LOCAL")
```

このスクリプトは、Mon Jan 06 17:24:05 2003 を返します。

```
UTCDateTime("Invalid")
```

文字列への日付と時刻情報の変換

スクリプトで、解釈や表示をより簡単にするため、日付および時刻情報を文字列に変換できます。以下の関数を使用できます。

使用場所	目的
<code>StringFromTime()</code>	UTC タイムスタンプをローカル時刻に変換し、結果を時刻文字列として返します。
<code>wwStringFromTime()</code>	ローカル タイムスタンプを UTC 時刻に変換して、結果を時刻文字列として返します。
<code>StringFromTimeLocal()</code>	タイムスタンプを時刻文字列として変換します。

StringFromTime() 関数

UTC 時刻のタイムスタンプをローカル時刻に変換して、結果を文字列として返します。この関数は、夏時間を考慮します。

注: この関数は、`StringFromGMTTimeToLocal()` 関数と同じです。

構文

```
result = StringFromTime (timestamp, format)
```

パラメータ

timestamp

UTC タイムゾーンで 1970 年 1 月 1 日の真夜中以降に経過した秒数を返します。リテラル整数値、整数型タグ名、または整数式です。

形式

文字列結果の表示方法を決めます。以下の意味を示す 1 ～ 5 のリテラル整数値、整数型タグ名、または整数式です。

- 1 - ローカル オペレーティング システムの [地域と言語のオプション] で設定された形式に従い日付を表示します。
- 2 - ローカル オペレーティング システムの [地域と言語のオプション] で設定された形式に従い時刻を表示します。
- 3 - 日付および時刻を 24 文字の文字列 (ddd mmm dd hh:mm:ss yyyy) で表示します。
- 4 - 曜日を短い形式で表示します。
- 5 - 曜日を長い形式で表示します。

例

この例では、ローカル ノードのタイムゾーンは太平洋標準時 (PST、UTC-0800) です。関数に渡される UTC 時間は 1970 年 1 月 2 日金曜日の 12:00:00 AM です。PST は UTC よりも 8 時間遅れているので、関数は以下の結果を返します。

以下のスクリプトは "1/1/70" を返します。

```
StringFromTime(86400,1)
```

以下のスクリプトは "04:00:00 PM" を返します。

```
StringFromTime(86400,2)
```

このスクリプトは "Thu Jan 01 16:00:00 1970" を返します。

```
StringFromTime(86400,3)
```

このスクリプトは "Thu" を返します。

```
StringFromTime(86400,4)
```

このスクリプトは "Thursday" を返します。

```
StringFromTime(86400,5)
```

wwStringFromTime() 関数

ローカル時刻のタイムスタンプを UTC 時刻に変換して、結果を文字列として返します。この関数は、夏時間を考慮します。

構文

```
result = wwStringFromTime (timestamp, format)
```

パラメータ

timestamp

ローカルタイムゾーンで 1970 年 1 月 1 日の真夜中以降に経過した秒数を返します。リテラル整数値、整数型タグ変数、または整数式です。

format

文字列結果の表示方法を決めます。以下の意味を示す 1 ～ 5 のリテラル整数値、整数タグ変数、または整数式です。

- 1 - ローカルオペレーティングシステムの [地域と言語のオプション] で設定された形式に従い日付を表示します。
- 2 - ローカルオペレーティングシステムの [地域と言語のオプション] で設定された形式に従い時刻を表示します。
- 3 - 日付および時刻を 24 文字の文字列 (ddd mmm dd hh:mm:ss yyyy) で表示します。
- 4 - 曜日を短い形式で表示します。
- 5 - 曜日を長い形式で表示します。

例

この例では、ローカルノードのタイムゾーンは太平洋標準時 (PST、UTC-0800) です。関数に渡されるローカル時間は 1970 年 1 月 1 日木曜日の 04:00:00 PM です。PST は UTC よりも 8 時間遅れているので、関数は以下の結果を返します。

以下のスクリプトは "1/2/70" を返します。

```
wwStringFromTime(57600,1)
```

以下のスクリプトは "12:00:00 AM" を返します。

```
wwStringFromTime(57600,2)
```

以下のスクリプトは "Fri Jan 02 00:00:00 1970" を返します。

```
wwStringFromTime(57600,3)
```

以下のスクリプトは "Fri" を返します。

```
wwStringFromTime(57600,4)
```

以下のスクリプトは "Friday" を返します。

```
wwStringFromTime(57600.5)
```

StringFromTimeLocal() 関数

タイムスタンプを時刻に変換して、結果を文字列として返します。

構文

```
result = StringFromTimeLocal (timestamp, format)
```

パラメータ

timestamp

1970 年 1 月 1 日の真夜中以降に経過した秒数を返します。リテラル整数値、整数型タグ変数、または整数式です。

format

文字列結果の表示方法を決めます。以下の意味を示す 1 ～ 5 のリテラル整数値、整数タグ変数、または整数式です。

- 1 - ローカル オペレーティング システムの [地域と言語のオプション] で設定された形式に従い日付を表示します。
- 2 - ローカル オペレーティング システムの [地域と言語のオプション] で設定された形式に従い時刻を表示します。
- 3 - 日付および時刻を 24 文字の文字列 (ddd mmm dd hh:mm:ss yyyy) で表示します。
- 4 - 曜日を短い形式で表示します。
- 5 - 曜日を長い形式で表示します。

例

以下のスクリプトは "1/2/70" を返します。

```
StringFromTimeLocal(86400,1)
```

以下のスクリプトは "12:00:00 AM" を返します。

```
StringFromTimeLocal(86400,2)
```

以下のスクリプトは "Fri Jan 02 00:00:00 1970" を返します。

```
StringFromTimeLocal(86400,3)
```

以下のスクリプトは "Fri" を返します。

```
StringFromTimeLocal(86400,4)
```

以下のスクリプトは "Friday" を返します。

```
StringFromTimeLocal(86400.5)
```

夏時間状態の確認

スクリプトで、wwIsDaylightSaving() 関数を使用すると、夏時間設定がアクティブになっているかどうかを確認できます。

wwIsDaylightSaving() 関数

夏時間が現在アクティブになっているかどうかを返します。

構文

```
result = wwIsDaylightSaving()
```

戻り値

ブール値の意味は以下のとおりです。

0 - 夏時間はアクティブになっていません。

1 - 夏時間はアクティブになっています。

他のアプリケーションとの対話

スクリプトで、さまざまな QuickScript を使用して、他の Windows アプリケーションと対話できます。たとえば、以下の操作を実行できます。

- メモ帳などのアプリケーションを起動する
- アプリケーション タイトル名を確認する
- 特定のアプリケーションが実行しているかどうか確認する
- 実行中のアプリケーションをアクティブにする
- キーボード入力をシミュレートする
- アプリケーション ウィンドウを閉じる、最小化する、または最大化する
- コマンドを実行し、DDE をサポートするアプリケーションとデータを交換する

Windows アプリケーションの起動

スクリプトで、StartApp コマンドを使用すると、Windows アプリケーションを起動できます。

構文

```
StartApp appname;
```

パラメータ

appname

起動するアプリケーションのパスおよびファイル名です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

注意：アプリケーションのパスおよびファイル名を把握する必要があります。アプリケーションが、Windows PATH 環境変数の一部であるディレクトリにある場合、ファイル名のみ（パスなし）を渡す必要があります。

例

このスクリプトは、Microsoft の電卓を起動します。

```
StartApp "calc"
```

実行中のアプリケーションのアプリケーション タイトルの取得

スクリプトで、InfoAppTitle() 関数を使用すると、指定した実行中のアプリケーションのアプリケーション タイトルまたは Windows タスク リスト名を検索できます。この情報は、たとえば、InTouch スクリプトで、指定したアプリケーションが現在実行中かどうかを確認するとき、またはそれをアクティブにするときに必要です。

注意：この機能では、Internet Explorer の値は返されません。回避策として、Google Chrome ブラウザを使用してください。

InfoAppTitle() 関数

指定した実行中のアプリケーションのアプリケーション タイトルまたは Windows タスク リスト名を返します。

構文

```
result = InfoAppTitle (appname)
```

パラメータ

appname

.exe 拡張子なしのアプリケーション名です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

例

このスクリプトは、"Calculator" を返します。

```
InfoAppTitle("calc")
```

このスクリプトは、"Microsoft Excel" を返します。

```
InfoAppTitle("excel")
```

アプリケーションが実行中かどうかの確認

スクリプトで、InfoAppActive() 関数を使用すると、特定のアプリケーションがすでに実行中かどうかを確認できます。特定のアプリケーションが実行中かどうかを確認するには、そのアプリケーション タイトルまたは Windows タスク リスト名を把握する必要があります。

InfoAppActive() 関数

アプリケーションの実行状態を返します。

構文

```
result = InfoAppActive (apptitle)
```

パラメータ

apptitle

実行状態を照会するアプリケーションのアプリケーション タイトルまたは Windows タスク リストです。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

戻り値

ブール値の意味は、以下のとおりです。

0 - アプリケーションは実行していません。

1 - アプリケーションは実行しています。

例

このスクリプトは、メモ帳アプリケーションを照会し、実行中である場合はアクティブにします。実行していない場合、メモ帳の新しいインスタンスを起動します。このようにして、メモ帳を複数回起動することを避けます。

```
IF InfoAppActive(InfoAppTitle("Notepad"))==1  
THEN  
    ActivateApp InfoAppTitle( "Notepad" );  
ELSE  
    StartApp "Notepad";  
ENDIF;
```

実行中の Windows アプリケーションのアクティブ化

スクリプトで、`ActivateApp()` 関数を使用すると、実行中の Windows アプリケーションをアクティブにできます。アクティブにすると、指定アプリケーションが前面に表示され、フォーカスが与えられます。

実行中の Windows アプリケーションをアクティブにするには、以下を実行する必要があります。

- アプリケーション タイトルまたは Windows タスク リスト名を検索する。「[実行中のアプリケーションのアプリケーション タイトルの取得](#)」を参照してください。
- Windows アプリケーションが実行中であることを確認する。「[アプリケーションが実行中かどうかの確認](#)」を参照してください。

ActivateApp 関数

すでに実行中の Windows アプリケーションをアクティブ化します。

構文

```
ActivateApp apptitle;
```

パラメータ

apptitle

アクティブ化する実行中のアプリケーションのアプリケーション タイトルまたは Windows タスク リスト名です。

例

このスクリプトは、コマンドプロンプト ウィンドウが開いているかを確認し、開いている場合はアクティブ化します。開いていない場合、コマンドプロンプト ウィンドウを起動します。

```
IF InfoAppActive( InfoAppTitle("cmd")) == 1 THEN  
    ActivateApp InfoAppTitle("cmd");  
ELSE  
    StartApp "cmd";  
ENDIF;
```

アプリケーションへのシミュレートしたキー入力の送信

スクリプトで、キーボードの一連のキーを押す操作をシミュレートできます。たとえば、以下の操作を実行するために使用できます。

- 開いているアプリケーションでデータを自動的に入力する
- アプリケーション (InTouch HMI など) を制御する

SendKeys 関数

一連のキー入力をシミュレートします。

重要： `SendKeys()` 関数は、64 ビットバージョンの Windows オペレーティングシステムでは機能しません。

構文

```
SendKeys sequence;
```

パラメータ

sequence

シミュレートする一連のキー入力です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

キーボードの通常の文字（英数字など）の他に、コードとして制御キーも指定できます。

```
{BACKSPACE} - Backspace キーをシミュレートします。  
{BREAK} - Break キーをシミュレートします。  
{CAPSLOCK} - Caps Lock キーをシミュレートします。  
{DELETE} - Delete キー（または {DEL}）をシミュレートします。  
{DOWN} - 下矢印キーをシミュレートします。  
{END} - End キーをシミュレートします。  
{ENTER} - Enter キー（または ~）をシミュレートします。  
{ESCAPE} - ESC キー（または {ESC}）をシミュレートします。  
{F1} .. {F12} - F1 .. F12 キーをシミュレートします。  
{HOME} - Home キーをシミュレートします。  
{INSERT} - Insert キーをシミュレートします。  
{LEFT} - 左矢印キーをシミュレートします。  
{NUMLOCK} - Num Lock キーをシミュレートします。  
{PGDN} - Page Down キーをシミュレートします。  
{PGUP} - Page Up キーをシミュレートします。  
{PRTSC} - Print Screen キーをシミュレートします。  
{RIGHT} - 右矢印キーをシミュレートします。  
{TAB} - Tab キーをシミュレートします。  
{UP} - 上矢印キーをシミュレートします。  
+ - Shift キーをシミュレートします。  
Shift キーと同時に押すキーは括弧で囲みます。  
^ : Ctrl キーをシミュレートします。  
Ctrl キーと同時に押すキーは括弧で囲みます。  
% - Alt キーをシミュレートします。  
Alt キーと同時に押すキーは括弧で囲みます。
```

備考

StartApp コマンドまたは ActivateApp() コマンド、あるいはその両方を使用すると、シミュレートしたキー入力送信する前に、別のアプリケーションをアクティブにできます。

例

このスクリプトは、B キーを押す操作をシミュレートしています。

```
SendKeys "b";
```

このスクリプトは、Ctrl キーを押しながら P を押す操作をシミュレートしています。これは、別のアプリケーションで [印刷] ダイアログ ボックスを起動するときに使用できます。

```
SendKeys "^ (p)";
```

このスクリプトは、F1（ヘルプ機能を開く）、Tab キー（カーソルを検索フィールドに置く）の順に押し、HAL と入力して、Enter キーを押す（検索を開始する）操作をシミュレートしています。

```
SendKeys "{F1}{TAB}HAL{ENTER}";
```

このスクリプトは、Ctrl、Shift、およびキー 1 を押す操作をシミュレートしています。これは、WindowMaker への切り替えと同じです。この強力な組み合わせは、自動修正（動的） InTouch HMI アプリケーションの開発に使用できます。

```
SendKeys "^(+(1))";
```

Windows アプリケーションを閉じる、最小化する、または最大化する

スクリプトで、WWControl() コマンドを使用すると、別の Windows アプリケーションを閉じたり、最小化したり、最大化したりできます。

Windows アプリケーションを閉じたり、最小化したり、最大化したりするには、以下を実行する必要があります。

- アプリケーション タイトルまたは Windows タスク リスト名を検索する。「[実行中のアプリケーションのアプリケーション タイトルの取得](#)」を参照してください。
- Windows アプリケーションが実行中であることを確認する。「[アプリケーションが実行中かどうかの確認](#)」を参照してください。

WWControl() 関数

Windows アプリケーションを復元、最小化、最大化、または終了します。

構文

```
WWControl (apptitle, control);
```

パラメータ

apptitle

元のサイズに戻す、最小化する、最大化する、または閉じる、実行中のアプリケーションのアプリケーション タイトルまたは Windows タスク リスト名です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

control

指定した Windows アプリケーションで実行するアクションを決めます。以下のいずれかの値のリテラル文字列値、メッセージ型タグ名、または文字列式です。

Restore - アプリケーション ウィンドウをアクティブにして、表示します。

Minimize - アプリケーション ウィンドウをアクティブにして、最小化します。

Maximize - アプリケーション ウィンドウをアクティブにして、最大化します。

Close - アプリケーションを閉じます。

例

このスクリプトは、すでに実行している場合、電卓アプリケーションを元のサイズに戻します。

```
WWControl ("Calculator","Restore");
```

このスクリプトは、WindowViewer を閉じます。

```
WWControl (InfoAppTitle("View"),"Close");
```

DDE を使用したコマンドの実行とデータの交換

DDE をサポートするアプリケーションと対話するスクリプトを記述できます。

使用	目的
WWExecute()	コマンドを送信および実行します。
WWRequest()	データを DDE アイテムから読み取ります。
WWPoke()	データを DDE アイテムに書き込みます。

WWExecute() 関数

コマンドをアプリケーションに送信し、それを実行して、状態結果を返します。これを使用すると、Excel でマクロを実行できます。

重要：WWExecute() 関数は、64 ビットバージョンの Windows オペレーティングシステムでは機能しません。

構文

```
Result = WWExecute (appname, topic, command)
```

パラメータ

appname

コマンドを送信するアプリケーションの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

topic

コマンドを送信するアプリケーション内のトピックの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

command

送信されるコマンドです。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

戻り値

以下を示す -1、0、または 1 の値です。

-1 - コマンドは正常に実行されませんでした。原因として、アプリケーションが実行中でない、トピックが存在しない、またはコマンドでエラーが発生したことが考えられます。

0 - アプリケーションがビジー状態であるため、コマンドは正常に実行されませんでした。

1 - コマンドは正常に実行されました。

例

このスクリプトは、コマンド [Run("Macro1",0)] を Excel に送信することで、Microsoft Excel にマクロ Macro1 を実行するように指示します。

```
Macro="Macro1";  
Command="[Run(" + StringChar(34) + Macro + StringChar(34) + ",0)]";  
WWExecute("excel","system",Command);
```

WWRequest() 関数

アプリケーションのアイテムからデータを読み取ります。たとえば、これを使用して、Microsoft Excel のスプレッドシートセルの値を読み取ることができます。

重要：WWRequest() 関数は、64 ビットバージョンの Windows オペレーティングシステムでは機能しません。

構文

```
Result = WWRequest(appname, topic, item, messagetag)
```

パラメータ

appname

アプリケーションの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

topic

アプリケーション内のトピックの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

item

トピックおよびアプリケーションに属すアイテムの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

messagetag

アイテムの値を受け取るメッセージ型タグ変数名です。メッセージ型タグ変数値は、StringToIntg() 関数または StringToReal() 関数を使用して、整数値または実数値に変換できます。

戻り値

以下を示す -1、0、または 1 の値です。

-1 - データは正常に読み取られませんでした。原因として、アプリケーションが実行中でない、またはトピックやアイテムが存在しないことが考えられます。

0 - アプリケーションがビジー状態であるため、データは正常に読み取られませんでした。

1 - データは正常に読み取られました。

例

このスクリプトは、Microsoft Excel ブック Book1.xls、シート Sheet1 の行 1、列 1 に含まれている値をメッセージ型タグ変数 MTag に読み取り、実数タグ変数 CellValue にこの値を配置します。

```
Result = WWRequest("excel","[Book1.xls]sheet1", "r1c1",Mtag);  
CellValue=StringToReal(MTag);
```

英語以外のオペレーティングシステムを使用している場合、StringReplace() 関数を使用して、別のデータタイプに変換する前に **MTag** の内容を変更しなければならないことがあります。たとえば、小数点としてカンマを使用するオペレーティングシステムでは、実数型データタイプに変換する前に **MTag** 内のカンマをすべて小数点に置換しなければならない場合があります。

WWPoke() 関数

データをアプリケーションのアイテムに書き込みます。たとえば、これを使用して、Excel のスプレッドシートセルに値を書き込むことができます。

重要: WWPoke() 関数は、64 ビットバージョンの Windows オペレーティングシステムでは機能しません。

構文

```
result = WWPoke (appname, topic, item, string)
```

パラメータ**appname**

アプリケーションの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

topic

アプリケーション内のトピックの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

item

トピックおよびアプリケーションに属すアイテムの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

string

書き込まれる値です。リテラル文字列値、メッセージ型タグ名、または文字列式です。StringFromIntg() 関数、StringFromReal() 関数、または Text() 関数を使用すると、整数型または実数型タグ名の値をメッセージ型タグ名に変換できます。

戻り値

以下を示す -1、0、または 1 の値です。

-1- データは正常に書き込まれませんでした。原因として、アプリケーションが実行中でない、またはトピックやアイテムが存在しないことが考えられます。

0- アプリケーションがビジー状態であるため、データは正常に書き込まれませんでした。

1- データは正常に書き込まれました。

備考

WWPoke() 関数または WWRequest() 関数を使用して、InTouch アプリケーション間の異なるノードまたはセッションに対してデータの読み取りや書き込みを実行しないでください。InTouch アプリケーション間でデータを読み取りおよび書き込むには、アクセス名を使用します。「[アクセス名の設定](#)」を参照してください。

例

このスクリプトは、実数型タグ名 CellValue の値をメッセージ型タグ名 Mtag に配置し、その値を Microsoft Excel ブック Book1.xls のシート Sheet1 のスプレッドシートセル行 1、カラム 1 に書き込みます。

```
Mtag = Text(CellValue,"0");
Result = WWPoke("excel","[Book1.xls]sheet1", "r1c1",Mtag);
```

ファイルの操作

さまざまなファイル管理およびアクセス操作を使用するスクリプトを記述できます。

使用	目的
FileCopy()	ファイルをコピーします。
FileDelete()	ファイルを削除します。
FileMove()	ファイルを移動します。
FileReadFields()、FileWriteFields()	csv データの読み取り／書き込みを行います。
FileReadMessage()、FileWriteMessage()	テキスト データの読み取り／書き込みを行います。

ファイルの管理

スクリプトでは、ファイルをコピー、削除、または移動できます。

FileCopy() 関数

ソース ファイルをコピー先ファイルにコピーして、結果状態を返します。この関数の実行には時間がかかることがあり、以下の複数の段階で実行されます。

1. FileCopy() 関数が呼び出され、ファイル コピー開始が成功または失敗したかを示す結果が返されます。
2. FileCopy() 関数は、バックグラウンドでコピー操作を実行し、InTouch スクリプトは、ファイルのコピー中も実行を続けます。ファイル コピーの進行状況は、整数型タグを使用して監視できます。
3. FileCopy() 関数は、ファイル コピー操作が成功または失敗したことを示すファイル コピーの結果を返します。

コピー先フォルダが使用できない場合（ネットワークの別のコンピュータなど）、関数はタイムアウトまで最大 10 秒間待機し、メッセージを **Logger** にポストします。

注記: `FileCopy()` 関数は、非同期クイック関数では使用しないでください。

構文

```
result = FileCopy (sourcefile, destfile, progresstag)
```

パラメータ

sourcefile

コピーされるファイルのフルパスおよびファイル名です。リテラル文字列値、メッセージ型タグ名、または文字列式です。このパラメータでワイルドカード文字（* および ?）を使用すると、指定条件に一致するファイルだけをコピーできます。パス名には、UNC パス名も使用できます。

destfile

コピー先のフルパスおよびファイル名（またはパス名のみ）です。リテラル文字列値、メッセージ型タグ名、または文字列式です。パス名には、UNC パスも使用できます。

progresstag

ファイルコピーの進行状況を示す値を含む、二重引用符で囲まれた整数型タグの名前です。リテラル文字列値、メッセージ型タグ名（値 "IntTag.Name" を含むメッセージ型タグなど）、または文字列式です。値の意味は以下のとおりです。

0 - `FileCopy()` 操作は現在進行中です。

1 - `FileCopy()` 操作は正常に完了しました。

-1 - `FileCopy()` 操作はエラーで終了しました。

戻り値

以下を示す -1、0、または 1 の値です。

1 - `FileCopy()` 関数は、正常に呼び出されました。

0 - 別の `FileCopy()` 操作がすでに実行されているため、`FileCopy()` 関数を呼び出すときにエラーが発生しました。

-1 - ソースファイルが存在しないか、コピー先ファイルが読み取り専用であるため、`FileCopy()` 関数を呼び出すときにエラーが発生しました。

例

このスクリプトは、ファイル `c:\MyData\output.log` をディレクトリ `d:\archive` にコピーして、ファイルの名前を `output.txt` に変更します。ファイルコピーの進行状況は、整数型タグ **Monitor** に書き込まれます。
`Status=FileCopy("c:\MyData\output.log","d:\archive\output.txt","Monitor");`

このスクリプトは、`c:\` ルートディレクトリにあり、名前が `.txt` で終わるすべてのファイルをコピー先ディレクトリ `c:\Backup` にコピーします。
`Status=FileCopy("c:*.txt", "c:\Backup", "Monitor");`

このスクリプトは、フルパスおよびファイル名がメッセージ型タグ **LogFile** に含まれるファイルをコピー先ディレクトリ `c:\results\` にコピーして、そのファイルの名前を `logxxx.txt` に変更します。ここで、`xxx` はタイムスタンプです。

```
Status=FileCopy(LogFile, "c:\results\log" + $DateString + $TimeString + ".txt", "Monitor");
```

FileDelete() 関数

個々のファイルを削除します。

構文

```
result = FileDelete (filename)
```

パラメータ

filename

削除するファイルのパス名およびファイル名です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。UNC パス名がサポートされています。

備考

FileDelete() 関数にはワイルドカード文字 (* および ?) を使用しないでください。また、**FileDelete()** 関数を非同期クイック関数で使わないでください。

FileDelete() 関数は、ディレクトリを削除しません。

戻り値

ファイル削除の成功または失敗を示す値です。

1 - ファイルは正常に削除されました。

0 - ファイルは正常に削除されませんでした。原因として、読み取り専用ファイルを削除しようとしたか、ファイルが存在しないことが考えられます。

例

このスクリプトは、ファイル **c:\Data.txt** を削除して、ファイルが検索あり、正常に削除された場合は **1** を返します。

```
Status=FileDelete("c:\Data.txt");
```

FileMove() 関数

ソース ファイルを移動先ファイルに移動して、結果状態を返します。また、ファイルの名前を変更するときにも使用できます。この関数の実行には時間がかかることがあり、以下の複数の段階で実行されます。

1. **FileMove()** 関数が呼び出され、ファイル移動開始が成功または失敗したかを示す結果が返されます。
2. **FileMove()** 関数は、バックグラウンドで移動操作を実行し、**InTouch** スクリプトは、ファイルの移動中も実行を続けます。ファイル移動の進行状況は、整数型タグを使用して監視できます。
3. **FileMove()** 関数は、ファイル移動操作が成功または失敗したかを示す、ファイル移動の結果を返します。

FileMove() 関数は、非同期クイック関数では使用しないでください。

構文

```
result = FileMove (sourcefile, destfile, progresstag)
```

パラメータ

sourcefile

移動されるファイルのフルパスおよびファイル名です。リテラル文字列値、メッセージ型タグ名、または文字列式です。このパラメータでワイルドカード文字 (* および ?) を使用すると、指定条件に一致するファイルだけを移動できます。パス名には、UNC パス名も使用できます。

destfile

コピー先のフルパスおよびファイル名（またはパス名のみ）です。リテラル文字列値、メッセージ型タグ名、または文字列式です。パス名には、UNC パスも使用できます。

progresstag

ファイル移動の進行状況を示す値を含む、二重引用符で囲まれた整数型タグの名前です。リテラル文字列値、メッセージ型タグ名（値 "IntTag" を含むメッセージ型タグなど）、または文字列式です。値の意味は以下のとおりです。

0 - FileMove() 操作は現在進行中です。

1 - FileMove() 操作は正常に完了しました。

-1 - FileMove() 操作はエラーで終了しました。

戻り値

以下を示す -1、0、または 1 の値です。

1 - FileMove() 関数は、正常に呼び出されました。

0 - 別の FileMove() 操作がすでに実行されているため、FileMove() 関数を呼び出すときにエラーが発生しました。

-1 - FileMove() 関数を呼び出すときにエラーが発生しました。存在しないファイルを移動しようとしたなどのエラーが考えられます。

例

このスクリプトは、ファイル c:\MyData\output.log をディレクトリ d:\archive に移動して、ファイルの名前を output.txt に変更します。ファイル移動の進行状況は、整数型タグ Monitor に書き込まれます。

```
Status=FileMove("c:\MyData\output.log","d:\archive\output.txt","Monitor");
```

このスクリプトは、c:\ ルート ディレクトリにあり名前が .txt で終わるすべてのファイルを移動先ディレクトリ c:\Backup に移動します。

```
Status=FileMove("c:\*.txt", "c:\Backup", "Monitor");
```

このスクリプトは、フルパスおよびファイル名がメッセージ型タグ LogFile に含まれるファイルを移動先ディレクトリ c:\results\ に移動して、そのファイルの名前を logxxx.txt に変更します。ここで、xxx はタイムスタンプです。

```
Status=FileMove(LogFile, "c:\results\log" + $DateString + $TimeString + ".txt", "Monitor");
```

CSV データの読み取りと書き込み

FileReadFields() 関数および FileWriteFields() 関数を使用すると、一連のタグ変数に対して csv（カンマ区切り形式）ファイルに含まれているデータを読み取りおよび書き込むスクリプトを作成できます。

FileReadFields() 関数および FileWriteFields() 関数は、カンマを区切り文字としてのみサポートしています。

FileReadFields() 関数

csv ファイルに含まれている値を一連のタグ変数に読み取ります。この関数を使用すると、一組のタグ変数値をロードできます。

サポートされている区切り文字はカンマだけです。

この関数は、同期呼び出しでのみ使用できます。

構文

```
[result = ] FileReadFields (filename, offset, starttag, numberoffields)
```

パラメータ

filename

データを読み取る csv ファイルの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

offset

読み取りを開始するファイルの位置（バイト単位）です。リテラル整数値、整数型タグ変数、または整数式です。

starttag

最初の読み取りデータ アイテムを受け取る最初のタグ変数の名前です。このタグ変数は、"MyTag1" のように、二重引用符で囲み、数値で終わる必要があります。リテラル文字列値、メッセージ型タグ変数名（値 "MyTag1" を含むメッセージ型タグ変数など）、または文字列式です。

numberoffields

csv ファイルから読み取るデータ アイテムの名前です。リテラル整数値、整数型タグ変数、または整数式です。最初のデータ アイテムは、**starttag** パラメータで定義されているタグ変数に読み取られ、その後のデータ アイテムは、**starttag** パラメータに接尾辞として増分値が付けられたタグ変数（MyTag1、MyTag2、MyTag3、...）に読み取られます。

戻り値

データ読み取り後のオプションの新しいファイル オフセット（バイト単位）です。これは、データの次のセットを読み取る際に使用できます。

例

この csv ファイル c:\set.csv に「Flour, 27.23,14,1」というデータが含まれていて、RecipeTag1:message、RecipeTag2:real、Recipe3:integer、RecipeTag4:discrete のタグ変数が定義されている場合、このスクリプトは、値 Flour を RecipeTag1 に、27.23 を RecipeTag2 に、14 を RecipeTag3 に、そして 1 を RecipeTag4 に読み取り、新しいオフセットを返します。

```
FileReadFields("c:\set.csv",0,"RecipeTag1",4);
```

FileWriteFields() 関数

一連のタグ名に含まれている値を csv ファイルに書き込みます。この関数を使用すると、タグ名の値のセットを保存できます。

サポートされている区切り文字はカンマだけです。

構文

```
[result = ] FileWriteFields (filename, offset, starttag, numberoffields)
```

パラメータ**filename**

データを書き込む csv ファイルの名前です。ファイルが存在しない場合、新しいファイルが作成されます。リテラル文字列値、メッセージ型タグ名、または文字列式です。

offset

書き込みを開始するファイルの位置（バイト単位）です。ファイルの最後に書き込むには（追加）、-1 を使用します。リテラル整数値、整数型タグ名、または整数式です。

starttag

書き込まれる最初のデータ アイテムを含む最初のタグ名の名前です。タグ名は、"MyTag1" のように、二重引用符で囲み、数値で終わる必要があります。リテラル文字列値、メッセージ型タグ名（値 "MyTag1" を含むメッセージ型タグなど）、または文字列式です。

numberoffields

csv ファイルに書き込むデータ アイテムの名前です。リテラル整数値、整数型タグ名、または整数式です。最初のデータ アイテムは、**starttag** パラメータで定義されているタグ名からファイルに書き込まれ、

その後のデータ アイテムは、**starttag** パラメータの増分数字の接尾辞付きのタグ名（MyTag1、MyTag2、MyTag3...）から書き込まれます。

戻り値

データ書き込み後のオプションの新しいファイル オフセット（バイト単位）です。これは、データの次のセットを書き込むときに使用できます。

例

一連の InTouch タグは以下のように定義されています。

タグ名	データ型	値
RecipeTag1	メッセージ型	Flour
RecipeTag2	実数型	27.23
RecipeTag3	整数型	14
RecipeTag4	論理型	1

このスクリプトは、RecipeTag1 ～ RecipeTag4 に含まれている値を csv ファイル c:\set.csv に書き込みます。
FileWriteFields("c:\set.csv",0,"RecipeTag1",4);

これで、ファイル c:\set.csv には以下のデータが含まれます。

Flour,27.23,14,1

テキストデータの読み取りと書き込み

FileReadMessage() 関数および FileWriteMessage() 関数を使用すると、ファイルに対してテキストデータを読み取りまたは書き込みを行うスクリプトを作成できます。指定バイト数、またはテキストのライン全体（ラインフィードで区切られる）を読み取りまたは書き込みすることができます。

FileReadMessage() 関数

ファイルから文字列データの指定バイト数（または1つの行）を読み取ります。

構文

```
[result = ] FileReadMessage (filename, offset, messagetag, charstoread)
```

パラメータ

filename

データを読み取るファイルの名前です。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

offset

読み取りを開始するファイルの位置（バイト単位）です。リテラル整数値、整数型タグ変数、または整数式です。

messagetag

ファイルから最初のラインまたはバイト数を受け取るメッセージ型タグ変数です。産業用グラフィック エディタのスクリプト エディタ内で関数を使用する場合は、タグ名を二重引用符で囲みます。

charstoread

ファイルから読み取るバイト数です。ラインフィード記号（LF）まで読み取る場合、0 に設定します。リテラル整数値、整数型タグ変数、または整数式です。

戻り値

読み取り後の新しいバイト位置を含みます。この値は、ファイルから続けて読み取りを行う場合に使用できます。

例

次のスクリプトは、ファイル `c:\Data\File.txt` のデータの最初のラインをメッセージ型タグ変数 `MsgTag` に読み取ります。

```
FileReadMessage ("c:\Data\File.txt",0,MsgTag, 0);  
FileReadMessage ("c:\Data\File.txt",0,"InTouch:MsgTag", 0);
```

FileWriteMessage() 関数

ファイルに文字列データの指定バイト数（または 1 つの行）を書き込みます。

構文

```
[result = ] FileWriteMessage (filename, offset, messagetag, linefeed)
```

パラメータ

filename

データを書き込むファイルの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

offset

書き込みを開始するファイルの位置（バイト単位）です。ファイルの最後にデータ書き込むには（追加）、`-1` に設定します。リテラル整数値、整数型タグ名、または整数式です。

messagetag

ファイルに書き込むデータを含むメッセージ型タグ名です。

linefeed

データをファイルに書き込んだ後、ラインフィード記号（LF）を書き込むかどうかを指定します。ラインフィード記号を書き込む場合、`1` に設定します。書き込まない場合、`0` に設定します。リテラルブール値、論理型タグ名、またはブール式です。

戻り値

書き込み後の新しいバイト位置を含みます。この値は、ファイルに続けて書き込みを行う場合に使用できます。

例

次のスクリプトは、メッセージ型タグ名 `MsgTag` の値をファイル `c:\Data\File.txt` の最後に書き込みます。

```
FileWriteMessage("c:\Data\File.txt",-1,MsgTag,1);
```

システム関連情報の取得

スクリプトで、以下のクイック関数を使用すると、システム関連情報を取得できます。

使用場所	目的
<code>GetNodeName()</code>	コンピュータのノード名を取得します。
<code>InfoDisk()</code>	ディスク空き容量情報を取得します。
<code>InfoFile()</code>	ファイル情報を取得します。

コンピュータのノード名の取得

スクリプトで、**GetNodeName()** 関数を使用すると、コンピュータのノード名を取得できます。この関数は、たとえばアクセス名を操作するときに、InTouch アプリケーションを動的に保つ場合に使用できます。

GetNodeName() 関数

コンピュータのノード名を返します。

構文

```
GetNodeName (messagetag, nodelist);
```

パラメータ

messagetag

ノード名を含むメッセージ型タグ変数です。産業用グラフィック エディタのスクリプト エディタ内で関数を使用する場合は、タグ名を二重引用符で囲みます。

nodelist

ノード名から取得する文字数です。0 ～ 131 のリテラル整数値、整数型タグ変数、または整数式です。

例

次のスクリプトは、ノード名を取得して、**NodeName** メッセージ型タグ変数に割り当てます。

```
GetNodeName(NodeName,131);  
GetNodeName("InTouch:NodeName",131);
```

ディスク容量情報の取得

スクリプトで、**InfoDisk()** 関数を使用すると、ディスク容量情報を取得できます。以下の情報を取得できます。

- ディスク ドライブの合計サイズ（バイトまたはキロバイト単位）
- ディスク ドライブの使用可能な空き容量（バイトまたはキロバイト単位）

また、トリガタグ変数を指定すると、（アニメーションリンクで）情報を取得するタイミングや方法を指定することもできます。

InfoDisk() 関数

ローカルまたはネットワーク ディスク ドライブの合計容量または空き容量を返します。

構文

```
result = InfoDisk (drive, infotype, trigger);
```

パラメータ

drive

情報を取得するドライブ文字です。文字列の最初の文字だけが使用されます。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

infotype

情報タイプを指定します。以下のいずれかの値のリテラル整数値、整数型タグ変数、または整数式です。

- 1 - ディスク ドライブの合計サイズを返します（バイト単位）。
- 2 - ディスク ドライブの空き容量を返します（バイト単位）。

3- ディスク ドライブの合計サイズを返します (キロバイト単位)。

4- ディスク ドライブの空き容量を返します (キロバイト単位)。

trigger

ディスク情報を再計算するトリガとして機能するタグ変数 (または式) です。トリガ値が変わると、ディスク情報が再計算されます。論理型またはアナログ型のタグ変数、または論理型またはアナログ型の式です。

備考

トリガタグ変数は、InfoDisk() 関数がアニメーション表示リンクで使用される場合だけ意味があります。この関数がスクリプトで使用される場合、リテラル数値、アナログ型タグ変数または数値式を指定できます。

例

アニメーション表示リンクで以下のスクリプトを使用すると、ディスク ドライブ C の空き容量を表示して、その情報を毎分更新できます。

```
InfoDisk("C", 4, $Minute)
```

ファイルまたはディレクトリの情報の取得

スクリプトで、InfoFile() 関数を使用すると、特定のファイルまたはディレクトリに関する情報を取得できます。さまざまなパラメータを使用することで、以下の情報を取得できます。

- ファイルが存在するかどうか
- 指定ファイル名が実際はディレクトリかどうか
- ファイルのサイズ (バイト単位)
- ファイルまたはディレクトリのタイムスタンプ
- ワイルドカード検索に一致するファイル数

InfoFile() 関数

ファイルまたはディレクトリのさまざまな情報を返します。

構文

```
result = InfoFile (filename, infotype, trigger)
```

パラメータ

filename

情報を取得するフル ファイル名またはディレクトリ名です。リテラル文字列値、メッセージ型タグ名、または文字列式です。 "*" および "?" などのワイルドカード文字を含めることもできます。

infotype

指定ファイルまたはディレクトリについて取得する情報のタイプです。リテラル整数値、整数型タグ名、または整数式です。値と意味は以下のとおりです。

1- 存在。InfoFile() 関数は、ファイルが存在する場合は **1**、ファイルがディレクトリの場合は **2**、ファイルまたはディレクトリが存在しない場合は **0** を返します。

2- サイズ。ファイルのサイズをバイト単位で返します。

3 - 作成タイムスタンプ。InfoFile() 関数は、タイムスタンプを 1970 年 1 月 1 日の真夜中以降に経過した秒数として返します。StringFromTimeLocal() 関数を使用すると、この値をメッセージタイムスタンプに変換できます。

4 - ワイルドカード検索一致。InfoFile() 関数は、指定したワイルドカード検索に一致するファイル数を返します。

trigger

ファイル情報を再計算するトリガとして機能するタグ名（または式）です。トリガ値が変わると、ファイル情報が再計算されます。論理型またはアナログ型のタグ名、または論理型またはアナログ型の式です。

備考

トリガタグは、InfoFile() 関数がアニメーション表示リンクで 사용되는場合だけ意味があります。この関数がスクリプトで 사용되는場合、リテラル数値、アナログ型タグ名または数式を指定できます。

例

このスクリプトは、ファイル c:\data\log.txt が存在する場合 1 を返します。

```
InfoFile("c:\data\log.txt",1,$minute)
```

このスクリプトは、ファイル c:\data\log.txt のファイルサイズが 14223 のバイトの場合 14223 を返します。

```
InfoFile("c:\data\log.txt",2,$minute)
```

このスクリプトは、ファイル c:\data\log.txt が 2006 年 1 月 26 日 11:14:26 AM に作成された場合 1138245266 を返します。

```
InfoFile("c:\data\log.txt",3,$minute)
```

このスクリプトは、c:\data\ ディレクトリに txt で終了するファイルが 14 個ある場合 14 を返します。

```
InfoFile("c:\data\*.txt",4,$minute)
```

InTouch 関連情報の取得

スクリプトで、以下の関数を使用すると、InTouch 関連情報を取得できます。

使用	目的
InfoInTouchAppDir()	開発している InTouch アプリケーションのディレクトリ情報を取得します。
InTouchVersion()	InTouch バージョン情報を取得します。

InTouch アプリケーションディレクトリの名前の取得

スクリプトで、InfoInTouchAppDir() 関数を使用すると、InTouch アプリケーションが実行されているディレクトリの名前を取得できます。この関数は、InTouch アプリケーションに同梱する外部ファイルを検索する場合に役立ちます。

InfoInTouchAppDir() 関数

現在の InTouch アプリケーションディレクトリを返します。

構文

```
result = InfoInTouchAppDir();
```

戻り値

現在実行している InTouch アプリケーションのディレクトリを含めるメッセージ型タグ変数です。

備考

文字数が 131 文字に制限されているため、メッセージ型タグ変数に渡されるとき、またはアニメーションリンクで表示されるときに、アプリケーションディレクトリ名が切り捨てられることがあります。

例

このスクリプトは、c:\documents and settings\user1\my documents\マイ InTouch アプリケーション\packaging を返します。

```
InfoInTouchAppDir()
```

InTouch バージョンの取得

スクリプトで、InTouchVersion() 関数を使用すると、現在実行している InTouch アプリケーションのバージョン番号を取得できます。

InTouchVersion() 関数

完全な InTouch バージョン番号、またはその一部を返します。

構文

```
result = InTouchVersion (infotype);
```

パラメータ

infotype

バージョン情報がどのように返されるかを指定します。リテラル整数値、整数型タグ変数、または整数式の意味は以下のとおりです。

- 0 - 完全なバージョン番号を返します。
- 1 - メジャー バージョン番号を返します。
- 2 - マイナー バージョン番号を返します。
- 3 - パッチ レベルを返します。
- 4 - ビルド レベルを返します。

例

機能	考えられる結果
InTouchVersion(0)	10.5.1626.0521.0045.0012
InTouchVersion(1)	10
InTouchVersion(2)	5
InTouchVersion(3)	0
InTouchVersion(4)	1626

セキュリティ関連スクリプト

さまざまな QuickScript 関数およびシステム タグ変数を使用すると、InTouch アプリケーションでセキュリティを追加および管理できます。セキュリティ機能の詳細については、『AVEVA™ InTouch HMI 管理ガイド』の「[InTouch のセキュリティ保護](#)」を参照してください。

ログオンとログオフ

以下の関数とシステム タグ変数を使用して、ログオンおよびログオフできます。

使用場所	目的
AttemptInvisibleLogon()	認証データをパラメータに提供することで、ユーザーをログオンします。
LogonCurrentUser()	現在 Windows ユーザーにログオンしているユーザーをログオンします（認証モードが "OS" の場合です）。
PostLogonDialog()	[ログオン] ダイアログ ボックスを表示します。
Logoff()	現在のユーザーをログオフします。
\$PasswordEntered	パスワードを設定します。
\$OperatorEntered	有効なユーザー名を設定します。
\$OperatorDomainEntered	有効なユーザー ドメイン名を設定します（認証モードが "OS" の場合）。

セキュリティ機能の詳細については、『AVEVA™ InTouch HMI 管理ガイド』の「[InTouch のセキュリティ保護](#)」を参照してください。

パスワードの変更と設定

以下の関数およびシステム タグ変数をパスワードの変更に使用できます。

使用場所	目的
ChangePassword()	現在ログオンしているユーザーに対して、[パスワードの変更] ダイアログ ボックスを呼び出します。
\$ChangePassword	現在ログオンしているユーザーに対して、[パスワードの変更] ダイアログ ボックスを呼び出します。

セキュリティ機能の詳細については、『AVEVA™ InTouch HMI 管理ガイド』の「[InTouch のセキュリティ保護](#)」を参照してください。

ユーザーの指定と設定

以下のシステム タグ変数をユーザーの指定および設定に使用できます。

使用場所	目的
\$ConfigureUsers	[ユーザーの設定] ダイアログ ボックスを呼び出します。

セキュリティ機能の詳細については、『AVEVA™ InTouch HMI 管理ガイド』の「[InTouch のセキュリティ保護](#)」を参照してください。

セキュリティとその他の情報の管理

以下のシステム タグ変数および関数をセキュリティの管理に使用できます。

使用場所	目的
\$AccessLevel	現在ログイン中のユーザーのアクセス レベルを取得します。
AddPermission()	アクセス レベルを特定のユーザーグループ（ローカル/ドメイン）に割り当てます。
GetAccountStatus()	アカウント情報（パスワードの使用期限、ロック、無効フラグ）を取得します。
\$InactivityTimeout	ユーザーが自動的にログオフされるまでに経過する時間を示します。
\$InactivityWarning	タイムアウト警告の時間を示します。
InvisibleVerifyCredentials()	OS ユーザーの InTouch アクセス レベル情報を取得します。
IsAssignedRole()	現在ログオンしているユーザーが、指定したユーザー役割であるかどうかを確認します。
QueryGroupMembership()	現在ログオンしているユーザーが指定したユーザー役割のメンバであるかどうかを確認します。

セキュリティ機能の詳細については、『AVEVA™ InTouch HMI 管理ガイド』の「[InTouch のセキュリティ保護](#)」を参照してください。

その他のスクリプト

InTouch スクリプトは、ヒューマンマシンインタラクションと音声に関連付けることができるように、音声出力をサポートしています。また、InTouch スクリプトは、ウィザードのプロパティの取得および設定もサポートしています。

InTouch アプリケーションからの音声ファイルの再生

スクリプトで、イベントおよび条件を特定の音声に関連付けることができます。たとえば、警告ダイアログボックスまたは重要な条件を警告音に関連付けることができます。

PlaySound() 関数

wave ファイルの音声または Windows のデフォルト音声を再生します。

構文

```
PlaySound (soundname, flag)
```

パラメータ

soundname

音声または wave ファイルの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。音声の名前として定義されている場合、たとえば、MC="c:\test.wav" のように、Win.ini ファイルの [Sounds] セクションで定義されている必要があります。

flag

音声の再生方法を指定します。リテラル整数値、整数型タグ名、または整数式です。値と意味は以下のとおりです。

- 0 - 音声を一度に同時に再生します（音声再生が終了するまで、スクリプトは実行を待機します）。
- 1 - 音声を一度に非同期に再生します（スクリプトは、音声再生の終了を待機しません）。
- 9 - 音声を連続して再生します（PlaySound() 関数が再び呼び出されるまでです）。

例

このスクリプトは、ファイル c:\welcome.wav の音声を一度再生して、再生が終了するまでスクリプト実行を停止します。

```
PlaySound("c:\welcome.wav",0);
```

このスクリプトは、音声 Alert を連続して再生します。win.ini ファイル [Sounds] セクションで、以下のよう、音声名 Alert と音声ファイルに関連付ける必要があります。

```
Alert=c:\alert.wav.  
PlaySound("Alert",9);
```

ウィザードのプロパティの取得と設定

分散アラームオブジェクトやウィンドウコントロールなどのいくつかのウィザードには、設定プロパティまたは読み取りプロパティが含まれています。これらのプロパティは、テキストボックスの値、またはチェックボックスのオンまたはオフ状態です。

スクリプトで、以下の関数を使用すると、これらのプロパティにアクセスできます。

使用	目的
SetPropertyD()、GetPropertyD()	論理型プロパティを設定または読み取ります。
SetPropertyI()、GetPropertyI()	整数型プロパティを設定または読み取ります。
SetPropertyM()、GetPropertyM()	メッセージ型プロパティを設定または読み取ります。

サポートされているプロパティのリストについては、ウィザードの説明を参照してください。

これらのプロパティを設定したり読み取ったりする一般的な方法を以下に示します。

GetPropertyD() 関数

ウィザードの論理型プロパティを読み取り、成功コードを返します。

構文

```
result = GetPropertyD (controlname.property, dtag)
```

パラメータ

controlname

プロパティをサポートするウィザードの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

property

読み取られるウィザードの論理型プロパティです。**controlname** と組み合わせて、リテラル文字列値、メッセージ型タグ名、または文字列式を使用できます。

dtag

論理型プロパティ値を取得する論理型タグ名です。

戻り値

整数型のエラー コードです。エラー コードの詳細については、「[ウィンドウ コントロール エラー メッセージの理解](#)」を参照してください。

例

チェック ボックス ウィザードが **Checkbox1**、論理型タグ名が **dtag** の場合、以下のスクリプト関数を使用すると、このチェック ボックスの表示または非表示を確認できます。

```
result=GetPropertyD("Checkbox1.visible",dtag);
```

このスクリプトは、チェック ボックス ウィザードが表示可能な場合、**dtag** を **1** に設定し、表示不可の場合 **0** に設定します。

SetPropertyD() 関数

ウィザードの論理型プロパティを設定して、成功コードを返します。

構文

```
result = SetPropertyD(controlname.property, Boolean)
```

パラメータ

controlname

プロパティをサポートするウィザードの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

property

設定されるウィザードの論理型プロパティです。**controlname** と組み合わせて、リテラル文字列値、メッセージ型タグ名、または文字列式を使用できます。

Boolean

ウィザードプロパティに渡すブール値です。リテラルブール値、論理型タグ名、またはブール式です。

戻り値

整数型のエラー コードです。エラー コードの詳細については、「[ウィンドウ コントロール エラー メッセージの理解](#)」を参照してください。

例

チェック ボックス ウィザードが **Checkbox1**、論理型タグ名が **dtag** の場合、以下のスクリプト関数を使用すると、このチェック ボックスの表示または非表示を制御できます。

```
result=SetPropertyD("Checkbox1.visible",dtag);
```

dtag を 0 に設定し、上記のスクリプト関数を呼び出すと、チェック ボックス ウィザードは非表示となります。

GetPropertyI() 関数

ウィザードで整数を読み取り、成功コードを返します。

構文

```
result = GetPropertyI (controlname.property, itag)
```

パラメータ

controlname

プロパティをサポートするウィザードの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

property

読み取られるウィザードの整数型プロパティです。**controlname** と組み合わせて、リテラル文字列値、メッセージ型タグ名、または文字列式を使用できます。

itag

整数型プロパティ値を取得する整数型タグ名です。

戻り値

整数型のエラー コードです。エラー コードの詳細については、「[ウィンドウ コントロール エラー メッセージの理解](#)」を参照してください。

例

ラジオ ボタン ウィザードが **Radiobutton1**、整数型タグ名が **itag** の場合、以下のスクリプト関数を使用すると、このラジオ ボタン グループで現在選択されているアイテムを確認できます。

```
result=GetPropertyI("Radiobutton1.value",itag);
```

このスクリプトは、最初（2 つ目、3 つ目...）のラジオ ボタンが選択されている場合、**itag** を 1（2、3...）に設定します。

SetPropertyI() 関数

ウィザードで整数型プロパティを設定して、成功コードを返します。

構文

```
result = SetPropertyI (controlname.property, integer)
```

パラメータ

controlname

プロパティをサポートするウィザードの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

property

設定されるウィザードの整数型プロパティです。**controlname** と組み合わせて、リテラル文字列値、メッセージ型タグ名、または文字列式を使用できます。

integer

ウィザードプロパティに渡す整数値です。リテラル整数値、整数型タグ名、または整数式です。

戻り値

整数型のエラーコードです。エラーコードの詳細については、「[ウィンドウコントロールエラーメッセージの理解](#)」を参照してください。

例

ラジオ ボタン ウィザードが **Radiobutton1** の場合、以下のスクリプト関数を使用すると、2 つ目のラジオ ボタンを設定できます。

```
result=SetPropertyI("Radiobutton1.value",2);
```

GetPropertyM() 関数

ウィザードのメッセージ型プロパティを読み取り、成功コードを返します。

構文

```
result = GetPropertyM (controlname.property, mtag)
```

パラメータ**controlname**

プロパティをサポートするウィザードの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

property

読み取られるウィザードのメッセージ型プロパティです。**controlname** と組み合わせて、リテラル文字列値、メッセージ型タグ名、または文字列式を使用できます。

mtag

メッセージ型プロパティ値を取得するメッセージ型タグ名です。

戻り値

整数型のエラーコードです。エラーコードの詳細については、「[ウィンドウコントロールエラーメッセージの理解](#)」を参照してください。

例

チェック ボックス ウィザードが **Checkbox1**、メッセージ型タグ名が **mtag** の場合、以下のスクリプト関数を使用すると、このチェック ボックスのキャプションを確認できます。

```
result=GetPropertyM("Checkbox1.caption",mtag);
```

このスクリプトは、**mtag** をチェック ボックスのキャプションに設定します。

SetPropertyM() 関数

ウィザードのメッセージ型プロパティを設定して、成功コードを返します。

構文

```
result = SetPropertyM (controlname.property, message)
```

パラメータ**controlname**

プロパティをサポートするウィザードの名前です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

property

設定されるウィザードのメッセージ型プロパティです。controlname と組み合わせて、リテラル文字列値、メッセージ型タグ名、または文字列式を使用できます。

message

ウィザードプロパティに渡すメッセージ値です。リテラル文字列値、メッセージ型タグ名、または文字列式です。

戻り値

整数型のエラーコードです。エラーコードの詳細については、「[ウィンドウコントロールエラーメッセージの理解](#)」を参照してください。

例

チェックボックスウィザードが Checkbox1 の場合、以下のスクリプト関数を使用すると、このチェックボックスウィザードのキャプションを動的に設定できます。

```
result=SetPropertyM("Checkbox1.caption","Start Engine 1");
```

このスクリプトは、チェックボックス Checkbox1 のキャプションを "Start Engine 1" に設定します。

OLE オブジェクトを使用したスクリプト

OLE オブジェクトを使用すると、InTouch HMI アプリケーションの機能を拡張できます。OLE オブジェクトを使用すると、以下の操作を実行できます。

- オペレータインターフェイスのポップアップダイアログボックスを作成する。
- コントロールパネルなど、オペレーティングシステムの機能にアクセスする。
- Manufacturing Execution Module のデータを InTouch HMI 内での処理に使用できるようにする。
Manufacturing Execution Module のマニュアルを参照してください。

OLE オブジェクトの作成、確認および解放

InTouch スクリプトで使用する OLE オブジェクトを作成および確認できます。OLE オブジェクトを使用したら、解放することによって、メモリを確保できます。

以下の関数を使用すると、OLE オブジェクトを作成、確認、および解放できます。

- [OLE_CreateObject\(\) 関数](#)
- [OLE_IsObjectValid\(\) 関数](#)
- [OLE_ReleaseObject\(\) 関数](#)

OLE_CreateObject() 関数

スクリプトで OLE オブジェクトを参照するには、最初に OLE オブジェクトを作成する必要があります。OLE オブジェクトを作成すると、それを参照するポインタを受け取ります。

スクリプトで、OLE_CreateObject() 関数を使用することによって、OLE オブジェクトを作成して、ポインタを割り当てることができます。

構文

```
OLE_CreateObject(%pointer, classname);
```

パラメータ

%pointer

OLE オブジェクトへのポインタの名前です。英数字（A ～ Z、0 ～ 9）およびアンダースコアを含めることができます。大文字と小文字は区別されません。

classname

OLE クラスの名前です。クラス名では大文字と小文字が区別されます。リテラル文字列値、メッセージ型タグ変数、または文字列式です。

備考

同じオブジェクト名を使用して別のオブジェクトを作成すると、そのオブジェクトは更新され、新しい OLE クラスを参照します。これは、古い OLE クラスから解放されます。

例

このスクリプトでは、クラス Wscript.Shell を参照する %WShell と呼ばれる OLE オブジェクトが作成されます。

```
OLE_CreateObject(%WShell, "Wscript.Shell");
```

OLE_IsObjectValid() 関数

スクリプトで、OLE_IsObjectValid() 関数を使用することによって、OLE オブジェクトが有効であるかどうかを確認できます。これは、OLE オブジェクトを操作する場合に必要な手順ではありませんが、OLE オブジェクトを操作する場合に問題が発生しないことを確認するために推奨します。

構文

```
result = OLE_IsObjectValid(%pointer)
```

引数

%pointer

テストされる OLE オブジェクトを参照するポインタです。

result

以下の内容を示すブール値です。

0 - ポインタが参照している OLE オブジェクトが無効です。

1 - ポインタが参照している OLE オブジェクトが有効です。

例

このスクリプトでは、Wscript.Shell クラスに基づいて OLE クラスが作成され、それを参照するポインタ %WS が作成されます。isvalid は、OLE オブジェクトが正常に作成された場合に TRUE となる論理型タグ変数です。それ以外の場合、FALSE となります。

```
OLE_CreateObject(%WS, "Wscript.Shell");  
isvalid = OLE_IsObjectValid(%WS);
```

OLE_ReleaseObject() 関数

OLE オブジェクトをスクリプトで使用した後でこれを解放し、そのポインタを削除して、システム リソースを確保できます。OLE オブジェクトを削除すると、そのポイントを使用して、関連付けられている OLE クラスのプロパティおよびメソッドにアクセスできなくなります。

構文

```
OLE_ReleaseObject(%pointer);
```

引数

%pointer

OLE オブジェクトを参照するポインタ名です。英数字（A ～ Z、0 ～ 9）およびアンダースコアを含めることができます。大文字と小文字は区別されません。

例

このスクリプトでは、ポイント %WShell に関連付けられている OLE オブジェクトが解放され、ポインタ %WShell が削除されます。

```
OLE_ReleaseObject(%WShell);
```

OLE オブジェクトのプロパティとメソッドの使用

スクリプトでは、ポインタを使用して、OLE プロパティに対して値の読み取りおよび書き込みを行うことができます。また、ポインタを使用して、OLE メソッドを呼び出すこともできます。使用できるプロパティおよびメソッドは、OLE オブジェクトにより異なります。

OLE オブジェクトのプロパティへのアクセス

スクリプトで、ほとんどのプログラム言語で行うように、OLE オブジェクトのプロパティにアクセスできます。プロパティは通常、ドット "." 演算子を使用して、識別されます。

注意：OLE オブジェクトプロパティをスクリプトで使用する場合、先頭の "%" を含め、リファレンスが 98 文字を超えないようにしてください。OLE ポインタ名はできるだけ短くしてください。

OLE オブジェクト プロパティの読み取り

スクリプトでは、プロパティをタグ変数に割り当てることによって、OLE オブジェクト プロパティを読み取ることができます。OLE オブジェクト プロパティへの直接リファレンスをアニメーション表示リンクで使用することはできません。

構文

```
tagname = %pointer.property;
```

引数

%pointer

OLE オブジェクトを参照するポインタです。OLE_CreateObject() 関数を使用して作成するか、プロパティを読み取る前に別のポインタに割り当てる必要があります。

property

読み取るプロパティの名前です。

tagname

値の書き込み先のタグ変数です。

例

このスクリプトでは、System.Random OLE クラスに基づいて OLE オブジェクトが作成され、そのオブジェクトを参照するポインタ %SR が作成され、Math.Random OLE オブジェクトの .NextDouble プロパティの値が実数型タグ変数 randtag に割り当てられます。

ランタイムでは、実数型タグ変数 Randtag は、0 ～ 1 の間のランダムな Double 浮動値を受け取ります。

```
OLE_CreateObject(%SR,"System.Random");  
randtag = %SR.NextDouble;
```

OLE オブジェクト プロパティへの書き込み

スクリプトでは、値をプロパティに割り当てることで、OLE オブジェクト プロパティに値を書き込むことができます。

構文

```
%pointer.property = value;
```

引数

%pointer

OLE オブジェクトを参照するポインタです。OLE_CreateObject() 関数を使用して作成するか、プロパティに書き込む前に別のポインタに割り当てる必要があります。

property

書き込まれるプロパティの名前です。

value

プロパティに書き込まれる値です。リテラル値、タグ変数、または式となります。アニメーション入力リンクから直接 OLE プロパティに書き込むことはできません。

OLE オブジェクトのメソッドの呼び出し

スクリプトで、OLE オブジェクト メソッドを呼び出すことができます。

構文

```
%pointer.method(parameters);
```

引数

%pointer

OLE オブジェクトを参照するポインタです。OLE_CreateObject() 関数を使用して作成するか、メソッドを呼び出す前に別のポインタに割り当てる必要があります。

method

OLE オブジェクトの一部であるメソッドの名前です。

parameters

メソッドに渡すパラメータのリストです。これらのパラメータは、カンマで区切る必要があります。リテラル値、タグ変数、または式です。

例

このスクリプトでは、OLE クラス Shell.Application に基づいて OLE オブジェクトが作成され、その OLE オブジェクトへのポインタ %sa が作成され、メソッド .MinimizeAll() が呼び出されます。このメソッドでは、デスクトップのすべてのウィンドウが最小化されます。

```
OLE_CreateObject(%SA,"Shell.Application");  
%SA.MinimizeAll();
```

注意： オプションパラメータは、OLE InTouch HMI スクリプトでは許可されていません。すべてのパラメータを指定する必要があります。

同じ OLE オブジェクトへの複数のポインタの割り当て

スクリプトでは、等号記号を使用することによって、複数のポインタを同じ OLE オブジェクトに割り当てることができます。

構文

```
%newpointer = %pointer
```

引数

%pointer

作成された OLE オブジェクトをすでに参照しているポインタの名前です。

%newpointer

同じ OLE オブジェクトを参照する新しいポインタの名前です。英数字（A ～ Z、0 ～ 9）およびアンダースコアを含めることができます。大文字と小文字は区別されません。

例

このスクリプトでは、Wscript.Shell クラスに基づいて OLE オブジェクトが作成され、このオブジェクトを参照するポインタ %WS が作成されます。ポインタ %WS2 は、%WS に設定されると、同じ OLE オブジェクトを参照します。これは、プロパティに対する読み取りおよび書き込み操作や、同じ OLE オブジェクトのメソッドの呼び出しに使用できます。

```
OLE_CreateObject(%WS,"Wscript.Shell");  
%WS2=%WS;
```

注意：メッセージ型タグ変数とポインタを合わせて使用できます。メッセージ型タグ変数をポインタに割り当てる場合、ID 値を取得できます。これを使用して、同じ OLE オブジェクトを参照する複数のポインタを作成できます。

OLE エラーのトラブルシューティング

スクリプトで、OLE 関数を使用して、OLE エラーをトラブルシューティングできます。

機能	説明
OLE_GetLastError() 関数	最後の OLE エラーのエラー番号を取得します。
OLE_GetLastErrorMessage() 関数	最後の OLE エラーの情報を取得します。
OLE_ResetObjectError() 関数	最後のエラーをリセットします。
OLE_ShowMessageOnObjectError() 関数	OLE エラー メッセージダイアログ ボックスを表示または非表示にします。
OLE_IncrementOnObjectError() 関数	InTouch HMI タグ変数を使用して、OLE エラー数をカウントします。

OLE_GetLastError() 関数

この関数は、最後の OLE エラーのエラー番号を返します。

構文

```
errnum = OLE_GetLastError();
```

引数

errnum

最後の OLE エラーの番号です。

OLE_GetLastObjectErrorMessage() 関数

この関数は、最後の OLE エラーのエラー メッセージを返します。

構文

```
errmsg = OLE_GetLastObjectErrorMessage();
```

引数

errmsg

最後の OLE エラーのエラー メッセージです。

OLE_ResetObjectError() 関数

スクリプトで、OLE_ResetObjectError() 関数を使用すると、最後の OLE エラーをリセットできます。これにより、最後の OLE エラー番号が 0 に設定され、最後の OLE エラー メッセージが空白に設定されます。

これは、OLE 関数のバッチで発生したエラーを特定するために使用できます。

構文

```
OLE_ResetObjectError()
```

OLE_ShowMessageOnObjectError() 関数

デフォルトでは、OLE エラーが発生すると、エラー メッセージ ダイアログ ボックスが表示されます。

スクリプトで、関数 OLE_ShowMessageOnObjectError() を使用すると、エラー メッセージ ダイアログ ボックスを表示するかどうかを指定できます。

構文

```
OLE_ShowMessageOnObjectError(Boolean)
```

引数

Boolean

OLE エラー メッセージ ダイアログ ボックスが表示されるかどうかを指定する値です。リテラルブール値、論理型タグ変数、またはブール式です。値の意味は以下のとおりです。

0 - OLE エラーが発生したとき、OLE エラー メッセージ ダイアログ ボックスは表示されません。

1 - OLE エラーが発生したとき、OLE エラー メッセージ ダイアログ ボックスが表示されます。

例

このスクリプトでは、すべての OLE エラー メッセージ ダイアログ ボックスが非表示となります。OLE エラーが発生したとき、エラー メッセージ ダイアログ ボックスは表示されません。

```
OLE_ShowMessageOnObjectError(0);
```

OLE_IncrementOnObjectError() 関数

スクリプトで、OLE_IncrementOnObjectError() 関数を使用すると、OLE エラーの数のカウンタとして、整数型タグ変数を指定できます。

構文

```
OLE_IncrementOnObjectError(integertag)
```

パラメータ

integertag

カウンタとして機能するタグ変数です。

備考

OLE エラー メッセージ ダイアログ ボックスが表示される場合、OLE エラー メッセージ ダイアログ ボックスが閉じられた後でのみ、カウンタ タグ変数の数が増加します。

例

このスクリプトでは、整数型タグ変数 **errorcount** がエラー カウンタとして指定され、エラー メッセージ ダイアログ ボックスが非表示にされ、無効な OLE クラス名に基づいて OLE オブジェクトが作成されます。ただし、この操作はエラーとなり、タグ変数値 **errorcount** の値は **1** に増加します。

```
errorcount = 0;  
OLE_IncrementOnObjectError(errorcount);  
OLE_ShowMessageOnObjectError(0);  
OLE_CreateObject(%WS,"InVaLiD.cLaSs.nAmE");
```

OLE を使用して行える操作

以下のスクリプトを使用すると、OLE オブジェクトを使用してアプリケーションに追加できる強力な機能について理解できます。

ランダム数の生成

スクリプトで、以下のコマンドを使用すると、0 ～ 255 までのランダム数を生成できます。

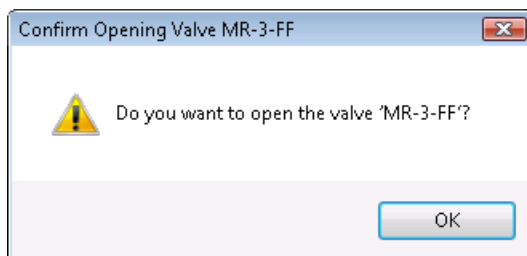
```
OLE_CreateObject(%SR,"System.Random");  
randtag = (%SR.NextDouble)*255;
```

ユーザー インターフェイス ダイアログ ボックスの作成

スクリプトで、以下のコマンドを使用すると、ユーザー インターフェイス ダイアログ ボックスを作成できます。

```
dim DlgBody as message;  
dim DlgTitle as message;  
dim Style as integer;  
dim Result as integer;  
DlgBody = "Do you want to open the valve 'MR-3-FF'?";  
DlgTitle = "Confirm Opening Valve MR-3-FF";  
Style = 48;  
OLE_CreateObject(%WS,"Wscript.Shell");  
result = %WS.Popup(DlgBody,1,DlgTitle,Style);
```

この例では、以下のユーザー インターフェイス ダイアログ ボックスが作成されます。



Style タグ変数は、ダイアログ ボックスに表示されるアイコンおよびボタンを決定します。以下の値を使用します。

アイコン	スタイル	値
(アイコンなし)	アイコンなし	0
	エラー アイコン	16
	疑問符アイコン	32
	警告アイコン	48
	情報アイコン	64

特定のボタンを使用するには、以下のいずれかの値を **Style** 値に追加します。

値	スタイル
0	[OK] ボタンのみ
1	[OK] ボタンおよび [キャンセル] ボタン
2	[中止] ボタン、[再試行] ボタン、および [無視] ボタン
3	[はい] ボタン、[いいえ] ボタン、および [キャンセル] ボタン
4	[はい] ボタンおよび [いいえ] ボタン
5	[再試行] ボタンおよび [キャンセル] ボタン
6	[キャンセル] ボタン、[再試行] ボタン、および [継続] ボタン

Result タグ変数には、ユーザーがクリックしたボタン番号が含まれます。これは、InTouch スクリプトの条件分岐に使用できます。以下の結果コードを使用できます。

結果値	意味
1	[OK] ボタンが押された
2	[キャンセル] ボタンが押された
3	[中止] ボタンが押された
4	[再試行] ボタンが押された
5	[無視] ボタンが押された

結果値	意味
6	[はい] ボタンが押された
7	[いいえ] ボタンが押された
10	[再試行] ボタンが押された
11	[継続] ボタンが押された

Windows の日付と時刻のプロパティ パネルを開く

スクリプトで、以下のコマンドを使用すると、Windows の [日付と時刻のプロパティ] パネルを開くことができます。

```
OLE_CreateObject(%WP,"Shell.Application");
%WP.SetTime();
```

同様のタスクを実行するには、異なるメソッドを呼び出して、これらを参照されている OLE オブジェクトに渡します。

メソッド	開くパネル
TrayProperties()	トレイ プロパティ
FileRun()	[ファイル実行] ダイアログ ボックス
FindFiles()	[ファイルの検索] ダイアログ ボックス
FindComputer()	[コンピュータの検索] ダイアログ ボックス
ShutdownWindows()	[Windows のシャットダウン] パネル

レジストリへの読み込みと書き込み

スクリプトで、以下の操作を行うことによって、OLE を使用して、Windows レジストリに対して読み取りや書き込みを行うことができます。

- Windows クラス Wscript.Shell に基づいて OLE オブジェクトを作成する
- OLE オブジェクトの RegRead() メソッドおよび RegWrite() メソッドを使用する

たとえば、以下のコマンドは、インストールされている InTouch HMI のバージョンを直接レジストリ キーから読み取り、その値を rkey メッセージタグ変数に保存します。

```
OLE_CreateObject(%WS,"Wscript.Shell");
rkey = %WS.RegRead("HKLM\SOFTWARE\Wonderware\InTouch\Installation\Version");
```

これらのコマンドでは、現在ログオンしているユーザーにファイル拡張子が非表示になっているかどうかを決めるレジストリ キーに値 1 が書き込まれます。

```
OLE_CreateObject(%WS,"Wscript.Shell");
%WS.RegWrite("HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\HideFileExt",1,"REG_DWORD");
```

ウィンドウの最小化

スクリプトで、以下のコマンドを使用すると、デスクトップのすべてのウィンドウを最小化できます。

```
OLE_CreateObject(%WA, "Shell.Application");  
%WA.MinimizeAll();
```

以下のメソッドを呼び出すと、同様のタスクを実行できます。

メソッド	ウィンドウの配置
TileHorizontally()	すべてのウィンドウを水平に整列します
TileVertically()	すべてのウィンドウを垂直に整列します
CascadeWindows()	すべてのウィンドウをカスケード表示します
UndoMinimizeALL()	すべてのウィンドウのサイズを元に戻します

ActiveX コントロールのスクリプト

ActiveX コントロールを使用すると、タグ名および I/O 参照の読み取りおよび書き込みを行えます。スクリプトでは、ActiveX コントロールを参照できます。

また、ActiveX コントロールに対してイベントが発生したときに実行するスクリプトを作成することもできます。これらのスクリプトは、他のアプリケーションで再使用およびインポートすることができます。

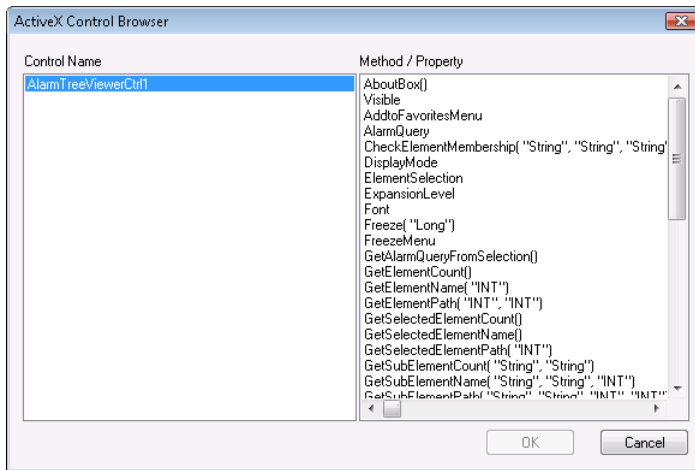
ActiveX コントロール メソッドの呼び出し

スクリプトでは、ActiveX コントロールのメソッドを呼び出して、ActiveX コントロールによりサポートされているアクションを実行できます。ActiveX メソッドは、任意のタイプの InTouch QuickScript または ActiveX イベント スクリプトから呼び出すことができます。

注記: ActiveX イベントが発生したときに ActiveX メソッドを呼び出すには、いくつかの必要条件があります。「[ActiveX イベント スクリプトの設定](#)」を参照してください。

ActiveX コントロール メソッドを呼び出すには

1. スクリプト ダイアログ ボックスの [挿入] メニューで、[ActiveX] をクリックします。
[ActiveX コントロール ブラウザ] ダイアログ ボックスが表示されます。



2. 左ペインから **ActiveX** コントロールの名前をクリックします。右ペインには、**ActiveX** コントロールによりサポートされているプロパティおよびメソッドの名前が含まれています。
3. 右ペインから使用するメソッドの名前をクリックし、続いて **[OK]** をクリックします。メソッド名およびデフォルトのパラメータが、スクリプト ウィンドウのカーソル位置に挿入されます。
4. メソッドパラメータは括弧内に設定します。

InTouch HMI からの **ActiveX** コントロール プロパティへのアクセス

スクリプトでは、**ActiveX** コントロール プロパティに対して読み込みおよび書き込みを実行して、**ActiveX** コントロールと InTouch タグ変数および表示リンク間でデータを交換できます。

データの読み取りや書き込みを行うための **ActiveX** コントロール プロパティの設定

スクリプトでは、**ActiveX** コントロールに対するデータの読み取りや書き込みを行うことができます。特定の **ActiveX** コントロールに関連付けられている **ActiveX** コントロール プロパティを使用できます。

以下の 2 つの方法があります。

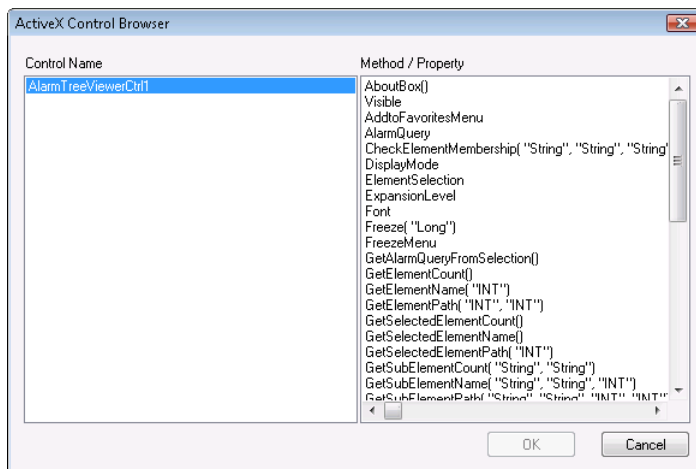
- **ActiveX** コントロール プロパティを InTouch HMI QuickScript または **ActiveX** イベント スクリプトで使用する。プロパティ値は、スクリプトが実行されるたびに読み取りまたは書き込みされます。
- **ActiveX** コントロール プロパティを InTouch HMI タグ変数または I/O リファレンスに直接リンクする。プロパティ値は、更新周期のたびに読み取りまたは書き込みされます。

ActiveX コントロール プロパティの読み取りや書き込みを行うためのスクリプトの設定

スクリプトでは、**ActiveX** コントロール プロパティを設定して、InTouch HMI タグ変数またはその他の式に対して値の書き込みや読み取りを行うことができます。

ActiveX コントロール プロパティに対してデータの読み取りや書き込みを行うには

1. スクリプト ウィンドウを開き、**[挿入]** をポイントして、**[ActiveX]** をクリックします。**[ActiveX コントロール ブラウザ]** ダイアログ ボックスが表示されます。



2. 左ペインから **ActiveX** コントロールの名前をクリックします。右ペインには、選択した **ActiveX** コントロールのプロパティおよびメソッドの名前が含まれます。
3. 右ペインから、使用するプロパティの名前をクリックします。プロパティ名が、スクリプトウィンドウのカーソル位置に挿入されます。
4. プロパティ名をタグ変数に割り当てるか、必要に応じて使用します。
5. **[OK]** をクリックします。

例

以下のスクリプトでは、**ActiveX** コントロール インスタンス **AlarmViewerCtrl1** の **ToPriority** プロパティが整数型タグ変数 **topri** に読み込まれます。

```
topri = #AlarmViewerCtrl1.ToPriority;
```

以下のスクリプトでは、値 **MS Comic** が **ActiveX** コントロールの **Font** プロパティである **AlarmViewerCtrl1** に書き込まれます。この例では、**AlarmViewer** **ActiveX** コントロールの表示フォントが動的に変更されます。

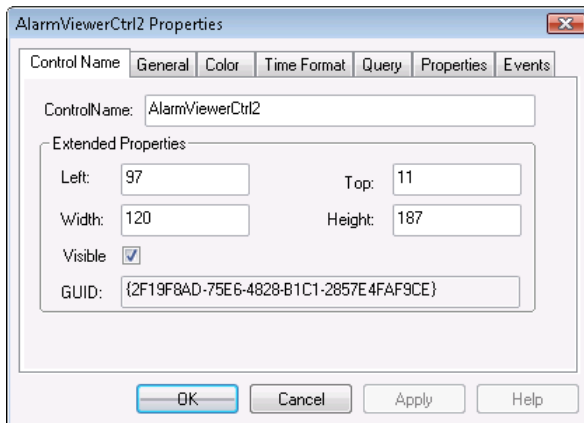
```
#AlarmViewerCtrl1.Font = "MS Comic";
```

タグ変数または I/O リファレンスへの **ActiveX** コントロール プロパティのリンク

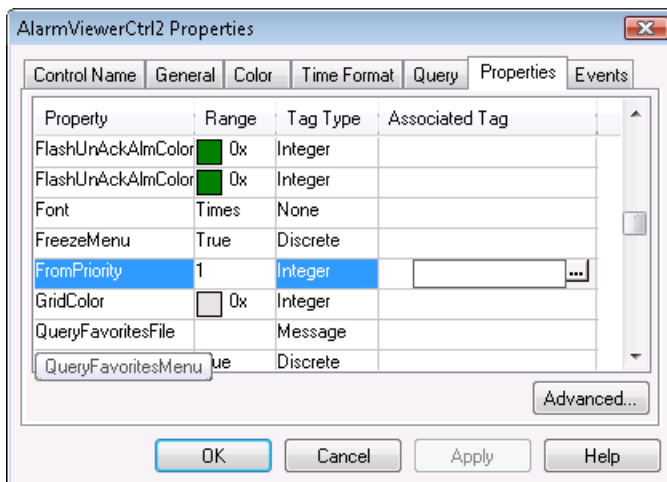
ActiveX コントロール プロパティを InTouch HMI タグ変数または I/O リファレンスにリンクできます。

ActiveX コントロール プロパティをタグ変数または I/O リファレンスにリンクするには

1. **ActiveX** コントロールをダブルクリックします。**ActiveX** コントロールのプロパティ ダイアログ ボックスが表示されます。



2. [プロパティ] タブをクリックして、右側にスクロールします。
3. 一覧からプロパティを選択します。



4. タグ変数または I/O リファレンスを割り当てます。以下のいずれかを実行します。
 - タグ変数 または I/O リファレンスを [関連タグ変数] カラムに直接入力します。
 - 角括弧の間の [関連タグ変数] カラムの参照ボタンをクリックします。[タグ変数を選択してください] ダイアログ ボックスが表示されます。タグ変数を選択して、[OK] をクリックします。
5. [OK] をクリックします。

ActiveX イベント スクリプトの作成と再使用

ActiveX コントロールでは、特定のアクションへの関連付けに使用できる、コントロールのシングルクリックなどのイベントがサポートされています。これらのアクションは、ActiveX イベント スクリプトに保存されます。

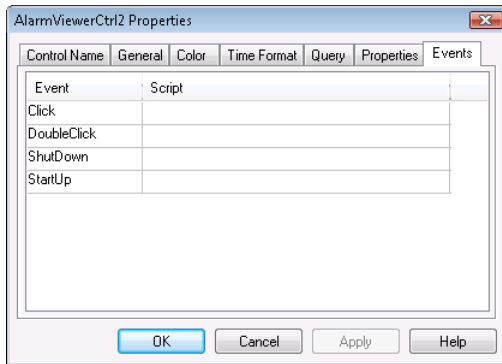
ActiveX イベント スクリプトの作成

ActiveX コントロールのクリックなど、特定の ActiveX コントロール イベントが発生するたびに実行されるイベント スクリプトを作成または再使用できます。

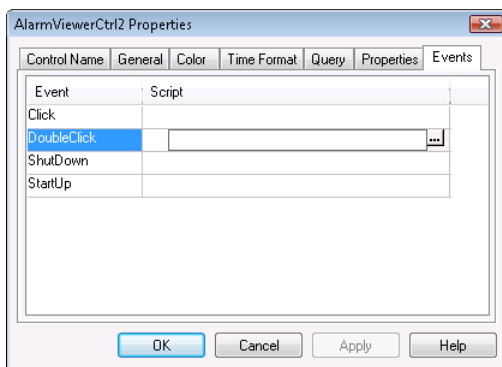
ActiveX イベント スクリプトを作成するには

1. ActiveX コントロールをダブルクリックします。プロパティ ダイアログ ボックスが表示されます。

2. [イベント] タブをクリックします。



3. 関連付けるイベントをクリックします。[スクリプト] カラムに括弧と参照ボタンが表示されます。



4. 対応する行の [スクリプト] カラムで、その括弧内をクリックします。
5. 新しい名前を入力して、[OK] をクリックします。メッセージが表示されたら、[OK] をクリックします。[ActiveX イベント スクリプト] ダイアログ ボックスが表示されます。
6. 必要に応じて、スクリプトを作成します。

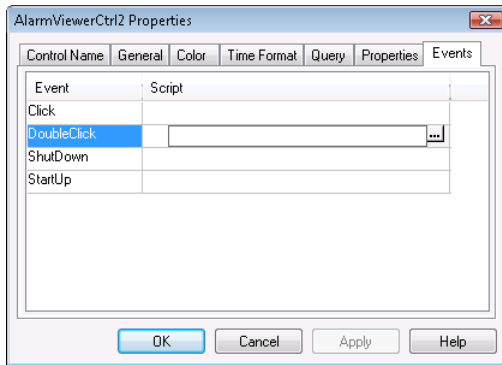
ActiveX イベント スクリプトの再使用

同じ ActiveX コントロールの親およびイベントにより作成された場合、ActiveX イベント スクリプトは再使用できます。

たとえば、アプリケーションに AlarmViewer ActiveX コントロールが複数ある場合、これらは、DoubleClick イベントのイベント スクリプトを共有できます。

ActiveX イベント スクリプトを再使用するには

1. ActiveX コントロールをダブルクリックします。プロパティ ダイアログ ボックスが表示されます。
2. [イベント] タブをクリックします。
3. 関連付けるイベントをクリックします。[スクリプト] カラムに括弧と参照ボタンが表示されます。



4. 対応する行の [スクリプト] カラムで、参照ボタンをクリックします。[ActiveX スクリプトの選択] ダイアログ ボックスが表示されます。
5. ActiveX スクリプトをクリックして、[OK] をクリックします。
6. [OK] をもう一度クリックします。

セルフリファレンス ActiveX イベント スクリプトの作成

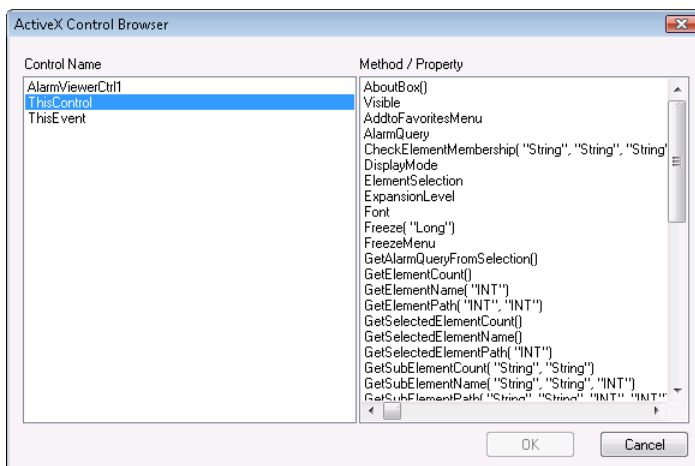
ActiveX イベント スクリプトを使用する場合、絶対的な ActiveX コントロール名ではなく、それら自身を参照するようにスクリプトを設定できます。これは、再使用される ActiveX イベント スクリプトを作成する場合に役立ちます。ActiveX イベント スクリプトは、以下のいずれかを参照できます。

- イベントを生成した特定の ActiveX コントロール (ThisControl)
- スクリプトを呼び出した特定のイベント (ThisEvent)

特定のイベントを参照すると、ActiveX コントロールでは、他のパラメータを ActiveX コントロール スクリプトに渡すことができます。

セルフリファレンス ActiveX イベント スクリプトを作成するには

1. 特定の ActiveX イベントの ActiveX イベント スクリプトを作成します。「[ActiveX イベント スクリプトの作成](#)」を参照してください。
2. [ActiveX イベント スクリプト] ダイアログ ボックスで、[挿入] をクリックして、[ActiveX] をクリックします。[ActiveX コントロール ブラウザ] ダイアログ ボックスが表示されます。



3. 左ペインで、以下のいずれかを実行します。

- **[ThisControl]** をクリックして、このコントロール（およびこのスクリプトを再使用できるその他のコントロール）と一緒に使用できるプロパティおよびメソッドを参照します。
 - **[ThisEvent]** をクリックして、セルフリファレンス イベントと一緒に使用できる **ActiveX** コントロールのプロパティおよびメソッドを参照します。
4. 右ペインで、いずれかのプロパティまたはメソッドをクリックして、**[OK]** をクリックします。選択したプロパティまたはメソッドが、スクリプト ウィンドウにコピーされます。
 5. スクリプトを設定します。
 6. **[OK]** をクリックします。

たとえば、次のステートメントは、ClicknRow イベント パラメータの値を ClickedRow タグ変数に書き込みます。

```
ClickedRow = ThisEvent.ClicknRow;
```

ActiveX イベント スクリプトのインポート

ActiveX イベント スクリプトを他の InTouch HMI アプリケーションからインポートして、現在開発中のアプリケーションで再使用できます。

他のアプリケーションから **ActiveX** イベント スクリプトをインポートするには

1. WindowMaker を開きます。
2. **[ファイル]** メニューの **[インポート]** をクリックし、**[表示]**、**[ウィンドウとスクリプト]** の順にクリックします。
[フォルダを開く] ダイアログ ボックスが表示されます。
3. インポートする **ActiveX** イベント スクリプトを含む InTouch HMI アプリケーションを参照します。
4. **[OK]** をクリックします。
1. **[ActiveX イベント スクリプト]** チェック ボックスをオンにし、**[インポート]** をクリックします。
すべての **ActiveX** イベントが現在の InTouch HMI アプリケーションにインポートされます。

章 12 ActiveX コントロール

ActiveX コントロールは、InTouch アプリケーションに追加の機能を提供するスタンドアロン ソフトウェア コンポーネントです。ActiveX コントロールには、コントロールの動作を変更するための、アプリケーションの実行中に変更可能なプロパティ、メソッド、およびイベントがあります。

InTouch アプリケーションでは、複数のタイプの ActiveX コントロールを使用できます。

- InTouch HMI にはアラーム用の ActiveX コントロールが含まれます。
- ActiveFactory などその他の製品は、データの操作および分析用のコントロールを提供しています。
- サードパーティの ActiveX コントロールを使用できます。
- Visual Basic または C で独自の ActiveX コントロールを開発できます。

InTouch アプリケーションでは、任意の数の ActiveX コントロールを使用できます。以下の操作を実行できます。

- ActiveX コントロールを選択して、任意のアプリケーション ウィンドウに貼り付ける
- コントロールによってサイズ変更がサポートされている場合、コントロールのサイズを調整する
- ActiveX コントロールの複製、切り取り、コピー、貼り付け、および削除を行う
- ActiveX コントロールを整列する：左寄せ、右寄せ、上端揃え、下端揃え、中心揃え
- ActiveX コントロールの追加
- セルを作成する場合、ActiveX コントロールを他のオブジェクトと組み合わせる
- 特定の ActiveX コントロール プロパティによってサポートされているプロパティ、メソッド、およびイベントを使用する

InTouch アプリケーションは以下のタイプの ActiveX コントロールはサポートしていません。

- ウィンドウのないコントロール
- シンプル フレーム サイトまたは領域ボックス
- コンテナ
- データ コントロール
- ディスパッチ オブジェクト

InTouch アプリケーションは、論理型（ブール）、整数型（32 ビット数）、実数型（32 ビット浮動小数点 IEEE 表記）、メッセージ型（131 文字までの文字列）のデータ型のみをサポートしています。サポートされていないデータ型には、バリエーション、ポインタ、配列、構造、およびパラメータ プロパティがあります。

ActiveX コントロールは、ウィンドウ コントロールまたはグラフィック オブジェクトなど、InTouch の他のオブジェクトと重ねることはできません。1 つのウィンドウに ActiveX コントロールが多すぎると、システム パフォーマンスが低下します。

ActiveX コントロールの使用

ActiveX コントロールを選択し、ウィンドウに貼り付け、ダブルクリックします。設定ダイアログ ボックスが表示されます。

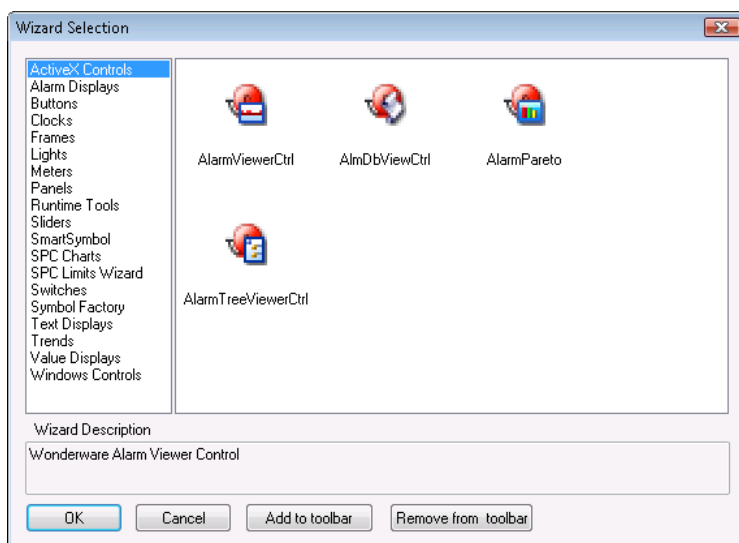
ActiveX コントロールを設定する際、固有のコントロール名を指定できます。コントロール名は、スクリプトで参照できるようになります。

ActiveX コントロールを InTouch アプリケーションで使用するには

1. ActiveX コントロールをインストールします。「[ActiveX コントロールのインストールとアンインストール](#)」を参照してください。
2. ActiveX コントロールを選択して、InTouch ウィンドウに貼り付けます。
3. ActiveX コントロール プロパティを設定して、タグにバインドします。
4. ActiveX イベントを ActiveX イベント スクリプトに関連付けます。
5. ActiveX メソッドを呼び出し、ActiveX コントロールのプロパティを ActiveX イベント スクリプトか、他の InTouch QuickScript に設定します。

ActiveX コントロールをウィンドウに配置するには

1. [描画] メニューの [挿入] グループで [ウィザード] をクリックします。
[ウィザードの選択] ダイアログ ボックスが表示されます。



2. ウィザードリストで、[ActiveX Controls] カテゴリをクリックします。使用できるすべての ActiveX コントロールが、表示領域に表示されます。
3. 使用する ActiveX コントロールをダブルクリックします。ダイアログ ボックスが閉じ、カーソルがコーナー シンボルに変わります。
4. ActiveX コントロールを貼り付ける位置をクリックします。

ActiveX コントロールをツールバーに追加するには

1. [描画] メニューの [挿入] グループで [ウィザード] をクリックします。
[ウィザードの選択] ダイアログ ボックスが表示されます。
1. 追加する ActiveX コントロールを選択します。
2. [ツールバーに追加] をクリックします。

ツールバーから **ActiveX** コントロールを削除するには

1. [描画] メニューの [挿入] グループで [ウィザード] をクリックします。
[ウィザードの選択] ダイアログ ボックスが表示されます。
2. [ツールバーから削除] をクリックします。[削除するウィザード] ダイアログ ボックスが表示されます。
3. 削除する **ActiveX** コントロールを選択します。
4. [OK] をクリックします。

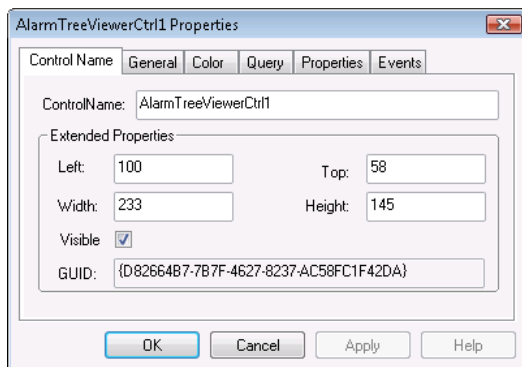
ActiveX コントロールの設定

特定の **ActiveX** コントロールに設定できるプロパティは、**ActiveX** コントロール自体によって決定されます。すべてのプロパティにはデフォルト値があります。

プロパティを設定する前に **ActiveX** コントロールを InTouch ウィンドウに貼り付ける必要があります。

- ウィンドウに **ActiveX** コントロールを貼り付けると、Calendar1 などのデフォルトのコントロール名が生成されます。コントロール名はスクリプトで参照できます。スクリプトを実行するには、開いているアプリケーション ウィンドウで **ActiveX** コントロールが実行されている必要があります。
- **ActiveX** コントロールプロパティは InTouch タグ変数に割り当てることができます。各プロパティタイプを対応する InTouch タグ変数タイプに割り当てする必要があります。

ActiveX コントロールには、[コントロール名]、[プロパティ]、および [イベント] という 3 つの標準タブがあります。



[イベント] タブを使用すると、ユーザーがマウスをダブルクリックしたときなど、スクリプトを使用できるコントロール イベントに割り当てることができます。

その他のタブおよび設定内容はコントロールによって固有であり、プロパティにより異なります。たとえば、コントロールの中には色およびフォントの設定が必要であるものがありますが、他のコントロールにはこれらのプロパティが存在しない場合もあります。

ActiveX コントロールの命名

以下の操作を実行すると、コントロールの新しいインスタンスが作成され、固有の名前が与えられます。

- [複製] を選択します。
- [切り取り] または [コピー] を選択し、[貼り付け] を選択します。

- [ウィンドウに名前を付けて保存] を選択します。
- [元に戻す] をクリックして、[やり直し] をクリックした場合
- コントロールを含むウィンドウをインポートした場合

ActiveX コントロールには、固有の名前を付ける必要があります。コントロールにすでに存在している名前を使用しようとすると、エラーメッセージが表示されます。

ActiveX コントロールに名前を付けるには

1. ActiveX コントロールを開発ウィンドウに貼り付けます。
2. コントロールをダブルクリックします。コントロールの [プロパティ] ダイアログ ボックスが表示されます。
3. [コントロール名] タブをクリックし、その ActiveX コントロールの名前を [コントロール名] ボックスに入力します。

ActiveX コントロールの標準操作

ActiveX コントロールの標準操作は、他の InTouch オブジェクトと同様に実行できます。詳細については、「[すべてのオブジェクトの特殊操作](#)」を参照してください。

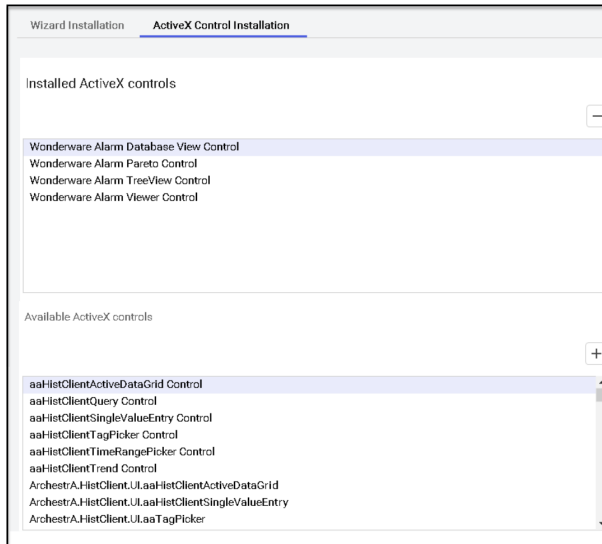
ActiveX コントロールのインストールとアンインストール

コンピュータにすでに ActiveX コントロールがインストールされている場合でも、WindowMaker にインストールして InTouch HMI に通知する必要があります。

コントロールは WindowMaker のリストから消去されても、コンピュータからは削除されません。単にメモリから消去されて、機能しなくなるだけです。

ActiveX コントロールをインストールするには

1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[ウィザード/ActiveX] をクリックします。
[ウィザード/ActiveX] 画面が表示されます。
3. [ActiveX コントロール] タブをクリックします。[インストールされた ActiveX コントロール] プロパティ シートが表示されます。



4. [インストールされた **ActiveX** コントロール] リストで、[インストール可能な **ActiveX** コントロール] リストからインストールするコントロールを選択し、[+] アイコンをクリックします。

ヒント: 複数のコントロールを選択するには、Shift キーまたは Ctrl キーを使用します。

5. [保存] をクリックします。

ActiveX コントロールを削除するには

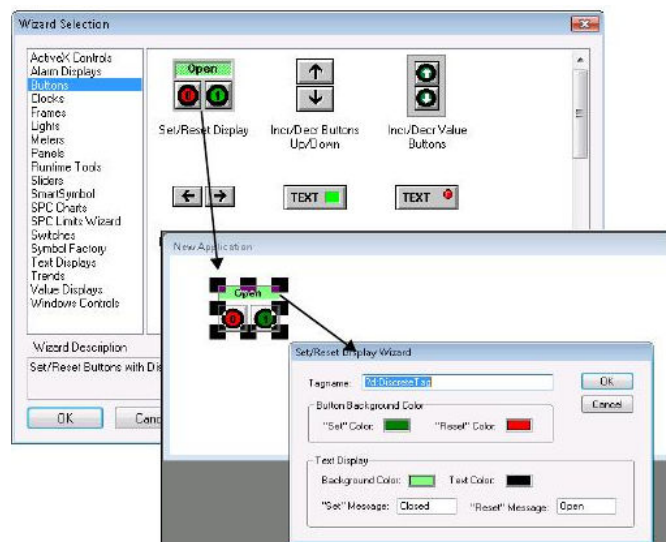
1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[ウィザード/ActiveX] をクリックします。[ウィザード/ActiveX] 画面が表示されます。
3. [ActiveX コントロール] タブをクリックします。[インストールされた **ActiveX** コントロール] プロパティ シートが表示されます。
4. [インストールされた **ActiveX** コントロール] リストで、アプリケーションから削除するコントロールを選択し、[-] アイコンをクリックします。

ヒント: 複数のコントロールを選択するには、Shift キーまたは Ctrl キーを使用します。

5. コントロールを削除するには、[はい] をクリックします。コントロールは [インストール可能な **ActiveX** コントロール] リストに移動されます。
6. [保存] をクリックします。

章 13 ウィザード

ウィザードは、アプリケーション用に選択、配置、および設定するだけで使用できる事前設計、事前構築、および事前プログラムされたオブジェクトです。



ウィザードの操作

ウィザードを使用すると、オブジェクトの個々のコンポーネントを描画したり、オブジェクトの値範囲を入力したり、オブジェクトをアニメーション化したりするために時間を費やす必要はありません。

- [ウィザード選択] ダイアログからウィザードを選択します。
- ウィザードは設定ダイアログ ボックス内にタグおよび **QuickScript** を入力することによって設定します。
- 選択したウィザードをウィンドウに貼り付けて、ダブルクリックすると、[設定] ダイアログ ボックスが表示されます。

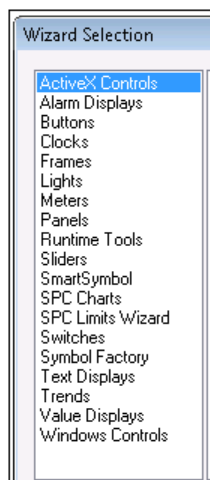
たとえば、スライダ ウィザードでは、有効なタグ名、スライダの移動範囲の最小値と最大値、塗りつぶし色などの項目を設定できます。必要な設定情報を入力すると、ウィザードは使用できるようになります。

また、背後で行われるようなタイプの操作を実行する独自の複合ウィザードを開発することもできます。これらの操作には、完全な表示ウィンドウの作成、データベースの作成や変換、**AutoCAD** 図面のインポート、および他のアプリケーションの設定などが含まれます。

独自のウィザードを作成する前に、プログラミングが不要なウィザード機能を提供する産業用グラフィックを調べてください。

ウィザードのタイプ

ウィザードのカテゴリは、[ウィザードの選択] ダイアログ ボックスに表示されます。



トレンドオブジェクトおよびウィンドウ コントロール ウィザードは、固有のパラメータを持つ特別なウィザードです。詳細については、「[トレンドオブジェクト](#)」および「[ウィンドウ コントロール ウィザード](#)」を参照してください。

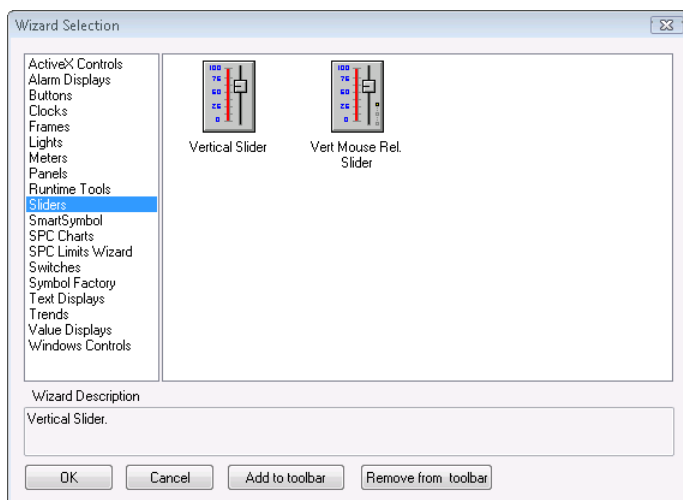
ツールバーへのウィザードの追加

頻繁に使用するウィザードをウィザード／ActiveX ツールバーに追加すると、素早く利用できるようになります。

ウィザードをウィザードツールバーに追加するには

1. [描画] メニューの [挿入] グループで [ウィザード] をクリックします。

[ウィザードの選択] ダイアログ ボックスが表示されます。



2. 左ペインで、[スライダ] などのウィザードカテゴリを選択します。
3. 右ペインでウィザードを選択し、[ツールバーに追加] をクリックします。ウィザード ボタンがツールバーに表示されます。

ウィザードをツールバーから削除するには

1. [描画] メニューの [挿入] グループで [ウィザード] をクリックします。

[ウィザードの選択] ダイアログ ボックスが表示されます。

2. [ツールバーから削除] をクリックします。[削除するウィザード] ダイアログ ボックスが表示されます。
3. 削除するウィザードを選択して、[OK] をクリックします。

ウィザード インスタンスの貼り付け

ウィザードのインスタンスはウィンドウに配置できます。

ウィザードをウィンドウに配置するには

1. [描画] メニューの [挿入] グループで [ウィザード] をクリックします。
[ウィザードの選択] ダイアログ ボックスが表示されます。
2. 左ペインで、ウィザード カテゴリを選択します。
3. 右ペインで、ウィザードを選択します。
4. [OK] をクリックします。

ダイアログ ボックスが閉じ、カーソルがコーナー シンボルに変わります。

5. ウィザードを配置する位置をクリックします。

ウィザードの設定

ウィザードをアプリケーションに配置した後で、ウィザードをダブルクリックしてプロパティを設定します。選択したウィザードに対してカスタムであるプロパティ ダイアログ ボックスが表示されます。

特定のタイプの各ウィザードの詳細については、ウィザードのヘルプを参照してください（ヘルプが用意されている場合）。

ウィザードの標準操作の実行

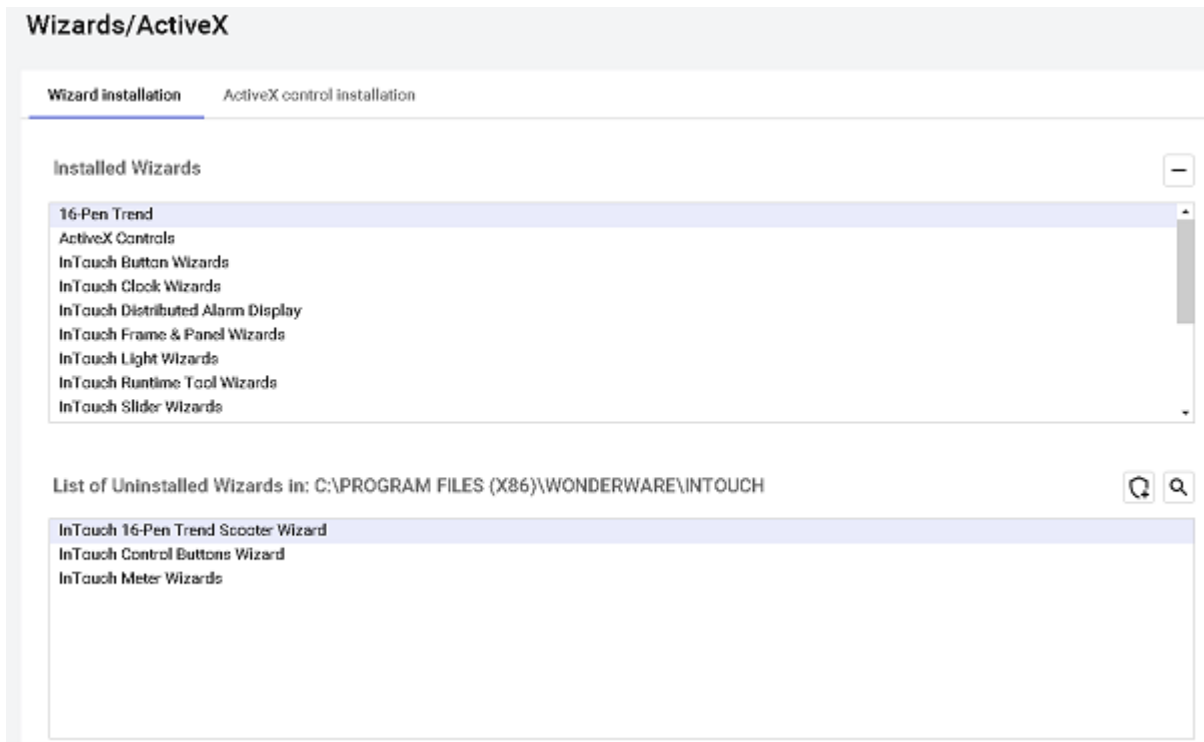
ウィザードの切り取り、コピー、貼り付け、削除、および更新を他のオブジェクトと同じ方法で、同じ結果で実行できます。

ウィザードのインストールとアンインストール

ウィザードを WindowMaker にインストールして、アプリケーションで使えるようにする必要があります。ウィザードは WindowMaker のリストから消去されても、コンピュータからは削除されません。

ウィザードをインストールするには

1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[ウィザード/ActiveX] をクリックします。
[ウィザード/ActiveX] 設定画面が表示されます。



3. [未インストールのウィザードリスト] から選択し、[+] アイコンをクリックします。
4. [保存] をクリックします。

ウィザードを削除するには

1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[ウィザード/ActiveX] をクリックします。
[ウィザード/ActiveX] 設定画面が表示されます。
3. [インストールされているウィザード] リストで、削除するウィザードを選択し、[-] アイコンをクリックします。

ヒント: Shift キーを押しながらクリックするか、Ctrl キーを押しながらクリックすると、複数選択ができます。

ウィザードは、[未インストールのウィザードリスト] に表示されます。

4. [保存] をクリックします。

別のディレクトリからウィザードをインポートするには

1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[ウィザード/ActiveX] をクリックします。[ウィザード/ActiveX] 画面が表示されます。
3. [検索] アイコンをクリックします。[フォルダの参照] ダイアログ ボックスが表示されます。
4. インストールするウィザードを含むディレクトリを参照して、[OK] をクリックします。[ウィザード] ダイアログ ボックスが再び表示され、インポートされたウィザードが [未インストールのウィザードリスト] に表示されます。

トレンド オブジェクト

トレンド オブジェクトは、タグの値を時間経過でチャートにするウィザードです。

トレンド オブジェクトには 3 つの主なタイプがあります。

- リアルタイムトレンドはリアルタイムで最大 4 つまでのタグ変数をチャートにします。
- 履歴トレンドは過去の期間で最大 8 つのタグ変数をチャートにします。
- 16 ペントレンドは最大 16 つまでのタグ変数のリアルタイムおよび履歴データをチャートにします。

ウィンドウに配置できるトレンド オブジェクト、リアルタイムまたは履歴の数に制限はありません。

以下のアイテムを使用して、トレンド オブジェクトを設定します。

- 時間範囲
- 値範囲
- グリッド線の精度
- 時間および値の目盛り位置
- ペンと色

トレンドウィザードを使用する前に、各タグ変数を追跡するためのログ記録を有効にして、InTouch アプリケーション内でのログ記録も有効にする必要があります。

タグ変数のログ記録を有効にするには

1. タグ変数ディクショナリから、タグ変数を選択して、[データ ログ] を選択します。
2. これを実行していない場合は、InTouch アプリケーションでログ記録を有効にします。
 - a. WindowMaker を開きます。
 - b. [ファイル] メニューで、[設定] をポイントし、[履歴ログ] をクリックします。
[履歴ログ] 画面が表示されます。
 - a. [履歴ログ実行] チェック ボックスまたは [Historian へのストレージを有効にする] チェック ボックスをオンにして、[OK] をクリックします。

トレンド オブジェクトの設定および使用の詳細については、「[タグデータのトレンド](#)」を参照してください。

ウィンドウ コントロール ウィザード

ウィンドウ コントロール ウィザードは、ドロップダウンリスト、コンボ ボックス、オプション選択、またはチェック ボックスなどのユーザー インターフェイス オブジェクトです。

ウィンドウ コントロール ウィザードを使用して、事前定義の選択肢セットをユーザーに提供します。たとえば、プロセス、レシピ、またはオペレータ ID のドロップダウンリストを作成できます。また、指定したコントロールを有効または無効にすることもできます。ドロップダウンリストの内容をロードおよび変更することもできます。

ウィンドウ コントロール ウィザードのランタイム プロパティは、アニメーション リンクの式ではなく QuickScript 関数を通じてアクセスされます。

ウィンドウ コントロール ウィザードには、InTouch タグ変数ドットフィールドと同様のプロパティがあり、読み取り／書き込み、または読み取り専用として設定できます。プロパティには、開発時やランタイム時にアクセスできるものもあります。これらのプロパティは「<コントロール名>.x」で識別され、xはプロパティを示します。

たとえば、ウィンドウ コントロールの [Visible] プロパティの値が 0 の場合、コントロールはウィンドウに表示されません。InTouch タグ変数と同様に、[Value] はウィンドウ コントロール ウィザードのデフォルトプロパティです。

注記: 産業用グラフィックを使用している場合は、より堅牢で柔軟な .NET ベースのウィンドウ コントロールセットが利用できます。

ウィンドウ コントロールの作成と設定

ウィンドウ コントロール ウィザードを作成および設定する際は、以下の点に注意してください。

- ウィンドウ コントロール ウィザードは互いにオーバーラップできません。
- ランタイム時、ウィンドウ内のその他のオブジェクトにウィンドウ コントロール ウィザードが重なる場合、ウィンドウ コントロール ウィザードは常に一番上に表示されます。
- 各ウィンドウ コントロール ウィザードには固有のコントロール名があり、タグ変数カウントには追加されません。
- リスト ボックスやコンボ ボックスに割り当てられているタグ変数の初期値では、リスト ボックスやコンボ ボックスの値は初期化されません。SetPropertyX QuickScript 関数をスクリプトで使用して、デフォルト値と異なる初期値を割り当てる必要があります。

ヒント: その他のウィザードと同様に、ウィンドウ コントロール ウィザードはウィンドウに貼り付けられます。読みやすさを最大限に向上させるには、ウィンドウ コントロールの背景色にグレーを選択してください。背景色をグレーにできない場合、グレーのパネル ウィザードをウィンドウ コントロール ウィザードの背面に置いてください。

各ウィンドウ コントロール ウィザードに対して、英数字コントロール名を指定する必要があります。最初の文字は英字となります。アンダースコアは使用できますが、その他の特殊文字は使用できません。たとえば、「Checkbox_1」は使用できますが、「Checkbox#1」は使用できません。

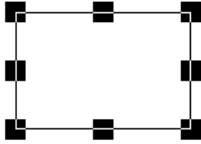
ウィンドウ コントロール ウィザードは、InTouch QuickScript を使用して設定できます。

テキスト ボックスの作成

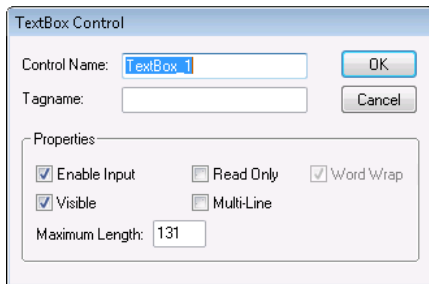
オペレーターは、アプリケーション内でテキスト ボックスを使用してテキスト文字列を入力できます。

テキスト ボックスを作成および設定するには

1. テキスト ボックスを作成して、配置します。以下の操作を行います。
 - a. [ウィザードの選択] ダイアログ ボックスで、[ウィンドウ コントロール ウィザード] を選択します。コントロール ウィザードアイコンが表示されます。
 - b. テキスト ボックスアイコンをダブルクリックします。アプリケーション ウィンドウが再表示され、カーソルが左コーナー シンボルに変わります。
 - c. アプリケーション ウィンドウ内をクリックして、ウィザードを配置します。コーナーに黒い大きなハンドルが付いたテキスト ボックス ウィザードが表示されます。



- a. アプリケーションに適合するようにウィザードをドラッグして、サイズ変更します。
2. ウィザードをダブルクリックします。[テキストボックス コントロール] ダイアログ ボックスが表示されます。



3. ダイアログ ボックスを設定します。以下の操作を行います。
 - a. `TextBox_1` などのコントロール名を [コントロール名] ボックスに入力します。
 - b. `New_Value` などのメモリ メッセージ型タグ変数を [タグ変数] ボックスに入力します。
 - c. [プロパティ] 領域で、[入力可能] および [表示可能] を選択します。
4. [OK] をクリックします。[テキストボックス コントロール] ダイアログ ボックスが閉じます。

リスト ボックスの作成

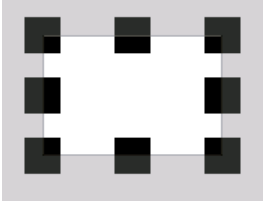
ユーザーがアイテムをオプション リストから選択することを可能にするアプリケーションを作成できます。

リスト ボックスをアプリケーションに追加する際、画面にコントロールを配置し、プロパティを設定してリストを設定して、必要なスクリプトを記述します。

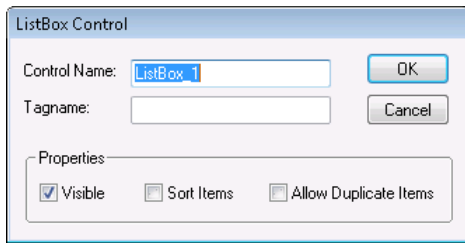
リスト ボックスはファイルから、またはランタイムでキーボード入力からアイテムと共にロードできます。詳細については、「[ウィンドウ コントロールのスクリプト記述](#)」を参照してください。

リスト ボックスを作成するには

1. リスト ボックス コントロールを作成して、配置します。以下の操作を行います。
 - a. [ウィザードの選択] ダイアログ ボックスで、[ウィンドウ コントロール] を選択します。
 - b. リスト ボックス アイコンをダブルクリックします。アプリケーション ウィンドウが再表示され、カーソルが左コーナー シンボルに変わります。
 - c. アプリケーション ウィンドウ内をクリックして、コントロールを配置します。リスト ボックス コントロールが表示されます。



2. コントロールをダブルクリックします。[リストボックス コントロール] ダイアログ ボックスが表示されます。



3. コントロールを設定します。以下の操作を行います。
 - a. `TextBox_1` などのコントロール名を [コントロール名] ボックスに入力します。
 - b. `LB1_Value` などのメモリ メッセージ型タグ変数を [タグ変数] ボックスに入力します。
 - c. [プロパティ] 領域で、コントロールの表示方法と機能を設定します。
4. [OK] をクリックします。

コンボ ボックスの作成

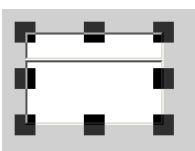
ユーザーがアイテムをコンボ ボックスを備えたオプション リストから選択することを可能にするアプリケーションを作成できます。コンボ ボックスは、テキスト ボックスとリスト ボックスを組み合わせたウィンドウ コントロールです。

コンボ ボックスをアプリケーションに追加する際、画面にコントロールを配置し、プロパティを設定してコンボ ボックスを設定して、必要なスクリプトを記述します。

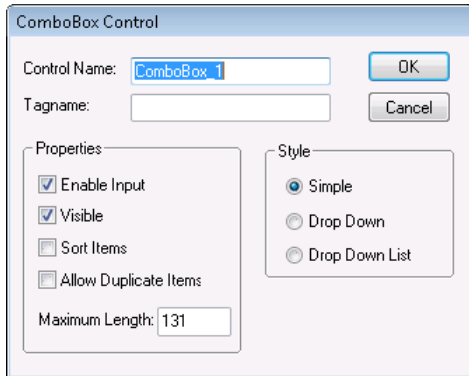
コンボ ボックスはファイルから、またはランタイムでキーボード入力からアイテムと共にロードできます。詳細については、「[ウィンドウ コントロールのスクリプト記述](#)」を参照してください。

コンボ ボックスを作成するには

1. コンボ ボックス コントロールを作成して、配置します。以下の操作を行います。
 - a. [ウィザードの選択] ダイアログ ボックスで、[ウィンドウ コントロール] を選択します。
 - b. コンボ ボックス アイコンをダブルクリックします。アプリケーション ウィンドウが再表示され、カーソルが左コーナー シンボルに変わります。
 - c. アプリケーション ウィンドウ内をクリックして、コントロールを配置します。コンボ ボックス コントロールが表示されます。



2. コントロールをダブルクリックします。[コンボ ボックス コントロール] ダイアログ ボックスが表示されます。



3. コントロールを設定します。以下の操作を行います。
 - a. *ComboBox_1* などのコントロール名を [コントロール名] ボックスに入力します。
 - b. *CB1_Value* などのメモリ メッセージ型タグ変数を [タグ変数] ボックスに入力します。
 - c. [プロパティ] 領域で、コントロールの表示方法と機能を設定します。
 - d. [スタイル] 領域で、コンボ ボックスのタイプを選択します。
4. [OK] をクリックします。

チェック ボックスの作成

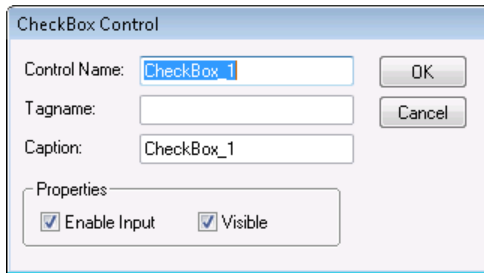
オペレータは、チェック ボックス コントロールを使用してオプションを選択できます。

チェック ボックス コントロールを設定するには

1. チェック ボックスを作成します。以下の操作を行います。
 - a. [ウィザードの選択] ダイアログ ボックスで、[ウィンドウ コントロール ウィザード] を選択します。コントロール ウィザード アイコンが表示されます。
 - b. チェック ボックス アイコンをダブルクリックします。アプリケーション ウィンドウが再表示され、カーソルが左コーナー シンボルに変わります。
 - c. アプリケーション ウィンドウ内をクリックして、ウィザードを配置します。チェックボックス ウィザードが表示されます。



- a. ウィザードをドラッグして、サイズ変更します。
2. ウィザードをダブルクリックします。[チェック ボックス コントロール] ダイアログ ボックスが表示されます。



3. ウィザードを設定します。以下の操作を行います。
 - a. コントロール名を入力します。
 - b. 論理型タグ変数を入力するか、または空白のタグ変数ボックスをダブルクリックして、**[タグ変数を選択してください]** ダイアログボックスを表示して、タグ変数を選択します。
 - c. ボタンの表面に表示されるキャプションを入力します。
4. **[OK]** をクリックします。

ラジオ ボタン グループの作成

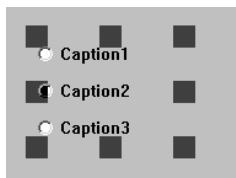
ユーザーが複数の選択肢から1つを選択する必要がある場合に、ラジオ ボタンを使用します。ユーザーがオプションを選択したら、以前に選択されたオプションは選択解除されます。

ユーザー用のオプションをラジオ ボタンのグループとして作成します。各ラジオ ボタンにはキャプションがあり、スクリプトに固有の値を提供します。

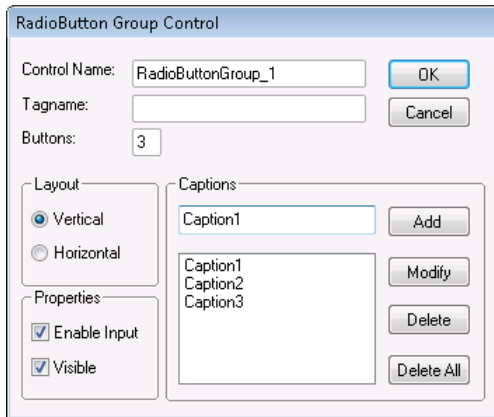
整数型タグ変数は、ラジオ ボタン コントロールにのみ割り当てることができます。

ラジオ ボタン グループを作成するには

1. ラジオ ボタン コントロール ウィザードを作成します。以下の操作を行います。
 - a. **[ウィザードの選択]** ダイアログボックスで、**[ウィンドウ コントロール ウィザード]** を選択します。コントロール ウィザードアイコンが表示されます。
 - b. ラジオ ボタン アイコンをダブルクリックします。アプリケーション ウィンドウが再表示され、カーソルが左コーナー シンボルに変わります。
 - c. アプリケーション ウィンドウ内をクリックして、ウィザードを配置します。3つのラジオ ボタンと共にラジオ グループ コントロール ウィザードが表示されます。



- a. アプリケーションに適合するようにウィザードをドラッグして、サイズ変更します。
2. ウィザードをダブルクリックします。**[ラジオ ボタン グループ コントロール]** ダイアログボックスが表示されます。



3. ウィザードを設定します。以下の操作を行います。
 - a. コントロール名を入力します。
 - b. 整数型タグ変数を入力して、このコントロールにリンクします。
 - c. 表示するボタンの数を入力します。
 - d. 各ボタンのキャプションを入力します。
 - e. レイアウトおよびプロパティを設定します。
4. [OK] をクリックします。

ウィンドウ コントロールのスクリプト記述

スクリプトで **QuickScript** 関数を使用して、以下の操作を行えます。

- コントロールの値の取得または設定
- コントロールの有効化、無効化、または非表示
- コンボ ボックス、リスト ボックス、テキスト ボックス、およびチェック ボックスのアイテムの操作

プロパティに基づいて、ランタイム プロパティは読み取り／書き込みまたは読み取り専用のいずれかにできます。

GetPropertyX() 関数および **SetPropertyX()** 関数を使用して、これらのプロパティを制御または取得します。

コントロールの値の取得または設定

.Value プロパティは、すべての InTouch ウィンドウ コントロール ウィザードのデフォルト プロパティです。

このプロパティに行われる変更は、InTouch タグ変数とウィンドウ コントロール ウィザードで同期されます。

.Value ドットフィールド

すべての InTouch ウィンドウ コントロール ウィザードのデフォルト プロパティです。このプロパティに行われる変更は、InTouch タグ変数とウィンドウ コントロール ウィザードで同期されます。

カテゴリ

ウィンドウ コントロール

使用法

M、I、および D は、GetProperty 関数および SetProperty 関数のメモリ型、整数型、および論理型バージョンです。

```
[ErrorNumber=]GetPropertyM("ControlName[.Value]", Tagname);  
[ErrorNumber=]SetPropertyM("ControlName[.Value]", Value);  
[ErrorNumber=]GetPropertyI("ControlName[.Value]", Tagname);  
[ErrorNumber=]SetPropertyI("ControlName[.Value]", Value);  
[ErrorNumber=]GetPropertyD("ControlName[.Value]", Tagname);  
[ErrorNumber=]SetPropertyD("ControlName[.Value]", Value);
```

パラメータ

ControlName

ウィンドウ コントロールの名前（例：ChkBox_4）

Tagname

プロパティの値を書き込む先のタグ変数です。

[.Value]

このプロパティはオプションです。指定されていない場合、関数は .Value プロパティが使用されていると想定します。

Value

関数が処理されるときに書き込まれるプロパティ値を保持する、書き込まれる実際の値または有効な InTouch タグ変数（書き込まれるプロパティと同じタイプ）。

備考

リスト ボックスまたはコンボ ボックスに割り当てられているタグ変数の初期値は、リスト ボックスまたはコンボ ボックスの値を初期化するために使用することはできません。

このドットフィールドは、開発時およびランタイム時ともに読み取り／書き込み可能です。タグ変数をリスト ボックスまたはコンボ ボックスに関連付けることによって .Value ドットフィールドにアクセスする場合は読み取り専用となります。 .Value ドットフィールドがチェック ボックス、ラジオ ボタン、またはテキスト ボックスに割り当てられる場合、読み取り／書き込みとなります。開発時に指定した値がランタイムでのデフォルトとなります。

データタイプ

テキスト ボックス、リスト ボックス、およびコンボ ボックスの場合はメッセージ型（読み取り／書き込み）

ラジオ ボタンの場合は整数型（読み取り／書き込み）

チェック ボックスの場合は論理型（読み取り／書き込み）

適用対象

テキスト ボックス、リスト ボックス、コンボ ボックス、チェック ボックス、ラジオ ボタン

例

以下のステートメントは、ラジオ ボタン オブジェクト RadioButton_1 の .Value ドットフィールドの値を 4 に設定します。

```
SetPropertyI( "RadioButton_1.Value", 4 );
```

参照項目

GetPropertyM()、SetPropertyM()、GetPropertyI()、SetPropertyI()、GetPropertyD()、SetPropertyD()

ユーザー入力用のコントロールの有効化または無効化

.Enabled プロパティを使用して、コントロール オブジェクトをユーザー生成イベントに応答できるようにするかどうか指定します。

.Enabled ドットフィールド

コントロール オブジェクトが、ユーザーによって生成されたイベントに反応するかどうかを指定します。

カテゴリ

ウィンドウ コントロール

使用法

```
[ErrorNumber=] GetPropertyD("ControlName.Enabled",  
Tagname);  
[ErrorNumber=] SetPropertyD("ControlName.Enabled",  
Discrete);
```

パラメータ

ControlName

ウィンドウ コントロールの名前。例：ChkBox_4

Tagname

要求されたプロパティを保持する論理型タグ変数

Discrete

関数が処理されるときに書き込まれる値を保持する論理型の値または論理型のタグ変数論理値：

0 = コントロール使用不可

1 = コントロール使用可能

備考

このプロパティは開発時およびランタイム時ともに読み取り／書き込みです。

データタイプ

論理型（読み取り／書き込み）

適用対象

テキスト ボックス、リスト ボックス、コンボ ボックス、チェック ボックス、ラジオ ボタン

例

以下のステートメントは、「ListBox_1」という名前のリスト ボックス オブジェクトを無効にします。

```
SetPropertyD("ListBox_1.Enabled", 0);
```

参照項目

GetPropertyD()、SetPropertyD()

ウィンドウ コントロールの動的非表示

.Visible プロパティを使用して、ウィンドウ コントロールをウィンドウに表示するかどうかを決定します。

.Visible ドットフィールド

ウィンドウ コントロールを表示するかどうかを決定します。

カテゴリ

ウィンドウ コントロール

使用法

```
[ErrorNumber=]GetPropertyD("ControlName.Visible", Tagname);  
[ErrorNumber=]SetPropertyD("ControlName.Visible", Number);
```

パラメータ

ControlName

ウィンドウ コントロールの名前。例 : ListBox_1

Tagname

関数が処理されるときにプロパティ値を保持するタグ変数（返されるものと同じタイプ）

Number

関数が処理されるときに書き込まれる値を保持する論理型の値または論理型のタグ変数論理値 :

0 = コントロールを非表示

1 = コントロールを表示

備考

このプロパティは開発時およびランタイム時ともに読み取り／書き込みです。

データタイプ

論理型（読み取り／書き込み）

有効値

適用対象

テキスト ボックス、リスト ボックス、コンボ ボックス、チェック ボックス、ラジオ ボタン

例

以下のステートメントにより、"TextBox_1" という名前のテキスト ボックスが非表示になります。
SetPropertyD("TextBox_1.Visible",0);

参照項目

GetPropertyD()、SetPropertyD()

コンボ ボックスのアイテムの追加と削除

以下のスクリプト関数を使用して、コンボ ボックスおよびリストからアイテムを追加したり、削除したりします。

スクリプト関数	効果
<code>wcAddItem()</code>	アイテムをリスト ボックスやコンボ ボックスのリストの最後に追加します。ソートが有効になっている場合、リストはアイテムの追加後にソートされます。
<code>wcInsertItem()</code>	アイテムをリスト ボックスまたはコンボ ボックスのリストの指定した位置に追加します。
<code>wcDeleteItem()</code>	アイテムをリスト ボックスまたはコンボ ボックスのリストの指定した位置から削除します。
<code>wcDeleteSelection()</code>	現在選択されているアイテムをリストまたはコンボ ボックスから削除します。
<code>wcClear()</code>	リストまたはコンボ ボックスからすべてのアイテムを削除します。

wcAddItem() 関数

アイテムをリスト ボックスやコンボ ボックスのリストの最後に追加します。ソートが有効になっている場合、リストはアイテムの追加後にソートされます。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcAddItem("ControlName", "MessageTag");
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

MessageTag

表示されるメッセージ文字列。実際の文字列またはメッセージ型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックスとコンボ ボックス

例

以下のステートメントは、リスト ボックスが含まれているウィンドウ（開く時ウィンドウ QuickScript を使用）を開くと、メッセージ文字列の内容をそのリスト ボックスに追加します。

```
wcAddItem("ListBox_1", "Chocolate");  
wcAddItem("ListBox_1", "Vanilla");  
wcAddItem("ListBox_1", "Strawberry");
```

参照項目

wcInsertItem()

wcInsertItem() 関数

指定した文字列をリスト ボックスまたはコンボ ボックスのリストの指定した位置に挿入します。
wcAddItem() 関数とは異なり、ソートされたリスト ボックスまたはコンボ ボックスとして作成された場合でも、wcInsertItem() 関数はリストをソートしません。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcInsertItem("ControlName", ItemPosition, "Message");
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

ItemPosition

アイテムが追加される位置を示す数。このパラメータが -1 の場合、文字列はリストの最後に追加されます。数値または整数型タグ変数です。

Message

ItemPosition で指定した位置に挿入する文字列が含まれます。実際の文字列またはメッセージ型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックスとコンボ ボックス

例

アクション スクリプトが実行されると、以下のステートメントにより、"Blueberry" という新しいアイテムが、リスト ボックスの先頭から 4 番目の場所に挿入されます。

```
wcInsertItem("ListBox_1", 4, "Blueberry");
```

参照項目

wcAddItem()

wcDeleteItem() 関数

アイテムをリスト ボックスまたはコンボ ボックスのリストの指定した位置から削除します。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcDeleteItem("ControlName", ItemPosition);
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

ItemPosition

アイテムの位置に対応する数。数値または整数型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロール エラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックスとコンボ ボックス

例

アクション スクリプトが実行されると、以下のステートメントにより、リストの 3 番目のアイテムが削除されます。

```
wcDeleteItem("ListBox_1", 3);
```

wcDeleteSelection() 関数

現在選択されているアイテムをリストから削除します。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber =]wcDeleteSelection("ControlName");
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロール エラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックスとコンボ ボックス

例

アクション スクリプトが実行されると、以下のステートメントにより、リストボックスで現在選択されているアイテムが削除されます。

```
wcDeleteSelection("ListBox_1");
```

wcClear() 関数

リスト ボックスまたはコンボ ボックスからすべてのアイテムを削除します。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcClear("ControlName");
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックスとコンボ ボックス

例

アクション スクリプトが実行されると、以下のステートメントにより、リスト ボックスのすべてのアイテムが削除されます。

```
wcClear("ListBox_1");
```

ファイルに対するリスト アイテムのロードと保存

以下のスクリプト関数を使用して、コンボ ボックスまたはリストのアイテムをファイルに対してロードしたり、保存したりします。

スクリプト関数 効果

wcLoadList()	新しいアイテムを持つリスト ボックスまたはコンボ ボックスをファイルからロードします。
wcSaveList()	リスト ボックスまたはコンボ ボックスの内容をファイルに保存します。

wcLoadList() 関数

新しいアイテムを持つリスト ボックスまたはコンボ ボックスをファイルからロードします。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcLoadList("ControlName", "Filename");
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

Filename

ファイルの名前が含まれています。メッセージパラメータの一部として完全なパス名が提供されない場合、この関数はアプリケーションディレクトリでメッセージファイルを探します。実際の文字列またはメッセージ型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

外部ファイルを使用して、リスト ボックスおよびコンボ ボックスの内容を入力する場合、特定の書式設定に従い、特定の情報を含む必要があります。書式：

ControlType, ListCount

ListItem, ItemIndex

ListItem, ItemIndex

::

::

ListItem, ItemIndex

ControlType は COMBOBOX または LISTBOX です。

たとえば、リスト ファイルをコンボ ボックスにロードする場合で、その中に選択できる 3 つのアイテムが含まれており、それらのアイテム データが割り当てられていない場合などがあります。ファイル形式には、以下の種類があります。

COMBOBOX, 3

Chocolate, 0

Vanilla, 0

Strawberry, 0

COMBOBOX は、コントロール タイプです。リスト カウントは、Chocolate、Vanilla、Strawberry の 3 つです。Chocolate は、最初のアイテムまたはポジション 1 として、Vanilla はポジション 2、Strawberry はポジション 3 として一覧されています。各アイテムはデータ値 0 を持ちます。

アイテム データの詳細については、「[wcSetItemData\(\) 関数](#)」を参照してください。

適用対象

リスト ボックスとコンボ ボックス

例

以下のステートメントは、適切に書式設定されたリスト (c:\wclist.txt) をコンボ ボックスにロードします。

```
wcLoadList("Combobox_1", "c:\wclist.txt");
```

参照項目

wcAddItem()、wcSaveList()

wcSaveList() 関数

リスト ボックスまたはコンボ ボックスのアイテムをファイルに保存します。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcSaveList("ControlName","Filename");
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

Filename

ファイルの名前が含まれています。ファイルが存在しない場合、作成されます。実際の文字列またはメッセージ型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロール エラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックスとコンボ ボックス

例

アクションスクリプトが実行されると、以下のステートメントにより、リストボックスの現在のアイテムがファイル (c:\newlist.txt) に保存されます。

```
wcSaveList("ListBox_1", "c:\newlist.txt");
```

参照項目

wcLoadList()、wcSetItemData()

コンボ ボックスまたはリストのアイテムの検索

wcFindItem() 関数を使用して、指定したアイテムをリスト ボックスまたはコンボ ボックスで検索します。アイテムが検索されたら、この関数は対応する位置を 4 番目のパラメータとして整数型タグ変数に返します。

wcFindItem() 関数

指定されたメッセージ文字列に一致するリスト ボックスまたはコンボ ボックスの最初のアイテムの対応する位置を決定します。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcFindItem ("ControlName", "MessageTag", CaseSens, Tagname);
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

MessageTag

比較されるメッセージ文字列。実際の文字列またはメッセージ型タグ変数です。

CaseSens

文字列比較のタイプを決定します。論理値またはタグ変数を指定となります。以下値が有効です。

0 = 大文字と小文字は区別されません。

1 = 大文字と小文字は区別されます。

Tagname

一致するアイテムの位置が返される整数型タグ変数。一致するアイテムが見つからない場合は、-1 が返されます。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックス、コンボ ボックス

例

ListBox_1 が "ItemA"、"ItemB"、および "ItemC" を含むリスト ボックスであると仮定すると、この関数は以下の値を結果として返します。

```
wcFindItem("ListBox_1", "ItemB", 0, Result);
```

2 を返します

```
wcFindItem("ListBox_1", "Itemb", 1, Result);
```

-1 を返します

```
wcFindItem("ListBox_1", "itemc", 0, Result);
```

3 を返します

```
wcFindItem("ListBox_1", "XYZ", 0, Result);
```

-1 を返します

コンボ ボックスまたはリストのアイテム インデックスの操作

以下のドットフィールドをリスト ボックスまたはコンボ ボックスのアイテム インデックスと共に使用します。

ドットフィールド	効果
.TopIndex	リスト ボックスの先頭アイテムの整数インデックス
.NewIndex	wcAddItem() 関数や wcInsertItem() 関数を使用して、リスト ボックスまたはコンボ ボックスに追加された最後のアイテムの整数インデックス (タグ変数)
.ListIndex	リストで現在選択されているアイテムのインデックス (タグ変数または数)

.TopIndex ドットフィールド

リスト ボックスの先頭アイテムの整数インデックスを設定するか読み取ります。

カテゴリ

ウィンドウ コントロール

使用法

```
[ErrorNumber=]GetPropertyI("ControlName.TopIndex", Tagname);  
[ErrorNumber=]SetPropertyI("ControlName.TopIndex", Number);
```

パラメータ

ControlName

ウィンドウ コントロールの名前。例：ListBox_1

Tagname

関数が処理されるときにプロパティ値を保持する整数型タグ変数

Number

リスト ボックスの先頭アイテムを定義するインデックス番号。整数値を提供するリテラル整数値または整数型タグ変数、または式のいずれかです。

備考

このプロパティは、ランタイムでのみ使用できます。

データタイプ

整数型（読み取り／書き込み）

適用対象

リスト ボックス

例

以下のステートメントは、"ListBox_1" というリスト ボックス オブジェクトの TopIndex を 14 に設定します。

```
SetPropertyI( "ListBox_1.TopIndex",14 );
```

参照項目

GetPropertyI()、SetPropertyI()、.ListIndex、.NewIndex

.NewIndex ドットフィールド

wcAddItem() 関数または wcInsertItem() 関数を使用して、リスト ボックスまたはコンボ ボックスに追加された最後のアイテムの整数インデックス（タグ変数）を返します。

カテゴリ

ウィンドウ コントロール

使用法

```
[ErrorNumber=]GetPropertyI("ControlName.NewIndex", Tagname);
```

パラメータ

ControlName

ウィンドウ コントロールの名前。例：ListBox_4

Tagname

リスト ボックスまたはコンボ ボックスに追加された最後のアイテムの整数インデックスを含むタグ変数。空のリストに対しては -1 が返されます。

備考

このプロパティは、ランタイムでのみ使用できます。

データタイプ

整数型（読み取り専用）

適用対象

リスト ボックスとコンボ ボックス

例

以下のステートメントにより、"ListBox_1" というリスト ボックスで最後に追加されたアイテムのインデックスを取得し、その値をメモリ整数型タグ変数 **NewItemIndex** に書き込みます。

```
GetPropertyI("ListBox_1.NewIndex", NewItemIndex);
```

参照項目

GetPropertyI(), wcAddItem(), wcInsertItem(), .ListIndex、.TopIndex

.ListIndex ドットフィールド

リストで現在選択されているアイテムのインデックス（Tagname または Number）を設定または読み取ります。

リスト ボックスを使用している場合、インデックス -1 は現在選択されているアイテムが存在しないことを示します。

コンボ ボックスを使用している場合、インデックス -1 はユーザーが新しいテキストをそのコントロールのテキスト入力フィールドに入力したことを示します。

構文

```
[ErrorNumber=]GetPropertyI("ControlName.ListIndex", Tagname);  
[ErrorNumber=]SetPropertyI("ControlName.ListIndex", Number);
```

パラメータ**ControlName**

ウィンドウ コントロールの名前。例：ListBox_4

Tagname

現在選択されているアイテムのインデックスを書き込む先のタグ変数です。

Number

リストで特定のアイテムを定義するインデックス番号

備考

リストで特定のアイテムを定義するインデックス番号。ListIndex ドットフィールドを使用して、リストまたはコンボ ボックスで現在選択されているアイテムのインデックスを設定または決定します。

このプロパティは、ランタイムでのみ使用できます。

データ タイプ

整数型（読み取り／書き込み）

適用対象

リスト ボックスとコンボ ボックス

例

このステートメントは、"ListBox_1" というリスト ボックスで現在選択されているアイテムのインデックスを取得し、その値をメモリ整数型タグ変数 *MyListBoxIndex* に書き込みます。

```
GetPropertyI( "ListBox_1.ListIndex", MyListBoxIndex );
```

参照項目

GetPropertyI(), SetPropertyI(), .NewIndex、.TopIndex

リスト ボックスまたはコンボ ボックス アイテムのカウント

.ListCount ドットフィールドには、リスト ボックスまたはコンボ ボックスのアイテム数が含まれています。

.ListCount ドットフィールド

リスト ボックスまたはコンボ ボックスのアイテム数を読み取ります。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]GetPropertyI("ControlName.ListCount", Tagname);
```

パラメータ

ControlName

ウィンドウ コントロールの名前。

Tagname

リストのアイテムの整数の数を含む有効なタグ変数

備考

このプロパティは、ランタイムでのみ使用できます。

データタイプ

整数型（読み取り専用）

適用対象

リスト ボックスとコンボ ボックス

例

以下のステートメントにより、*ListBox_1* というリスト ボックスからアイテム数を取得し、その値をメモリ整数型タグ変数 *MyListBoxCount* に書き込みます。

```
GetPropertyI("ListBox_1.ListCount", MyListBoxCount);
```

参照項目

GetProperty()、.ListIndex

リスト アイテムの値の取得または設定

wcGetItemData() 関数を使用して、アイテム インデックスによって識別されるリスト アイテムに関連付けられている整数値を検索します。

wcSetItemData() 関数を使用して、アイテム インデックスで指定されるリストのアイテムに整数値を割り当てます。これにより文字列に数字が割り当てられます。

wcGetItemData() 関数

ItemIndex パラメータで識別されるリスト アイテムに関連付けられている整数値を読み取ります。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcGetItemData("ControlName", ItemIndex, Tagname);
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

ItemIndex

アイテムの位置に対応する数。数値または整数型タグ変数です。

Tagname

実数型または整数型タグ変数の実際の名前。wcGetItemData() 関数は、関数から返されたアイテムに対応する数値をこのタグ変数に配置します。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックスとコンボ ボックス

例

アクション スクリプトが実行されると、以下のステートメントはリスト ボックスの 5 番目のアイテムに関連付けられている数値を取得して、整数型タグ変数 ItemValue に返します。

```
wcGetItemData("ListBox_1", 5, ItemValue);
```

リスト ボックスの 5 番目のアイテムに整数値 4500 が割り当てられると、タグ変数 ItemValue には 4500 が含まれます。

参照項目

wcSetItemData()

wcSetItemData() 関数

アイテム (Number パラメータ) の整数値を ItemIndex パラメータで指定したリストのアイテムに割り当てます。この関数では、文字列に数値を割り当てることができます。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcSetItemData("ControlName", ItemIndex, Number);
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例: ListBox_1 実際の文字列またはメッセージ型タグ変数です。

ItemIndex

編集するリスト アイテムを指定する整数値です。数値または整数型タグ変数です。

Number

アイテム データを表す整数値。数値または整数型タグ変数です。

備考

メモ帳などのプログラムを使用して、アイテムを含む完全なリストを作成して、1つの関数呼び出しを使用してそれらをロードできます。wcSaveList() 関数に従ってリストを書式設定します。

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

wcGetItemData() 関数を使用して、リスト アイテムに関連付けられている値 (アイテム データ) を返します。タグ変数パラメータには、返された数値が含まれます。このパラメータには、実際のデバイスに直接書き込む I/O 整数型タグ変数を使用できます。

例

小麦粉、砂糖、塩を使用するレシピがあるとします。使用する小麦粉の量は 4500 グラム、砂糖と塩はそれぞれ 1500 グラムと 325 グラムです。選択されるレシピ (タグ変数、レシピ名) によって切り替えられるデータ変化スクリプトを使用して、リスト ボックスの各アイテムに値が割り当てられます。

```
wcSetItemData("ListBox_1", 1, 4500); {リスト内の最初のアイテムを (flour)=4500 に設定します}  
wcSetItemData("ListBox_1", 2, 1500); {リスト内の 2 番目のアイテムを (sugar)=1500 に設定します}  
wcSetItemData("ListBox_1", 3, 325); {リスト内の 3 番目のアイテムを (salt)=325 に設定します}
```

参照項目

wcLoadList()、wcSaveList()、wcGetItemData()

リスト アイテムの名前の取得

wcGetItem() 関数を使用して、リスト ボックスまたはコンボ ボックスの対応するアイテム インデックスに関連付けられているアイテム文字列を返します。

wcGetItem() 関数

リスト ボックスまたはコンボ ボックスの ItemIndex に対応するアイテムの内容を含む文字列を返します。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcGetItem("ControlName", ItemIndex, Tagname);
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

ItemIndex

アイテムの位置に対応する数。数値または整数型タグ変数です。

Tagname

メッセージ型タグ変数 **wcGetItem** 関数は、関数から返されたアイテム インデックスに対応するデータをこのタグ変数に配置します。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロール エラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックスとコンボ ボックス

例

アクション スクリプトが実行されると、以下のステートメントは、コンボ ボックスの 10 番目のアイテムの文字列値をメッセージ型タグ変数 **ListSelection** に返します。

```
wcGetItem("Combobox_1", 10, ListSelection);
```

リストの 10 番目のアイテムが "Vanilla" の場合、タグ変数 **ListSelection** には "Vanilla" という文字列が含まれます。

テキスト ボックスの内容のロード

wcLoadText() 関数を使用して、テキスト ボックスの内容をファイルからロードします。**wcSaveText()** 関数を使用して、テキスト ボックスの内容をテキスト ファイルに保存します。

注意：タグ変数に最高文字数が定義されている場合、その文字数だけがテキスト ボックスの内容からタグ変数に割り当てられます。テキスト ボックスにタグ変数が割り当てられていない場合、テキスト ボックスには最大 65,535 文字まで入力できます。

wcLoadText() 関数

テキスト ボックスの内容をファイルの内容で置き換えます。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcLoadText("ControlName", "Filename");
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

Filename

ファイルの名前が含まれています。メッセージパラメータの一部として完全なパス名が提供されない場合、この関数はアプリケーションディレクトリでファイルを検索します。実際の文字列またはメッセージ型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

適用対象

テキスト ボックス

例

以下のステートメントは、テキスト ボックスを含むウィンドウ（開く時ウィンドウ スクリプト）が開くときにテキスト ファイル（c:\InTouch.32\readme.txt）をテキスト ボックスにロードします。

```
wcLoadText("Textbox_1", "c:\InTouch.32\readme.txt");
```

wcSaveText() 関数

テキスト ボックスに含まれる文字列を指定ファイルに保存します。ファイルが存在しない場合、作成されます。ファイルが存在する場合、読み取り／書き込みである必要があります。

カテゴリ

ウィンドウ コントロール

構文

```
[ErrorNumber=]wcSaveText("ControlName", "Filename");
```

パラメータ

ControlName

ウィンドウ コントロール オブジェクトの名前。例：ListBox_1 実際の文字列またはメッセージ型タグ変数です。

Filename

ファイルの名前が含まれています。完全なパス名が提供されていない場合、ファイルはアプリケーションディレクトリに保存されます。ファイルが存在する場合は上書きされます。ファイルが存在しない場合、作成されます。作成されたファイルは後から **wcLoadText()** 関数を使用してテキスト ボックス オブジェクトにロードできます。実際の文字列またはメッセージ型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

適用対象

テキスト ボックス

例

アクションスクリプトが実行されると、以下のステートメントは、テキスト ボックスに入力されている現在の情報を c:\InTouch.32\newtext.txt ファイルに保存します。

```
wcSaveText("Textbox_1", "c:\InTouch.32\newtext.txt");
```

参照項目

wcLoadText()

テキスト ボックスが読み取り専用であるかどうかの確認

.ReadOnly ドットフィールドを使用して、テキスト ボックスの内容が読み取り専用であるか読み取り／書き込みであるかを確定できます。

.ReadOnly ドットフィールド

テキスト ボックスの内容が読み取り専用であるか読み取り／書き込みであるかを確定できます。

カテゴリ

ウィンドウ コントロール

使用法

```
[ErrorNumber=]GetPropertyD("ControlName.ReadOnly", Tagname);
```

パラメータ

ControlName

ウィンドウ コントロールの名前。例 : Textbox_1

Tagname

関数が処理されるとき、プロパティ値を保持する論理型タグ変数

0 = テキスト ボックスの内容は読み取り／書き込みです。

1 = テキスト ボックスの内容は読み取り専用です。

備考

このプロパティは開発時およびランタイム時ともに使用できます。

データタイプ

論理型（読み取り専用）

適用対象

テキスト ボックス

例

以下のステートメントは、"TextBox_1" という名前のテキスト ボックスの読み取り専用ステータスを取得します。

```
GetPropertyD( "TextBox_1.ReadOnly", A_Tagname );
```

参照項目

GetPropertyD()、 SetPropertyD()

チェック ボックスのラベルの取得または設定

.Caption ドットフィールドはチェック ボックスのメッセージテキストを定義します。

.Caption ドットフィールド

チェック ボックスに表示するメッセージを決定します。

カテゴリ

ウィンドウ コントロール

使用法

```
[ErrorNumber=]GetPropertyM ("ControlName.Caption", Tagname);  
[ErrorNumber=]SetPropertyM ("ControlName.Caption", "Message");
```

パラメータ

ControlName

ウィンドウ コントロールの名前。例：ChkBox_4

Tagname

要求されたプロパティを保持するメッセージ型タグ変数

Message

引用符で囲まれたメッセージ文字列

備考

このプロパティは開発時およびランタイム時ともに読み取り／書き込みです。

データタイプ

メッセージ型（読み取り／書き込み）

適用対象

チェック ボックス

例

このステートメントは、チェック ボックス オブジェクト "CheckBox_1" のキャプションを "Blue Paint Option." に設定します。

```
SetPropertyM( "CheckBox_1.Caption","Blue Paint Option" );
```

参照項目

GetPropertyM()、SetPropertyM()

ウィンドウ コントロール エラー メッセージの理解

wcErrorMessage() は、指定されたエラー番号のエラーを説明するメッセージ文字列を返します。リスト ボックス、テキスト ボックス、コンボ ボックス、ラジオ ボタン、およびチェック ボックスに適用されます。

ウィンドウ コントロール関数は、QuickScript 関数処理した結果に基づいて値を返します。戻り値はエラー診断に使用されます。これらの値を整数型タグ変数に割り当てることができます。次に例を示します。

```
ErrorNumber = wcGetItem("ControlName", Number, Tagname);
```

このスクリプトでは、**ErrorNumber** は、返されたエラー値を含む整数型タグ変数です。この関数の戻り値は、**wcErrorMessage()** 関数に渡すことができます。**wcErrorMessage()** 関数は、そのエラーの説明文字列を返します。次に例を示します。

```
ErrorMsg = wcErrorMessage(ErrorNumber);
```

このスクリプトでは、**ErrorMsg** は、返されたエラーのテキストを含むメッセージ型タグ変数です。以下の表では、エラー数値およびその定義を識別しています。

エラー メッセージ 定義

0	成功
-1	一般エラー
-2	使用できるメモリが不十分です
-3	プロパティは読み取り専用です
-4	指定したアイテムはすでに存在します
-5	オブジェクト名が不明です
-6	プロパティ名が不明です
-x	不明なエラー

wcErrorMessage() 関数

エラーを説明するメッセージ文字列を返します。

カテゴリ

ウィンドウ コントロール

構文

```
ErrorMessage=wcErrorMessage(ErrorNumber);
```

パラメータ

ErrorMessage

メッセージ型タグ変数

ErrorNumber

すべてのウィンドウ コントロール関数によって返される数。数値または整数型タグ変数です。

備考

返されるエラー番号のリストについては、「[ウィンドウ コントロールエラー メッセージの理解](#)」を参照してください。

適用対象

リスト ボックス、テキスト ボックス、コンボ ボックス、チェック ボックス、およびラジオ ボタン

例

リストのロード中にエラーが発生した場合、メッセージ型タグ変数 **ErrorDescription** にエラーのテキストによる説明を表示します。この例では、エラー メッセージを表示するために、値の表示のアニメーションリンクがタグ変数 **ErrorDescription** に割り当てられます。

"開く時" ウィンドウ QuickScript :

```
ErrorNumber=wcLoadList("ListBox_1","c:\recipe.txt");  
ErrorDescription=wcErrorMessage(errornumber);
```

この関数はすべてのウィンドウ コントロール関数と使用して、エラー メッセージを表示できます。

```
ErrorNumber=wcAddItem("ListBox_1","AM_4A4356");  
ErrorMsg=wcErrorMessage(ErrorNumber);
```

XML ファイルを使用したウィンドウのインポート

コマンドファイルを作成して、コマンドファイルで **WindowMaker** を実行し、XML ファイルからウィンドウをインポートします。ウィンドウのインポートに加えて、コマンドファイルを使用して次を行うことができます。

- 新しいアプリケーションを作成する。
- ウィンドウを削除する。
- ウィンドウの名前を変更する。
- 印刷情報をファイルまたはプリンタに送信する。
- クロス リファレンスをファイルに送信する。

XML のインポート中は、以下の操作は参照されているシンボルでサポートされません。

- シンボルの削除
- シンボルの名前変更
- シンボル名のスワップ

しかし、これらの操作は XML インポートが完了した後に実行できます。

以下のシステム レベルの操作はサポートされません。

- コンソールセッションの起動
- 他のアプリケーションへのフォーカスの切り替え

InTouch DBDump および DBLoad ユーティリティはコマンドファイルから実行できますしかし、マネージド InTouch アプリケーションの場合、**System Platform IDE** 拡張機能から **DBDump** および **DBLoad** ユーティリティを実行できます。DBDump は、InTouch アプリケーションからのタグ情報を含むエクスポートファイルを作成します。DBLoad は、タグ情報を InTouch アプリケーションにインポートします。DBDump および DBLoad ユーティリティの詳細については、『AVEVA™ InTouch HMI アプリケーション メンテナンス ガイド』参照してください。

XML ファイルの準備

次の手順は、XML ファイルを作成し、ウィンドウ定義を InTouch アプリケーションにインポートする方法を示します。

1. XML ファイルを作成します。このガイドでは使用するツールや基本的な XML 構文を定義しません。
2. XML ファイルが XML 書式設定の標準に準拠しているかどうかを確認してください。XML 検証ツールを使用して XML ファイルの構文を検証できます。XML ファイルを表示するには、Internet Explorer で開きます。
3. アプリケーションでタグを準備します。InTouch DBDump および DBLoad ユーティリティを使用して、タグを抽出、変更および再ロードします。DBDump、DBLoad、および CSV ファイルの詳細については、『AVEVA™ InTouch HMI アプリケーションメンテナンス ガイド』を参照してください。
4. 必要な機能を実行するための WindowMaker コマンドファイルを準備します。
5. InTouch アプリケーション、WindowMaker、および WindowViewer を終了します。
6. コマンドラインプロンプトから WindowMaker コマンドファイルを実行するか、プログラムまたは IDE 拡張機能からコマンドファイルを実行します。
7. エラーを確認します。エラー ログ ファイルを定義した場合、エラー ログ ファイルとロガーを確認します。プログラムからコマンドファイルを実行した場合、戻りコードを確認します。

XML インポート機能を使用してウィンドウ定義を作成およびインポートする際は、以下のサンプルが役に立ちます。

- InTouch アプリケーションにウィンドウ定義をロードするサンプル XML ファイル。
- サンプル コマンドファイル。
- 2つのスキーマ ファイル。インポート パーサーでは、これらのスキーマ ファイルは使用されません。スキーマ ファイルの制限は XML インポート パーサーよりも厳密です。

注記: スキーマ ファイルは、インストールが完了した後に InTouch インストール フォルダに配置されます。

コマンド ファイルの準備

WindowMaker コマンド ファイルはテキスト ファイルです。このファイルからコマンドが読み取られ、InTouch アプリケーション データ セットに対してアクションが実行されます。

コマンド ファイルはシングルバイトの ANSI 文字セットを使用します。マルチバイト文字セット (MBCS) および Unicode 文字は使用できません。

テキスト ファイルの構文の詳細を以下に示します。

- 8 ビット ANSI 文字セットが使用されますが、使用できるのは 127 文字までです。ほとんどの制御文字は削除されます。
- # 文字で始まる行はコマンドとして扱われます。
- 空白行は無視されます。
- 行はキャリッジ リターン ライン フィード (CRLF) シーケンスで終了します。
- スペースは、行の最初、行の最後、コマンドとコマンドの間、等号文字、およびその引数に使用できます。

- 引用符が必要な引数では、引用符とその他のテキストの間にスペースを使用することはできません。
- 行の最初および最後のホワイト スペースは、各行が処理される前に削除されます。
- 各コマンドはピリオドで開始します。
- 各コマンドファイルは WindowMaker コマンドファイル コマンド (.WINDOWMAKERCOMMANDFILE) で始まる必要があります。

1つのコマンドファイルに複数のコマンドを含めることができます。コマンドが失敗した場合、コマンドプロセッサはファイル内の次のコマンドの処理を続けます。

最小コマンド ファイルの作成

最小のコマンド ファイルの例を次に示します。

```
.WINDOWMAKERCOMMANDFILE  
.VERSION=1
```

ファイルへの印刷情報の送信

InTouch アプリケーション全体のリストをプリンタまたはテキスト ファイルに送信できます。InTouch ウィンドウに産業用グラフィックが含まれる場合、出力は HTML ファイルに送信されます。

次の例のコマンド ファイルは、アプリケーション情報を Printer01 というプリンタに出力します。

```
.WINDOWMAKERCOMMANDFILE  
.VERSION=1  
.PRINTAPPLICATIONINFORMATION  
.OUTPUTTARGET=Printer  
.OUTPUTTARGETNAME=Printer01  
.GO
```

ファイルに出力するコマンド ファイルの例を以下に示します。ファイルが存在する場合は上書きされます。

```
.WINDOWMAKERCOMMANDFILE  
.VERSION=1  
.PRINTAPPLICATIONINFORMATION  
.OUTPUTTARGET=TextFile  
.OUTPUTTARGETNAME=C:\MyApps\AppInfo.txt  
.GO
```

ファイルへのクロス リファレンスの送信

クロス リファレンスを作成してファイルに送信できます。ファイルが存在する場合は上書きされます。ユーザーの操作は必要ありません。

コマンド ファイルの例を以下に示します。

```
.WINDOWMAKERCOMMANDFILE  
.VERSION=1  
.CROSSREFERENCE  
.SEARCHFOR=TagName  
.REFERENCETYPE=ByWindow  
.OUTPUTFILE=C:\MyApps\AppCrossRef.Csv  
.GO
```

ログ ファイルの作成

情報メッセージ、警告、およびエラーをテキスト ファイルに記録できます。このコマンドを指定しない場合、処理ステータスはロガーに送信されます。ファイルが既に存在する場合は上書きされます。

エラーをファイルに記録するコマンドの例を以下に示します。

```
.WINDOWMAKERCOMMANDFILE  
.VERSION=1  
.COMMANDLOGFILE=C:\MyApps\LogFile.Txt
```

MyApps フォルダが存在することを確認してください。MyApps フォルダが存在しない場合、ログ ファイルは作成されません。

コマンドの構文

以下の表には、WindowMaker コマンド ファイルで使えるコマンドのリストが一覧表示されています。

コマンド	説明
.WINDOWMAKERCOMMANDFILE	ファイルが WindowMaker コマンド ファイルであることを識別します。このコマンドが見つからない場合、ファイルは処理されず、WindowMaker はエラー コードで終了します。
.VERSION=1	このコマンドは、読み取るファイルがあまり新しくないことを WindowMaker に伝えます。
.COMMANDLOGFILE=<フル ファイル パス>	指定したファイルに情報、警告、およびエラーが記録されます。ファイルが既に存在する場合は上書きされます。このコマンドを省略した場合、ロガーで処理ステータスに関する情報を確認します。
.GO	コマンドを実行します。1 つ以上のパラメータを含むコマンドの後に必要です。
.WINDOWCREATE	XML ファイルからウィンドウを作成します。
.XMLFILEPATH=<フル ファイル パス>	ウィンドウ仕様を含む XML ファイルへのフル ファイル パス。WINDOWCREATE コマンドを使用する場合に必要です。
.WINDOWDELETE	名前で指定した InTouch アプリケーションを削除します。ウィンドウ名が見つからない場合、エラーは生成されません。
.WINDOWNAME=<ウィンドウ名>	削除するウィンドウの名前 WINDOWDELETE コマンドを使用する場合に必要です。

コマンド	説明
.WINDOWRENAME	ウィンドウの名前を変更します。古いウィンドウ名が見つからない場合、警告が表示されます。新しい名前と同じ名前のウィンドウが存在する場合、エラーメッセージが生成されます。ウィンドウの名前を変更できない場合、エラーが発生します。古い名前と新しい名前が同じ場合、何の処理も行われません。
.OLDWINDOWNAME=<既存のウィンドウ名>	現在のウィンドウ名。 WINDOWRENAME コマンドを使用する場合に必要です。
.NEWWINDOWNAME=<新しいウィンドウ名>	新しいウィンドウ名。この名前のウィンドウが存在しないことを確認してください。 WINDOWRENAME コマンドを使用する場合に必要です。
.PRINTAPPLICATIONINFORMATION	InTouch アプリケーション データ セットをテキスト ファイルに出力またはダンプします。これは、 WindowMaker から [ファイル] メニュー コマンドの [印刷] オプションをクリックした場合と同じ動作です。
.OUTPUTTARGET=Printer TextFile	出力をプリンタまたはテキスト ファイルに送信します。
.OUTPUTTARGETNAME=<プリンタ名> <フルテキスト ファイル パス>	プリンタまたは出力ファイルの名前。ファイルが存在する場合は上書きされます。
.CROSSREFERENCE	クロス リファレンス情報をカンマ区切り変数 (CSV) 形式で生成します。
.SEARCHFOR= TagName QuickFunctions	タグまたはクイック関数を名前で検索します。
.REFERENCETYPE= ByTagName ByWindow	タグまたはウィンドウ名によるクロス リファレンス。
.OUTPUTFILE=<フル ファイル パス>	出力ファイルへのフルパス。ファイルが存在する場合は上書きされます。

アプリケーションの作成

WindowMaker コマンドファイルを使用して、新しいデフォルト **InTouch** アプリケーションを作成できます。最小のコマンドファイルを使用して空のアプリケーションを作成します。しかし、最小のコマンドファイルを使用して空のマネージド **InTouch** アプリケーションを作成することはできません。詳細については、「[最小コマンドファイルの作成](#)」を参照してください。

コマンドファイルは、アプリケーションを作成するフォルダに配置する必要があります。フォルダに既存の InTouch.ini ファイルが含まれる場合、アプリケーションは生成されません。

新しいアプリケーションフォルダへのパスには、埋め込まれた "-I" および "-L" 文字シーケンスを使用することはできません。たとえば、C:\MyApps\App-Large というフォルダを作成することはできません。

作成された InTouch.ini ファイルには、次の例のような内容が含まれます。ウィンドウの位置は、アプリケーションを表示する画面解像度により異なります。アプリケーションのタイトルと説明は "Generated InTouch Application" です。デフォルト言語は英語です。

InTouch.ini ファイルの内容の例を以下に示します。

```
[InTouch]
AppMode=2
AppName0=Generated InTouch Application
AppName1=
AppName2=
AppName3=
AppDesc0=Generated InTouch Application
AppDesc1=
AppDesc2=
AppDesc3=
LanguageBase=English (United States)
LanguageBaseID=1033
InTouchView=0
SAOConverted=1
WinFullScreen=1
WinLeft=-4
WinTop=-4
WinWidth=1288
WinHeight=1004
SnapOn=1
```

新しく作成されたアプリケーションへのタグの追加

新しい InTouch アプリケーションにはシステム タグだけが含まれます。ウィンドウをインポートする前にアプリケーションにタグを追加する必要がある場合、次の手順に従います。

1. 新しいアプリケーションを作成する。
2. DBLoad を実行してタグを新しいアプリケーションにインポートします。
3. ウィンドウをインポートします。

次に例を示します。

```
WM.Exe C:\MyApps\App001 COMMANDFILE="C:\BlankFile.Txt"
DBLoad C:\MyApps\App001,C:\TagDumps\App001.Csv,0
WM.Exe C:\MyApps\App01 COMMANDFILE="C:\Commands.Txt"
```

ウィンドウの削除

同じ名前のウィンドウがある既存のアプリケーションにウィンドウを追加する場合、既存のウィンドウを削除する必要があります。既存のウィンドウは WindowMaker コマンドファイルを使用して削除することができます。

重要: 削除するウィンドウが存在しない場合、エラー メッセージは表示されません。

既存のウィンドウを削除できない場合、ログにメッセージが表示されます。ユーザーの操作は必要ありません。

InTouch アプリケーションからウィンドウを削除するには、コマンドファイルにコマンドのシーケンスを入力します。コマンドファイルには、複数のウィンドウ削除コマンドシーケンスを含めることができます。

次の例は、InTouch アプリケーションからウィンドウを削除するコマンドシーケンスを示します。

```
.WINDOWMAKERCOMMANDFILE  
.VERSION=1  
.WINDOWDELETE  
.WINDOWNAME=Window002  
.GO
```

ウィンドウの名前の変更

InTouch アプリケーション内のウィンドウの名前を変更できます。

InTouch アプリケーションからウィンドウの名前を変更するには、コマンドファイルにコマンドのシーケンスを入力します。コマンドファイルには、複数のウィンドウ名前の変更コマンドシーケンスを含めることができます。

名前を変更するウィンドウが存在しない場合、エラーメッセージは表示されません。

新しい名前と同じ名前のウィンドウが既に存在する場合、何の処理も行われず、警告メッセージが記録されます。

古い名前のウィンドウが存在し、新しい名前と同じ名前のウィンドウが存在しなくても何らかの理由で名前の変更が失敗した場合、エラーメッセージが記録されます。

ユーザーの操作は必要ありません。警告とエラーメッセージがログに表示されます。

ウィンドウの名前を既存のウィンドウの名前に変更した場合、警告メッセージがログファイルまたは Logger に記録されます。以下の例は、Window005 の名前を既存のウィンドウ Window006 の名前に変更すると、警告メッセージがログファイルまたはログに記録されます。

```
.WINDOWRENAME  
.OLDWINDOWNAME=Window005  
.NEWWINDOWNAME=Window006  
.GO
```

ウィンドウのインポート

ウィンドウを InTouch アプリケーションにインポートできます。InTouch アプリケーションからウィンドウの名前を変更するには、コマンドファイルにコマンドのシーケンスを入力します。新しいウィンドウ名は XML ファイル内に含まれます。

次の場合、ウィンドウはインポートされません。

- 同じ名前のウィンドウがアプリケーション内に存在する場合。
- ウィンドウ スクリプト内で解決できないエラーが発生した場合。
- オブジェクトの一部をウィンドウに追加しようとしたときに解決できないエラーが発生した場合。

次の場合、ユーザーの操作が必要です。

- スクリプトまたは式で指定した要素にエラーが含まれる場合、または要素が欠落している場合。

- スクリプトに構文エラーが含まれる場合。

ユーザーが問題を修正するためのメッセージは表示されないことがあります。詳細については、ログファイルまたはロガーを確認してください。

ウィンドウをインポートするには、コマンドファイルにコマンドのシーケンスを入力します。

次に例を示します。

```
.WINDOWCREATE  
.XMLFILEPATH=C:\WMCommandTest\WMCreateFile.Xml  
.GO
```

コマンドファイルには、複数のウィンドウ インポート コマンドおよびいくつかのウィンドウ作成コマンドシーケンスを含めることができます。

エラーの処理

XML ファイルは一般的な XML 書式設定ルールに従う必要があります。XML ファイルを Internet Explorer で開けない場合、XML アイルに XML 書式設定エラーが含まれています。WindowMaker でファイルを使用する前に、すべてのエラーを修正します。

一般的な書式設定エラーを含むファイルを使用すると、WindowMaker は「XML ファイルをロードできませんでした」というエラー メッセージがロガーに記録されます。

スクリプトで要素が欠落している場合、スクリプト エラーがある場合、またはその他の要素仕様エラーが発生した場合、WindowMaker は停止します。次に例を示します。

- タグの欠落
- 外部 WindowMaker スクリプト拡張 DLL の欠落
- ActiveX コントロールの欠落
- ウィザード DLL の欠落

特定のアニメーション リンクまたはカスタム プロパティの上書きなどの場合、タグが欠落していてもメッセージ ボックスは表示されません。ロガーにメモが書き込まれ、アニメーション リンクとオブジェクトは作成されず、ウィンドウは作成されません。その他の場合、スクリプトおよび式の解析は停止します。その場合、タグを作成することができます。タグが正常に作成されると、処理が継続されます。

SmartSymbol の欠落

SmartSymbol テンプレートがターゲット アプリケーションに存在しない場合、ウィンドウはインポートされません。エラー メッセージがロガーおよび出力テキスト ファイルに表示されます。

何らかの理由で SmartSymbol インスタンスの作成に失敗した場合、ウィンドウは作成されません。

SmartSymbol テンプレートが存在していても SmartSymbol のインポートが失敗することもあります。失敗に関する追加情報がロガーに記録されている可能性があります。

産業用グラフィックの欠落

産業用グラフィック参照が Galaxy リポジトリに存在しない場合、ウィンドウは作成されません。エラーメッセージがロガーおよび出力テキスト ファイルに表示されます。

産業用グラフィック参照が存在していても XML のインポート プロセスが失敗することもあります。失敗に関する追加情報がロガーに記録されます。

式、タグ名、およびスクリプト

式、タグ名、およびスクリプトが必要な場合、空のテキストやブランク テキストは使用できません。これらのテキスト項目が正しく作成されていない場合、式、タグ名、およびスクリプトは正しく解析されず、エラーが発生します。

タグ名の場合、タグ タイプはオブジェクトで予期されるタイプに一致する必要があります。ほとんどの場合、ドットフィールドを使用してタグ プロパティにアクセスできます。たとえば、整数型タグ `iTag005` の 2 番目のビットには次のようにしてアクセスできます。

```
iTag005.02
```

その場合、タグとドットフィールドは論理値に解決され、論理型タグ名が必要な場所で使用できますが、`iTag005` だけではエラーの結果になります。

コマンド プロンプトからの WindowMaker の実行

コマンド プロンプトから **WindowMaker** を実行できます。

コマンド ラインは以下のとおりです。

```
WM.EXE AppPath,Commandfile="Command.txt"
```

パラメータ

AppPath

InTouch アプリケーションのフルパスを定義します。**WindowMaker** をアプリケーションのフォルダで実行している場合、このパラメータはオプションです。

CommandFile

コマンド ファイルのフルパスを定義します。パス名は引用符で囲む必要があります。等号の周囲に余分なスペースを挿入することはできません。

InTouch アプリケーションが `C:\MyApps` フォルダにあり、コマンド ファイルが `C:\WMCommandFile.txt` である場合を考えてみます。次の例は、コマンド プロンプトから入力する **WindowMaker** コマンドを示します。

```
WM.EXE C:\MyApps,COMMANDFILE="C:\WMCommandFile.Txt"
```

注: コマンド プロンプトからマネージド InTouch アプリケーションを作成することはできません。

System Platform IDE 拡張機能

System Platform IDE 拡張機能を使用してマネージド アプリケーションに対して InTouch XML インポート機能を実行できます。IDE 拡張機能を使用して XML インポート プロセスのコマンド ファイルを選択できます。コマンド ファイルを選択した後、**WindowMaker** が起動し、関連付けられた XML が解析されます。

重要: XML インポートが進行中のときに新しいコンソールセッションを開始した場合、または既存のコンソールセッションを最大化した場合、**WindowMaker** は応答を停止します。

System Platform IDE には、XML インポートからコマンド ファイルを選択するために使用するコンテキスト メニュー項目があります。InTouchViewApp 派生テンプレートを右クリックすると、**[InTouch コマンド ファイルの処理]** メニュー項目がコンテキスト メニューに表示されます。しかし、以下のものを選択した場合、このメニュー項目は表示されません。

- 複数の InTouchViewApp テンプレート
- InTouchViewApp ベース テンプレート

- InTouchViewApp インスタンス

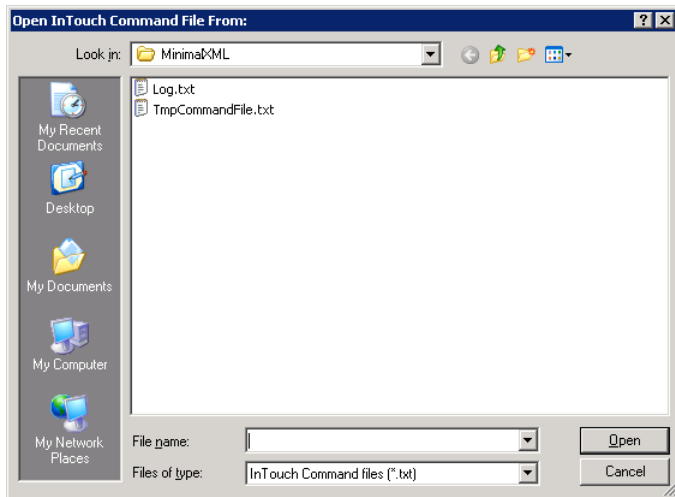
古いバージョンの InTouch からマネージド InTouch アプリケーションを移行した場合、コマンド ファイルを処理する前に少なくとも 1 回は WindowMaker で InTouch アプリケーションを開く必要があります。

マネージド InTouch アプリケーションからの WindowMaker の実行

マネージド InTouch アプリケーションから WindowMaker 実行できます。

マネージド InTouch アプリケーションから WindowMaker を実行するには

1. System Platform IDE を開きます。
2. [テンプレート ツールボックス] で、InTouchViewApp テンプレートから派生テンプレートを作成します。
3. 新しい InTouch アプリケーションを作成するか、既存のスタンドアロン InTouch アプリケーションをインポートすることによって InTouch アプリケーションを派生テンプレートに関連付けます。
4. 派生テンプレートを右クリックし、[InTouch コマンド ファイルの処理] をクリックします。[InTouch コマンド ファイルを開く:] ダイアログ ボックスが表示されます。



注: [InTouch コマンド ファイルの処理] コマンドを実行する前に WindowMaker で少なくとも 1 回 InTouchViewApp 派生テンプレートを設定する必要があります。

1. コマンド ファイルの場所を参照して [開く] をクリックします。WindowMaker が起動します。

コマンド プロンプトからの DBDump の実行

DBDump を使用して InTouch アプリケーションから外部タグ情報を抽出します。コマンド プロンプトから DBDump を実行できます。DBDump を実行する前に WindowMaker を停止する必要があります。

複数の DBDump コマンド パラメータの間にはカンマが必要です。パラメータは位置に依存します。含まれるパラメータ間で 1 つのパラメータを省略する場合、省略するパラメータのカンマを含める必要があります。次の例は、コマンド プロンプトから DBDump コマンドを実行する場合の構文を示します。

```
DBDump AppPath,CsvPath,GroupTypes,OverwriteCsvFile, MessageBoxes
```

パラメータ

AppPath

InTouch アプリケーションのパスを定義します。

CsvPath

InTouch アプリケーションのタグ名ディクショナリからのタグ定義を含むエクスポート ファイルのパスを定義します。

GroupTypes

DBDump エクスポート ファイルでタグを InTouch タグ タイプでグループ化するかどうかを指定します。1 を指定すると、タグ データベース名がタグ グループ タイプでソートされます。0 を指定すると、タグ名はタイプでソートされません。

OverwriteCsvFile

エクスポート ファイルを上書きするかどうかを指定します。1 を指定すると、エクスポート ファイルが上書きされます。0 を指定すると、エクスポート ファイルは上書きされません。

MessageBoxes

DBDump ユーティリティがアプリケーションのタグ名ディクショナリの内容をエクスポートするとき、メッセージを表示するかどうかを指定します。1 を指定すると、メッセージボックスが表示されます。0 を指定すると、メッセージボックスは表示されません。

例

InTouch アプリケーションが C:\MyInTouchApps\App001 フォルダにあり、タグ データベースの内容を C:\TagDumps\App001.csv ファイルに書き込む場合を考えてみます。メッセージボックスを表示せず、既存のターゲット エクスポート ファイルを上書きするとします。コマンドを次に示します。

```
DBDump C:\MyInTouchApps\App001,C:\TagDumps\App001.Csv,1,1,0
```

マネージド InTouch アプリケーションに対する DBDump の実行

マネージド InTouch アプリケーションに対して DBDump を実行できます。

マネージド InTouch アプリケーションに対して DBDump を実行するには

1. System Platform IDE を開きます。
2. [テンプレート ツールボックス] で InTouchViewApp 派生テンプレートを右クリックし、[エクスポート] をポイントして [CSV] を選択します。
[オートメーションオブジェクトを CSV にエクスポート] ダイアログ ボックスが表示されます。
3. CSV ファイルの名前を入力して、[保存] をクリックします。アプリケーション タグ データが CSV ファイルにダンプされます。

コマンド プロンプトからの DBLoad の実行

DBLoad を使用してタグ情報を InTouch アプリケーションにインポートします。コマンド プロンプトから DBLoad を実行できます。DBLoad を実行する前に WindowMaker を停止する必要があります。

DBLoad コマンド パラメータは位置に依存します。含まれるパラメータ間で 1 つのパラメータを省略する場合、省略するパラメータのカンマを含める必要があります。次の例は、コマンド プロンプトから DBLoad コマンドを実行する場合の構文を示します。

```
DBLoad AppPath,CsvPath,MessageBoxes
```

パラメータ**AppPath**

InTouch アプリケーションのパスを指定します。

CsvPath

アプリケーションのタグ名ディクショナリにインポートするタグ定義を含むファイルへのパスを指定します。

MessageBoxes

DBLoad ユーティリティがインポート ファイルの内容をアプリケーションのタグ名ディクショナリにインポートするときにメッセージを表示するかどうかを指定します。**1** を指定すると、メッセージボックスが表示されます。**0** を指定すると、メッセージボックスは表示されません。

例

InTouch アプリケーションが C:\MyInTouchApps\App001 フォルダにあり、タグ データベース情報を C:\TagDumps\App001.Csv ファイルから読み取る場合を考えてみます。その際にメッセージボックスは表示しないようにします。次の例は、コマンドプロンプトに入力する DBLoad コマンドを示します。

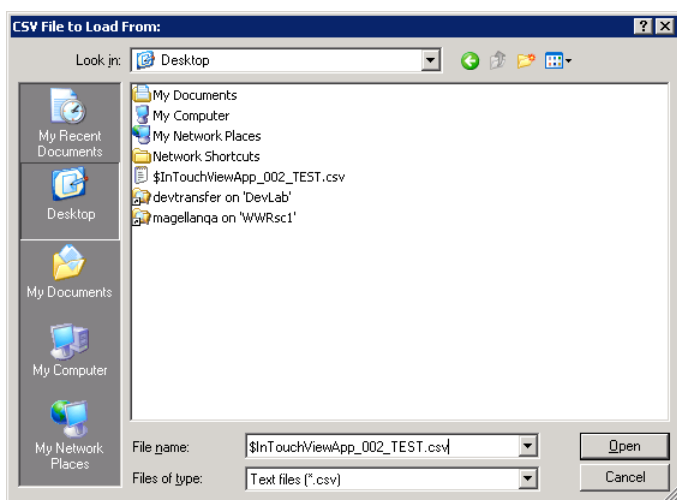
```
DBLoad C:\MyInTouchApps\App001,C:\TagDumps\App001.Csv,0
```

マネージド InTouch アプリケーションに対する DBLoad の実行

マネージド InTouch アプリケーションに対して DBLoad を実行できます。

マネージド InTouch アプリケーションに対して DBLoad を実行するには

1. System Platform IDE を開きます。
2. [テンプレート ツールボックス] で InTouchViewApp 派生テンプレートを右クリックし、[インポート] をポイントして [DB ロード] をクリックします。[ロード元 CSV ファイル名:] ダイアログ ボックスが表示されます。



3. CSV ファイルの場所を参照して [開く] をクリックします。タグ データが InTouch アプリケーションに正常にロードされます。

XML の形式

XML ファイルで定義されているウィンドウおよびほとんどのウィンドウ要素をインポートできます。タグを XML ファイルからインポートすることはできません。DBLoad ユーティリティを使用する必要があります。

一般的な XML ファイル形式

XML ファイルは一般的なすべての XML 書式設定ルールに従う必要があります。XML ファイルが Internet Explorer で開けることを確認してください。開けない場合、XML ファイルに書式設定エラーが含まれています。

スキーマの使用

XML ファイルをスキーマ ファイルで検証できます。スキーマ ファイルで XML ファイルを検証すると、ほとんどの XML 書式設定エラーが検出されます。

スキーマの制限は XML 入力パーサーよりも厳密です。

- スキーマでは、要素の順序がこのガイドの表に一覧表示されている順序に一致する必要があります。
- 要素名と値では大文字と小文字が区別されます。
- スキーマでは要素を明示的に定義する必要がある場合があります。

スキーマ検証はウィンドウ要素から呼び出すことができます。スキーマの指定方法の詳細については、「[ウィンドウの定義](#)」を参照してください。

XML ファイルヘッダー

ファイルの上部に XML 宣言を含める必要があります。最小の宣言を以下に示します。

```
<?xml version="1.0"?>
```

非必須の XML 要素

InTouch の XML インポート機能では、未定義の要素、属性、およびノードに関するエラーと警告は生成されません。

すべての要素、属性、およびノードを正しく入力する必要があります。<ONRIGHTDOWN> アクション スクリプトを使用する際に間違って <ONRITEDOWN> と入力すると、警告は生成されず、アクション スクリプトはウィンドウ オブジェクトに追加されません。

必須の要素が欠落している XML 定義に関する警告とエラーは作成されます。

ユーザー指定のテキスト

スクリプトで XML フィールド区切り文字が使用されている場合、スクリプト テキストを CDATA 要素内にカプセル化する必要があります。CDATA 要素内のテキストは XML ファイルインポート機能によって解析されません。

テキストに XML フィールド区切り文字が含まれない場合、CDATA 要素内でカプセル化せずにテキストを入力できます。

テキスト ホワイトスペースの維持

CDATA 要素にないテキストが処理されると、先頭と末尾のスペースは削除されます。その結果、<Title> MyWindowName</Title> などの要素は、先頭にスペースのない 'MyWindowName' になります。

テキストの先頭と末尾のホワイトスペースを維持するには、テキストを CDATA 要素内に囲みます。たとえば、<Title></Title> という要素は、"MyWindowName" というウィンドウ名になります。

共通の要素定義

一部の要素または定義は多くの要素で共有できます。

色要素

色要素は、RGB 値、名前、参照値、または整数値で指定できます。色要素は、R、G、B、Name、Ref、および Value です。これらの要素は、その他の要素（FillColor、TextColor、BGColor など）で使用されます。

RGB 要素

RGB 値は色を指定するために使用します。RGB 要素に割り当てられる値は 0 から 255 の範囲です。要素がない場合は、デフォルト値が使用されます。デフォルト値はウィンドウ オブジェクトに応じて異なります。

例:

```
<FillColor>  
<R>192</R>  
<G>192</G>  
<B>192</B>  
</FillColor>  
<TextColor> <R>0</R><G>0</G><B>0</B> </TextColor>
```

色名要素

Internet Explorer バージョン 3.0 以降でサポートされる名前を使用して色を指定できます。色名では大文字と小文字が区別され、既知の HTML 色に一致する必要があります。

色名と値のリストについては、以下の Web サイトを参照してください。

http://www.w3schools.com/html/html_colornames.asp

<http://www.learningwebdesign.com/colornames.html>

http://www.oreilly.com/catalog/wdnut/excerpt/color_names.html

色名のリストについては、以下の Web サイトを参照してください。

<http://www.geocities.com/SiliconValley/Pines/6986/colortbl.html>

例:

```
<FillColor>  
<Name>White</Name>  
</FillColor>  
<TextColor> <Name>White</Name> </TextColor>
```

色参照要素

16 進数の色値を使用して色を指定できます。色値には、常に 6 桁の 16 進数が必要です。

名前とそれに対応する 16 進数値の色のリストについては、以下の Web サイトを参照してください。

http://www.w3schools.com/html/html_colornames.asp

<http://www.learningwebdesign.com/colornames.html>

次の Web サイトでは、ディザリング処理なしの色値のリストが公開されています。

<http://www.htmlgoodies.com/tutorials/colors/article.php/3479001>

次のサイトでは 4096 色の 16 進数コードが紹介されています。

<http://yorktown.cbe.wvu.edu/sandvig/MIS314/Assignments/A03/ColorHexCodes.asp>

例:

```
<FillColor>
```

```
<Ref>#FF00FF</Ref>
</FillColor>
<TextColor> <Ref>#FF00FF</Ref> </TextColor>
```

色値要素

正の整数値の色値を使用して色を指定できます。値は 0 から 16777215 まで範囲である必要があります。

例:

```
<FillColor>
<Value>16711935</Value>
</FillColor>
<TextColor> <Value>16711935</Value> </TextColor>
```

TextInfo 要素

TextInfo 要素を使用して画面にテキストを表示する方法を指定できます。

次の要素を使用してテキストを指定できます。

要素	説明
Font	フォントファミリーの名前 (System や Arial など)。
FontStyle	値は、Regular、Italic、Bold、BoldItalic です。
FontSize	フォント サイズの測定単位を指定する必要があります。通常は、ポイントを使用します。値は 0 またはそれ以上です。
下線	下線付きテキスト: true または false。
取り消し線	テキストの取り消し線: true または false。
TextColor	テキスト色の色要素。
TextJustify	テキストの位置揃え: Left、Center、または Right

例:

```
<TextInfo>
<Font>Arial</Font>
<FontStyle>Regular</FontStyle>
<FontSize>12</FontSize>
<Underline>>false</Underline>
<Strikeout>>false</Strikeout>
<TextColor>
<R>0</R>
<G>0</G>
<B>0</B>
</TextColor>
<TextJustify>Left</TextJustify>
</TextInfo>
```

Point 要素

Point 要素を使用して、その他の要素の位置を定義します。Point 要素には、2 つの要素 (X および Y) 含まれます。X と Y の要素は、-32000 から 32000 の範囲の値を含む必要があります。

要素	説明
X	ピクセル単位での左座標。必須。
Y	ピクセル単位での上座標。必須

例:

```
<Point>  
<X>10</X>  
<Y>25</Y>  
</Point>
```

Pen 要素

Pen 要素は、オブジェクトの境界線の特徴を指定するために使用します。

次の要素を使用して Pen 要素を指定できます。

フィールド	説明
PenColor	次の色要素を使用します。RGB 要素、名前要素、参照要素、または値要素。
PenWidth	ピクセル単位のペン幅。値は、1、2、4、6、9、または 11 です。
PenStyle	値は、None、Solid、Dash、Dot、DashDot、DashDotDot です。

例:

```
<Pen>  
<PenColor>  
<R>0</R>  
<G>0</G>  
<B>0</B>  
</PenColor>  
<PenWidth>4</PenWidth>  
<PenStyle>Solid</PenStyle>  
</Pen>
```

Dimension 要素

Dimension 要素は、画面上のオブジェクトの左上隅の座標を指定する要素を含みます。Dimension 要素には、長方形オブジェクトの幅と高さを指定する要素も含みます。

WindowMaker では、座標の制限は、垂直と水平の両方向で -32000 から +32000 です。(-32000, -32000, 32000, 32000) 境界の外にある計算済み座標を配置する幅または高さ値で X 位置と Y 位置の組み合わせを指定すると、警告が表示され、値は最大値に設定されます。高さと幅の値は正である必要があります。

Dimension 要素内で以下の要素を使用して画面の領域を指定します。

要素	説明
Left	左端の座標。必須。
Top	上端の座標。必須。
Width	ピクセル単位の幅。
Height	ピクセル単位の高さ。

例:

```
<Dimension>
<Left>4</Left>
<Top>4</Top>
<Width>632</Width>
<Height>278</Height>
</Dimension>
```

式

式テキストは **CDATA** セクション内で囲みます。そうすることにより、式で有効な **XML** 区切り文字で解析が失敗することを回避します。

例:

```
<EXPRESSION>
<![CDATA[
式テキストの最初の行
式テキストの 2 番目の行
式テキストの N 番目の行
]]>
</EXPRESSION>
```

仮想キーコードと仮想キーフラグ

一部の InTouch アニメーションリンクはキーボード入力をサポートします。

XML パーサーは、仮想キーの名前を仮想キーコードに変換します。また、数値ビットの組み合わせでなく、テキストを使用して修飾キーのフラグが指定されます。キー名では大文字と小文字は区別されません。

InTouch アプリケーションは、以下の表に示す仮想キー名を使用します。

表されるキー	仮想キー名
追加	追加
アルファベット キー	A ~ Z
BACKSPACE	Backspace
キャンセル	CtrlBreak
クリア	クリア
コピー	コピー

表されるキー	仮想キー名
10 進数	10 進数
削除	削除
除算	除算
下矢印	下
空の文字列は割り当てられていないことを示します。	<Blank>
END	End
ENTER	Return
ESC	エスケープ
実行	実行
F1	F1
F2	F2
F3	F3
F4	F4
F5	F5
F6	F6
F7	F7
F8	F8
F9	F9
F10	F10
F11	F11
F12	F12
F13	F13
F14	F14
F15	F15
F16	F16
ヘルプ	ヘルプ
HOME	Home
挿入	挿入

表されるキー	仮想キー名
左矢印	左
乗算	乗算
数値キー	1 ～ 9
数値キーパッド 0	NUMPAD0
数値キーパッド 1	NUMPAD1
数値キーパッド 2	NUMPAD2
数値キーパッド 3	NUMPAD3
数値キーパッド 4	NUMPAD4
数値キーパッド 5	NUMPAD5
数値キーパッド 6	NUMPAD6
数値キーパッド 7	NUMPAD7
数値キーパッド 8	NUMPAD8
数値キーパッド 9	NUMPAD9
NUM LOCK	NumLock
PAGE UP	PageUp
PAGE DOWN	PageDown
PRINT SCREEN	印刷
右矢印	右
選択	選択
区切り記号	区切り記号
Space キー	Space
減算	減算
TAB	Tab
上矢印	上

2 つの修飾キーがサポートされています。修飾キーは **CKEYFLAGS** 要素に対して個別に指定するか、組み合わせで指定することができます。修飾された名前が要素値です。修飾キーは、修飾名が属性値文字列内にある場合に適用されます。

名前	表されるキー
CTRL	通常のキーと一緒に CONTROL キーを押す必要があります。
SHIFT	通常のキーと一緒に SHIFT キーを押す必要があります。
CTRL + SHIFT	CONTROL と SHIFT の両方のキーを押す必要があります。

CTRL+A シーケンスを生成する XML を以下に示します。

```
<VirtualKeyType>  
  <KeyCode>A</KeyCode>  
  <KeyFlags>CTRL</KeyFlags>  
</VirtualKeyType>
```

ウィンドウ要素

XML ファイルに含めることができるウィンドウ要素は **1** つだけです。ウィンドウ要素は XML ファイルで指定する最初の要素である必要があります。

ウィンドウ名が InTouch アプリケーションの既存のウィンドウ名に一致しないことを確認してください。

ウィンドウの定義

以下の要素を使用してウィンドウを指定できます。

要素	説明
Title	空でない文字列としてのウィンドウ名最大 32 文字で、末尾のスペースはカウントされません。必須。
Comment	コメントテキスト。最大 59 文字。
Dimension	オプションのスキーマを使用する場合に必須。スキーマを使用しない場合、 Dimension 要素がなければデフォルトの寸法が適用されます。デフォルト値は 4 4 632 278 です。
WindowStyle	ウィンドウのタイプ = Replace Overlay Popup 。
BackgroundColor	次の色要素で指定されたウィンドウの背景色。RGB 要素、名前要素、参照要素、または値要素。
FrameStyle	Single Double None 。

要素	説明
TitleBar	タイトル バーの有効化 = true false。
CloseButton	ウィンドウの閉じるボタンの有効化 = true false。 タイトル バーが有効になっている場合にのみ有効にすることができます。
SizeControls	サイズ コントロールの有効化 = true false。
ScriptOnShow	スクリプト要素。
ScriptWhileShowing	スクリプト要素。
ScriptOnHide	スクリプト要素。
ObjectList	ウィンドウに含まれるオブジェクト (SmartSymbol を含む)。

次の例では、空のウィンドウが作成されます。

```
<?xml version="1.0" encoding="UTF-8"?>
<iw:InTouchWindow
xmlns:iw="http://www.wonderware.com/InTouch/Window"
xmlns:itc="http://www.wonderware.com/InTouch/Common"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.wonderware.com/InTouch/Window
wwInTouchWindow.xsd" Version="1">
<Title>ApplicationSecondWindow</Title>
<Comment>Main application window</Comment>
<Dimension>
<Left>4</Left>
<Top>4</Top>
<Width>632</Width>
<Height>278</Height>
</Dimension>
<BackgroundColor>
<R>192</R>
<G>192</G>
<B>192</B>
</BackgroundColor>
<WindowStyle>Overlay</WindowStyle>
<FrameStyle>Single</FrameStyle>
<TitleBar>true</TitleBar>
<CloseButton>True</CloseButton>
<SizeControls>true</SizeControls>
</iw:InTouchWindow>
```

最小の例:

```
<iw:InTouchWindow
xmlns:iw="http://www.wonderware.com/InTouch/Window"
xmlns:itc="http://www.wonderware.com/InTouch/Common"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.wonderware.com/InTouch/Window
```

```
wwInTouchWindow.xsd" Version="1">
<Title>ApplicationSecondWindow</Title>
<Dimension><Left>4</Left><Top>4</Top>
  <Width>632</Width>
<Height>278</Height></Dimension>
</iw:InTouchWindow>
```

スキーマ検証のアクティブ化

スキーマの検証と処理は、InTouchWindow 要素内に特定のデータを配置することによってアクティブ化されます。スキーマをアクティブ化する場合、InTouchCommon.Xsd および InTouchWindow.Xsd ファイルを XML ファイルと同じフォルダに配置する必要があります。

スキーマの使用例:

```
<iw:InTouchWindow
xmlns:iw="http://www.wonderware.com/InTouch/Window"
xmlns:itc="http://www.wonderware.com/InTouch/Common"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.wonderware.com/InTouch/Window wwInTouchWindow.xsd"
Version="1">
</iw:InTouchWindow>
```

スキーマを使用しない場合の例:

```
<InTouchWindow Version="1">
</InTouchWindow>
```

ウィンドウ スクリプト

XML インポート ファイル内には 3 つのタイプのウィンドウ スクリプト（OnShow、WhileShowing、および OnHide）を作成できます。ウィンドウ要素内にスクリプト要素を配置します。

スクリプトテキストは CDATA セクションに囲んで、スクリプト内の XML 区切り文字が XML ファイル解析に干渉することを回避できます。

OnShow ウィンドウ スクリプト要素

OnShow スクリプト要素にはスクリプトテキストが含まれます。

例:

```
<ScriptOnShow><![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]></ScriptOnShow>
```

最小の例:

```
<ScriptOnShow>単一行のスクリプト テキスト</ScriptOnShow>
```

WhileShowing ウィンドウ スクリプト要素

WhileShowing スクリプトには 2 つの要素があります。スクリプトテキストは CDATA セクション内に配置できます。

要素	説明
Text	ウィンドウ スクリプトのテキスト。必須。

要素	説明
FREQUENCY	ミリ秒単位でのスクリプト実行頻度。オプションのスキーマを使用する場合に必須。デフォルトは 1000 です。

例:

```
<ScriptWhileShowing>
<Text><![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]></Text>
<Frequency>1000</Frequency>
</ScriptWhileShowing>
```

最小の例:

```
<ScriptWhileShowing>
<Text>単一行のスクリプト テキスト</Text>
<Frequency>1000</Frequency>
</ScriptWhileShowing>
```

OnHide ウィンドウ スクリプト要素

OnHide ウィンドウ スクリプトには **1** つの要素があります。スクリプト テキストは CDATA セクション内に配置できます。

例:

```
<ScriptOnHide>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
</ScriptOnHide>
```

最小の例:

```
<ScriptOnHide>スクリプト テキストの行</ScriptOnHide>
```

ウィンドウ オブジェクト

ウィンドウ オブジェクトは、ObjectList 要素によって定義されたオブジェクト リスト内に配置されます。

例:

```
<?xml version="1.0" encoding="UTF-8"?>
<iw:InTouchWindow xmlns:iw="http://www.wonderware.com/InTouch/Window" xmlns:itc="http://www.wonderware.com/InTouch/Common" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.wonderware.com/InTouch/Window.xsd" Version="1">
<Title>Window0001</Title>
<Comment>Sample simple window</Comment>
<Dimension>
  <Left>10</Left>
  <Top>10</Top>
  <Width>400</Width>
  <Height>400</Height>
</Dimension>
```

```

<BackgroundColor>
  <R>255</R>
  <G>0</G>
  <B>255</B>
</BackgroundColor>
<WindowStyle>Replace</WindowStyle>
<FrameStyle>Double</FrameStyle>
<TitleBar>true</TitleBar>
<SizeControls>true</SizeControls>
<ObjectList>
  <Rectangle>
    <Title>Rectangle1</Title>
    <Pen>
      <PenColor>
        <Name>Black</Name>
      </PenColor>
      <PenWidth>4</PenWidth>
      <PenStyle>Solid</PenStyle>
    </Pen>
    <Dimension>
      <Left>100</Left> <Top>50</Top>
      <Width>270</Width> <Height>80</Height>
    </Dimension>
    <FillColor>
      <R>128</R> <G>128</G> <B>128</B>
    </FillColor>
  </Rectangle>
</ObjectList>
</iw:InTouchWindow>

```

各ウィンドウ オブジェクトにはアニメーション リンクを含めることができます。アニメーション リンクを含むウィンドウ オブジェクトには、1つのアニメーション リンク要素が含まれます。

```
<AnimationLinks> </AnimationLinks>
```

デフォルト要素値

ウィンドウ オブジェクト要素を指定しない場合、以下の表の要素がデフォルト値として使用されます。

要素	デフォルト値
FILLCOLOR	<R>212</R><G>208</G> 200
PENWIDTH	1
PENCOLOR	Black
PENSTYLE	Solid
CORNERDIMENSION	<Width>20</Width> <Height>20</Height>
TEXTCOLOR	Black
TEXTJUSTIFY	Left

要素	デフォルト値
ROTATION	0
FONT	System
FONTSTYLE	regular
FONTWEIGHT	無視されます
FONTSIZE	10
UNDERLINE	False
STRIKEOUT	False
FLIP	なし
TRANSPARENT	<R>0</R><G>255</G> 0

ペン スタイルの制限

実線以外のペン スタイルを指定した場合、ペン幅は **1** に設定されます。**1** 以上のペン幅を指定するには、ペン スタイル オプションを削除するか、**SOLID** に設定する必要があります。

ペンの寸法

寸法仕様はオブジェクトの中央線に適用されます。オブジェクトのペン幅が **1** よりも大きい場合、指定された寸法内にオブジェクトが収まらないことがあります。その場合、ペン幅がオブジェクトの境界をまたぎます。一部のピクセルがオブジェクトの境界内に配置され、その他のピクセルは外側に配置されます。

長方形オブジェクト

長方形の要素に対して以下の要素を指定できます。

要素	説明
FillColor	長方形の塗りつぶしの色。色要素で指定します。デフォルトは rgb(212, 208, 200) です。
Pen	Pen 要素。
Dimension	オブジェクトの位置とサイズ。要素は、 top 、 left 、 width 、 height です。 Top 、 left 、 width 、および height の値はピクセルです。結果の座標は -32000 ~ +32000 の間である必要があります。 width と height は両方とも 0 以外の値を指定する必要があります。必須。

要素	説明
	Dimension 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。
Title	オブジェクトの名前。オプション。
AnimationLinks	オプションのアニメーション リンク リスト。

例:

```
<Rectangle>
<Title>Rectangle1</Title>
<Pen>
<PenColor><Name>Black</Name></PenColor>
<PenWidth>4</PenWidth>
<PenStyle>Solid</PenStyle>
</Pen>
<Dimension>
<Left>100</Left> <Top>50</Top>
<Width>270</Width> <Height>80</Height>
</Dimension>
<FillColor>
<R>128</R> <G>128</G> <B>128</B>
</FillColor>
<AnimationLinks> </AnimationLinks>
</Rectangle>
```

最小の例:

```
<Rectangle>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
</Rectangle>
```

角丸長方形オブジェクト

角丸長方形オブジェクトに対して以下の要素を指定できます。

要素	説明
FillColor	塗りつぶしの色。色要素で指定します。デフォルトは rgb(212, 208, 200) です。

要素	説明
Pen	Pen 要素。
Dimension	<p>オブジェクトの位置とサイズ。要素は、top、left、width、height です。Top、left、width、および height の値はピクセルです。結果の座標は -32000 ～ +32000 の間である必要があります。width と height は両方とも 0 以外の値を指定する必要があります。必須。</p> <p>Dimension 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。</p>
CornerDimension	width および height 要素角の幅は長方形の幅以下である必要があります。角の高さは長方形の高さ以下である必要があります。角の幅と高さは 1 以上である必要があります。デフォルトは 20 、 20 です。
Title	オブジェクトの名前。オプション。
AnimationLinks	オプションのアニメーション リンク リスト。

例:

```
<RoundedRectangle>
<Title>RoundedRectangle1</Title>
<Pen>
<PenColor><Name>Black</Name></PenColor>
<PenWidth>4</PenWidth>
<PenStyle>Solid</PenStyle>
</Pen>
<Dimension>
<Left>100</Left>
<Top>50</Top>
<Width>270</Width>
<Height>80</Height>
</Dimension>
<FillColor>
<R>128</R> <G>128</G> <B>128</B>
</FillColor>
<AnimationLinks>
</AnimationLinks>
<CornerDimension>
<Width>8</Width>
<Height>8</Height>
</CornerDimension>
```

```
</RoundedRectangle>
```

最小の例:

```
<RoundedRectangle>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
</RoundedRectangle>
```

楕円形オブジェクト

楕円形オブジェクトに対して以下の要素を指定できます。

要素	説明
FillColor	塗りつぶしの色。色要素で指定します。デフォルトは rgb(212, 208, 200) です。
Pen	Pen 要素。
Dimension	<p>オブジェクトの位置とサイズ。要素は、top、left、width、height です。Top、left、width、および height の値はピクセルです。結果の座標は -32000 ~ +32000 の間である必要があります。width と height は両方とも 0 以外の値を指定する必要があります。必須。</p> <p>Dimension 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。</p>
Title	オブジェクトの名前。オプション。
AnimationLinks	オプションのアニメーション リンク リスト。

例:

```
<Ellipse>
<Title>Ellipse1</Title>
<Pen>
<PenColor><Name>Black</Name></PenColor>
<PenWidth>4</PenWidth>
<PenStyle>Solid</PenStyle>
</Pen>
<Dimension>
<Left>100</Left>
<Top>50</Top>
<Width>270</Width>
<Height>80</Height>
</Dimension>
<FillColor>
```

```
<R>128</R> <G>128</G> <B>128</B>
</FillColor>
<AnimationLinks>
</AnimationLinks>
</Ellipse>
```

最小の例:

```
<Ellipse>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
</Ellipse>
```

線オブジェクト

線オブジェクトに対して以下の要素を指定できます。開始ポイントと終了ポイントに同じポイントを指定することはできません。

要素	説明
Pen	Pen 要素
Title	オブジェクトの名前。オプション。
AnimationLinks	オプションのアニメーション リンク リスト。
ポイント	Point 要素。2 つ含む必要があります。余分な Point 要素は無視されます。

例:

```
<Line>
<Title>Line1</Title>
<Pen>
<PenColor><Name>Black</Name></PenColor>
<PenWidth>1</PenWidth>
<PenStyle>Solid</PenStyle>
</Pen>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>150</X><Y>160</Y></Point>
</Points>
<AnimationLinks>
</AnimationLinks>
</Line>
```

最小の例:

```
<Line>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>150</X><Y>160</Y></Point>
</Points> </Line>
```

水平線オブジェクト

水平線オブジェクトに対して以下の要素を指定できます。その場合、WindowMaker 水平/垂直線オブジェクトが作成されます。

要素	説明
Pen	Pen 要素。
Title	オブジェクトの名前。オプション。
AnimationLinks	オプションのアニメーション リンク リスト。
ポイント	2つのポイントを含む必要があります。2番目のポイントのY座標は無視され、最初のポイントのY座標に設定されます。余分な Point 要素は無視されます。必須。 Points 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。

例:

```
<HorizontalLine>
<Title>Line1</Title>
<Pen>
<PenColor><Name>Black</Name></PenColor>
<PenWidth>1</PenWidth>
<PenStyle>Solid</PenStyle>
</Pen>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>150</X><Y>150</Y></Point>
</Points>
<AnimationLinks>
</AnimationLinks>
</HorizontalLine>
```

最小の例:

```
<HorizontalLine>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>150</X><Y>150</Y></Point>
</Points>
</HorizontalLine>
```

垂直線オブジェクト

垂直線オブジェクトに対して以下の要素を指定できます。その場合、WindowMaker 水平/垂直線オブジェクトが作成されます。

要素	説明
Pen	Pen 要素。
Title	オブジェクトの名前。オプション。
AnimationLinks	オプションのアニメーション リンク リスト。
ポイント	<p>2つのポイントを含む必要があります。2番目のポイントのX座標は無視され、最初のポイントのX座標に設定されます。余分な Point 要素は無視されます。必須。</p> <p>Points 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。</p>

例:

```
<VerticalLine>
<Title>Line1</Title>
<Pen>
<PenColor><Name>Black</Name></PenColor>
<PenWidth>1</PenWidth>
<PenStyle>Solid</PenStyle>
</Pen>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>50</X><Y>250</Y></Point>
</Points>
<AnimationLinks>
</AnimationLinks>
<VerticalLine>
```

最小の例:

```
<VerticalLine>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>50</X><Y>250</Y></Point>
</Points>
<VerticalLine>
```

折れ線オブジェクト

折れ線オブジェクトをロードするには、少なくとも 2 つのポイントを定義する必要があります。2 つのポイントに同じ座標を使用することは推奨されません。

折れ線オブジェクトに対して以下の要素を指定できます。

要素	説明
ポイント	Point 要素。少なくとも 2 つ必要です。 Points 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。
Pen	Pen 要素。
Title	オブジェクトの名前。オプション。
AnimationLinks	オプションのアニメーション リンク リスト。塗りつぶしの色、テキスト色、塗りつぶしパーセント、値の表示は使用できません。

例:

```
<Polyline>
<Title>polyline1</Title>
<Pen>
<PenColor><Name>Black</Name></PenColor>
<PenWidth>1</PenWidth>
<PenStyle>Solid</PenStyle>
</Pen>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>60</X><Y>250</Y></Point>
<Point><X>70</X><Y>350</Y></Point>
<Point><X>80</X><Y>450</Y></Point>
</Points>
<AnimationLinks>
</AnimationLinks>
</Polyline>
```

最小の例:

```
<Polyline>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>80</X><Y>450</Y></Point>
</Points>
</Polyline>
```

多角形オブジェクト

多角形をロードするには、少なくとも 2 つのポイントを定義する必要があります。2 つのポイントに同じ座標を使用することは推奨されません。

多角形オブジェクトに対して以下の要素を指定できます。

要素	説明
ポイント	Point 要素を含みます。少なくとも 2 つのポイントが必要です。X および Y の値はピクセル単位です。

要素	説明
FillColor	多角形オブジェクトの塗りつぶしの色。1つの色要素を含みます。デフォルトは <code>rgb(212, 208, 200)</code> です。
Pen	Pen 要素。
Title	オブジェクトの名前。オプション。
AnimationLinks	オプションのアニメーションリンクリスト。テキスト色、値の表示のアニメーションリンクは使用できません。

例:

```
<Polygon>
<Title>polygon1</Title>
<Pen>
<PenColor><Name>Black</Name></PenColor>
<PenWidth>1</PenWidth>
<PenStyle>Solid</PenStyle>
</Pen>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>60</X><Y>250</Y></Point>
<Point><X>70</X><Y>350</Y></Point>
<Point><X>80</X><Y>450</Y></Point>
</Points>
<FillColor>
<R>128</R> <G>128</G> <B>128</B>
</FillColor>
<AnimationLinks>
</AnimationLinks>
</Polygon>
```

最小の例:

```
<Polygon>
<Points>
<Point><X>50</X><Y>150</Y></Point>
<Point><X>80</X><Y>450</Y></Point>
</Points>
</Polygon>
```

テキストオブジェクト

テキストオブジェクトに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。

要素	説明
TextString	表示されるテキスト文字列。必須。 TextString 要素に無効な値を割り当てた場合、または値がない場合、テキストオブジェクトは作成されません。
TextInfo	テキストの表示方法を定義します。
Rotation	テキストの回転を度で定義します。使用できる値は、 0 、 90 、 180 、 270 です。デフォルトは 0 です。
AnimationLinks	オプションのアニメーション リンク リスト。
Dimension	オブジェクトの位置とサイズ。要素は、 top 、 left 、 width 、 height です。 Top 、 left 、 width 、および height の値はピクセルです。結果の座標は -32000 ～ +32000 の間である必要があります。 width と height は両方とも 0 以外の値を指定する必要があります。必須。 Dimension 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。

例:

```
<Text>
<Title>text1</Title>
<TextString>これは表示するテキストです</TextString>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
<TextInfo>
<Font>Arial</Font>
<FontStyle>Regular</FontStyle>
<FontSize>12</FontSize>
<Underline>>false</Underline>
<Strikeout>>false</Strikeout>
<TextColor>
<R>0</R>
<G>0</G>
<B>0</B>
</TextColor>
<TextJustify>Left</TextJustify>
</TextInfo>
<Rotation>0</Rotation>
```



```
<AnimationLinks>  
</AnimationLinks>  
</Text>
```

最小の例:

```
<Text>  
<TextString>これは表示するテキストです</TextString>  
<Dimension>  
<Left>100</Left><Top>50</Top>  
<Width>270</Width><Height>80</Height>  
</Dimension>  
</Text>
```

ビットマップ オブジェクト

指定されたソース画像が存在する必要があります。サポートされるグラフィック ファイル形式は、JPG、JPEG、BMP、TIF、PCX、BIF、および TGA です。

指定されたソース画像のないビットマップは、デフォルトの長方形要素としてウィンドウに表示されます。

透明色

ビットマップに対する InTouch の内部アプリケーション オブジェクト データ構造には、透明色を適用することを指定するブール型フィールドが含まれています。実際の透明色は別のフィールドに格納されます。

WindowMaker で新しいビットマップを作成する場合、黒のデフォルト透明色が最初に設定されますが、これは使用できません。透明色ツールを選択して、ビットマップ オブジェクトに対して恒久的な透明色を適用します。透明色ツールからは、ビットマップに透明色が割り当てられているかどうかを判断することはできません。

XML ファイルからインポートされたビットマップ オブジェクトの場合、ビットマップの透明色ノードはオプションのエントリですが、存在する場合、透明色が有効化されてビットマップ オブジェクトに割り当てられます。エンドユーザーが WindowMaker でウィンドウを開いてビットマップ オブジェクトを選択し、透明色を無効化することはできません。

ビットマップ オブジェクト要素

ビットマップ オブジェクトに対して以下の要素を指定できます。

要素	説明
Title	ビットマップ オブジェクトの名前。オプション。
FillColor	ビットマップ オブジェクトの内部 RGB 色。1 つの色要素を含みます。デフォルトは <code>rgb(0, 0, 0)</code> です。
SourceImage	ビットマップ ファイルのパス。デフォルトは <code>Null</code> です。
透明	透明色の RGB 色。デフォルトは <code>rgb(0, 255, 0)</code> です。

要素	説明
Pen	ビットマップ オブジェクトの周囲の線を定義します。
AnimationLinks	オプションのアニメーション リンク リスト。
Dimension	<p>オブジェクトの位置とサイズ。要素は、top、left、width、height です。Top、left、width、および height の値はピクセルです。結果の座標は -32000 ～ +32000 の間である必要があります。width と height は両方とも 0 以外の値を指定する必要があります。必須。</p> <p>Dimension 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。</p>

例:

```
<Bitmap>
<Title>bitmap1</Title>
<Pen>
<PenColor><Name>Black</Name></PenColor>
<PenWidth>1</PenWidth>
<PenStyle>Solid</PenStyle>
</Pen>
<Dimension>
<Left>100</Left>
    <Top>50</Top>
<Width>270</Width>
    <Height>80</Height>
</Dimension>
<FillColor>
<R>128</R>
    <G>128</G> <
    B>128</B>
</FillColor>
<Transparent>
<R>0</R> <G>128</G> <B>255</B>
</Transparent>
<SourceImage>C:\MyPictures\hello.jpg</SourceImage>
<AnimationLinks>
</AnimationLinks>
</Bitmap>
```

例:

```
<Bitmap>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
```

```
<SourceImage>C:\MyPictures\hello.jpg</SourceImage>  
</Bitmap>
```

空の SourceImage ノードの例:

```
<Bitmap>  
<Dimension>  
<Left>100</Left><Top>50</Top>  
<Width>270</Width><Height>80</Height>  
</Dimension>  
<FillColor>  
<R>128</R> <G>128</G> <B>128</B>  
</FillColor>  
<SourceImage></SourceImage>  
</Bitmap>
```

最小の例:

```
<Bitmap>  
<Dimension>  
<Left>100</Left><Top>50</Top>  
<Width>270</Width><Height>80</Height>  
</Dimension>  
</Bitmap>
```

ボタンオブジェクト

ボタンオブジェクトに対して以下の要素を指定できます。

要素	説明
Title	ボタンオブジェクトの名前。オプション。
TextString	表示されるキャプション。デフォルト文字列は "Text" です。
TextInfo	キャプションの表示方法を定義します。
AnimationLinks	オプションのアニメーションリンクリスト。線の色、塗りつぶしの色、テキスト色、塗りつぶしパーセント、向きのアニメーションリンクは使用できません。

要素	説明
Dimension	<p>オブジェクトの位置とサイズ。要素は、top、left、width、height です。Top、left、width、および height の値はピクセルです。結果の座標は -32000 ～ +32000 の間である必要があります。width と height は両方とも 0 以外の値を指定する必要があります。必須。</p> <p>Dimension 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。</p>

例:

```
Button>
<Title>button1</Title>
<TextInfo>
<Font>Arial</Font>
<FontStyle>Regular</FontStyle>
<FontSize>12</FontSize>
<Underline>false</Underline>
<Strikeout>false</Strikeout>
<TextColor>
<R>0</R>
<G>0</G>
<B>0</B>
</TextColor>
<TextJustify>Left</TextJustify>
</TextInfo>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
<TextString>Stop All Robots</TextString>
<AnimationLinks>
</AnimationLinks>
</Button>
```

最小の例:

```
<Button>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
</Button>
```

SmartSymbol

SmartSymbol テンプレートを使用するウィンドウをインポートする前に、アプリケーション内で SmartSymbol テンプレートを定義する必要があります。SmartSymbol テンプレートが存在しない場合、インポートは失敗し、ウィンドウは作成されません。

1 つのウィンドウに対して複数の SmartSymbol を指定できます。各 SmartSymbol は個別の `<SmartSymbol>` `</SmartSymbol>` 要素内で宣言されます。

SmartSymbol に対して以下の要素を指定できます。

要素	説明
SymbolName	SmartSymbol の名前。必須。 名前がアプリケーション内の SmartSymbol テンプレート名に一致しない場合、または名前がない場合、オブジェクトは作成されません。
Dimension	オブジェクトの位置とサイズ。要素は、 <code>top</code> 、 <code>left</code> 、 <code>width</code> 、 <code>height</code> です。 <code>Top</code> 、 <code>left</code> 、 <code>width</code> 、および <code>height</code> の値はピクセルです。結果の座標は <code>-32000</code> ～ <code>+32000</code> の間である必要があります。 <code>width</code> と <code>height</code> は両方とも <code>0</code> 以外の値を指定する必要があります。必須。 Dimension 要素に無効な値を割り当てた場合、または値がない場合、オブジェクトは作成されません。
TagReplace	インスタンス タグの置換。
StringReplace	インスタンス文字列の置換。

例:

```
<SmartSymbol>
<SymbolName>MyCoolSymbol</SymbolName>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
<TagReplace>
<Find>Tag001</Find>
<Replace>Tag007</Replace>
</TagReplace>
<StringReplace>
<Find>Find This Text</Find>
<Replace>Replace it with this text</Replace>
</StringReplace>
</SmartSymbol>
```

最小の例:

```
<SmartSymbol>
<SymbolName>MyCoolSymbol</SymbolName>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
</SmartSymbol>
```

タグの置換

SmartSymbol インスタンス内のタグを置換できます。アプリケーション内に現在存在する SmartSymbol インスタンス内のタグのみを置換してください。

タグの置換では大文字と小文字が区別されず、タグ名文字列全体が一致する必要があります。1 つまたは複数の <TagReplace> 要素を SmartSymbol ノートに追加して複数のタグを置換できます。

次に例を示します。

```
<TagReplace>
<Find>Tag1</Find>
<Replace>DTagA</Replace>
</TagReplace>
<TagReplace>
<Find></Find>
<Replace></Replace>
</TagReplace>
```

置換に指定されたタグ名がアプリケーション内に存在しない場合、SmartSymbol は作成されません。また、SmartSymbol を含むウィンドウは作成されません。

置換後のタグのタイプは、元のタグのタイプと同じである必要があります。

文字列の置換

SmartSymbol インスタンス内の文字列を置換できます。文字列の置換では大文字と小文字が区別され、ソース文字列全体が一致する必要があります。単一の SmartSymbol 要素内で複数の文字列置換要素を使用できます。

```
<StringReplace>
<FIND>
</FIND>
<REPLACE>
</REPLACE>
</StringReplace>
<StringReplace>
<FIND>Open</FIND>
<REPLACE>Off</REPLACE>
</StringReplace>
```

SmartSymbol の例

タグおよび文字列の置換要素を含む SmartSymbol 要素の例を以下に示します。

```
<SmartSymbol>
<SymbolName>MyCoolSymbol</SymbolName>
<Dimension>
<Left>100</Left><Top>50</Top>
<Width>270</Width><Height>80</Height>
</Dimension>
```

```
<TagReplace>
<Find>DTag1</Find>
<Replace>DTagA</Replace>
</TagReplace>
<TagReplace>
<Find>ATag001</Find>
<Replace>ATag002</Replace>
</TagReplace>
<StringReplace>
<Find> </Find>
<Replace> </Replace>
</StringReplace>
<StringReplace>
<FIND>
</FIND>
<REPLACE>
</REPLACE>
</StringReplace>
</SmartSymbol>
```

産業用グラフィック

産業用グラフィック参照を使用するウィンドウをインポートする前に、**Galaxy** 内で産業用グラフィックを定義する必要があります。産業用グラフィック参照が存在しない場合、インポートは失敗し、ウィンドウは作成されません。

シンボル参照内のインスタンスまたはグラフィック ツールボックスからシンボルを指定できます。しかし、シンボル参照内のテンプレートシンボルを指定することはできません。

1 つのウィンドウに対して複数の産業用グラフィックを指定できます。各産業用グラフィックは個別の `<Industrial Graphic>` `</Industrial Graphic>` 要素内で宣言されます。

産業用グラフィックに対して以下の要素を指定できます。

要素	説明
SymbolReference	<p>Galaxy 内のシンボルを検索する参照。必須。</p> <p>シンボル参照によって指定された産業用グラフィックが存在しない場合、またはシンボル参照フィールドが欠落している場合や空の場合、オブジェクトは作成されません。</p>

要素	説明
Dimension	<p>オブジェクトの位置とサイズ。 Dimension サブ要素は、top、left、width、および height です。必須。</p> <p>width と height の指定はオプションです。width と height を指定しない場合、産業用グラフィックの元の幅と高さが使用されます。width だけを指定した場合、縦横比を使用して height が計算されます。height だけを指定した場合も同様に width が計算されます。たとえば、元の幅が 200 で、高さが 100 の場合、width に 100 を指定すると、height は 50 になります。</p> <p>Dimension 要素に無効な値を割り当てた場合、または値が含まれない場合、オブジェクトは作成されません。このシンボルを含むウィンドウも作成されません。</p>
Title	<p>要素のテキスト名。オプション。</p> <p>この名前は XML ファイル内で一意である必要があります。そうでない場合は無視されます。</p>
Flip	要素の反転タイプ。オプション。
Rotation	要素の回転角度。オプション。
StringReplace	1 つまたは複数の文字列置換ノード。オプション。
CustomPropertyOverride	1 つまたは複数のカスタム プロパティの上書き。オプション。
AnimationLinks	<p>アニメーション リンクのリスト。オプション。</p> <p>線の色、塗りつぶしの色、テキスト色、塗りつぶしパーセント、向き、スライダー、ツールヒン</p>

要素	説明
	ト、値の表示、点滅、およびユーザー入力は使用できません。

```

<ArchestrASymbol>
<Title>EmbedSym1</Title>
<Dimension>
  <Left>200</Left>
  <Top>200</Top>
  <Width>150</Width>
  <Height>150</Height>
</Dimension>
<Flip>None</Flip>
<Rotation>0</Rotation>

  <SymbolReference>ButtonChromeMomentaryRed</SymbolReference>
  <AnimationLinks>
  </AnimationLinks>
  <StringReplace>
    <Find>LABEL</Find>
    <Replace>OFF</Replace>
  </StringReplace>
</ArchestrASymbol>

```

最小の例:

```

<ArchestrASymbol>
<Dimension>
  <Left>200</Left>
  <Top>200</Top>
</Dimension>
  <SymbolReference>ButtonChromeMomentaryRed</SymbolReference>
</ArchestrASymbol>

```

CustomPropertyOverride

上書きできるのは、産業用グラフィックに対して既に定義されているカスタムプロパティだけです。

複数のカスタムプロパティを上書きするには、1 つまたは複数の <CustomPropertyOverride> 要素を産業用グラフィック ノードに追加します。

例:

```

<ArchestrASymbol>
  <Title>EmbedSym1</Title>
  <Dimension>
    <Left>200</Left>
    <Top>200</Top>
    <Width>150</Width>
    <Height>150</Height>

```

```

</Dimension>
<Flip>None</Flip>
<Rotation>0</Rotation>
<SymbolReference>ButtonChromeMomentaryRed</SymbolReference>
<AnimationLinks>
</AnimationLinks>
<CustomPropertyOverride> <CustomPropertyName>cp1</CustomPropertyName>
  <OverrideValue>DTagA</OverrideValue>
  <IsConstant>>false</IsConstant>
</CustomPropertyOverride>
</ArchestraSymbol>

```

この例では、cp1 は既存のカスタムプロパティの名前です。上書きは、新しい値が DTagA に設定されたカスタムプロパティに適用されます。IsConstant はオプションのフィールドで、値を定数として解釈するかどうかを指定します。IsConstant フラグは、カスタムプロパティのタイプが文字列、時刻、または経過時間の場合にのみ適用できます。デフォルトでは、IsConstant フラグは false に設定されます。

注: OverrideValue に対して指定されたタグ名がタグデータベースに存在しない場合、産業用グラフィック参照はウィンドウに作成されず、その特定のウィンドウのインポートは失敗します。エラーメッセージがログファイルまたはロガーに記録されます。

産業用グラフィック文字列置換タイプ

産業用グラフィック インスタンス内の既存の文字列を置換できます。文字列の置換では大文字と小文字が区別されます。単一の産業用グラフィック ノード内で複数の文字列置換ノードを使用できます。

```

<StringReplace>
<FIND>
</FIND>
<REPLACE>
</REPLACE>
</StringReplace>
<StringReplace>
<FIND>Open</FIND>
<REPLACE>Off</REPLACE>
</StringReplace>

```

サポートされないウィンドウ オブジェクト

セル、リアルタイムトレンド、および履歴トレンドオブジェクトはインポートできません。サポートされないオブジェクトの要素が XML ファイルに含まれる場合、その要素は無視されます。

XML ユーティリティを使用したウィンドウのインポート

XML ユーティリティを使用してウィンドウをインポートする際に、閉じるボタンのスクリプトを XML スクリプトに追加することができます。

注意： インポートするウィンドウは、バッチファイル（wm.bat）で指定します。

```
<CloseButton>True</CloseButton>
```

XML ファイルにスクリプトが含まれていると、インポートするウィンドウのタイトルバーにウィンドウを閉じるボタンが表示されます。

ウィンドウ アニメーション リンク

ウィンドウ オブジェクト アニメーション リンクは <AnimationLinks> </AnimationLinks> 要素内で宣言されます。

1つのウィンドウ要素内に複数のアニメーションリンクを設定できますが、必ずしも設定する必要はありません。

すべての要素ですべてのアニメーションリンクがサポートされるわけではありません。産業用グラフィックは、以下のアニメーションリンクをサポートします。

- ObjSize_Height
- ObjSize_Width
- Location_Vert
- Location_Hori
- TPushB_Disc
- TPushB_Action
- TPushB_ShowWin
- TPushB_HideWin
- Misc_Visib
- Misc_Disable

ウィンドウアニメーションリンクの詳細については、InTouch HMIのマニュアルを参照してください。

アニメーションリンクの一部のタイプでは、その他のタイプのアニメーションリンクを作成できません。アニメーションリンクの処理は、XMLファイルで指定された順序で行われます。

スクリプト/式/タグ名要件のマトリックス

各アニメーションリンクには1つのコントロールフィールドがあります。コントロールフィールドは、スクリプト、式、またはタグ名です。一部のコントロールフィールドは、使用できるタグまたは式のタイプによって制限されています。以下の表には、各アニメーションリンクに必要なコントロールフィールドとコントロールフィールドの制限が一覧表示されています。

アニメーションリンク タイプ	コントロールフィールド	コントロールフィールドの制限
論理型ユーザー入力	タグ	論理型タグ
アナログユーザー入力	タグ	アナログ型タグ
文字列ユーザー入力	タグ	文字列タグ
論理線の色	式	論理型式
アナログ線の色	式	アナログ式
論理型アラーム線の色	タグ	論理型タグ アラーム ステータス

アニメーションリンク タイプ	コントロール フィールド	コントロール フィールドの制限
アナログ アラーム 線の色	タグ	アナログ型タグ アラーム ステータス
論理塗りつぶしの色	式	論理型式
アナログ塗りつぶしの色	式	アナログ式
論理型アラーム塗りつぶしの色	タグ	論理型タグ アラーム ステータス
アナログ アラーム 塗りつぶしの色	タグ	アナログ型タグ アラーム ステータス
論理テキスト色	式	論理型式
アナログ テキスト 色	式	アナログ式
論理型アラーム テキスト色	タグ	論理型タグ アラーム ステータス
アナログ アラーム テキスト色	タグ	アナログ型タグ アラーム ステータス
垂直スライダー	タグ	有効なアナログ型タグ
水平スライダー	タグ	有効なアナログ型タグ
オブジェクト サイズ高さ	式	アナログ値
オブジェクト サイズ幅	式	アナログ値
垂直位置	式	有効な式
水平位置	式	有効な式
垂直塗りつぶしパーセント	式	アナログ値
水平塗りつぶしパーセント	式	アナログ値
論理値タッチ押しボタン	タグ	論理値

アニメーション リンク タイプ	コントロール フィールド	コントロール フィールドの 制限
アクション タッチ 押しボタン	スクリプト	有効なスクリプト
ウィンドウの表示 タッチ押しボタン	ウィンドウ名	ウィンドウが存在する必要があります。
ウィンドウの非表示 タッチ押しボタン	ウィンドウ名	ウィンドウが存在する必要があります。
表示オン/オフ	式	論理値
点滅	式	論理値
向き	式	アナログ値
無効化	式	論理値
ツールヒント	式	文字列タグ
論理値表示	式	論理型式
アナログ値表示	式	アナログ式
文字列値表示	式	文字列式

論理型ユーザー入力

論理型ユーザー入力アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Message	ユーザーに表示するメッセージ テキスト。デフォルトではメッセージ テキストはありません。オプションのスキーマを使用する場合に必要です。
InputOnly	入力のみかどうかを指定します。 true 、 false 。デフォルトは false です。
OnMessage	ユーザーに表示するオン メッセージ テキスト。デフォルトは On です。空のテキストを指定することはできません。
OffMessage	ユーザーに表示するオフ メッセージ テキスト。デフォルトは Off です。空

要素	説明
	のテキストを指定することはできません。
SetPrompt	ユーザーに表示するセットプロンプトメッセージテキスト。デフォルトは On です。空のテキストを指定することはできません。
ResetPrompt	ユーザーに表示するリセットプロンプトメッセージテキスト。デフォルトは Off です。空のテキストを指定することはできません。
KeyAssignment	仮想キー要素。デフォルトでは割り当ては設定されません。空の文字列は割り当てが発生しないことを示します。
Expression	論理型タグ。必須。無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<UserInputDiscrete>
<Title>UserInputDiscrete1</Title>
<InputOnly>false</InputOnly>
<KeyAssignment>
<KeyCode>F1</KeyCode>
<KeyFlags>Ctrl</KeyFlags>
</KeyAssignment>
<Message>Pump Valve State</Message>
<Expression></Expression>
<OnMessage>On Message Text</OnMessage>
<OffMessage>Off Message Text</OffMessage>
<ResetPrompt>
</ResetPrompt>
<SetPrompt>Set Prompt Text</SetPrompt>
</UserInputDiscrete>
```

最小の例:

```
<UserInputDiscrete>
<Message>Pump Valve State</Message>
<Expression>dTag001</Expression>
</UserInputDiscrete>
```

アナログ ユーザー入力

アナログ ユーザー入力アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。

要素	説明
Message	ユーザーに表示するメッセージテキスト。オプションのスキーマを使用する場合に必要です。デフォルトではメッセージテキストはありません。
InputOnly	入力のみかどうかを指定します (true、false)。デフォルトは false です。
MinAnalogValue	使用できる最小の浮動小数点値。オプションのスキーマを使用する場合に必要です。デフォルトは 0.0 です
MaxAnalogValue	使用できる最大の浮動小数点値。オプションのスキーマを使用する場合に必要です。デフォルトは 100.0 です。 MINANALOGVALUE 要素に割り当てられた値よりも大きい値である必要があります。
KeyPadEnabled	キーパッドを表示するかどうかを指定します。使用できる値は True または False です。デフォルトは False です。
KeyAssignment	仮想キー要素、空の文字列、または存在しない。デフォルトでは割り当ては設定されません。空の文字列は割り当てが発生しないことを示します。
Expression	アナログ型タグ名。必須。無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<<UserInputAnalog>  
<Title>UserInputAnalog1</Title>  
<InputOnly>false</InputOnly>  
<KeyAssignment>  
<KeyCode>F1</KeyCode>  
<KeyFlags>Ctrl</KeyFlags>  
</KeyAssignment>  
<Message>Flush Pump Speed</Message>  
<Expression>aTag001</Expression>  
<KeyPadEnabled>false</KeyPadEnabled>
```

```
<MinAnalogValue>0.0</MinAnalogValue>  
<MaxAnalogValue> 100.0</MaxAnalogValue>  
</UserInputAnalog>
```

最小の例:

```
<UserInputAnalog>  
<Message>Pump Valve State</Message>  
<Expression></Expression>  
<MinAnalogValue>0.0</MinAnalogValue>  
<MaxAnalogValue>100.0</MaxAnalogValue>  
</UserInputAnalog>
```

文字列ユーザー入力

EchoEnabled または **EchoMode** のいずれかを使用します。両方使用することはできません。**EchoMode** ではパスワードモードを指定できます。**EchoEnabled** では、有効または無効の状態のみを指定できます。

ランタイム時にユーザー入力の表示方法を制御するには、エコー文字要素を使用します。使用できる値は、**Yes**、**No**、および **Password** です。

- 要素を **Yes** に設定すると、文字列がランタイム時に入力編集ボックスに表示されます。入力が有効になります。パスワード文字と暗号化は無効になります。
- 要素を **No** に設定すると、入力文字はランタイム時に入力編集ボックスに表示されません。入力が有効になります。パスワード文字と暗号化は無効になります。
- 要素を **Password** に設定すると、ユーザーが入力されたパスワードではなく、パスワード文字がランタイム時に表示されます。パスワード文字はオプションですが、指定した場合は空にすることはできません。デフォルトのパスワード文字はアスタリスクです。暗号化はオプションで、デフォルトで無効化されています。「入力のみ」は必須で、有効化されています。

要素が上記の条件を満たさない場合、デフォルトのオプションが使用されます。

文字列ユーザー入力アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Message	ユーザーに表示するメッセージテキスト。オプションのスキーマを使用する場合に必要です。デフォルトではメッセージテキストはありません。
InputOnly	入力のみかどうかを指定します。 true 、 false 。デフォルトは false です。
EchoCharacters	エコー文字 (no 、 yes 、 password)。オプションのスキーマを使用する場合に必要です。デフォルトは yes です。

要素	説明
KeyPadEnabled	キーパッドを表示するかどうかを指定します (true、false)。デフォルトは false です。
PasswordCharacter	パスワード文字。デフォルトは "*" です。
EncryptEnabled	暗号化の有効化 (yes、no)。デフォルトは no です。
KeyAssignment	仮想キー要素、空の文字列、または存在しない。デフォルトでは割り当ては設定されません。空の文字列は割り当てが発生しないことを示します。
Expression	メッセージ型タグ。必須。無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<UserInputString>
<Title>UserInputString1</Title>
<InputOnly>false</InputOnly>
<KeyAssignment>
<KeyCode>F1</KeyCode>
<KeyFlags>Ctrl</KeyFlags>
</KeyAssignment>
<Message>Select Pump</Message>
<Expression>mTag001</Expression>
<EchoCharacters>>true</EchoCharacters>
<EncryptEnabled>false</EncryptEnabled>
<PasswordCharacter>*</PasswordCharacter>
</UserInputString>
```

最小の例:

```
<UserInputString>
<Message>Select Pump</Message>
<Expression>mTag001</Expression>
<EchoCharacters>>true</EchoCharacters>
</UserInputString>
```

論理線の色

論理線の色アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Oncolor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。

要素	説明
Offcolor	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Expression	論理型タグまたは式。必須。 Expression 要素値がない場合、または無効な場合、オブジェクトは作成されません。

例:

```
<LineColorDiscrete>
<Title>LineColorDiscrete1</Title>
<Expression>
</Expression>
<OnColor><Name>Green</Name></OnColor>
<OffColor><Name>Red</Name></OffColor>
</LineColorDiscrete>
```

最小の例:

```
<LineColorDiscrete>
<Expression>dTag001</Expression>
</LineColorDiscrete>
```

アナログ線の色

必須のブレークポイント値には増分値が必要です。そうでない場合、警告がログに記録されます。無効な値は、以前の値に **1** を追加したものです。

必須の値がない場合または無効な場合、アニメーションリンクを含むオブジェクトは作成されません。

アナログ線の色アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Color1	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color2	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color3	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color4	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color5	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color6	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color7	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。

要素	説明
Color8	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Color9	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Color10	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Values	定数アナログ値。9つの Value 要素を含む必要があります。必須。
Expression	アナログ型タグまたは式。必須。

例:

```
<LineColorAnalog>
<Title>LineColorAnalog1</Title>
<Expression>aTag001</Expression>
<Colors>
<Color1><Name>White</Name></Color1>
<Color2><Name>Red</Name></Color2>
<Color3><Name>Orange</Name></Color3>
<Color4><Name>Yellow</Name></Color4>
<Color5><Name>Green</Name></Color5>
<Color6><Name>Blue</Name></Color6>
<Color7><Name>Cyan</Name></Color7>
<Color8><Name>Magenta</Name></Color8>
<Color9><Name>Violet</Name></Color9>
<Color10><Name>Black</Name></Color10>
</Colors>
<Values>
<Value>10.0</Value>
<Value>20.0</Value>
<Value>30.0</Value>
<Value>40.0</Value>
<Value>50.0</Value>
<Value>60.0</Value>
<Value>70.0</Value>
<Value>80.0</Value>
<Value>90.0</Value>
</Values>
</LineColorAnalog>
```

最小の例:

```
<LineColorAnalog>
<Expression>aTag001</Expression>
<Values>
<Value>10.0</Value>
<Value>20.0</Value>
<Value>30.0</Value>
<Value>40.0</Value>
<Value>50.0</Value>
```

```

<Value>60.0</Value>
<Value>70.0</Value>
<Value>80.0</Value>
<Value>90.0</Value>
</Values>
</LineColorAnalog>

```

論理型アラーム線の色

論理型アラーム線の色アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
NormalColor	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
AlarmColor	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Expression	論理型タグ。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```

<LineColorDiscreteAlarm>
<Title>LineColorDiscreteAlarm1</Title>
<Expression>
</Expression>
<NormalColor><Name>Black</Name></NormalColor>
<AlarmColor><Name>Red</Name></AlarmColor>
</LineColorDiscreteAlarm>

```

最小の例:

```

<LineColorDiscreteAlarm>
<Expression>dTag001</Expression>
</LineColorDiscreteAlarm>

```

アナログアラーム線の色

アナログアラーム線の色アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Value: LOLOCOLOR	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Value: LOCOLOR	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Value: NORMALCOLOR	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。

要素	説明
Value:HICOLOR	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Value: HIHICOLOR	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Deviation: NORMALCOLOR	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Deviation: MINORCOLOR	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Deviation: MAJORCOLOR	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
ROC: NORMALCOLOR	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
ROC: ROCCOLOR	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Expression	アナログ型タグ。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

値アラームの例:

```
<LineColorAnalogAlarm>
<Title>LineColorAnalogAlarm1</Title>
<Expression>
</Expression>
<Value>
<LoLoColor><Name>Red</Name></LoLoColor>
<LoColor><Name>DarkRed</Name></LoColor>
<NormalColor><Name>Black</Name></NormalColor>
<HiColor><Name>DarkGreen</Name></HiColor>
<HiHiColor><Name>Green</Name></HiHiColor>
</Value>
</LineColorAnalogAlarm>
```

偏差アラームの例:

```
<LineColorAnalogAlarm>
<Title>LineColorAnalogAlarm1</Title>
<Expression>
</Expression>
  <Deviation>
<NormalColor><Name>Black</Name></NormalColor>
<MinorColor><Name>Green</Name></MinorColor>
<MajorColor><Name>Red</Name></MajorColor>
</Deviation>
</LineColorAnalogAlarm>
```

ROC アラームの例:

```
<LineColorAnalogAlarm>
<Title>LineColorAnalogAlarm1</Title>
<Expression>
</Expression>
<ROC>
<NormalColor><Name>Black</Name></NormalColor>
<ROCColor><Name>Red</Name></ROCColor>
</ROC>
</LineColorAnalogAlarm>
```

論理塗りつぶしの色

論理塗りつぶしの色アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
OnColor	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
OffColor	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Expression	論理型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<FillColorDiscrete>
<Title>FillColorDiscrete1</Title>
<Expression>
</Expression>
<OnColor><Name>Green</Name></OnColor>
<OffColor><Name>Red</Name></OffColor>
</FillColorDiscrete>
```

最小の例:

```
<FillColorDiscrete>
<Expression>dTag001</Expression>
</FillColorDiscrete>
```

アナログ塗りつぶしの色

必須の値がない場合または無効な場合、アニメーションリンクを含むオブジェクトは作成されません。
論理型塗りつぶしの色アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Color1	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color2	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。

要素	説明
Color3	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Color4	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Color5	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Color6	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Color7	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Color8	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Color9	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Color10	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Values	定数アナログ値。9 つの Value 要素を含む必要があります。必須。
Expression	アナログ型タグまたは式。必須。

例:

```
<FillColorAnalog>
<Title>FillColorAnalog1</Title>
<Expression>aTag001</Expression>
<Colors>
<Color1><Name>White</Name></Color1>
<Color2><Name>Red</Name></Color2>
<Color3><Name>Orange</Name></Color3>
<Color4><Name>Yellow</Name></Color4>
<Color5><Name>Green</Name></Color5>
<Color6><Name>Blue</Name></Color6>
<Color7><Name>Cyan</Name></Color7>
<Color8><Name>Magenta</Name></Color8>
<Color9><Name>Violet</Name></Color9>
<Color10><Name>Black</Name></Color10>
</Colors>
<Values>
<Value>10.0</Value>
<Value>20.0</Value>
<Value>30.0</Value>
<Value>40.0</Value>
<Value>50.0</Value>
<Value>60.0</Value>
```

```
<Value>70.0</Value>
<Value>80.0</Value>
<Value>90.0</Value>
</Values>
</FillColorAnalog>
```

最小の例:

```
<FillColorAnalog>
<Expression>aTag001</Expression>
<Values>
<Value>10.0</Value>
<Value>20.0</Value>
<Value>30.0</Value>
<Value>40.0</Value>
<Value>50.0</Value>
<Value>60.0</Value>
<Value>70.0</Value>
<Value>80.0</Value>
<Value>90.0</Value>
</Values>
</FillColorAnalog>
```

論理型アラーム塗りつぶしの色

論理型アラーム塗りつぶしの色アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
NormalColor	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
AlarmColor	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Expression	論理型タグ。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<FillColorDiscreteAlarm>
<Title>FillColorDiscreteAlarm1</Title>
<Expression>
</Expression>
<NormalColor><Name>Black</Name></NormalColor>
<AlarmColor><Name>Red</Name></AlarmColor>
</FillColorDiscreteAlarm>
```

最小の例:

```
<FillColorDiscreteAlarm>
<Expression>dTag001</Expression>
</FillColorDiscreteAlarm>
```


アナログアラーム塗りつぶしの色

アナログアラーム塗りつぶしの色アニメーションリンクに対して以下の要素を指定できます。必須の値がない場合または無効な場合、アニメーションリンクを含むオブジェクトは作成されません。

要素	説明
Title	オブジェクトの名前。オプション。
Value: LoLoColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Value: LoColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Value: NormalColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Value: HiColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Value: HiHiColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Deviation: NormalColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Deviation: MinorColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Deviation: MajorColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
ROC: NormalColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
ROC: ROCColor	1つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Expression	アナログ型タグ。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

値アラームの例:

```
<FillColorAnalogAlarm>
<Title>FillColorAnalogAlarm1</Title>
<Expression>
</Expression>
<Value>
<LoLoColor><Name>Red</Name></LoLoColor>
<LoColor><Name>DarkRed</Name></LoColor>
<NormalColor><Name>Black</Name></NormalColor>
<HiColor><Name>DarkGreen</Name></HiColor>
<HiHiColor><Name>Green</Name></HiHiColor>
</Value>
```

```
</FillColorAnalogAlarm>
```

偏差アラームの例:

```
<FillColorAnalogAlarm>
<Title>FillColorAnalogAlarm1</Title>
<Expression>
</Expression>
<Deviation>
<NormalColor><Name>Black</Name></NormalColor>
<MinorColor><Name>Green</Name></MinorColor>
<MajorColor><Name>Red</Name></MajorColor>
</Deviation>
</FillColorAnalogAlarm>
```

ROC アラームの例:

```
<FillColorAnalogAlarm>
<Title>FillColorAnalogAlarm1</Title>
<Expression>
</Expression>
<ROC>
<NormalColor><Name>Black</Name></NormalColor>
<ROCColor><Name>Red</Name></ROCColor>
</ROC>
</FillColorAnalogAlarm>
```

論理テキスト色

論理テキスト色アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
OnColor	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
OffColor	1 つの色要素を含みます。デフォルトは <code>rgb(0,0,0)</code> です。
Expression	文字列タグまたは式。必須。

例:

```
<TextColorDiscrete>
<Title>TextColorDiscrete1</Title>
<Expression>
</Expression>
<OnColor><Name>Green</Name></OnColor>
<OffColor><Name>Red</Name></OffColor>
</TextColorDiscrete>
```

最小の例:

```
<TextColorDiscrete>
<Expression>dTag001</Expression>
</TextColorDiscrete>
```

アナログテキスト色

アナログテキスト色アニメーションリンクに対して以下の要素を指定できます。必須の値がない場合または無効な場合、アニメーションリンクを含むオブジェクトは作成されません。

要素	説明
Title	オブジェクトの名前。オプション。
Color1	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color2	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color3	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color4	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color5	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color6	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color7	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color8	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color9	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Color10	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Values	定数アナログ値。9つの Value 要素を含む必要があります。必須。
Expression	アナログ型タグまたは式。必須。

例:

```
<TextColorAnalog>  
<Title>TextColorAnalog1</Title>  
<Expression>aTag001</Expression>  
<Colors>  
<Color1><Name>White</Name></Color1>  
<Color2><Name>Red</Name></Color2>  
<Color3><Name>Orange</Name></Color3>  
<Color4><Name>Yellow</Name></Color4>  
<Color5><Name>Green</Name></Color5>  
<Color6><Name>Blue</Name></Color6>
```

```

<Color7><Name>Cyan</Name></Color7>
<Color8><Name>Magenta</Name></Color8>
<Color9><Name>Violet</Name></Color9>
<Color10><Name>Black</Name></Color10>
</Colors>
<Values>
<Value>10.0</Value> <Value>20.0</Value>
<Value>30.0</Value> <Value>40.0</Value>
<Value>50.0</Value> <Value>60.0</Value>
<Value>70.0</Value> <Value>80.0</Value>
<Value>90.0</Value>
</Values>
</TextColorAnalog>

```

最小の例:

```

<TextColorAnalog>
<Expression>aTag001</Expression>
<Values>
<Value>10.0</Value>
<Value>20.0</Value>
<Value>30.0</Value>
<Value>40.0</Value>
<Value>50.0</Value>
<Value>60.0</Value>
<Value>70.0</Value>
<Value>80.0</Value>
<Value>90.0</Value>
</Values>
</TextColorAnalog>

```

論理型アラーム テキスト色

論理型アラーム テキスト色アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
NormalColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
AlarmColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Expression	論理型タグ。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```

<TextColorDiscreteAlarm>
<Title>TextColorDiscreteAlarm1</Title>
<Expression>
</Expression>
<NormalColor><Name>Black</Name></NormalColor>
<AlarmColor><Name>Red</Name></AlarmColor>

```

```
</TextColorDiscreteAlarm>
```

最小の例:

```
<TextColorDiscreteAlarm>  
<Expression>dTag001</Expression>  
</TextColorDiscreteAlarm>
```

アナログアラーム テキスト色

アナログアラーム テキスト色アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Value: LoLoColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Value: LoColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Value: NormalColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Value: HiColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Value: HiHiColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Deviation: NormalColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Deviation: MinorColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Deviation: MajorColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
ROC: NormalColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
ROC: ROCColor	1 つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Expression	アナログ型タグ。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<TextColorAnalogAlarm>  
<Title>TextColorAnalogAlarm1</Title>  
<Expression>  
</Expression>  
<Value>  
<LoLoColor><Name>Red</Name><LoLoColor>
```

```
<LoColor><Name>DarkRed</Name></LoColor>
<NormalColor><Name>Black</Name></NormalColor>
<HiColor><Name>DarkGreen</Name></HiColor>
<HiHiColor><Name>Green</Name></HiHiColor>
</Value>
</TextColorAnalogAlarm>
```

偏差型アラームの例:

```
<TextColorAnalogAlarm>
<Title>TextColorAnalogAlarm1</Title>
<Expression>
</Expression>
<Deviation>
<NormalColor><Name>Black</Name></NormalColor>
<MinorColor><Name>Green</Name></MinorColor>
<MajorColor><Name>Red</Name></MajorColor>
</Deviation>
</TextColorAnalogAlarm>
```

ROC 型アラームの例:

```
<TextColorAnalogAlarm>
<Title>TextColorAnalogAlarm1</Title>
<Expression>
</Expression>
<ROC>
<NormalColor><Name>Black</Name></NormalColor>
<ROCColor><Name>Red</Name></ROCColor>
</ROC>
</TextColorAnalogAlarm>
```

垂直スライダー

垂直スライダー アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
ReferenceLocation	垂直参照: top 、 middle 、 bottom 。デフォルトは bottom です。
TopValue	上の値。BottomValue と同じ値にすることはできません。デフォルトは 0 です。
BottomValue	下の値。デフォルトは 100 です。
UpwardMovement	上への移動。値の範囲は 0 から 32767 です。範囲外の値は固定され、エラー メッセージが記録されます。デフォルトは 50 です。

要素	説明
DownwardMovement	下への移動。値の範囲は 0 から 32767 です。範囲外の値は固定され、エラーメッセージが記録されます。デフォルトは 50 です。
Expression	アナログ型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<SliderVertical>  
<Title>SliderVertical1</Title>  
<ReferenceLocation>Bottom</ReferenceLocation>  
<TopValue>10.0</TopValue>  
<BottomValue> 110.0</BottomValue>  
<UpwardMovement> 20.0</UpwardMovement>  
<DownwardMovement> 120.0</DownwardMovement>  
<Expression> </Expression>  
</SliderVertical>
```

最小の例:

```
<SliderVertical>  
<Expression>aTag001</Expression>  
</SliderVertical>
```

水平スライダー

水平スライダー アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
ReferenceLocation	参照位置: left、center、right。デフォルトは left です。
LeftValue	スライダーの左の値。デフォルトは 0 です。
RightValue	スライダーの右の値。デフォルトは 100 です。
LeftMovement	左の水平移動。値の範囲は 0 から 32767 です。範囲外の値は固定され、エラーメッセージが記録されます。デフォルトは 50 です。
RightMovement	右の水平移動。値の範囲は 0 から 32767 です。範囲外の値は固定され、エラーメッセージが記録されます。デフォルトは 50 です。

要素	説明
Expression	アナログ型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<SliderHorizontal>  
<Title>SliderHorizontal1</Title>  
<ReferenceLocation>Left</ReferenceLocation>  
<LeftValue>10.0</LeftValue>  
<RightValue>120.0</RightValue>  
<LeftMovement>20.0</LeftMovement>  
<RightMovement>150.0</RightMovement>  
<Expression>  
</Expression>  
</SliderHorizontal>
```

最小の例:

```
<SliderHorizontal>  
<Expression></Expression>  
</SliderHorizontal>
```

オブジェクト高さ

オブジェクト高さアニメーションリンクに対して以下の要素を指定できます。オブジェクト高さアニメーションリンクは向きアニメーションリンクと一緒に使用できません。

要素	説明
Title	オブジェクトの名前。オプション。
SizeAnchor	オブジェクト上でアンカーを配置する位置を定義します。値は、 bottom 、 middle 、 top です。デフォルトは bottom です。
SizeMin	最小高さの値。デフォルトは 0 です。
SizeMax	最大高さの値。デフォルトは 100 です。SizeMin 要素に割り当てられた値よりも大きい値である必要があります。
MinPercent	高さの最小パーセンテージ。デフォルトは 0 です。範囲は 0 から 100 です。
MaxPercent	高さの最大パーセンテージ。デフォルトは 100 です。最小パーセントよりも大きい値である必要があります。範囲は 0 から 100 です。

要素	説明
Expression	アナログ型タグ名または式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<ObjectSizeHeight>  
<Title>ObjectSizeHeight1</Title>  
<Expression> </Expression>  
<SizeMin>0.0</SizeMin>  
<SizeMax>100.0</SizeMax>  
<MinPercent>0.0</MinPercent>  
<MaxPercent>100.0</MaxPercent>  
<SizeAnchor>Top</SizeAnchor>  
</ObjectSizeHeight>
```

最小の例:

```
<ObjectSizeHeight>  
<Expression>aTag001</Expression>  
</ObjectSizeHeight>
```

オブジェクト幅

オブジェクトサイズ幅アニメーションリンクに対して以下の要素を指定できます。オブジェクト幅アニメーションリンクは向きアニメーションリンクと一緒に使用できません。

要素	説明
Title	オブジェクトの名前。オプション。
SizeAnchor	オブジェクト上でアンカーを配置する位置を定義します。値は、 left 、 center 、 right です。デフォルトは left です。
SizeMin	最小幅の値。デフォルトは 0 です。
SizeMax	最大幅の値。デフォルトは 100 です。 SizeMin よりも大きい値である必要があります。
MinPercent	幅の最小パーセンテージ。デフォルトは 0 です。範囲は 0 から 100 です。
MaxPercent	幅の最大パーセンテージ。デフォルトは 100 です。 MinPercent よりも大きい値である必要があります。範囲は 0 から 100 です。
Expression	アナログ型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
< ObjectSizeWidth>
<Title>ObjectSizeWidth1</Title>
<Expression> </Expression>
<SizeMin>0.0</SizeMin>
<SizeMax>100.0</SizeMax>
<MinPercent>0.0</MinPercent>
<MaxPercent>100.0</MaxPercent>
<SizeAnchor>Center</SizeAnchor>
</ObjectSizeWidth>
```

最小の例:

```
<ObjectSizeWidth>
<Expression>aTag001</Expression>
</ObjectSizeWidth>
```

垂直位置

垂直位置アニメーションリンクに対して以下の要素を指定できます。オブジェクト垂直位置 アニメーションリンクは向きアニメーションリンクと一緒に使用できません。

要素	説明
Title	オブジェクトの名前。オプション。
MinValue	上の値。デフォルトは 0 です。下と同じ値にすることはできません。
MaxValue	下の値。デフォルトは 100 です。上と同じ値にすることはできません。
DecreaseMovement	上への垂直移動。デフォルトは 0 です。 0 より大きい値である必要があります。
IncreaseMovement	下への垂直移動。デフォルトは 100 です。 100 より小さい値である必要があります。
Expression	アナログ型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<LocationVertical>
<Title>LocationVertical1</Title>
<Expression>
</Expression>
<MinValue>0.0</MinValue>
<MaxValue>100.0</MaxValue>
<DecreaseMovement>0</DecreaseMovement>
```

```
<IncreaseMovement>100</IncreaseMovement>
</LocationVertical>
```

最小の例:

```
<LocationVertical>
<Expression>aTag001</Expression>
</LocationVertical>
```

水平位置

水平位置アニメーションリンクに対して以下の要素を指定できます。オブジェクト水平位置アニメーションリンクは向きアニメーションリンクと一緒に使用できません。

要素	説明
Title	オブジェクトの名前。オプション。
MinValue	左の値。デフォルトは 0 です。右端と同じ値にすることはできません。
MaxValue	右の値。デフォルトは 100 です。左端と同じ値にすることはできません。
DecreaseMovement	左への水平移動。デフォルトは 0 です。 0 より大きい値である必要があります。
IncreaseMovement	右への水平移動。デフォルトは 100 です。 100 より小さい値である必要があります。
Expression	アナログ型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<LocationHorizontal>
<Title>LocationHorizontal1</Title>
<Expression>
</Expression>
<MinValue>0.0</MinValue>
<MaxValue>100.0</MaxValue>
<DecreaseMovement>0</DecreaseMovement>
<IncreaseMovement>100</IncreaseMovement>
</LocationHorizontal>
```

最小の例:

```
<LocationHorizontal>
<Expression>aTag001</Expression>
</LocationHorizontal>
```

垂直塗りつぶしパーセント

垂直塗りつぶしパーセントアニメーションリンクに対して以下の要素を指定できます。垂直塗りつぶしパーセントアニメーションリンクは向きアニメーションリンクと一緒に使用できません。

要素	説明
Title	オブジェクトの名前。オプション。
FillDirection	移動方向を定義します。可能な値は以下のとおりです。up、down。デフォルトは up です。必須。
FillColor	純色の塗りつぶしの色要素。デフォルトは rgb(0, 0, 0) です。
FillMin	最小値を定義します。デフォルトは 0 です。
FillMax	最大値を定義します。デフォルトは 100 です。最小塗りつぶしよりも大きい値である必要があります。
FillMinPercent	最小値をパーセンテージとして定義します。デフォルトは 0 です。範囲は 0 から 100 です。
FillMaxPercent	最大値をパーセンテージとして定義します。デフォルトは 100 です。範囲は 0 から 100 です。最小パーセンテージよりも大きい値である必要があります。
Expression	アナログ型タグまたは式。必須。

例:

```
<PercentFillVertical>
<Title>PercentFillVertical1</Title>
<Expression></Expression>
<FillMin>0.0</FillMin>
<FillMax>100.0</FillMax>
<FillMinPercent>0</FillMinPercent>
<FillMaxPercent>100</FillMaxPercent>
<FillColor><Name>Purple</Name></FillColor>
<FillDirection>Up</FillDirection>
</PercentFillVertical>
```

最小の例:

```
<PercentFillVertical>
<Expression>aTag001</Expression>
<FillDirection>Up</FillDirection>
</PercentFillVertical>
```

水平塗りつぶしパーセント

水平塗りつぶしパーセントアニメーションリンクに対して以下の要素を指定できます。水平塗りつぶしパーセントアニメーションリンクは向きアニメーションリンクと一緒に使用できません。

要素	説明
Title	オブジェクトの名前。オプション。
FillDirection	left、right。デフォルトは right です。
FillColor	1 つの色要素を含みます。デフォルトは rgb(0, 0, 0) です。
FillMin	最小塗りつぶしの値。デフォルトは 0 です。
FillMax	最大塗りつぶしの値。デフォルトは 100 です。最小塗りつぶしよりも大きい値である必要があります。
FillMinPercent	オブジェクトを塗りつぶす最小パーセント。デフォルトは 0 です。範囲は 0 から 100 です。
FillMaxPercent	オブジェクトを塗りつぶす最大パーセント。デフォルトは 100 です。範囲は 0 から 100 です。最小塗りつぶしパーセントよりも大きい値である必要があります。
Expression	アナログ型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<PercentFillHorizontal>  
<Title>PercentFillHorizontal1</Title>  
<Expression></Expression>  
<FillMin>0.0</FillMin>  
<FillMax>100.0</FillMax>  
<FillMinPercent>0</FillMinPercent>  
<FillMaxPercent>100</FillMaxPercent>  
<FillColor><Name>Purple</Name></FillColor>  
<FillDirection>Right</FillDirection>  
</PercentFillHorizontal>
```

最小の例:

```
<PercentFillHorizontal>  
<Expression>aTag001</Expression>  
<FillDirection>Left</FillDirection>  
</PercentFillHorizontal>
```

論理値押しボタン

論理値押しボタン アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
ButtonType	論理値ボタンのタイプ: direct 、 reverse 、 toggle 、 reset 、または set 。必須。無効な場合、または欠落している場合、オブジェクトは作成されません。
Expression	論理型タグまたは式。必須。 Expression 要素が無効な場合、または欠落している場合、オブジェクトは作成されません。
KeyAssignment	仮想キー要素、空の文字列、または存在しない。デフォルトでは割り当ては設定されません。空の文字列は割り当てが発生しないことを示します。

例:

```
<ButtonDiscreteValue>
<Title>ButtonDiscreteValue1</Title>
<ButtonType>Reverse</ButtonType>
<KeyAssignment>
<KeyCode>F2</KeyCode>
<KeyFlags>Shift</KeyFlags>
</KeyAssignment>
<Expression>
</Expression>
</ButtonDiscreteValue>
```

最小の例:

```
<ButtonDiscreteValue>
<ButtonType>Reverse</ButtonType>
<Expression></Expression>
</ButtonDiscreteValue>
```

ウィンドウの表示押しボタン

リンクを生成したときに名前付きウィンドウが存在しない場合、警告がログに記録されます。ウィンドウに対するアクションなしでリンクが生成されます。少なくとも 1 つの名前付きウィンドウが存在する必要があります。そうでない場合、**ShowWindow** アニメーション リンクはインポートされません。

ウィンドウの表示押しボタン アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。

要素	説明
WindowName	表示するウィンドウの名前。少なくとも 1 つのウィンドウ名を指定する必要があります。そうでない場合、このアニメーションリンクはオブジェクトに追加されません。

例:

```
<ButtonShowWindow>
<Title>ButtonShowWindow1</Title>
<WindowName>
</WindowName>
<WindowName>Window007</WindowName>
<WindowName></WindowName>
</ButtonShowWindow>
```

ウィンドウの非表示押しボタン

リンクを生成したときに名前付きウィンドウが存在しない場合、警告がログに記録されます。ウィンドウに対するアクションなしでリンクが生成されます。少なくとも 1 つの名前付きウィンドウが存在する必要があります。そうでない場合、**HideWindow** アニメーションリンクはインポートされません。

ウィンドウの非表示押しボタン アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
WindowName	非表示にするウィンドウの名前。少なくとも 1 つのウィンドウ名を指定する必要があります。そうでない場合、このアニメーションリンクはオブジェクトに追加されません。

例:

```
<ButtonHideWindow>
<Title>ButtonHideWindow1</Title>
<WindowName>
</WindowName>
<WindowName>Window002</WindowName>
<WindowName> </WindowName>
</ButtonHideWindow>
```

表示オン/オフ

表示オン/オフ アニメーションリンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
State	オブジェクトを表示するかどうかを定義します。可能な値は以下のとおりです。on、off。デフォルトは On です。オ

要素	説明
	プシヨンのスキーマを使用する場合に必要です。
Expression	論理型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<Visibility>  
<Title>Visibility1</Title>  
<Expression>dTag001</Expression>  
<State>On</State>  
</Visibility>
```

最小の例:

```
<Visibility>  
<Expression>dTag001</Expression>  
<State>On</State>  
</Visibility>
```

点滅

点滅アニメーションリンクに対して以下の要素を指定できます。式が真の場合、アニメーションリンクが点滅します。

要素	説明
Title	オブジェクトの名前。オプション。
BlinkAttribute	Invisible、Visible。デフォルトは Visible です。
BlinkSpeed	オブジェクトの点滅レートを定義します。可能な値は以下のとおりです。Slow、Medium、Fast。デフォルトは Medium です。
TextColor	点滅するテキストの色を定義します。1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
LineColor	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
FillColor	1つの色要素を含みます。デフォルトは rgb(0,0,0) です。
Expression	論理型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:


```
<Blink>
<Title>Blink1</Title>
<Expression>dTag001</Expression>
<TextColor><R>0</R><G>0</G><B>0</B></TextColor>
<LineColor><R>0</R><G>0</G><B>0</B></LineColor>
<FillColor><R>0</R><G>255</G><B>0</B></FillColor>
<BlinkAttribute>Invisible</BlinkAttribute>
<BlinkSpeed>Slow</BlinkSpeed>
</Blink>
```

最小の例:

```
<Blink>
<Expression>dTag001</Expression>
<BlinkAttribute>Visible</BlinkAttribute>
<BlinkSpeed>Fast</BlinkSpeed>
</Blink>
```

向き

向きアニメーションリンクに対して以下の要素を指定できます。向きアニメーションリンクは、スライダー、サイズ、位置、および塗りつぶしパーセントアニメーションリンクと一緒に使用することはできません。

要素	説明
Title	オブジェクトの名前。オプション。
X	オブジェクトの中心点からの水平オフセット。オプション。デフォルトは 0 です。
Y	オブジェクトの中心点からの垂直オフセット。オプション。デフォルトは 0 です。
CWMax	最大時計回りの回転の値。デフォルトは 100 です。
CWRotation	時計回りの回転。デフォルトは 360 です。値の範囲は 0 から 360 です。 CWROTATION+CCWROTATION は 360 以下である必要があります。
CCWMax	最大反時計回りの回転の値。デフォルトは 0 です。
CCWRotation	反時計回りの回転。デフォルトは 0 です。値の範囲は 0 から 360 です。 CWRotation+CCWRotation は 360 以下である必要があります。
Expression	アナログ型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<Orientation>
```

```
<Title>Orientation1</Title>
<Expression> </Expression>
<X>0</X> <Y>0</Y>
<CWMax>100.0</CWMax>
<CWRotation>360.0</CWRotation>
<CCWMax>0.0</CCWMax>
<CCWRotation>0.0</CCWRotation>
</Orientation>
```

最小の例:

```
<Orientation>
<Expression>aTag001</Expression>
</Orientation>
```

無効化

無効化アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
State	on、off。デフォルトは On です。オプションのスキーマを使用する場合に必要です。
Expression	論理型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<Disable>
<Title>Disable1</Title>
<Expression>dTag001</Expression>
<State>On</State>
</Disable>
```

最小の例:

```
<Disable>
<Expression>dTag001</Expression>
<State>On</State>
</Disable>
```

静的ツールヒント

静的ツールヒント アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Message	静的テキスト メッセージ。必須。無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<TooltipStatic>
```

```
<Title>TooltipStatic1</Title>
<Message>
<![CDATA[ Click here to win a million dollars!
]]>
</Message>
</TooltipStatic>
```

動的ツールヒント

動的ツールヒント アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Expression	Expression 要素。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<TooltipTag>
<Title>TooltipTag1</Title>
<Expression>
</Expression>
</TooltipTag>
```

論理値表示

論理値アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
OnMessage	アニメーション リンクの値が真のときに表示する文字列。デフォルトテキストは On です。
OffMessage	アニメーション リンクの値が偽のときに表示する文字列。デフォルトテキストは Off です。
Expression	論理型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<ValueDisplayDiscrete>
<Title>ValueDisplayDiscrete1</Title>
<Expression>
</Expression>
<OnMessage>
</OnMessage>
<OffMessage>
</OffMessage>
</ValueDisplayDiscrete>
```

最小の例:

```
<ValueDisplayDiscrete>
<Expression>dTag001</Expression>
</ValueDisplayDiscrete>
```

アナログ値表示

アナログ値アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Expression	アナログ型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<ValueDisplayAnalog>
<Title>ValueDisplayAnalog1</Title>
<Expression>
</Expression>
</ValueDisplayAnalog>
```

文字列値表示

文字列値アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Title	オブジェクトの名前。オプション。
Expression	メッセージ型タグまたは式。必須。式が無効な場合、または欠落している場合、オブジェクトは作成されません。

例:

```
<ValueDisplayString>
<Title>ValueDisplayString1</Title>
<Expression>
</Expression>
</ValueDisplayString>
```

押しボタンアクションスクリプト

押しボタンアクション スクリプト要素は、アニメーション リンク要素内で 0 宣言されたボタン アクション スクリプト要素内で宣言されます。

例:

```
<AnimationLinks>
<ButtonActionScripts>
<OnLeftDown>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
```

```
]]>
</OnLeftDown>
</ButtonActionScripts>
</AnimationLinks>
```

左のクリック時/押した時

同じオブジェクトで **OnLeftDown** と **OnKeyDown** アニメーション リンクの両方を使用しないでください。

例:

```
<OnLeftDown>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
</OnLeftDown>
```

左クリック/左を押している間

WhileLeftDown または **WhileKeyDown** アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Text	スクリプト テキスト。必須。
頻度	ミリ秒単位でのスクリプト実行頻度。 範囲は 1 から 360000 です。オプション のスキーマを使用する場合に必要です。 デフォルトは 1000 です。

同じオブジェクトで **WhileLeftDown** と **WhileKeyDown** アニメーション リンクの両方を使用しないでください。

例:

```
<WhileLeftDown>
<Text>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
</Text>
<Frequency>1000</Frequency>
</WhileLeftDown>
```

左のクリック時/離れた時

OnLeftUp 要素内にスクリプトのテキストを配置します。同じオブジェクトで **OnLeftUp** と **OnKeyUp** アニメーション リンクを使用しないでください。

例:

```
<OnLeftUp>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
```

```
]]>  
</OnLeftUp>
```

左のダブルクリック時

OnLeftDoubleClick アニメーション リンク内にスクリプトのテキストを配置します。

例:

```
<OnLeftDoubleClick>  
<![CDATA[  
スクリプト テキストの最初の行  
スクリプト テキストの 2 番目の行  
スクリプト テキストの N 番目の行  
]]>  
</OnLeftDoubleClick>
```

右のクリック時/押した時

OnRightDown 要素内にスクリプトのテキストを配置します。同じオブジェクトで OnRightDown と OnRightKeyDown アニメーション リンクを使用しないでください。

例:

```
<OnRightDown>  
<![CDATA[  
スクリプト テキストの最初の行  
スクリプト テキストの 2 番目の行  
スクリプト テキストの N 番目の行  
]]>  
</OnRightDown>
```

右クリック/右を押している間

WhileRightKeyDown または WhileRightDown アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Text	スクリプト テキスト。必須
頻度	ミリ秒単位でのスクリプト実行頻度。1 から 360000 の数値。デフォルトは 1000 です。

同じオブジェクトで WhileRightDown と WhileRightKeyDown アニメーション リンクの両方を使用しないでください。

例:

```
<WhileRightDown>  
<Text>  
<![CDATA[  
スクリプト テキストの最初の行  
スクリプト テキストの 2 番目の行  
スクリプト テキストの N 番目の行  
]]>  
</Text>  
<Frequency>1000</Frequency>  
</WhileRightDown>
```

右クリック/右を離した時

OnRightUp および OnRightKeyUp 要素内にスクリプトのテキストを配置します。同じオブジェクトで OnRightUp と OnRightKeyUp アニメーション リンクの両方を使用しないでください。

例:

```
<OnRightUp>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
</OnRightUp>
```

右のダブルクリック時

OnRightDoubleClick 要素内にスクリプトのテキストを配置します。

例:

```
<OnRightDoubleClick>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
</OnRightDoubleClick>
```

中央のクリック時/押した時

OnMiddleKeyDown 要素内にスクリプトのテキストを配置します。同じオブジェクトで OnMiddleDown と OnMiddleKeyDown アニメーション リンクを使用しないでください。

例:

```
<OnMiddleDown>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
</OnMiddleDown>
```

中央クリック/中央を押している間

WhileMiddleKeyDown または WhileMiddleDown アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Text	スクリプト テキスト。必須
頻度	ミリ秒単位でのスクリプト実行頻度。1 から 360000 の数値。オプションのスキーマを使用する場合に必要です。デフォルトは 1000 です。

同じオブジェクトで WhileMiddleDown と WhileMiddleKeyDown アニメーション リンクを使用しないでください。

例:

```
<WhileMiddleDown>
<Text>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
</Text>
<Frequency>1000</Frequency>
</WhileMiddleDown>
```

中央のクリック時/離れた時

OnMiddleKeyUp および OnMiddleUp アニメーション リンク内にスクリプトのテキストを配置します。同じオブジェクトで OnMiddleUp と OnMiddleKeyUp アニメーション リンクを使用しないでください。

例:

```
<OnMiddleUp>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
</OnMiddleUp>
```

中央のダブルクリック時

OnMiddleDoubleClick 要素内にスクリプトのテキストを配置します。

例:

```
<OnMiddleDoubleClick>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
スクリプト テキストの N 番目の行
]]>
</OnMiddleDoubleClick>
```

マウス ホバー時

OnMouseOver アニメーション リンクに対して以下の要素を指定できます。

要素	説明
Text	スクリプト テキスト。必須。
タイムアウト	ミリ秒単位のタイムアウト。オプションのスキーマを使用する場合に必要です。デフォルトは 250 です。

例:

```
<OnMouseOver>
<Text>
<![CDATA[
スクリプト テキストの最初の行
スクリプト テキストの 2 番目の行
```



```
スクリプト テキストの N 番目の行  
]]>  
</Text>  
<Timeout>250</Timeout>  
</OnMouseOver>
```

左キーアクションの割り当て

左キーアクションの割り当てアニメーションリンクに対して以下の要素を指定できます。

要素	説明
KeyCode	仮想キー コード名または空の文字列。必須。空の文字列は割り当てが発生しないことを示します。
KeyFlags	仮想キー フラグ組み合わせまたは空の文字列。必須。空の文字列は割り当てが発生しないことを示します。 KeyCode を指定しない場合、無効化されます。

"CTRL+B" の例:

```
<LeftKey>  
<KeyCode>B</KeyCode>  
<KeyFlags>Ctrl</KeyFlags>  
</LeftKey>
```

"L" に対するキー フラグ修飾子なしの例:

```
<LeftKey>  
<KeyCode>L</KeyCode>  
</LeftKey>
```

CTRL+SHIFT+F7 に対する両方のキー フラグ修飾子の例:

```
<LeftKey>  
<KeyCode>F7</KeyCode>  
<KeyFlags>CtrlShift</KeyFlags>  
</LeftKey>
```

右キーアクションの割り当て - サポートされていません

右キーアクションの割り当ては、InTouch XML インポート機能でサポートされていません。

右キーアクションの割り当てアニメーションリンクに対して以下の要素を指定できます。

要素	説明
KeyCode	仮想キー コード名または空の文字列。必須。空の文字列は割り当てが発生しないことを示します。
KeyFlags	仮想キー フラグ組み合わせまたは空の文字列。必須。空の文字列は割り当てが発生しないことを示します。 KeyCode を指定しない場合、無効化されます。

例:

```
<RightKey>  
<KeyCode>B</KeyCode>  
<KeyFlags>Ctrl</KeyFlags>  
</RightKey>
```

中央キーアクションの割り当て - サポートされていません

中央キーアクションの割り当ては、InTouch XML インポート機能でサポートされていません。

中央キーアクションの割り当てアニメーションリンクに対して以下の要素を指定できます。

要素	説明
KeyCode	仮想キーコード名または空の文字列。必須。空の文字列は割り当てが発生しないことを示します。
KeyFlags	仮想キーフラグ組み合わせまたは空の文字列。必須。空の文字列は割り当てが発生しないことを示します。KeyCode を指定しない場合、無効化されます。

```
<MiddleKey>  
<KeyCode>B</KeyCode>  
<KeyFlags>Ctrl</KeyFlags>  
</MiddleKey>
```

サポートされていない InTouch 機能

ウィザードと ActiveX コントロールは、XML インポート機能を使用してインポートできません。

タグまたはタグデータベースは、XML インポート機能を使用してインポートできません。タグ定義を含むファイルを作成し、DBLoad ユーティリティを使用してアプリケーションのタグ名ディクショナリにインポートすることができます。DBLoad の使用の詳細については、『AVEVA™ InTouch HMI アプリケーションメンテナンスガイド』参照してください。コマンドプロンプトからの DBLoad の実行の詳細については、「[コマンドプロンプトからの DBLoad の実行](#)」を参照してください。

章 14 AVEVA Connect での産業用グラフィックの操作

AVEVA Connect は、産業用グラフィック、コントロール、ウィジェットなどの共通コンテンツをクラウドで管理できる共通クラウドリポジトリです。

グラフィックは、オンデマンドでダウンロードおよびアップロードできます。グラフィックは AVEVA Connect の「ストア」に格納されます。グローバル、テナント、およびユーザー固有の 3 種類のストアがあります。各ストア内で複数のドライブを設定できます。クラウドでグラフィックを管理するには AVEVA Connect ユーザー アカウントが必要です。各ドライブには、さまざまなユーザーに別のアクセスレベルを設定できます。

管理者が付与したアクセスに応じてユーザーに読み取り専用または読み取り/書き込みのアクセスが設定されます。

- 読み取り専用アクセスを持つユーザーは、クラウドでグラフィックを表示できます。このユーザーは、クラウドからグラフィックをダウンロードできますが、そのグラフィックは読み取り専用になります。グラフィックを変更することやアップロードすることはできません。
- 読み取り/書き込みアクセスを持つユーザーは、グラフィックの表示、ダウンロード、変更、およびアップロードを行うことができます。

複数のユーザーが同じドライブ内のグラフィックにアクセスできますが、1 つのグラフィックを複数のユーザーが同時に編集することはできません。

AVEVA Connect へのログイン

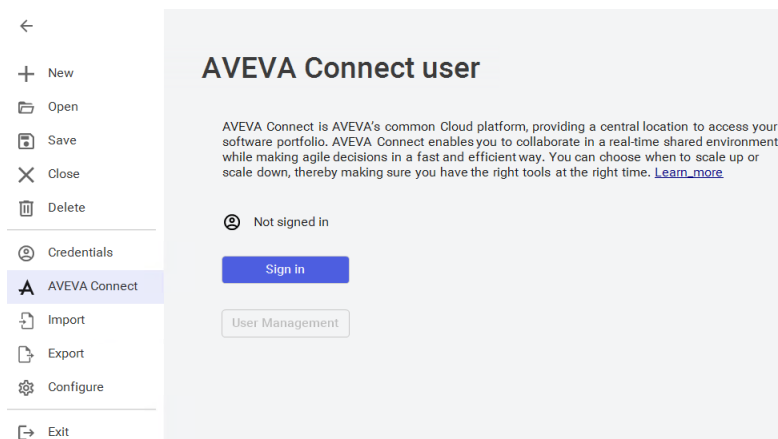
AVEVA Connect にアクセスするには、AVEVA Connect アカウントに加えて管理者からの承認が必要です。

1. InTouch を使用している場合、WindowMaker を起動し、[ファイル] メニューから [AVEVA Connect] を選択します。

System Platform IDE を使用している場合、[Galaxy] メニューを選択し、[AVEVA Connect] を選択します。

注記: コンフィグレータの [ライセンス モード] タブで [接続エクスペリエンス] モードが選択されている場合、AVEVA Connect オプションは使用できません。

AVEVA Connect ユーザーの設定画面が表示されます。



2. [サインイン] をクリックします。
3. AVEVA Connect 電子メール アドレスを入力します。
4. パスワードを入力します。
資格情報が検証されてサインイン処理が完了します。
5. **AVEVA Cloud Integration Studio** ソリューションを選択します (Pym Technologies など)。
6. AVEVA Drive が [視覚化] フォルダ内の個別のリポジトリとして表示されます。

注記: 選択した Integration Studio ソリューションはブラウザ メモリにキャッシュされます。Integration Studio ソリューションを切り替えるには、ブラウザ キャッシュをクリアする必要があります。

共有ドライブ間のナビゲーション

特定のストアにログインした後、ドライブ間を移動してグラフィックのアップロードとダウンロードを行うことができます。

1. [表示] メニューの [クラウド ドライブ] グループで [共有ドライブ] をクリックします。
2. 使用可能なオプションからドライブを選択します。

グラフィック ツールボックスが更新され、選択したドライブが表示されます。

注記: 別のユーザーによってロックされているグラフィックを移動することはできません。

グラフィックのアップロード/ダウンロード

AVEVA Drive 機能は、ローカルのグラフィック ツールボックスに似ています。AVEVA Drive にフォルダをアップロードまたは作成してグラフィックを整理できます。読み取り/書き込みアクセスがあるユーザーは、グラフィックをローカルのグラフィック ツールボックスにダウンロードしてアプリケーションで使用できます。グラフィックをダウンロードまたはアップロードするとき、コンテンツは選択した上書きオプションに従って上書きされます。

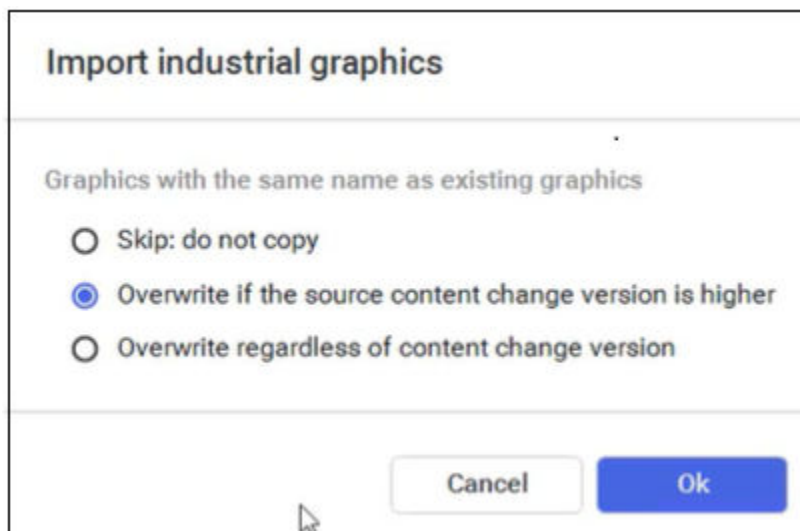
- ターゲット ドライブにコンテンツが既に存在する場合、コンテンツは同じツールセットの場所で保持されます。そうでない場合、コンテンツは同じツールセット パスにソースとして追加されます。

選択した埋め込みコンテンツに対して次の操作を行います。

- ターゲット ドライブに埋め込みコンテンツが既に存在する場合、コンテンツはターゲットの同じツールセットの場所で保持されます。
- 埋め込みコンテンツが親コンテンツと同じツールセットの場所または同じフォルダ階層内に存在する場合、埋め込みコンテンツは親として同じツールセット パスに追加されます。そうでない場合、埋め込みコンテンツは同じツールセット パスにソースとして追加されます。

AVEVA Connect へのグラフィックのアップロード

1. ローカルのグラフィック ツールボックスからグラフィックを AVEVA Drive ツールボックスにドラッグアンドドロップします。
2. ドライブにグラフィックが既に存在する場合、上書きするかアップロードをスキップするかを尋ねるポップアップ ダイアログが表示されます。



3. 適切なオプションを選択し、[OK] をクリックします。
4. [コンテンツをアップロード] ダイアログが表示されて処理の進捗状況が表示されます。[詳細の表示] をクリックすると進捗状況が表示されます。
5. [閉じる] をクリックします。

複数のグラフィックを含むフォルダをアップロードすることもできます。

Download Graphics to the Local Visualization Folder

Users with Read-Write access can download graphics from the AVEVA Drive to their local Visualization folder.

1. Drag the graphic(s) or folder from the AVEVA Drive and drop it to the local Visualization folder.

The **Download Content** dialog box requests confirmation of the download.



- Click **Yes** to continue with the download.
 - Click **No** to cancel the operation.
 - Click **View Details** to view the list of controls and namespaces used. Any errors during downloading are also displayed.
2. If the graphic already exists, the Import industrial graphic dialog box appears.

3. Select the appropriate option.

4. The graphic(s) are downloaded to the local Visualization folder. Embedded graphics are also copied.

さまざまなクラウド コンテンツのアップロード/ダウンロードのサポート

さまざまなクラウド コンテンツのアップロード/ダウンロードのサポートを以下の表に示します。

クラウド コンテンツ	アップロード サポート	ダウンロード サポート
埋め込まれたクライアント コントロールおよび HTML5 ウィジェット	アップロードでは、埋め込まれたグラフィック コンテンツ（クライアント コントロールや HTML5 ウィジェットなど）が aaPKG 形式でパッケージ化され、クラウドにコピー/上書きされます。	ダウンロードでは、埋め込まれたグラフィック コンテンツ（クライアント コントロールや HTML5 ウィジェットなど）が抽出され、コントロールがローカル アプリケーションにインポート/上書きされます。
埋め込まれたアプリケーションのオブジェクトシンボル	サポートされていません。	サポートされていません。
カスタム スクリプト ライブラリ	サポートされていません。	サポートされていません。
アプリケーション スタイル ライブラリ	サポートされていません。	サポートされていません。

AVEVA Connect でのグラフィックの管理

ローカル グラフィックの管理と同様に、AVEVA Drive 上でグラフィックの作成、名前の変更、更新、複製、および削除を直接行うことができます。

- AVEVA Drive または任意のフォルダを右クリックして新しいグラフィックまたはツールセットを作成します。
- グラフィックをダブルクリックし、グラフィック エディタを使用して編集します。
- 右クリックして目的のオプションを選択し、グラフィックの名前の変更、複製、または削除、およびサムネイルの更新を行います。

クラウドでサポートされないコンテキスト メニュー オプションはグレー表示されます（[Web Client ホーム シンボルの作成]、[Web Client ルート フォルダの設定]、[シンボルのエクスポート]、[ローカリゼーションのエクスポート] など）。

注記: 埋め込まれたグラフィックのサムネイルを AVEVA Drive で直接更新すると、多くのクラウドグラフィックが発生します。したがって、サムネイルはローカルで更新することをお勧めします。

Managing Graphics with Multiple Users

Multiple users can use the graphics available on AVEVA Drive at a time. However, a graphic can be edited by only one user at a time. When a graphic in the Cloud repository is being edited by one user, the graphic is checked-out and locked, preventing other users from making any edits. All other users can only view the read-only

unmodified graphic when it is checked-out. The latest changes reflect only when it is checked-in the the modifying user.

- A lock icon against the graphic icon in the folder indicates that the graphic is checked out by the logged in user.
- An edit icon against the graphic icon in the folder indicates that the graphic is being edited by another user.

After the first user saves and checks in the graphic, the second user can check it out and make changes to it. Once the graphic is saved and checked-in, all users can view the modified graphic.

クラウドグラフィックのバージョン管理

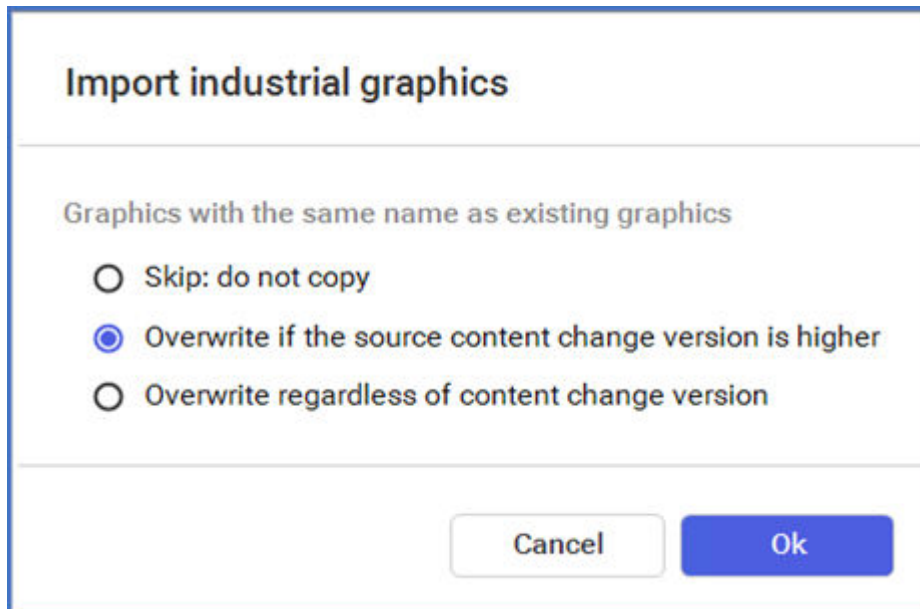
AVEVA Drive グラフィックは、グラフィック サブシステムのさまざまなバージョンの製品からクラウドにパブリッシュ済み産業用グラフィック、クライアント コントロール、およびウィジェットをサポートします。

クラウドグラフィック バージョンのサポートの主な概念

- 産業用グラフィックは、グラフィック サブシステムで使用できるすべての機能をサポートする特定の形式でローカルおよびクラウドで保存されます。
- グラフィック サブシステムに新機能を追加した場合、グラフィックを保存して、その他のユーザーが使用できるようにチェックインする必要があります。
- 産業用グラフィック サブシステムのバージョンは、一意の内部バージョン番号によって識別されます。ユーザーが視覚的に識別できるものではありません。各製品はリリース時に産業用グラフィック サブシステムの最新バージョンを使用します。
- 産業用グラフィック サブシステムのバージョン番号は、保存された各産業用グラフィックの一部として含まれ、一致するバージョンまたはより高いバージョンの産業用グラフィック サブシステムに接続された製品のみがその産業用グラフィックの編集、埋め込み、またはダウンロードを行うことが許可されるように使用されます。
- 同じクラウドレポジトリに接続されたすべてのユーザーには、同じセットのグラフィックが表示されます。クラウドグラフィックの編集、埋め込み、またはダウンロードできる機能は、ユーザーが接続されている製品内の産業用グラフィック サブシステムのバージョンに依存します。
- 新しいバージョンの産業用グラフィック サブシステムに接続された製品によってクラウドレポジトリに保存された産業用グラフィックは、同じクラウドにアクセスする古いバージョンの製品のグラフィック ツールボックスでは無効化された状態で表示されます。古い製品バージョンのユーザーは、グラフィック レポジトリの編集、埋め込み、およびダウンロードを行うことができません。
- ローカルのグラフィック ツールボックスと同様に、グラフィック名は1つのクラウドレポジトリ内で一意である必要があります。これは、グラフィックが産業用グラフィック サブシステムの別のバージョンに接続された製品によって保存された場合でも同じです。

クラウドでのグラフィック コンテンツの上書き

アップロードまたはダウンロードしたグラフィック コンテンツが既に存在する場合、グラフィックを上書きするかどうかを尋ねるメッセージが表示されます。



上書きのオプションは、ソースとターゲットのグラフィック サブシステムのバージョン、およびグラフィックの変更ログのバージョンに応じて異なります。

- オプション 1 では、グラフィック サブシステムのバージョンやソースとターゲットの変更ログのバージョンに関係なく上書きがスキップされます。
- グラフィック サブシステムのバージョンがソースとターゲットで異なる場合、オプション 2 と 3 の両方では、変更ログのバージョンに関係なくグラフィックが上書きされます。
- ソースとターゲットのグラフィック サブシステムのバージョンが同じである場合、次の処理が行われます。
 - オプション 2 では、ソースの変更ログのバージョンが高い場合にのみ上書きされます。
 - オプション 3 では、変更ログのバージョンにかかわらず上書きされます。

クラウドでのカスタム クライアント コントロールとウィジェットの上書き

クラウドグラフィックのバージョン管理のガイドラインは、このリリースの時点では、カスタム クライアント コントロールとウィジェットには適用されません。クライアント コントロールとウィジェットは、ダウンロードおよびアップロード操作の両方でサポートされます。しかし、上書きはサポートされません。カスタム クライアント コントロールまたはウィジェットがターゲットの場所に既に存在する場合、上書きはクライアント コントロールの内部バージョンに関係なくスキップされます。ソース リポジトリのクライアント コントロールを上書きするには、最初にソース リポジトリのクライアント コントロールを削除する必要があります。その後、クライアント コントロールのアップロードまたはダウンロードが正常に行われます。

注記: カスタム クライアント コントロールまたはウィジェットをクラウドにアップロードした場合、最新のカスタム クライアント コントロールまたはウィジェットを使用するには、そのクラウド リポジトリに接続された他のすべてのユーザーは **WindowMaker** を再起動する必要があります。

Cloud Graphic Versioning - Scenarios and Examples

This section provides a non-exclusive list of high level scenarios and expectation for Cloud graphic version handling.

Let's consider two products P1_Old and P2_New, of different versions, which connect to the same tenant in AVEVA Cloud.

- P1_Old has the older version of Graphic Subsystem and P2_New has the newer version of Graphic Subsystem.
- To begin with, the cloud repository has graphics S1 and S2 uploaded by P1_Old.

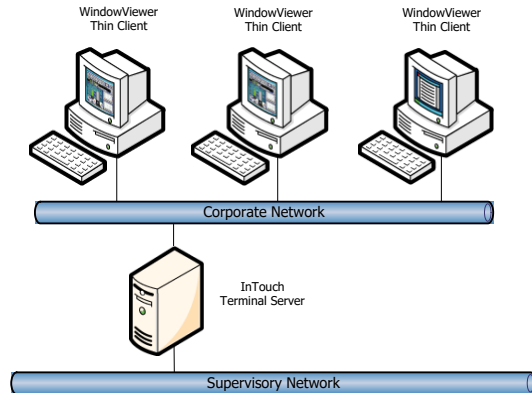
Scenario / Steps	Expectation
Scenario 1: P2_New creates S3. P1_Old attempts to edit S3.	S3 is saved in newer graphic version and hence it cannot be edit, embed, or download in older version. P1_Old receives a message stating that S3 is saved in newer version and cannot be opened.
Scenario 2: P2_New opens S2. P2_New saves S2 in Cloud. Then, P1_Old attempts to open S2.	S2 created in older graphic version can be opened and saved in the newer graphic version. S2 is saved in newer graphic version and hence it cannot be edit, embed, or download in older version. P1_Old receives a message stating that S3 is saved in newer version and cannot be opened.
Scenario 3: P1_Old saves S1 and S2, where S2 is embedded in S1 P2_New saves and updates only S2. P1_Old attempts to edit S1.	S1 is opened because it is still the same version in P1_Old. The embedded S2 will fail to load.
Scenario 4: P2_New creates a new S4 in Cloud. P2_New embeds S1, S2, and S3 in S4.	S1 is in older graphic version but still can be loaded in S4 (newer version). S2 and S3 can be embedded because they are the same graphic version as S4.
Scenario 5: P1_Old creates a new S5 in Cloud. P1_Old embeds S1, S2, and S3 in S5.	S1 can be embedded because they are the same graphic version. S2 and S3 cannot be embedded due to newer version.
Scenario 6: P1_Old download S3.	S3 cannot be downloaded because it is in newer graphic version.
Scenario 7: Cloud repository has a graphic S3 which is in newer graphic version. P1_Old attempts to upload a graphic with same name S3.	By selecting Option 3, the graphic can be overwritten.
Scenario 8:	S9 is uploaded to Cloud.

P2_New has a graphic S8. P1_Old creates locally S8, S9, and embed S8 in S9. P1_Old uploads S9 to Cloud.	S8 is prevented upload because a newer version exists in the Cloud. By selecting Option 3, the content can be overwritten.
---	--

Deploy

章 15 ターミナル サービスとリモート デスクトップ サービスの配置と操作

ターミナル サービスとは、Microsoft Windows Server オペレーティング システムに含まれている設定可能なサービスであり、サーバーを中心として Windows ベースのアプリケーションを実行できます。ターミナル サービスでは、InTouch ソフトウェア アプリケーションの複数のインスタンスが同時に実行されるサーバー ノードにクライアント コンピュータがアクセスします。



ターミナル サービスの環境には、次のような主な 3 つの部分があります。

- **ターミナル サービス サーバー。** このサーバーは、各クライアントセッションのコンピュータ処理用リソースを管理し、クライアント ユーザーに対し、各ユーザー自身の環境を提供します。サーバーは、リモートクライアントで実行されるキー入力、およびマウス操作をすべて処理し、オペレーティング システムとアプリケーションの両方の表示出力を適切なクライアントに送信します。ターミナル サービス アプリケーションのすべての処理がサーバーで発生します。
- **リモート デスクトップ プロトコル (RDP)。** リモート デスクトップ プロトコル (RDP) クライアント アプリケーションにより、キー入力、マウスの動きなどの入力データがサーバーに渡されます。
- **クライアント。** ターミナル サービス クライアントでは、ローカル アプリケーションの処理は実行されません。アプリケーション出力が表示されるだけです。Windows の [プログラム] メニューから [ターミナル サービス クライアント] コマンドを実行し、クライアントからターミナル サービスにアクセスします。ターミナル サーバーに接続すると、クライアント環境が Windows サーバーと同じように見えます。ただし、アプリケーション自体がローカルで実行されているわけではないということが分かります。

ターミナル サービスの機能および利点を含む詳細については、Microsoft のマニュアルを参照してください。

ターミナル サーバー アプリケーションの計画上の考慮事項

重要: アプリケーションを生産環境に配置する前に、テスト サーバーにインストールすることをお勧めします。

ターミナル サービスをインストールする前に：

- サーバーにインストールするクライアント アプリケーション (InTouch など) を決定する
- クライアントに対するハードウェアの必要条件を確認する

- クライアントのサポートに必要なサーバー設定を決定する
- ターミナル サービスやその他のアプリケーションの実行に必要なライセンスを確認する
- ターミナル サービスの下で実行される InTouch アプリケーションのアラーム、セキュリティ、I/O、およびスクリプトなどを理解する

ターミナル サービス環境へのマネージド InTouch アプリケーションの配置

InTouch アプリケーションをターミナル サービス環境に配置する場合、各ノードに個別の InTouch アプリケーションを配置してください。

ターミナル サービス環境のアラーム

分散アラーム システムを **Terminal Services for InTouch** と併用することで、異なるターミナルセッションで実行しているアラーム クライアント上でどのアラームをどのように表示するかを選択できます。

アラーム プロバイダは、アプリケーションに固有な名前とアプリケーションのインスタンスで識別できます。この情報は、アラーム プロバイダまたはアラーム コンシューマが分散アラーム システムに登録するときに提供されます。

アラーム プロバイダを実行しているノードは、システム内のコンピュータ ノードを識別する固有な名前前で識別されます。この情報は、そのコンピュータ ノードでそのインスタンスが起動したときに分散アラーム システムに提供されます。

アラーム イベントがログ記録されると、ノードと完全なアラーム プロバイダの名前によってアラームのソースが識別されます。

ターミナル サービス環境でアラームが確認されると、記録されるオペレータ ノードは、オペレータによって使用されるターミナル サービスセッションが実行されているクライアント コンピュータの名前になります。コンピュータのノード名を取得できない場合、代わりにその IP アドレスが使用されます。

注意：また、アラーム プロバイダもターミナルセッションではサポートされていません。これらは、ターミナル コンソールでしかサポートされていないので注意してください。

ターミナル サービス環境のセキュリティ

InTouch アプリケーション、IndustrialSQL Server、およびその他のシステムに関する機密情報を保護するには、アプリケーションセキュリティを使用します。

- アプリケーションを保護するには、**\$Operator** システム タグ変数を使用します。これで、内部タグ変数に特定の関数をリンクすることにより、オペレータを制御してこれらの特定の関数にアクセスできるようになります。

\$Operator システム タグ変数の使用の詳細については、「InTouch のセキュリティ保護の概要」を参照してください。

- クライアント コンピュータを識別するには、**GetNodeName()** 関数を新しい **TseGetClientId()** 関数に置き換えます。ターミナル サービスを使用すると **GetNodeName()** 関数は、クライアント コンピュータ名ではなく、ターミナル サーバー名を返します。

侵入の試行を監視するには、セキュリティの監査を使用します。システム侵入のような攻撃を受けている疑いがある場合は、一連の監査可能なイベントをログできます。デフォルトでは、セキュリティ ログや監査は、通常は余分な処理能力を必要とするため、無効にされています。

注意：セキュリティの監査には大量のリソースを必要とします。InTouch アプリケーションのハードウェアの必要条件を正確に見積もるには、パイロットサーバーを評価するときに監査を有効にします。

ターミナル サービス環境の I/O

ターミナル サービス環境では、InTouch HMI は I/O サーバーを起動できません。表示セッションが開始される順序によっては、IOReinitialize() 関数を使用する必要があります。すべてのサーバー (I/O デバイスまたは表示アプリケーション) を実行してから、アプリケーションを起動してください。アプリケーションは、これらのサーバーから値を読み込みます。

WindowViewer の起動時に「I/O の初期化」のエラー メッセージが表示されないようにするには、

1. WindowMaker を開きます。

1. [ファイル] メニューの [設定] をクリックし、[WindowViewer] をクリックします。

WindowViewer の設定画面が表示されます。

1. [環境設定] タブの [I/O] セクションで [ローカルサーバーを起動] チェック ボックスをオフにします。



ターミナル サービス環境でのスクリプトの実行

ターミナル サービスで実行されるすべてのアプリケーションでは、単一の基準タイミング (サーバー クロック) が使用されるため、CPU 負荷が異常に高い場合はスクリプトが実行されない場合があります。異常な CPU 負荷は、過度なビデオ処理を行う場合や複数のアプリケーションに同じスクリプトのトリガ (就労時間の終了イベントなど) が定義されている場合に発生する可能性があります。サーバーが多数のクライアントからのスクリプト処理で忙しくなり、タイマーで一定時間ごとに起動される別のクライアントのスクリプトが起動できなくなることがあります。これにより、スクリプトがクライアントで実行されなくなります。

スクリプトを適切に実行するには、複数のスクリプトを共通のトリガに結合し、それらをタグ変数サーバーのような単一のアプリケーションに移します。このような理由から、パイロット導入を行います。パイロットを配置すると、十分なハードウェアを選択しているかどうかを決定するための耐久テストを実行できます。

InTouch を実行するためのターミナル セッションへの適切なログオン

各セッションは一意のアカウントでログオンする必要があります。これは、手動で行うか、一意のログオンを強制するようターミナル サービスを設定することによって行うことができます。

注意：複数のセッションで同じログオン アカウントを使用すると、破損やその他の予期しない結果が生じることがあります。

ターミナル サービス環境のアラーム クエリーの構文

セッションのアラームのアラーム クエリーの構文を以下に示します。

```
\\ServerNodename\InTouch!$System
```

コンソールアラームのアラーム クエリーの構文の場合、ノード名の後にコロン (:) を含めます。次に例を示します。

```
\\ServerNodename\InTouch!$System
```

ターミナル サービス環境のその他の制限

以下の表では、ターミナル サーバーでアプリケーションを実行する際の制限および提案するソリューションを説明します。

機能	サポート	コメント
WindowViewer	あり	WindowViewer は、ターミナル サービスでのサービスとしての実行はサポートされていません。
I/O デバイスまたは MS Office (例 : Excel) への DDE	なし	タグ変数サーバーを使用します(コンソールまたは別のコンピュータ)。これには、次の DDE QuickScripts が含まれます。 WWExecute()、WWpoke() および WWRequest()
MS Office からの DDE (例 : Excel で設定したホットリンク)	あり	Excel と InTouch HMI は、同一のセッションで実行する必要があります。
履歴トレンド	あり	値をログするには、タグ変数サーバーまたは NAD を使用します。複数のセッションが同じ履歴ファイルを読み取りますが、コンソールのみが履歴ファイルに書き込むことができます。
InTouch Alarm Logger	あり	--
MEM OLE オートメーション	あり	--
アラームの印刷	なし	--
変数値保持タグ変数	あり	NAD を使用する必要があります。
SQL アクセス (ODBC)	あり	データベースは、別のコンピュータ上に格納されている必要があります。
I/O デバイスまたは別の InTouch アプリケーションへの SuiteLink	あり	別の表示セッションと通信する場合は、ターミナル サーバー ノード名を含め、必要なセッションの IP アドレスをアプリケーション名に追加する必要があります (view10.103.25.6 など)。I/O Servers はクライアントセッションでサポートされていません。

スクリプトを使用した InTouch クライアント セッションの情報の取得

ターミナルサービスでは、次の InTouch QuickScript 関数を使用できます。

- [TseGetClientId\(\) 関数](#)
- [TseGetClientNodeName\(\) 関数](#)
- [TseQueryRunningOnConsole\(\) 関数](#)
- [TseQueryRunningOnClient\(\) 関数](#)

TseGetClientId() 関数

WindowViewer アプリケーションがターミナル サーバー クライアントで稼動している場合は、クライアント ID（クライアントの TC/IP アドレス）の文字列を返します。この ID は内部で使用され、SuiteLink サーバー名およびロガー ファイル名が生成されます。そうでない場合、TseGetClientId() 関数は空の文字列を返します。

構文

```
MessageResult=TseGetClientId();
```

例

MsgTag タグ変数に、クライアント IP アドレス 10.103.202.1 が保存されます。

```
MsgTag=TseGetClientID();
```

TseGetClientNodeName() 関数

Windows によって識別可能な名前が割り当てられたターミナル サーバー クライアントで表示アプリケーションが実行されている場合、クライアントのノード名を返します。そうでない場合、TseGetClientNodeName() 関数は空の文字列を返します。

構文

```
MessageResult=TseGetClientNodeName();
```

例

MsgTag タグ変数に割り当てられた値として、クライアントのノード名が返されます。

```
MsgTag=TseGetClientNodeName();
```

TseQueryRunningOnConsole() 関数

表示アプリケーションがターミナル サービス コンソールで実行されているかどうかを示すために、スクリプトから TseQueryRunningOnConsole() 関数を実行できます。

構文

```
Result=TseQueryRunningOnConsole();
```

戻り値

WindowViewer アプリケーションがターミナル サービス コンソールで稼動している場合は、ゼロ (0) 以外の整数値を返します。そうでない場合、TseQueryRunningOnConsole() 関数はゼロを返します。

例

WindowViewer がターミナル サービス コンソールで実行されている場合、IntTag は 1 に設定されます。


```
IntTag=TseQueryRunningOnConsole();
```

TseQueryRunningOnClient() 関数

WindowViewer アプリケーションがターミナル サービス クライアントで実行されている場合は、ゼロ (0) 以外の整数値を返します。そうでない場合、ゼロを返します。

構文

```
Result=TseQueryRunningOnClient();
```

戻り値

WindowViewer がターミナル サーバー クライアントで実行されていない場合は、0 を返します。

例

WindowViewer がターミナル サービス クライアントで実行されている場合、IntTag は 1 に設定されます。

```
IntTag=TseQueryRunningOnClient;
```

リモート デスクトップ サービスの概要

リモート デスクトップ サービス (以前の名称はターミナル サービス) は、Windows Server® 2008 R2 以降のバージョンにおいて、リモート デスクトップ セッション ホスト (RD セッション ホスト) サーバー上にインストールされた Windows ベースのプログラム、または完全な Windows デスクトップにアクセスすることのできるサーバーの役割です。リモート デスクトップ サービスでは、ユーザーは、企業ネットワーク内またはインターネットから RD セッション ホスト サーバーにアクセスできます。

ユーザーが RD セッション ホスト サーバー上のプログラムにアクセスすると、プログラムはサーバー上で実行します。各ユーザーには、自身のセッションだけが表示されます。セッションは、他のクライアント セッションから独立していて、サーバー オペレーティング システムによって等価的に管理されます。さらに、Hyper-V™ を使用するようにリモート デスクトップ サービスを構成して、仮想マシンをユーザーに割り当てることや、接続時に使用可能な仮想マシンをリモート デスクトップ サービスでユーザーに動的に割り当てることができます。

リモート デスクトップ サービスの詳細については、Windows Server 2008 R2 TechCenter (<http://go.microsoft.com/fwlink/?LinkId=138055>) のリモート デスクトップ サービスのページを参照してください。

リモート デスクトップ サービスの役割サービス

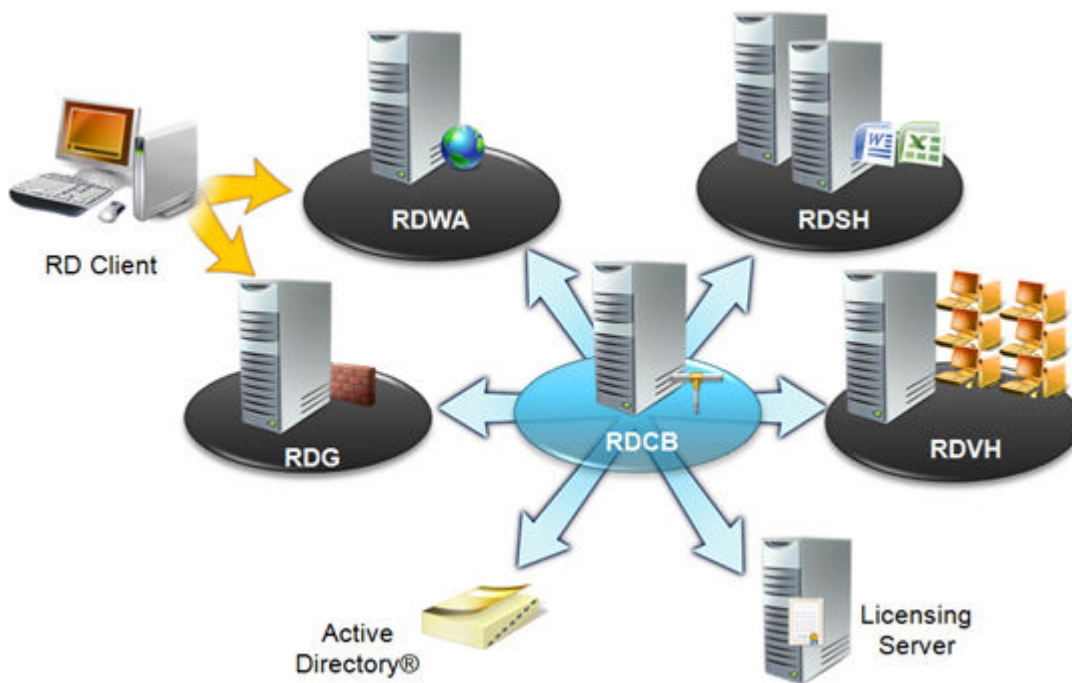
リモート デスクトップ サービスは、いくつかの役割サービスで構成されるサーバーの役割です。

Windows Server 2008 R2 以降のバージョンでは、リモート デスクトップ サービスは以下の役割サービスで構成されます。

- **RD セッション ホスト**：リモート デスクトップ セッション ホスト (RD セッション ホスト) (以前の名称はターミナル サーバー) では、サーバーで Windows ベースのプログラムまたは完全な Windows デスクトップをホストできます。ユーザーは、RD セッション ホスト サーバーに接続して、プログラムを実行することに加えて、ファイルの保存およびそのサーバー上のネットワーク リソースの使用を行うことができます。
- **RD Web アクセス**：リモート デスクトップ Web アクセス (RD Web アクセス) (以前の名称は TS Web アクセス) では、Windows 7 以降を実行するコンピュータの [スタート] メニューまたは Web ブラウザから RemoteApp およびデスクトップ接続にアクセスできます。RemoteApp およびデスクトップ

プ接続は、RemoteApp プログラムおよび仮想デスクトップのカスタム ビューをユーザーに提供します。

- **RD ライセンス**：リモートデスクトップライセンス (RD ライセンス) (以前の名称は TS ライセンス) は、RD セッション ホスト サーバーに 接続する各デバイスまたはユーザーに必要なリモート デスクトップ サービス クライアント アクセス ライセンス (RDS (CAL) を管理します。RD ライセンス を使用して、 リモート デスクトップ ライセンス サーバーで RDS CAL のインストール、発行、および 可用性の追跡を行います。
- **RD ゲートウェイ**：リモートデスクトップゲートウェイ (RD ゲートウェイ) (以前の名称は TS ゲートウェイ) では、承認済みのリモートユーザーがインターネットに接続されたデバイスから社内ネットワーク上のリソースに接続できます。
- **RD 接続ブローカー**：リモートデスクトップ接続ブローカー (RD 接続ブローカー) (以前の名称は TS セッションブローカー) は、負荷分散された RD セッション ホスト サーバー ファームでのセッション負荷分散とセッション再接続をサポートします。RD 接続ブローカーは、RemoteApp および デスクトップ接続からの RemoteApp プログラムおよび仮想デスクトップへのユーザー アクセスも提供します。
- **RD 仮想化ホスト**：リモートデスクトップ仮想化ホスト (RD 仮想化ホスト) は Hyper-V と統合し、仮想マシンをホストして仮想デスクトップとしてユーザーに提供します。一意の仮想デスクトップを組織内の各ユーザーに割り当てることや、仮想デスクトップのプールへの共有アクセスを提供することができます。



リモート デスクトップ サービス (RDS) 接続の保護

攻撃に対する保護を提供するために、以下のセキュリティ プラクティスが推奨されます。

1. 強力なパスワードを使用する

リモート デスクトップにアクセスするすべてのアカウントで強力なパスワードを使用します。

2. ソフトウェアを更新する

Microsoft Update の自動実行を有効にして監査することによって、クライアント ソフトウェアとサーバー ソフトウェアの両方の最新バージョンを実行します。

3. アカウント ロックアウト ポリシーを設定する

特定の回数のログイン試行が失敗した場合に一定期間アカウントをロックすることによって、"ブルートフォース" 攻撃を防止できます。

4. 2 要素認証を使用する

RD ゲートウェイは、スマートカードを使用した 2 要素認証をサポートします。

5. リモート デスクトップのライセンス ポートを変更する

デフォルトのリモート デスクトップ ポート (TCP 3389) へのアクセスを試みるネットワーク攻撃およびワームを防止します。

6. RD ゲートウェイを使用する

RD ゲートウェイは、リモート デスクトップ ポートへのアクセスを制限する一方で、単一の "ゲートウェイ" サーバーを介したリモート デスクトップ接続をサポートします。RD ゲートウェイ サーバーを使用する場合、デスクトップおよびワークステーション上のすべてのリモート デスクトップサービスは RD ゲートウェイを介してルーティングされます。RD ゲートウェイ サーバーは HTTPS (ポート 443) でリモート デスクトップ リクエストをリスンし、クライアントをターゲット マシン上のリモート デスクトップ サービスに接続します。手順については、<http://technet.microsoft.com/ja-jp/library/cc770601.aspx> を参照してください。

7. リモート デスクトップ サービス接続にネットワーク レベルの認証を設定する

ネットワーク レベルの認証では、ユーザーはセッションを作成する前に RD セッション ホスト サーバーに対して認証される必要があります。ネットワーク レベルの認証を使用すると、RD セッション ホスト サーバーの可用性が向上します (RD セッション ホスト サーバーのサービス拒否攻撃のリスクが削減されます)。詳細については、<https://technet.microsoft.com/ja-jp/library/hh831778.aspx> を参照してください。

8. サーバー認証および暗号化レベルを設定する

デフォルトでは、ターミナル サービス セッションでは、ネイティブのリモート デスクトップ プロトコル (RDP) 暗号化が使用されます。しかし、RDP では、ターミナル サーバーの ID を検証する認証は提供されません。サーバー認証にトランスポート層セキュリティ (TLS) 1.0 を使用してターミナル サービス セッションのセキュリティを強化して、ターミナル サーバー通信を暗号化できます。セキュリティを強化するには、RDS およびクライアント コンピュータを TLS に対して正しく構成する必要があります。デフォルトでは、クライアントとサーバーの間の RDS 接続が最高のセキュリティレベル (128 ビット) で暗号化され、転送されるデータの整合性と機密性が保証されます。

Windows Server 2016 のリモート デスクトップ サービスのベスト プラクティス

Windows Server 2016 環境では、リモート デスクトップ サービスに対して以下のベスト プラクティスを実装できます。

- 多要素認証を使用する

多要素認証で Active Directory のパワーを活用して高いセキュリティ保護を適用します。詳細については、Microsoft のドキュメント (<https://docs.microsoft.com/ja-jp/windows-server/remote/remote-desktop-services/rds-plan-mfa>) を参照してください。

- ユーザー プロファイル ディスク (UPD) でデータ ストレージを保護する

ユーザー プロファイル ディスク (UPD) では、ユーザーは単一のコレクション内のユーザー データ、カスタマイズ、およびアプリケーション設定を使用できます。UPD はユーザーおよびコレクション単位の VHD ファイルです。UPD は、ユーザーがサインインするときにユーザーのセッションにマウントされる一元的な共有に保存され、セッション中はローカルディスクとして扱われます。詳細については、Microsoft のドキュメント (<https://docs.microsoft.com/ja-jp/windows-server/remote/remote-desktop-services/rds-plan-secure-data-storage>) を参照してください。

章 16 アプリケーションの分散

通常、分散アプリケーションには中央開発ステーション、データの中央記憶装置、およびクライアントステーションがあります。分散アプリケーションを構築および保持するには、InTouch ネットワークアプリケーション開発 (NAD) を使用します。NAD を使用すると、アプリケーションの開発に支障をきたすことなく、多くのクライアントステーション上にそのアプリケーションのコピーを保持できます。アプリケーションの個々のコピーを使用すると、ビューアが冗長になります。アプリケーションが変更されると、クライアントステーションに自動的に通知されます。単一のコンピュータ、クライアントベースおよびサーバーベースの InTouch アプリケーションを作成できます。

System Platform IDE を使用して InTouch アプリケーションを管理および配置することもできます。

サポートされている InTouch アーキテクチャ

次の InTouch ネットワーク アーキテクチャがサポートされています。

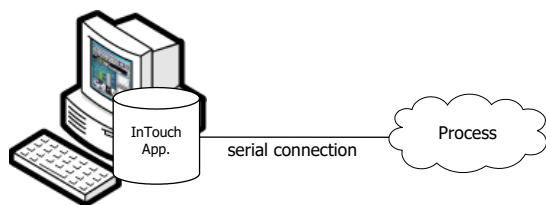
- 単一のコンピュータ
- クライアントベース
- サーバーベース
- NAD

単一のコンピュータのアーキテクチャ

通常、単一のコンピュータアプリケーションは、ネットワークに接続されていない、メインのオペレータインターフェイスとして機能する 1 つのコンピュータから構成されています。このコンピュータは、シリアルケーブルなどにより、生産プロセスに直接接続されています。

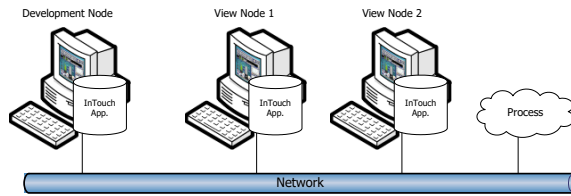
このアーキテクチャでは、単一のコンピュータで InTouch アプリケーションを開発します。アプリケーションを他のコンピュータにコピーしてから変更し、後で元のコンピュータにコピーを戻すことができます。

Development and View Node



クライアントベースのアーキテクチャ

クライアントベースのアーキテクチャでは、WindowViewer が実行されている各コンピュータに対して (表示ノード)、またはネットワークサーバー上の一意の場所に、1 つの InTouch アプリケーションの一意のコピーがあります。次の例では、アプリケーションが開発ノードで開発およびテストされ、次に各表示ノードにコピーされています。



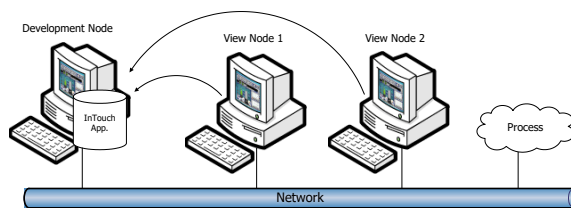
各ノード上にアプリケーションを重複して保持しているため本質的に冗長であり、使用できる表示ノードの数に制限はありません。

各表示ノードは、同じアプリケーションのコピーを持ち、I/O Servers や IndustrialSQL Server などのネットワーク データ ソースに対して同じアクセス権を持っている必要があります。ただし、各表示ノードが共有サーバーと個別に通信することになるので、ネットワークの負荷が増大します。

実行中のプロセスに影響を与えずに、開発ノードでアプリケーションを変更およびテストできます。ただし、変更されたアプリケーションを表示ノードに分散する必要があります。各表示ノードをローカルでシャットダウンし、新規アプリケーションをコピーし、次に再起動する必要があります。

サーバー ベースのアーキテクチャ

サーバー ベースのアーキテクチャにより、共通の InTouch アプリケーションが複数の表示ノードに分散されます。次の図では、2つの表示ノードが、開発ノードから同じアプリケーションにアクセスしています。



各表示ノードの場合：

- 開発ノードの共有ネットワーク ドライブに論理ドライブをマップする必要があります。
- InTouch プログラムで、共有アプリケーションを登録する必要があります。
- コンピュータは、アプリケーションによって参照されるどのデータ ソースにも同じアクセス権を持っている必要があります。スクリプトの組み合わせを使用してノード名を識別し、その名前に基づいて各データの場所を変更することによって、データ ソースの場所を定義する方法もあります。

このアーキテクチャでは、単一のアプリケーションを保持します。アプリケーションが変更されると表示ノードが自動的に更新され、WindowViewer が再起動します。

このアーキテクチャの短所は次のとおりです。

- アプリケーションの開発が制限される
- 開発ノードがダウンした場合、冗長性がない
- すべてのノードを同じ画面解像度にする必要がある

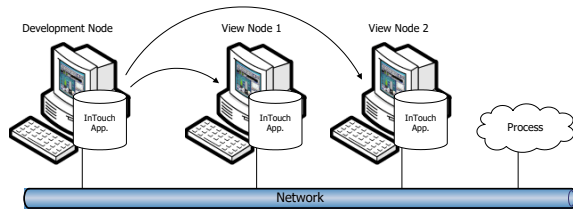
ネットワーク アプリケーション開発 (NAD)

ネットワーク アプリケーション開発 (NAD) アーキテクチャでは、中央のネットワーク ノードにアプリケーションのマスタ コピーを保持します (通常は開発ノード)。各表示ノードによって、ユーザーが定義した場所にアプリケーションがコピーされ、実行されます。

(WindowMaker の [システム] メニューの [クライアントに通知] コマンドを使用して) アプリケーションの変更がクライアントに通知されると、アプリケーションディレクトリにフラグが設定され、表示ノードによって読み取られます。

アプリケーションの変更が表示ノードでどのように処理されるのかを設定できます。「変更を無視する」、「表示ノードを自動的に終了して再起動する (マスタ アプリケーションを再ロードする)」などを定義できます。

次の図では、2 つの表示ノードには開発ノードから登録したマスタ アプリケーションがありますが、実際には各コンピュータ上でローカルで実行されています。



注意：マスタ アプリケーション ノードのアプリケーションディレクトリに履歴データを書き込むようアプリケーションを設定した場合、すべての NAD ノードが履歴データをマスタ アプリケーションに書き込もうとします。この問題を避けるには、各 NAD ノードで、マスタ アプリケーション ノードではなくローカルディレクトリに書き込まれるように履歴データを設定します。

大規模で複雑なアプリケーションを多くのノードに分散すると、最初のダウンロード時にシステムの応答時間が明らかに長くなります。ただし、更新は最適化されます。アプリケーションの転送は、低速ネットワークやシリアル接続では問題となる可能性があります。

また、特定のタイプのネットワーク トラフィックおよびアドレスを除去するルーターのユーザーなど、ネットワークに関するその他の制限に注意してください。

ネットワーク アプリケーションの計画上の考慮事項

InTouch アプリケーションを構築する際に選択したアーキテクチャにかかわらず、次のことを考慮してください。

- I/O データ ソースへのアクセス
- 共有ファイルへのアクセス
- データがログ記録される場所
- 特殊なネットワークの必要条件

ネットワーク アプリケーションの I/O データ アクセス

InTouch HMI では、リアルタイムの I/O データのリファレンスとして、アクセス名を使用します。各アクセス名は、ノード名、アプリケーション、およびトピックから構成されている I/O アドレスと同等です。分散アプリケーションでは、I/O リファレンスをネットワーク I/O Server のグローバルアドレスとして、またはローカル I/O Server のローカルアドレスとして設定できます。

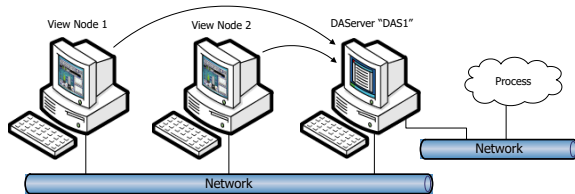
注意：InTouchView では、単一の Galaxy アクセス名しか使用できません。InTouchView では他のアクセス名を作成できません。InTouchView の制限の詳細については、「[ランタイムのアプリケーションの表示](#)」を参照してください。

表示ノードは、データ ソースに対して開発ノードと同じアクセス権を持っている必要があります。

グローバル I/O アドレスの使用

I/O データのグローバルアドレスを使用すると、すべての表示ノードで、ネットワーク ベースの I/O サーバーを共有できます。これにより複数の I/O サーバーを持つ必要がなくなりますが、耐故障性が低下し、結果的にはパフォーマンスが低下します。

以下の図では、2つの表示ノードで同じアプリケーションのコピーが実行されています。両方の表示ノードが同じ I/O データソースを参照しています。このアプリケーションは、完全にマッピングされた I/O アドレスを使用してデータソースを参照するので、すべてのリファレンスが同じ I/O サーバーを指すことになります。



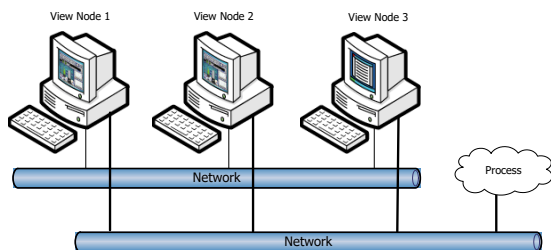
アクセス名の3つの部分からなるアドレス形式を使用して、別のノードに保存されているデータ要素を識別するように、InTouch アプリケーションを設定できます。アクセス名のアドレス形式には、リモートデータがあるノード名、アプリケーション名、およびトピック名が含まれています。InTouch アプリケーションは、アクセス名とアイテム名を組み合わせ使用してリモートデータを取得します。リモート I/O サーバーのアクセス名の定義の詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[I/O を使用したデータ アクセス](#)」を参照してください。

注: WindowMaker でアクセス名を作成するとき、アクセス名で SuiteLink プロトコルが使用されている場合、アクセス名は同じノード、アプリケーション、およびトピックにアクセスできません。ランタイム時にアクセス名を複製する際に、IOSetAccessName() 関数を使用してアクセス名にリダイレクトしないでください。リダイレクトされたアクセス名が機能しなくなります。

ローカル I/O アドレスの使用

各表示ノードに独自の I/O サーバーがある場合、I/O データに対するローカルアドレスが使用されます。このアーキテクチャでは、ネットワークで障害が発生しても各表示ノードが独立して動作し続けるため、耐故障性の高い操作を実現できます。

以下の図では、2つの表示ノードで、独自の I/O データソースを参照する同じアプリケーションのコピーが実行されています。各アプリケーションはローカルの I/O アドレスを使用してデータソースを参照するため、各参照はローカルの I/O サーバーを指すことになります。



ローカルの I/O サーバーを使用すると、プロセス接続ネットワークの負荷が大幅に増加します。たとえば、3つのノードの場合、各ノードのリクエストを別々に処理しなければならないため、トラフィック量は、1つのノードの場合の3倍になります。

ローカル I/O サーバーのアクセス名の定義の詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[I/O を使用したデータ アクセス](#)」を参照してください。

SuiteLink

SuiteLink 通信プロトコルは、TCP/IP プロトコルをベースにしています。高速の産業用アプリケーションでは、次のような機能を持つ SuiteLink を使用してください。

- タイムスタンプと品質インジケータがすべてのデータ値に関連付けられている VTQ (Value Time Quality) が、VTQ 対応のクライアントに送信されます。InTouch HMI は、VTQ インジケータと共にタグ変数データが送信される VTQ 対応のクライアントです。
- データのスループット、サーバーのロード、コンピュータ リソースの消費、およびネットワークの転送に対する包括的な診断を、Microsoft Windows オペレーティング システムのパフォーマンス モニタから行うことができます。
- アプリケーションが 1 つのノード上にある場合や、数多くのノードに分散されている場合にかかわらず、アプリケーション間で一貫した大容量のデータを維持できます。

SuiteLink は DDE、FastDDE、または NetDDE に取って代わるものではありません。クライアントとサーバー間の接続はそれぞれ、ご使用のネットワーク要件によって異なります。

共有ファイルへのアクセス

分散アプリケーションでは、ファイル参照を次のように設定できます。

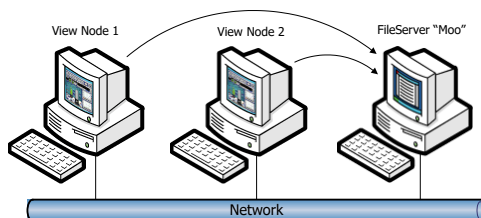
- ネットワーク ファイル サーバーに対するグローバルアドレス
- ローカル ファイルに対するローカルアドレス

表示ノードは、データ ソースに対して開発ノードと同じアクセス権を持っている必要があります。

ファイルデータに対するグローバルアドレスの使用

ファイルデータに対するグローバルアドレスを設定すると、共通のネットワーク ベースの一連のファイルをすべての表示ノードで共有できます。これによりファイルを集中管理できますが、ローカル コピーを持つ場合に比べると耐故障性が低下します。

次の図では、同じアプリケーションを実行している 2 つの表示ノードが同じレシピ ファイルを参照しています。各アプリケーションは、完全に規定どおりのネットワーク パスにマッピングされたドライブ文字を使用するため、すべての参照が同じファイルを指すことになります。



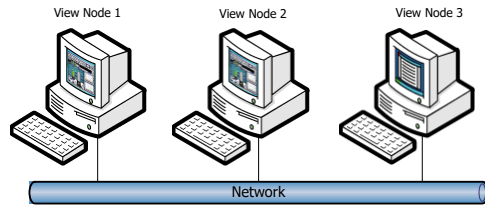
共有ファイルを設定するには

1. 参照ファイルを含む共有パスに、ネットワーク ドライブをマッピングします。たとえば、G:\Directory\Recipe.csv では、「G:\」は \\Moo\Share を参照するようにマッピングされたドライブ文字です。すべての表示ノードで、この同じドライブをマッピングする必要があります。
2. スクリプトでは、共有パスを参照します。次に例を示します。

```
RecipeSelectRecipe("G:\Directory\Recipe.csv", "review", "RecipeName");
```

ファイルデータに対するローカルアドレスの使用

各表示ノードにファイルの独自のコピーがある場合、ファイルデータに対するローカルアドレスを使用できます。次の図では、3つの各表示ノードで同じアプリケーションのコピーが実行されており、レシピファイルのローカルコピーを参照しています。



この例では、ローカルアドレスは次のとおりです。

```
C:\Directory\Recipe.csv
```

「C:\」はローカルドライブです。

スクリプトでは、ローカルパスを参照します。次に例を示します。

```
RecipeSelectRecipe("C:\Directory\Recipe.csv", "review", "RecipeName");
```

このアーキテクチャは高い耐故障性を持っています。ただし、ファイルの変更をすべての表示ノードにコピーする必要があります。

すべてのファイルアクセスを「読み取り専用」に設定し、ローカルファイルへの変更を禁止する必要があります。

UNCを経由した共有ファイルへのアクセス

通常、アプリケーションディレクトリのエントリ、アイテムの設定、および分散アラームなどのためにファイルパスを入力する場合に、UNC (Universal Naming Convention) アドレスを使用できます。UNC名を使用した場合、マッピングされたドライブを作成する必要はありません。

UNCアドレスの形式は「\\Node\Share\Path」です。各要素の意味を以下に示します。

- 「Node」は、ファイル共有が含まれているコンピュータの名前です。
- 「Share」は、そのコンピュータ上の共有フォルダに割り当てられている論理名です。
- 「Path」は共有に対する通常のパスです。

注意：SuiteLinkを使用する場合、ノード名は15文字に制限されています。

UNCを使用してファイルにアクセスするには、アクセス先のコンピュータ上にファイル共有を作成する必要があります。詳細については、Windowsのマニュアルを参照してください。

たとえば、「EngineRm」というネットワーク名のコンピュータがあり、共有名「Root」とルートドライブ「C:\」を共有しているとします。「C:\IT\Apps\Boiler」アプリケーションのUNCパスを設定するには、次のUNCを使用する必要があります。

```
\\EngineRm\Root\IT\Apps\Boiler
```

「Boiler」ディレクトリ自体が「Boiler」として共有されている場合、UNCは以下のように省略できます。

```
\\EngineRm\Boiler
```

PATH環境変数で指定したパスに共有がある場合、パスは必要ありません。

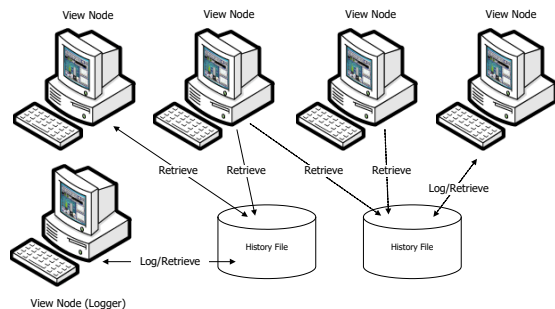
注意：UNCアドレスで参照されるファイルに書き込む必要がある場合、ローカルノードのときでもその共有は読み書き共有でなければなりません。UNCを使用してパスワード保護された共有にアクセスす

るには、まずネットワーク ドライブのマッピングを設定しなければなりません。Windows エクスプローラを使用すれば、リモート ノードからドライブのマッピングを設定できます。

分散環境でのデータのログ

InTouch 分散履歴システムを使用すると、ネットワーク上のどの InTouch アプリケーションからでも履歴データを取得できます。また、このシステムでは、複数の履歴データベースから同時にリモートでデータを取得することもできます。これらのデータベースは、履歴プロバイダと呼ばれます。

分散履歴ファイルには 1 つの InTouch ノードのみがログギングできます。ただし、ファイルのコンテンツを表示できる InTouch ノードの数には制限はありません。



履歴ファイルのデータを検索するリモート ノードから、最新時刻のデータ（ログギング ノードの時間）を参照できないことがあります。リモートトレンドグラフはログギング ノードディスクに書き込まれたデータしか表示できません。

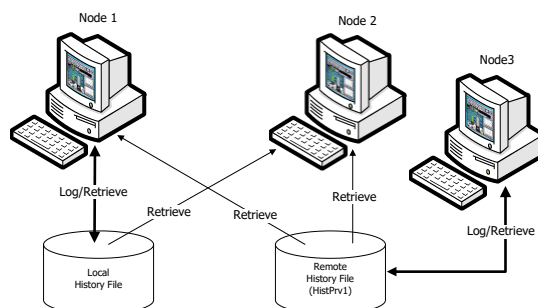
「データログ」用に指定された各タグ変数のデータは、そのタグのサンプルが 22 個集められたときに自動的にディスクに書き込まれます。HTUpdateToCurrentTime() 関数が呼び出された場合は、集められたサンプルの数に関係なく、データがディスクに書き込まれます。デフォルトでは、1 時間に 1 回データがディスクに書き込まれます。この間隔を変更するには、INTOUCH.ini ファイルに次の行を追加します。

ForceLogging=X

X は分単位の時間で、5 ～ 120 の範囲で設定することができます。

注: 分散履歴システムを使用するときは、NetDDE Helper サービスが実行されていなければなりません。

以下の図は、ネットワーク アプリケーション開発 (NAD) を使用してアプリケーションを分散する一般的な分散履歴システムの設定を示しています。



ノード 1 とノード 2 には、同じ InTouch アプリケーションのコピーが存在します。どちらのノードもローカルまたはリモートの履歴データ ファイルから検索できますが、ノード 1 のアプリケーションだけがローカルの履歴データ ファイルにログを出力するよう設定されています。ノード 3 も、同じリモートの履歴データ ファイルに対してログの出力と検索ができます。履歴プロバイダであるノード 3 には、

HistPrv1 という名前が割り当てられています。ノード 1 は開発とランタイムの両ステーションを兼ねていますが、ノード 2 はランタイム専用ステーションです。

この種類のアプリケーションを作成するには、次の手順を実行します。

1. 履歴プロバイダ リストを作成します。「[リモート履歴プロバイダの設定](#)」を参照してください。
2. 履歴トレンドオブジェクトを作成および設定します。詳細については、『*AVEVA™ InTouch HMI アプリケーション開発ガイド*』の「[タグデータのトレンド](#)」を参照してください。
3. 分散ログ出力用アプリケーションを設定します。「[分散履歴ログの設定](#)」を参照してください。
4. アプリケーションを配布します。「[NAD 用 InTouch アプリケーションの設定](#)」を参照してください。

アプリケーションは、手作業または NAD（Network Application Development）を使用して配布できます。アプリケーションを配布すると、アプリケーションの一部である履歴プロバイダ リスト ファイルも自動的に配布されます。

アプリケーションをいったん配布すると、表示ノードからローカルのタグ変数とリモートの履歴プロバイダのタグ変数を検索できます。アプリケーションはすべての表示ノード上で動作しますが、履歴ログ ファイルにログを出力できるのはロギング ノードだけで、他のノードは履歴ログ ファイルの読み取りしかできません。

リモート履歴プロバイダの設定

InTouch HMI と共に使用する各リモート履歴プロバイダの名前およびネットワーク ノードを指定する必要があります。リモートの InTouch 履歴プロバイダまたはリモートの IndustrialSQL Server 履歴プロバイダのどちらかを使用できます。

注記: InTouchView アプリケーションでは、リモート履歴プロバイダを設定できません。InTouchView アプリケーションの制限事項の詳細については、「[InTouchView アプリケーション](#)」を参照してください。

ローカルの InTouch アプリケーションは履歴プロバイダと見なされるため、アプリケーションで定義する必要はありません。

アプリケーションで未定義の履歴プロバイダを参照する場合、参照が WindowViewer によって無視され、エラー メッセージがロガーに書き込まれます。

HistData ユーティリティは、Historian プロバイダから履歴情報を取得できません。

履歴プロバイダを設定するには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[分散名前マネージャ] をクリックします。
[分散アラーム] のタブおよび [分散履歴] のタブを含む [分散名前マネージャ] 設定画面が表示されます。
3. [分散履歴] タブで [+] アイコンをクリックして履歴プロバイダを追加します。または、Alt+A を押します。
[履歴プロバイダの追加] ダイアログが表示されます。

Add history provider

Provider name
TankFarmServer

☐ InTouch provider UNC name

☒ Historian

AVEVA history provider

Provider name: TankFarmServer

Data source: Galaxy01 Credentials: Credential1

Test connection

Cancel Add

4. **[プロバイダ名]** ボックスに、新しい履歴プロバイダの名前を入力します。
プロバイダ名には、16 文字以下の英数字を使用できます。
5. InTouch 履歴プロバイダを設定するには、以下の手順を実行します。
 - a. **[InTouch プロバイダ]** をクリックします。
 - b. **[UNC 名]** ボックスに InTouch アプリケーションディレクトリへの UNC パスを入力して、**[追加]** をクリックします。
UNC パスの形式は以下のとおりです。
`\\node_name\volume_name\directory\`
UNC の場所はパスワードによって保護されているため、Windows エクスプローラを使用して、まずノード接続を確立する必要があります。
6. AVEVA 履歴プロバイダを設定するには、以下の手順を実行します。
 - a. **[Historian]** をクリックします。
 - b. **[データ ソース]** ボックスに、Historian Server がインストールされているサーバーのノード名を入力します。
 - c. **[資格情報]** ドロップダウン リストから、認証に使用する資格情報を選択します。

注記: スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報マネージャから取得されます。詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[資格情報マネージャの操作](#)」を参照してください。

7. [接続テスト] をクリックして、Historian Server データベースへの接続を確認します。データベースへの接続が成功したかどうかを通知するメッセージが表示されます。
8. [追加] をクリックして、ダイアログ ボックスを閉じます。

[履歴プロバイダ] リストに Historian Server ノードが表示されます。

リモート履歴プロバイダの動的な設定

ランタイム中に履歴トレンドグラフのリモート履歴プロバイダを動的に設定することもできます。これはスクリプトで HTSetPenName 関数を使用して、リモート履歴プロバイダのタグ変数の参照を指定します。次に例を示します。

```
HTSetPenName("HistTrendTag", 1, "HistPrv1.Boiler1");
```

ここで、「1」は指定したリモート履歴プロバイダ タグ変数を表示するトレンドペンの番号を表します。

ランタイム中の [履歴トレンドの設定] ダイアログ ボックスと Pen ドットフィールドは、リモート履歴プロバイダに対してはサポートされていません。

分散履歴ログの設定

履歴ファイルには 1 つの InTouch ノードのみをログ記録できます。ただし、複数の InTouch ノードにファイルを表示できます。

注記: InTouchView アプリケーションでは、履歴ログを設定できません。InTouchView アプリケーションの制限の詳細については、「[InTouchView アプリケーション](#)」を参照してください。

分散履歴ログを設定するには

1. [ファイル] メニューの [設定] をクリックし、[履歴ログ] をクリックします。

[履歴ログ] 設定画面が表示されます。

Historical logging

Historical logging Historian logging Printing control

☒ Enable historical logging

Keep log files for: 0 days

☒ Store log files in application directory

☐ Store log files in specific directory

Directory
C:\Users\Public\Wonderware\Intouch Applicator

Logging node

2. [履歴ログ実行] チェック ボックスをオンにし、グローバル タグのログ機能を有効にします。
3. [他のディレクトリにログ ファイルを保存] をオンにし、この入力ボックスにログ ファイルの保存場所のパスを入力します。

ここでは有効な UNC（汎用名前付け規則）パスを入力する必要があります。例: \\Node\Share\Path
NAD を使用している場合、アプリケーション フォルダ以外のフォルダをパスがポイントするようにしてください。

4. [ロギング ノード名] ボックスに、履歴ログ ファイルに記録を行うノード名を入力します。

ここで指定したノードだけが、ログ ファイルに記録されます。

5. [OK] をクリックします。

注記: [履歴ログ実行] オプションを選択したアプリケーションを WindowViewer ノードに分散すると、ノードはこのオプションをチェックしてログ ファイルへの記録を行うかどうかを判断します。[履歴ログ実行] を選択すると、次のような設定が可能です。[フィールドがノードの名前に等しい場合 - 履歴ログを実行する]、[フィールドがノードの名前に等しくない場合 - 履歴ログを実行しない]。

特殊なネットワークの考慮事項

低速なネットワークで作業をしていて、InTouch HMI の起動または情報の保存に長時間かかる場合、NAD クライアントの win.ini の設定を次のように変更します。

```
ViewNadClearNADCopyDirectory=0  
ViewNADCopyApplicationOnStartup=1  
ViewNADOnApplicationChanged=3（または 4）  
ViewNADThreadPriority=2
```

ViewNADOnApplicationChanged パラメータでは、3 の設定は、InTouch アプリケーション マネージャの [ノードの設定] ダイアログ ボックスの [WindowViewer に変更をロード] オプションに対応します。4 の設定は、[WindowViewer へ変更ロード時にメッセージ表示] オプションに対応します。これらの設定により、NAD のダウンロードが並行して発生している間でも、アプリケーションは個別の実行スレッドで実行を続行できます。

NAD がアプリケーションへの更新を実行している場合は、マスタから変更されたファイルのみがコピーされます。NAD は、ランタイム言語の切り替えのために、開発時の SmartSymbol のディクショナリ ファイルをコピーすることはしません。

NAD 用 InTouch アプリケーションの設定

ネットワーク アプリケーション開発（NAD）は、クライアント ベースとサーバー ベースのアーキテクチャの利点を組み合わせたものです。NAD を使用すれば、アプリケーションの変更を自動的に通知し、更新されたアプリケーションを表示ノードに自動的に分散できます。

NAD 用にアプリケーションを設定する場合、WindowViewer によるマスタ アプリケーションのコピー先のフォルダを指定する必要があります。

- これが開発ノードの場合は、ローカル フォルダ パス（c:\InTouch\NAD など）を入力することができます。また、ネットワーク上のリモート UNC パス（\\node\share\path など）を入力することもできます。これは、ファイルの大半が中央に保存されるファイル サーバー ベースのネットワークの場合に便利です。
- これがクライアント ノード（ランタイムのみ）である場合、通常ローカル フォルダ パスを使用します。

WindowViewer の動作がネットワークの遅延や障害の影響を受けないように、できるだけローカル フォルダを指定することが推奨されます。

注意: ルートフォルダまたはルートフォルダを指す UNC パス名は指定しないでください。表示ノードは、マスタ アプリケーションフォルダをコピーする前に、指定されたコピー先アプリケーションフォルダの下にあるファイルとサブフォルダをすべて削除してしまいます。そのため、マスタ アプリケーションフォルダのパス、またはマスタ アプリケーションフォルダへの UNC は決して使用しないでください。

フォルダを指定しない場合、WindowViewer が起動するフォルダ内に、NAD という名前のローカル サブフォルダが WindowViewer によって自動的に作成されます。NAD フォルダは、一時フォルダと見なされます。NAD 自体によってコピーされたもの以外のファイルはこのディレクトリに保存しないでください。

NAD 用アプリケーションを設定するには

1. アプリケーション マネージャ を起動します。
2. [ツール] タブの [ノードのプロパティ] をクリックします。
[ノードのプロパティ] 画面が表示されます。

Node properties

The screenshot shows the 'Node properties' dialog box with the 'App development' tab selected. The 'Application path' is set to 'C:\ProgramData\InTouchDemos\demoapp1_1280'. The 'Enable network application development' radio button is selected. Under 'Network application development', the 'Local working directory' is set to 'C:\Users\wwuser\AppData\Local\NAD'. The 'Polling period' is set to 10 seconds. Under 'Change mode', the 'Prompt user to restart WindowViewer' radio button is selected.

App development Resolution Memory settings Performance Security

☐ None

☐ Start following application in WindowViewer as a service

Application path: C:\ProgramData\InTouchDemos\demoapp1_1280

☒ Enable network application development

Network application development

Local working directory: C:\Users\wwuser\AppData\Local\NAD

Polling period: 10 sec

Change mode

☐ Ignore changes

☐ Restart WindowViewer

☒ Prompt user to restart WindowViewer

☐ Load changes into WindowViewer

☐ Prompt user to load changes into WindowViewer

Cancel Ok

3. [ネットワーク アプリケーション開発をオン] ラジオ ボタンを選択します。
4. [ローカルディレクトリ] フィールドに、WindowViewer によるマスタ アプリケーションのコピー先フォルダへのパスを入力します。

5. **[ポーリング周期 (秒)]** ボックスに、表示ノードが開発ノードの更新をチェックする間隔を秒単位で入力します。
- ここでは、小さすぎる値を設定しないようにしてください。WindowViewer がマスタ アプリケーションの変更を必要以上に頻繁にチェックすると、実行中のアプリケーションへのサービスが妨げられる可能性があります。
6. **[モードの変更]** 領域で、マスタ アプリケーションが変更された場合の WindowViewer の動作を決定するオプションを選択します。
- 開発ノードで行われた変更を WindowViewer ノードが無視するように設定するには、**[変更を無視]** をクリックします。
 - 更新されたマスタ アプリケーションを WindowViewer ノードにコピーし (そのように設定されている場合)、次に再起動するように設定するには、**[WindowViewer の再起動]** をクリックします。
 - アプリケーションが変更されたというメッセージをオペレータに表示するには、**[WindowViewer の再起動時にメッセージ表示]** をクリックします。オペレータは、更新されたアプリケーションで WindowViewer を再起動するのか、または現在のアプリケーションの使用を続行するのかを選択できます。
 - 開発ノードで行われた変更を WindowViewer に動的にロードするには、**[WindowViewer に変更をロード]** をクリックします。大量の更新が行われた場合には、ロードする速度が遅くなることがあります。

注記: **[WindowViewer に変更をロード]** オプションは、アプリケーションの変更がマイナーで数が少ない場合にのみ使用することが推奨されます。マイナーな変更の例には、既存のウィンドウ内の変更、グラフィック ツールバー要素のサイズ変更、新しいグラフィック ツールバー要素の追加、および参照の変更などがあります。WindowViewer の再起動が必要な変更 (新しいタグの追加、新しいウィンドウの追加、設定の変更など) を行った場合、または再起動が必要かどうか分からない場合は、いずれかの再起動オプションを使用してください。

- アプリケーションが変更されたというメッセージをオペレータに表示するには、**[WindowViewer へ変更ロード時にメッセージ表示]** をクリックします。変更をロードするよう、オペレータにメッセージが表示されます。

7. **[OK]** をクリックします。

自動 NAD 更新の実行

アプリケーション開発中に、自動 NAD 更新を起動できます。

[クライアントに通知] コマンドを実行すると、マスター アプリケーションが変更されたことをすべてのリモート表示ノードに通知するフラグが設定されます。クライアントは、各ノードに対して定義された **[モードの変更]** オプションに基づいて、更新プロセスを自動的に開始できます。

スタンドアロン アプリケーション (産業用グラフィック埋め込み済み) を表示ノードで初めて開くとき、グラフィックが表示されず、エラーがロガーに記録されることがあります。この問題を回避するには、マスター ノードから **[クライアントに通知]** コマンドを実行します。この操作により、産業用グラフィックが **[変更モード]** オプションに基づいて表示ノードに読み込まれます。

自動更新を実行するには

1. WindowMaker でアプリケーションを開きます。

2. [プロパティ] メニューの [クライアント] グループで次の操作を行います。

- a. クライアントに直ちに通知を行うには、[今すぐ通知] をクリックします。
- b. WindowMaker を終了するときに NAD クライアントに通知を送信するには、[終了時に通知] をクリックします。

注記: [終了時に通知] オプションを選択すると、WindowMaker が閉じるたびに、最後の通知以降に行われた変更がないかが検証されます。変更が検出された場合、NAD クライアントに通知するかどうかを確認するメッセージを含むダイアログ ボックスが表示されます。クライアントに通知する場合は、[はい] をクリックします。変更を無視する場合は、[いいえ] をクリックします。

手動による NAD 更新の実行

稼働中の表示ノード上でオペレータが手動で NAD 更新を開始できるようにするスクリプトを記述できます。

NAD を持つアプリケーションを手動で更新するには、[ノードのプロパティ] ダイアログ ボックスで [モードの変更] オプションを [変更を無視] に設定する必要があります。詳細については、「[NAD 用 InTouch アプリケーションの設定](#)」を参照してください。

NAD 更新を手動で実行するには、スクリプトで次のシステム タグ変数および関数を使用します。

- [\\$ApplicationChanged システム タグ変数](#)
- [\\$ApplicationVersion システム タグ変数](#)
- [RestartWindowViewer\(\) 関数](#)
- [ReloadWindowViewer\(\) 関数](#)

\$ApplicationChanged システム タグ変数

ネットワーク アプリケーション開発 (NAD) アーキテクチャで、マスタ アプリケーションに変化があったことを通知します。

カテゴリ

アプリケーション

使用法

\$ApplicationChanged

備考

このシステム タグ変数は、WindowMaker の [システム] メニューにある [クライアントに通知] コマンドの選択により、更新シグナルが発生する度に 1 に変化します。アプリケーションが更新されると、\$ApplicationChanged が 0 にリセットされます。マスタ アプリケーションが変更されたことをオペレータに通知するメッセージを生成するために、このタグ変数を使用できます。

また、\$ApplicationChanged システム タグ変数をデータ変更スクリプト内で使用すれば、ノード更新通知スクリプトを作成できます。このスクリプトにより、独自のダイアログ ボックスを起動したり、実行中のプロセスを停止できます。その後、ReloadWindowViewer() 関数を使用して更新プロセスを開始できます。

データタイプ

論理型 (読み取り専用)

例

データ変換スクリプトのタグ変数ボックスで次のステートメントを使用すると、スクリプトの本文が実行されます。変化を有効にするには **WindowViewer** の再起動が必要ですが、スクリプトの本文を使って、この通知をユーザーへウィンドウで表示することもできます。

```
$ApplicationChanged
```

参照項目

\$ApplicationVersion

\$ApplicationVersion システム タグ変数

アプリケーションの現在のバージョン番号です。この番号は、保存したり元に戻すことができるすべての変更により変化します。

カテゴリ

アプリケーション

使用法

```
$ApplicationVersion
```

備考

\$ApplicationVersion システム タグ変数に関連する値は、InTouch アプリケーションの現在のバージョンに設定されます。保存したり元に戻すことができるアプリケーションに対するすべての変更により、バージョンが変化します。マスタ アプリケーションが変更されたことをオペレータに通知するメッセージを生成するために、このタグ変数を使用できます。

データタイプ

実数型（読み取り専用）

例

アナログ データ表示リンクに使用すると、このシステム タグ変数により、**WindowViewer** で実行中のアプリケーションの現在のバージョン番号が表示されます。

```
$ApplicationVersion
```

参照項目

\$ApplicationChanged

RestartWindowViewer() 関数

WindowViewer をシャットダウンし、更新されたマスタ アプリケーションをコピーし（そのように設定されている場合）、次に **WindowViewer** を再起動します。

カテゴリ

システム

構文

```
RestartWindowViewer();
```

備考

自動更新のネットワーク アプリケーション開発（NAD）機能がオフの場合、アプリケーションの更新にこの関数を使用します。

NAD 更新がいつ発生するのかを決定するには、`$ApplicationChanged` システム タグ変数を使用します。

NAD 更新を開始するには、**[クライアントに通知]** コマンドを使用します。ただし、オペレータが更新を遅らせ、後で実行したい場合もあります。ボタンの動作スクリプトと共にこの関数を使用すると、オペレータの都合がよいときに **WindowViewer** を再起動するようにできます。

WindowViewer をシャットダウンせずに表示ノードを更新するには、`ReloadWindowViewer()` 関数を使用できます。

参照項目

`$ApplicationChanged`、`ReloadWindowViewer()`

ReloadWindowViewer() 関数

更新されたマスタ NAD アプリケーションを使用して、サービスを中断せずに動的に **WindowViewer** を更新します。

カテゴリ

システム

構文

```
ReloadWindowViewer();
```

この関数は、ユーザーが **WindowViewer** の再ロードを管理することを可能にします。

備考

自動更新のネットワーク アプリケーション開発 (NAD) 機能がオフの場合、アプリケーションの更新にこの関数を使用します。

NAD 更新がいつ発生するのかを決定するには、`$ApplicationChanged` システム タグ変数を使用します。

NAD 更新を開始するには、**[クライアントに通知]** コマンドを使用します。ただし、オペレータが更新を遅らせ、後で実行したい場合もあります。ボタンの動作スクリプトと共にこの関数を使用すると、オペレータの都合がよいときに **WindowViewer** でアプリケーションを再ロードするようにできます。

参照項目

`$ApplicationChanged`

アプリケーション編集のロック

複数の開発者が 1 つのアプリケーションを同時に編集しないように、**WindowMaker** は編集作業中にアプリケーションをロックします。ロックされたアプリケーションを開こうとすると、エラーメッセージが表示されます。メッセージには、アプリケーションを編集しているノード名が含まれています。

アプリケーションがロードされている状態で **WindowMaker** が異常終了すると、`appedit.lock` ファイルが削除されない場合があります。`appedit.lock` ファイルをアプリケーションディレクトリから削除することにより、ロックを手動解除することができます。

NAD 更新中のアプリケーションの変更

WindowViewer ノードは、アプリケーションを更新するとき、マスタ アプリケーションの属性（読み取り専用、システム、隠しファイルなど）を維持しようとします。

また、**WindowViewer** は、すべてのマスタ アプリケーションのファイルおよびサブフォルダをコピーします (`*.WWW`、`*.DAT`、`*.LGH`、`*.IDX`、`*.LOG`、`*.LOK`、`*.FSM`、`*.STG`、`*.DBK`、`*.CBK`、`*.HBK`、`*.KBK`、

.LBK、.NBK、*.OBK、*.TBK、*.WBK、*.XBK、*.\$\$\$、RETENTIV.X、RETENTIV.D、RETENTIV.A、RETENTIV.S、RETENTIV.H、RETENTIV.T、SSD_、WM.INI、DB.INI、LINKDEFS.INI、TBOX.INI、GROUP.DEF、および ITOCX.CFG を除く)。

注意：WindowViewer は、ランタイム言語の切り替えに必要なものを除く、コピー先のアプリケーション内のすべてのファイルおよびサブフォルダ フォルダを削除します。このフォルダは、一時フォルダと見なされます。その他のファイルはその場所に配置することはできません。

NAD クライアントは、クライアント アプリケーション ディレクトリに表示される、ファイルおよびサブディレクトリのローカル リストを作成することにより、更新を開始します。マスタ ファイルのリストの更新を探るとき、NAD クライアントは、ローカル リストの各マスタ ファイルに対する対応するクライアント ファイルを削除します。ローカル リストの残りのエントリは、アプリケーションから削除される必要がある、使われていないファイルおよびサブディレクトリです。

すべてのダウンロードされたファイルは、NAD_Temp と呼ばれる一時サブディレクトリにコピーされます。新しい、更新されたすべてのファイルが再試行制限内で正常にコピーされた場合にのみ、ファイルは、NAD_Temp からアプリケーション ディレクトリにコピーされます。NAD クライアントが更新を中止する必要がある場合、新しい、更新されたファイルを部分的にコピーしたことによって、実行中のアプリケーションが壊れることはありません。

すべての新しい、更新されたファイルがダウンロードされた後、NAD マスタとの連携が失敗した場合でも、NAD_Temp の更新をコピーし、使われていないファイルを削除することにより更新を完了できます。これにより、接続が失われたことによって、マスタ アプリケーション上にあることを確認できないため、ファイルは消去されません。

NAD は、アプリケーションのダウンロード中に、マスタ アプリケーションに追加の変更が行われたかどうかを検出できます。そのような状況が発生した場合、NAD はアプリケーションのダウンロードを中止します。最後の更新後に [クライアントに通知] コマンドを実行すると、NAD は、次のポーリング周期に最新のアプリケーション ファイルのダウンロードを自動的に開始します。そうでない場合、アプリケーションのダウンロードが発生する前に、次の [クライアントに通知] コマンドが発行されるまで待ちます。

ランタイム時のアプリケーションの解像度のスケーリング

ダイナミック レゾリューション機能 (DRC) を使用すると、作成した分散アプリケーションを異なる画面解像度で実行できます。

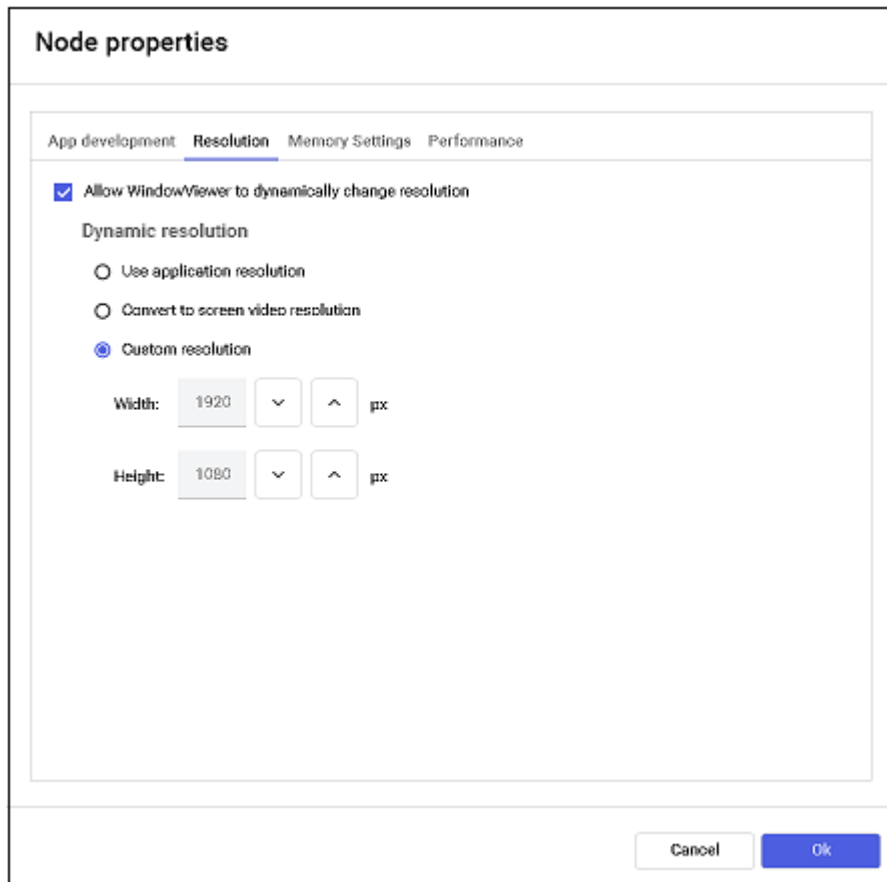
各表示ノードでは、解像度のカスタマイズを含め、アプリケーションを適切にスケーリングできます。このスケーリングは、WindowViewer がアプリケーションをコンパイルするときに行われるため、WindowMaker は必要ありません。各表示ノードはそれぞれ異なる DRC 設定を使用できるため、それぞれのノードで独自の設定を行う必要があります。

注意: DRC を使用してアプリケーションをスケーリングしない場合、WindowViewer は、ノードの画面解像度がアプリケーション開発ノードの画面解像度と同じ場合のみ、アプリケーションを実行します。2 つの画面解像度が異なる場合、WindowViewer はオペレータに WindowMaker を実行してノードの解像度に合わせてアプリケーションを変換するよう指示します。UNC パスをマスタ アプリケーション ディレクトリに設定してある場合、元のアプリケーションだけが変更されるため、上記の操作には注意が必要です。

DRC 用アプリケーションを設定するには

1. アプリケーション マネージャを起動します。

2. [ツール] メニューで、[ノードのプロパティ] をクリックします。[ノードのプロパティ] ダイアログボックスが表示されます。
3. [解像度] タブをクリックします。



4. WindowViewer でマスタ アプリケーションをローカルでスケーリングするには、[WindowViewer で動的に解像度を変更] チェック ボックスをオンにします。
5. [解像度] 領域で、以下のいずれかを選択します。
 - WindowViewer がアプリケーションを開発されたときの解像度で実行し、ノードの解像度を無視するようにするには、[アプリケーションの解像度を使用] を選択します。たとえば、アプリケーションが 800 x 600 の解像度で開発され、ノードの解像度は 1024 x 768 であるという場合でも、WindowViewer はアプリケーションを動的にスケーリングしません。アプリケーションの解像度は 800x600 のままになります。
 - WindowViewer が、ノードの解像度でアプリケーションを実行し、アプリケーションが開発されたときの解像度を無視するようにするには、[スクリーン ビデオの解像度に変換] を選択します。たとえば、ノードの解像度が 800 x 600 で、アプリケーションが 1280 x 1024 で開発された場合、WindowViewer はアプリケーションをノードの 800 x 600 の解像度に合わせて自動的にスケーリングします。
 - アプリケーションが作成されたときに画面解像度とターゲット解像度が異なる場合、元のアプリケーションの解像度から現在の画面解像度にスケールされます。元のアプリケーション解像度は、ターゲット解像度の設定に関係なく、アプリケーションが作成されたときの画面解像度です。たとえば、1280x1024 のターゲット解像度でアプリケーションが 1920x1080

で開発され、表示ノードが 800x600 の解像度でアプリケーションを実行している場合、WindowViewer では、1920x1080 の元のアプリケーション解像度を使用するようアプリケーションが動的にスケールされます。詳細については、「[元のアプリケーション解像度](#)」を参照してください。

- **[幅 (X)]** および **[高さ (Y)]** (整数値) ボックスに指定された特定の解像度で WindowViewer でアプリケーションを実行するには、**[解像度のカスタマイズ]** を選択します。アプリケーションの解像度とノードの解像度はどちらも無視されます。たとえば、**[幅 (X)]** が 512 に設定され、**[高さ (Y)]** が 384 に設定されている場合、アプリケーションはノードの画面上の 512 x 384 のピクセル領域に合わせて動的にスケーリングされます。
- アプリケーションが作成されたときに画面解像度とターゲット解像度が異なる場合、元のアプリケーションの解像度から現在の画面解像度にスケールされます。元のアプリケーション解像度は、ターゲット解像度の設定に関係なく、アプリケーションが作成されたときの画面解像度です。

6. **[OK]** をクリックします。

アプリケーションの解像度のロック

WindowMaker プロパティを設定することで、InTouch アプリケーション ウィンドウのサイズをロックすることができます。この機能は、ウィンドウとグラフィックのスケーリングをせずに、アプリケーションの解像度を変換できます。

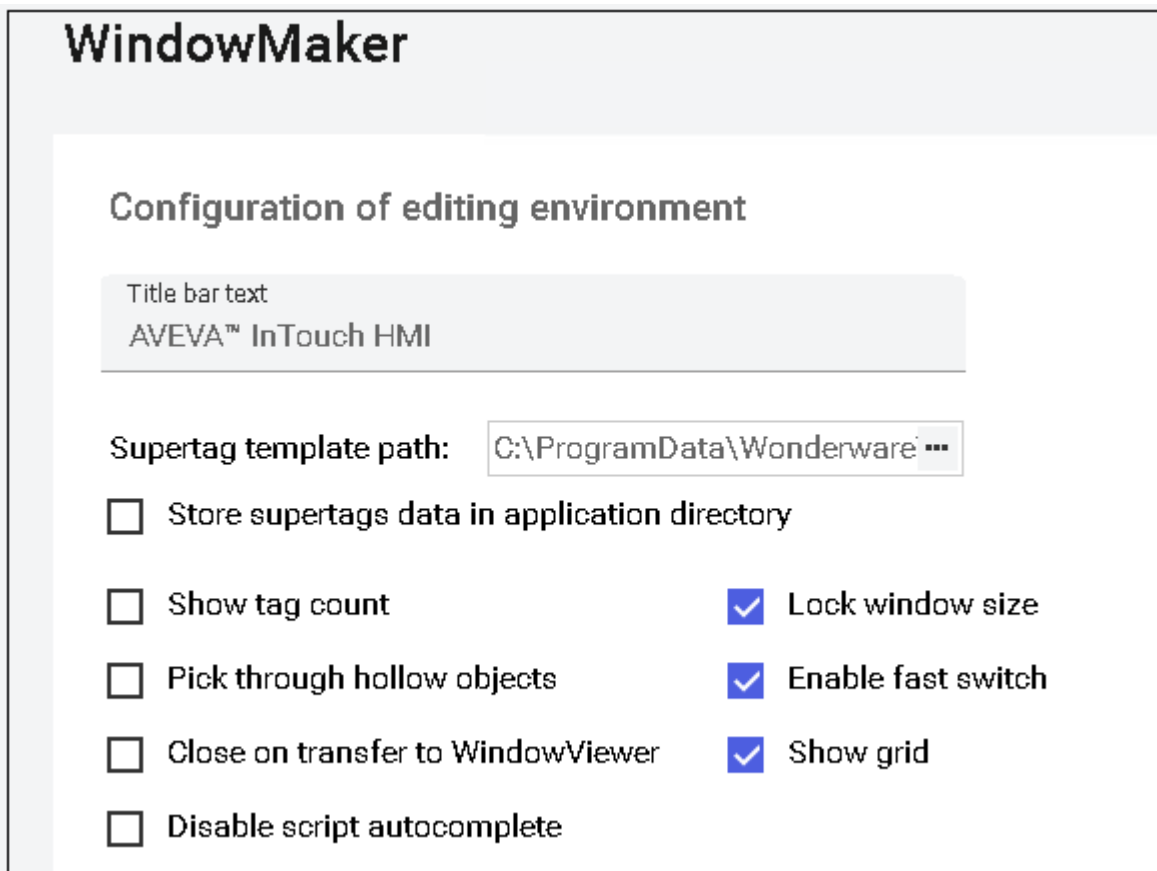
このオプションを選択した場合、異なる解像度のコンピュータでアプリケーションを開くと、ウィンドウとグラフィックのスケーリングを行わずにアプリケーションを新しい解像度に変換するかどうかが見つけられます。

アプリケーションの解像度は、WindowMaker 内から、またはアプリケーション マネージャからロックできます。

アプリケーションの解像度を WindowMaker からロックするには

1. WindowMaker を開きます。
2. **[ファイル]** メニューで、**[設定]** をポイントし、**[WindowMaker]** をクリックします。

WindowMaker の設定画面が表示されます。



WindowMaker

Configuration of editing environment

Title bar text
AVEVA™ InTouch HMI

Supertag template path: C:\ProgramData\Wonderware ...

☐ Store supertags data in application directory

☐ Show tag count ☒ Lock window size

☐ Pick through hollow objects ☒ Enable fast switch

☐ Close on transfer to WindowViewer ☒ Show grid

☐ Disable script autocomplete

3. [ウィンドウサイズをロック] チェック ボックスをオンにします。デフォルトでは、このチェック ボックスはオンになっていません。

4. [保存] をクリックします。

アプリケーションの解像度をアプリケーション マネージャからロックするには

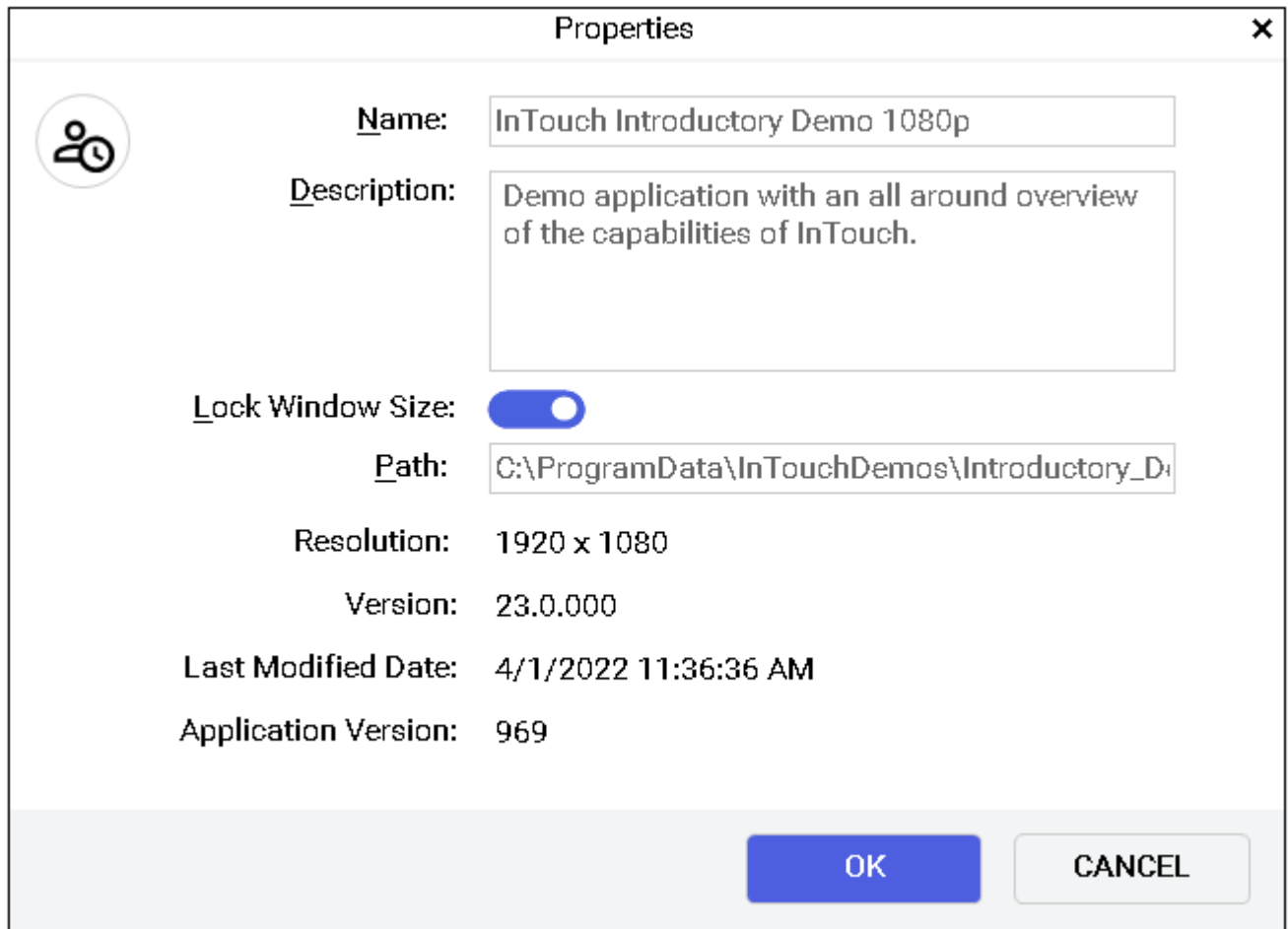
1. アプリケーション マネージャを開きます。

2. 設定するアプリケーションをクリックして選択します。


3. メニュー バーの [ファイル] をクリックし、[プロパティ] をクリックします。

[プロパティ] ダイアログ ボックスが表示されます。

4. [ウィンドウサイズをロック] スイッチを選択します。デフォルトでは、このチェック ボックスはオンになっていません。



Properties

 **Name:** InTouch Introductory Demo 1080p

Description: Demo application with an all around overview of the capabilities of InTouch.

Lock Window Size: ☒

Path: C:\ProgramData\InTouchDemos\Introductory_Demo

Resolution: 1920 x 1080

Version: 23.0.000

Last Modified Date: 4/1/2022 11:36:36 AM

Application Version: 969

OK **CANCEL**

5. [OK] をクリックします。

リモート ノードへのアプリケーションのパブリッシュ

アプリケーション パブリッシャを使用すると、別のコンピュータ上に InTouch アプリケーションをインストールするために必要な関連するすべてのファイルおよび設定手順が含まれている、圧縮された自己解凍形式のパッケージファイルを作成できます。アプリケーション パブリッシャを使用すると、スタンドアロンの InTouch アプリケーションをパブリッシュできます。マネージド InTouch アプリケーションは System Platform IDE を使用してパブリッシュします。

アプリケーションをパブリッシュするには、次の 2 つのオプションがあります。

- **ランタイムのみ。**ランタイムのみのパッケージには、アプリケーションを実行するために必要なファイルが含まれていますが、アプリケーションを編集するためのファイルは含まれていません。
- **開発時およびランタイム。**開発時およびランタイムのパッケージには、アプリケーションを編集および実行するために必要なすべてのファイルが含まれています。コンパイル済みの *.www ファイルなどのいくつかのランタイム ファイルは、開発時のファイルから再作成できるため除外されています。

パブリッシュ済みアプリケーションは、ダウンロードおよびインストール可能な Web Server にポストできます。ポストされたアプリケーションに対する、次のパッケージ情報が表示されます。

- パッケージの説明

- パブリッシャ名
- パブリッシュ済みファイル名（実行可能ファイル）
- アプリケーションの解像度

次に例を示します。

説明	乳製品処理アプリケーション
発行者	Navin Johnson
ファイル名	Dairy.exe / ビデオ解像度...（1024x768）

説明	乳製品処理アプリケーション
発行者	Navin Johnson
ファイル名	Dairy_2.exe / Video Resolution...(800x600)

発行したファイルのコンテンツ

以下の表は、すべての発行済みスタンドアロン InTouch アプリケーションの含まれているフォルダとファイル、および除外されたファイルを示します。

含まれているフォルダ	含まれているファイル	除外されたファイル
メイン アプリケーション フォルダ	全て	バックアップ ファイル。これ らのファイルのファイル名に は <code>?.bk</code> という拡張子が付いて います。
	これらの拡張子を持つファイ ル :	特殊ディレクトリ リストにな いサブフォルダ
	<code>.win</code> 、 <code>.dat</code> 、 <code>.lgh</code> 、 <code>.idx</code> 、 <code>.log</code> 、 <code>.fs m</code> 、 <code>.stg</code> 、 <code>.\$\$\$</code>	
	<code>retentiv.x</code> <code>retentiv.d</code> <code>retentiv.a</code> <code>retentiv.s</code> (ピリオドが 2 つ) <code>retentiv.h</code> <code>wm.ini</code> <code>db.ini</code> <code>linkdefs.ini</code> <code>tbox.ini</code> <code>group.def</code> <code>itocx.cfg</code>	WindowMaker でアプリケーシ ョンが開かれていることを示 す <code>appedit.lok</code> ファイル
	<code>SSD_*.xml</code> という形式の名前を 持つすべてのファイル	<code>.www</code> という拡張子が付いたフ ァイル名を持つコンパイルさ れたウィンドウ ファイル
ランタイム言語の切り替 えのためのディクショナ リ サブフォルダ	<code>.xml</code> という拡張子を持つすべ てのファイル	
Symbol サブフォルダ	すべてのファイルおよびサブ フォルダ	
	ウィザードがインストールさ れている場合は、 <code>wiz.ini</code> ファイ ル	
	ウィザードの実行可能ファイ ルのコピー	
	<code>.dll</code> ファイル、 <code>.wdo</code> ファイル <code>.wdf</code> ファイル	

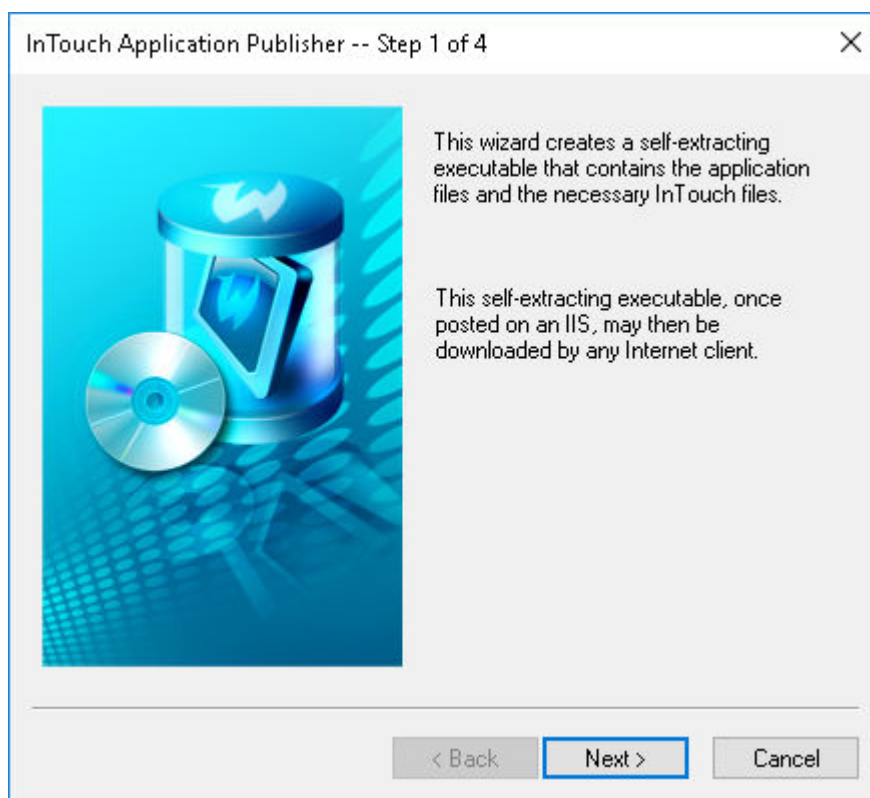
ランタイムのみのアプリケーションでは、SSD_*.xml というファイル名を持つすべてのファイルが除外されます。

スタンドアロン InTouch アプリケーションのパブリッシュ

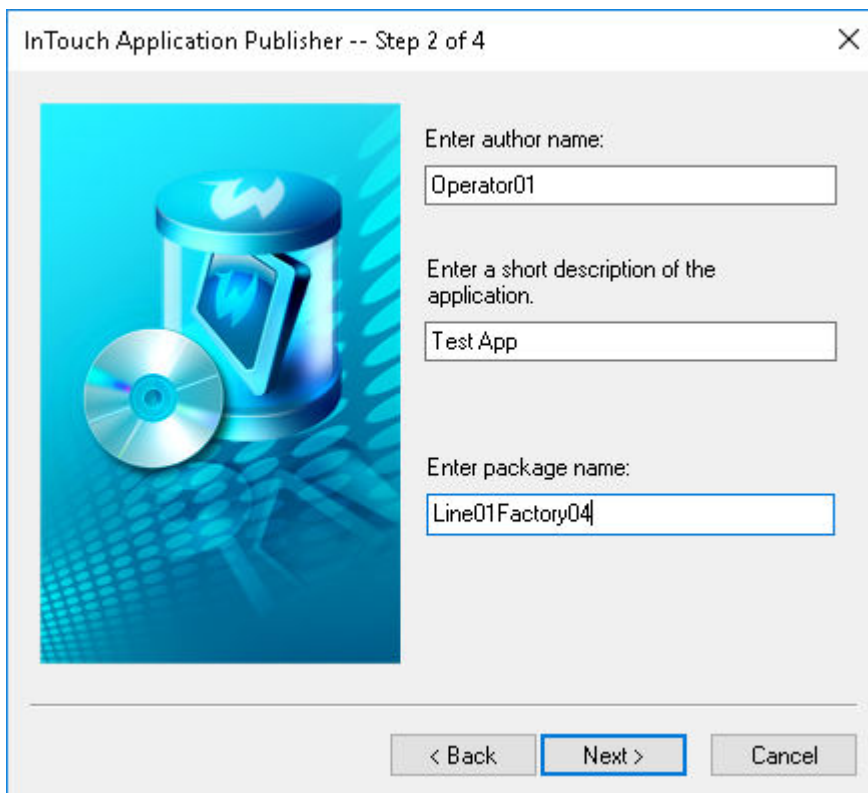
アプリケーション パブリッシャを使用すると、スタンドアロン InTouch アプリケーションをパブリッシュできます。パブリッシュ済みアプリケーションを特定の画面解像度で実行する場合、元のアプリケーションをパブリッシュする前に解像度を設定します。マネージド InTouch アプリケーションをパブリッシュするには、System Platform IDE 使用します。

スタンドアロン InTouch アプリケーションをパブリッシュするには

1. アプリケーション パブリッシャを起動します。
 - a. WindowMaker を開きます。
 - b. [ツール] パネルを展開し、[アプリケーション] を展開します。
 - c. [Application Publisher] をダブルクリックします。
- [InTouch アプリケーション パブリッシャ – ステップ 1/4] ダイアログ ボックスが表示されます。



2. [次へ] をクリックします。[InTouch アプリケーション パブリッシャ – ステップ 2/4] ダイアログ ボックスが表示されます。



InTouch Application Publisher -- Step 2 of 4

Enter author name:
Operator01

Enter a short description of the application:
Test App

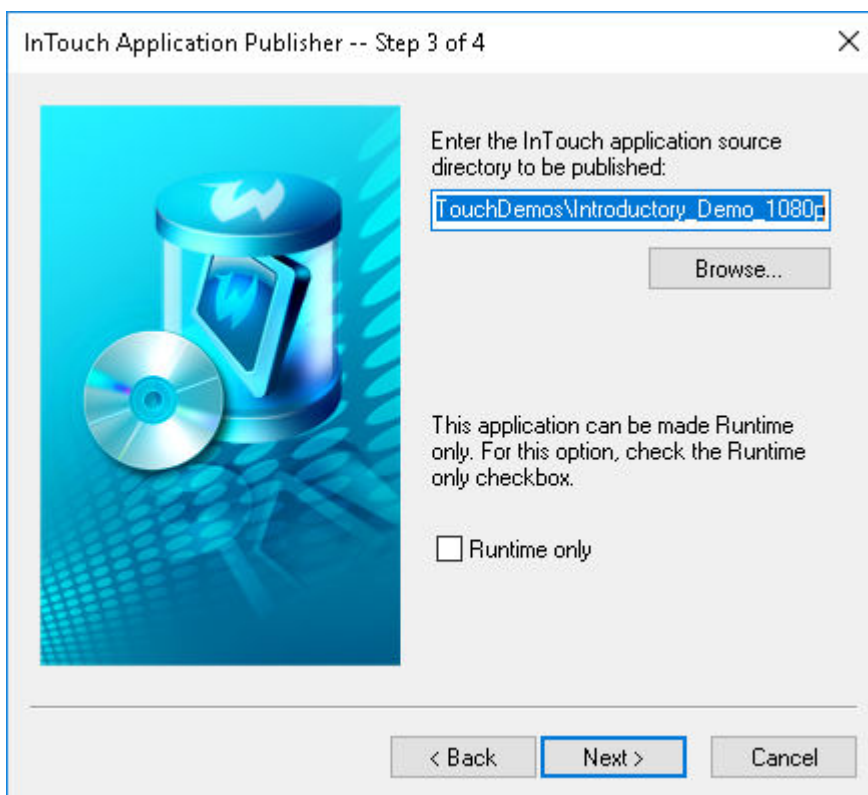
Enter package name:
Line01Factory04

< Back Next > Cancel

1. パッケージの詳細を設定します。

- [作成者名を入力してください] ボックスで、アプリケーションに関する連絡先の名前を入力します。入力できる名前は 256 文字までです。
- [このアプリケーションの説明を入力してください] ボックスに、アプリケーションの説明を入力します。入力できる文字数は 256 文字までです。
- [パッケージ名を入力してください] ボックスに、パブリッシュ済みアプリケーションパッケージの一意的な名前を入力します。入力できる文字数は 32 文字までです。既存のパッケージの名前を使用している場合、既存のパッケージが上書きされます。

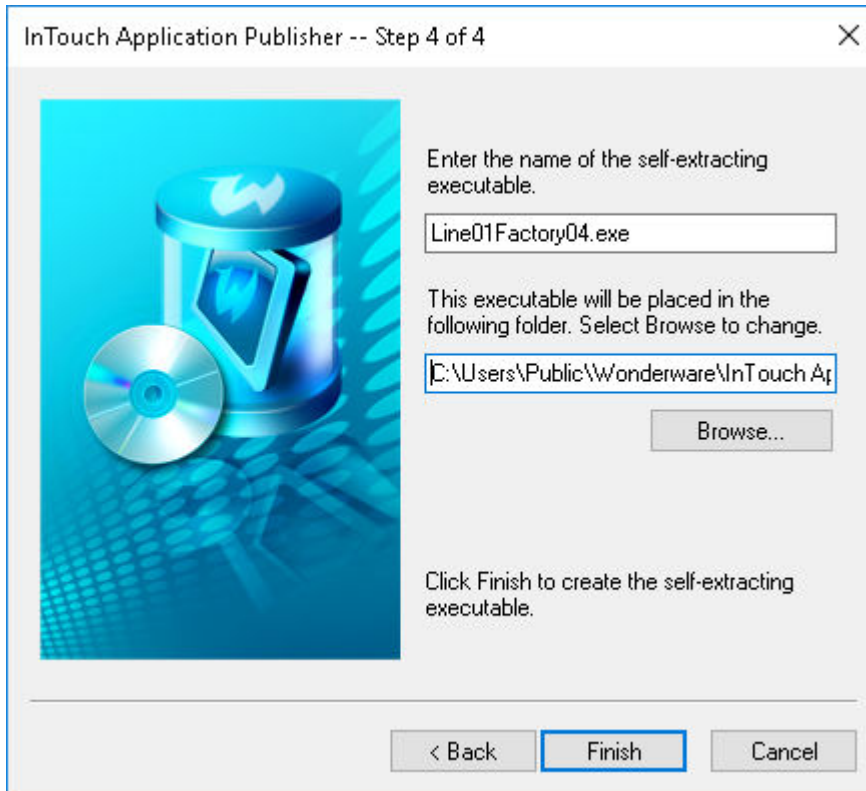
2. [次へ] をクリックします。[InTouch アプリケーションパブリッシャーステップ 3/4] ダイアログボックスが表示されます。



3. パブリッシュの詳細を設定します。

- ボックスに、InTouch アプリケーション フォルダへのパスを入力します。デフォルトのパスは WindowMaker のアプリケーション フォルダです。
- パブリッシュ済みファイルで開発 WindowMaker ファイルを除外するには、[ランタイム専用] チェック ボックスをオンにします。

4. [次へ] をクリックします。[InTouch アプリケーション パブリッシャー - ステップ 4/4] ダイアログ ボックスが表示されます。



5. 実行可能ファイルのアプリケーションパッケージの詳細を設定します。

- 最初のボックスで、最初のボックスの実行可能ファイル名が正しいことを確認します。デフォルトでは、実行可能ファイル名はパッケージ名と同じです。
- 2 番目のボックスで、実行可能ファイルを保存するフォルダへのパスを入力するか、または [参照] をクリックして別のフォルダを選択します。デフォルトでは、実行可能ファイルは現在の一時フォルダに保存されます。

6. [終了] をクリックします。

マネージド InTouch アプリケーションのパブリッシュ

マネージド InTouch アプリケーションをパブリッシュできます。パブリッシュ済み InTouch アプリケーションは、InTouchViewApp テンプレートには関連付けられていません。

パブリッシュ済みアプリケーションを IDE 内で編集したり、他の InTouchViewApp テンプレートにインポートすることはできません。つまり、IDE で管理したり再パブリッシュすることはできません。

パブリッシュ済み InTouch アプリケーションは、埋め込まれた産業用グラフィックを使用して、引き続き Galaxy と通信できます。たとえば、Galaxy にデータを書き込んだり、Galaxy データを表示することができます。

コピー、切り取り、貼り付け、複製、移動、サイズ変更、反転、回転、および InTouch アニメーションリンクの設定などの基本的な InTouch の操作を使用して、産業用グラフィックを編集できます。

ただし、産業用グラフィックは変更できません。また、InTouch アプリケーションに新しい産業用グラフィックを埋め込むことはできません。これらの操作は、マネージド InTouch アプリケーションを使用した場合のみ可能です。

これは、ArchestrA の処理要件をサポートしていない環境で行うことができます。たとえば、リモートでの工場操業または小規模ネットワークなどです。

マネージド InTouch アプリケーションのパブリッシュ

関連付けられている InTouchViewApp オブジェクトからマネージド InTouch アプリケーションをパブリッシュできます。

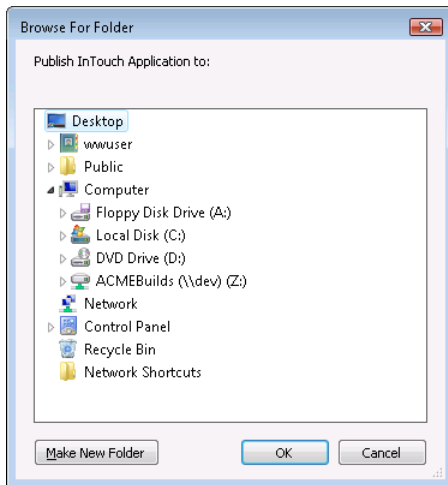
エクスポートは、オブジェクトおよびマネージド InTouch アプリケーションに関する情報が含まれているフォルダで構成されています。

これは、InTouchViewApp オブジェクト自体のエクスポートとは異なります。詳細については、「[InTouchViewApp オブジェクトのインポートとエクスポート](#)」を参照して下さい。

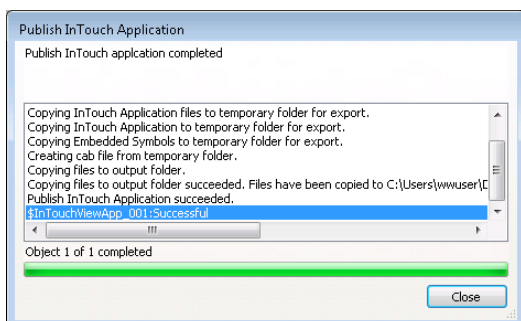
パブリッシュ済みの InTouch アプリケーションは、InTouchViewApp オブジェクトに再インポートできません。パブリッシュ済みの InTouch アプリケーションは編集できません。

マネージド InTouch アプリケーションをパブリッシュするには

1. System Platform IDE を開きます。
2. パブリッシュするマネージド InTouch アプリケーションが含まれている InTouchViewApp オブジェクトを検索します。
3. オブジェクトを右クリックし、[InTouch アプリケーションのパブリッシュ] をクリックします。[フォルダの参照] ダイアログ ボックスが表示されます。



4. InTouch アプリケーションのパブリッシュ先のフォルダを指定します。以下のいずれかを実行します。
 - 既存のフォルダを参照します。
 - 新しいフォルダまたはフォルダ構造を作成するには、[新規フォルダの作成] をクリックします。
5. [OK] をクリックします。[InTouch アプリケーションのパブリッシュ] 進捗ダイアログ ボックスが表示されます。



6. パブリッシュが完了したら、[閉じる] をクリックします。新しいパブリッシュ済み InTouch アプリケーションが含まれているディレクトリが、選択したフォルダに作成されます。任意のランタイムノードにコピーできるようになりました。

Insight へのアプリケーションのパブリッシュ

Insight Publisher を使用して、アプリケーションを Insight Web サイトにパブリッシュできます。アプリケーションマネージャまたは WindowMaker を使用できます。

Insight にアプリケーションをパブリッシュするには

1. アプリケーションマネージャを開きます。
2. [ツール] メニューの [ツール] グループで [Insight Manager] をクリックします。

[Insight Publisher] ウィンドウが表示されます。

InTouch WindowMaker では、[アプリケーション] の下にある [ツール] ペインから [AVEVA Insight Publisher] を選択できます。

次のいずれかのオプションを選択して、画面に表示される手順に従います。

- [パブリッシュ] - 既存の InTouch アプリケーションから新しい Insight データ ソースを作成します。
- [インポート] - OPC、MQTT、または OI Server から Excel スプレッドシート リスト項目をインポートします。
- [権限付与] - データ ソースを作成します。

詳細については、Historian のマニュアルを参照してください。

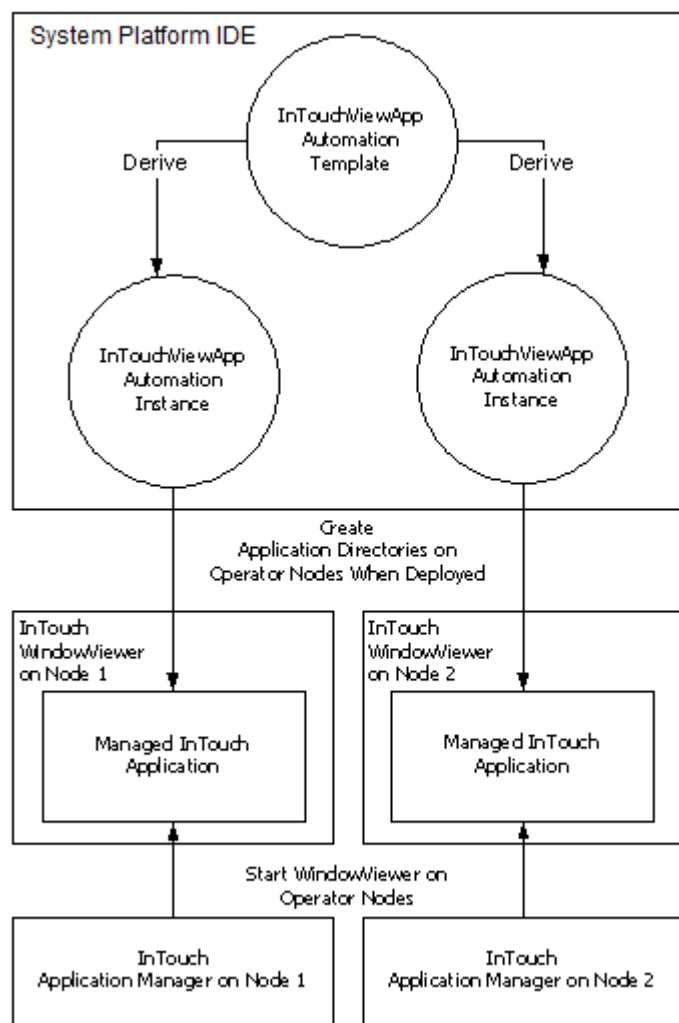
注記: アプリケーションをパブリッシュするには、Insight アカウントが必要です。

章 17 ランタイム時のマネージド InTouch アプリケーションの使用

InTouchViewApp インスタンスをリモート ノードに配置すると、マネージド InTouch アプリケーションをリモート ノードで実行できます。

また、これらのノードに InTouch アプリケーションや含まれている産業用グラフィックを配置することや、各ノードの変更を受け入れるのか拒否するのかを選択することもできます。

以下の図は、マネージド InTouch アプリケーションがランタイム ノードにどのように配置されるのかを示しています。



マネージド InTouch アプリケーションの配置

System Platform IDE からローカル ノードまたはリモート ノードに、マネージド InTouch アプリケーションを配置することができます。アプリケーションを配置した後、リモート ノード上の WindowViewer で実行できます。

注記: WindowViewer で一度に開くことのできるアプリケーションは1つだけです。プラットフォームがローカルノードに配置されている場合、他のスタンドアロンアプリケーションまたはマネージドアプリケーションで設定されているスタイルよりも Galaxy で設定したスタイルが優先されます。

初めての InTouchViewApp オブジェクトの配置

InTouchViewApp オブジェクトを初めて配置する場合、関連付けられている InTouch アプリケーションが、オブジェクトをホストするプラットフォームのノードにコピーされます。これはオペレータ ノードと呼ばれます。

マネージド InTouch アプリケーションを配置するには

1. System Platform IDE を開きます。
2. マネージド InTouch アプリケーションを配置する InTouchViewApp のインスタンスを選択します。
3. [オブジェクト] メニューで、[配置] をクリックします。[配置] ダイアログボックスが表示されます。
4. [OK] をクリックします。完全な InTouch アプリケーションがオペレータ ノードにコピーされます。

マネージド InTouch アプリケーションへの変更の配置

注記: [WindowViewer のデフォルト設定を上書き] オプションは、英語以外のオペレーティングシステムではテストされていません。Web Client では、このオプションがサポートされません。

マネージド InTouch アプリケーションは以下のように変更できます。

- マネージド InTouch アプリケーションで使用されている産業用グラフィックの参照、コンテンツ、またはサイズを変更する。
- InTouchViewApp テンプレートから WindowMaker を開いて、マネージド InTouch アプリケーション自体を変更する。

どちらの場合でも、変更を保存するときに、更新されたテンプレートから派生したインスタンスに変更が反映されます。変更は [変更の保留] アイコンによって識別されます。

変更は、実行中の WindowViewer セッションにすぐには反映されません。各ノードのオペレータは、変更を受け入れるのかまたは拒否するのかを選択できます。詳細については、「[オペレータ ノードでの新しいアプリケーションバージョンの受け入れ](#)」を参照してください。

マネージド InTouch アプリケーションへの変更を配置するには

1. System Platform IDE を開きます。
2. マネージド InTouch アプリケーションの変更を配置する InTouchViewApp のインスタンスを選択します。
3. [オブジェクト] メニューで、[配置] をクリックします。[配置] ダイアログボックスが表示されます。
4. [OK] をクリックします。変更がオペレータ ノードにコピーされます。

マネージド InTouch アプリケーションの起動

ユーザー ノードから、アプリケーションマネージャを起動し、実行するマネージド InTouch アプリケーションを選択できます。

また、異なる解像度でマネージド InTouch アプリケーションを実行するために、アプリケーション マネージャの解像度を設定することもできます。

マネージド InTouch アプリケーションを起動するには

1. InTouchViewApp オブジェクトが配置されるノードで、InTouch アプリケーション マネージャを起動します。
2. アプリケーション リストで、WindowViewer で実行するマネージド InTouch アプリケーションを選択します。
3. [WindowViewer] アイコンをクリックします。しばらくすると、WindowViewer でアプリケーションが起動します。

ダイナミック レゾリューション機能の設定をするには

1. InTouch アプリケーション マネージャを開きます。
2. [ツール] メニューで、[ノードのプロパティ] をクリックします。[ノードの設定] ダイアログ ボックスが表示されます。
3. [解像度] タブをクリックします。
4. [WindowViewer で動的に解像度を変更] チェック ボックスをオンにします。
[WindowViewer で動的に解像度を変更] がオフになっている場合、開発された解像度でマネージド アプリケーションが実行されます。
5. アプリケーションを実行する方法を設定します。以下のいずれかを実行します。
 - 開発されたときの解像度と同じ解像度でマネージドアプリケーションを実行するには、[アプリケーションの解像度を使用] をクリックします。
 - 画面の解像度を使用して実行できるようにマネージドアプリケーションを変換するには、[スクリーン ビデオの解像度に変換] をクリックします。
 - マネージドアプリケーションを指定した解像度に変換するには、[解像度のカスタマイズ] をクリックします。
6. [OK] をクリックします。

マネージド アプリケーションを配置するときの WindowViewer の再起動待機時間の制御

再起動が必要なマネージドアプリケーションを配置するときに WindowViewer が再起動するまでの遅延時間を制御できます。遅延時間を実装すると、アラーム サブシステムの適切な動作を保証できます。

WindowViewer の再起動遅延は、C:\Windows folder にある win.ini ファイルの以下のパラメータによって制御できます。すべてのパラメータの単位は秒です。

- ViewManagedRestartWaitPeriod=0

このパラメータは、WindowViewer の再起動待機時間を制御します。デフォルト値は 0 で、待機時間はありません。

- ViewNADShutdownWaitPeriod=30

ネットワーク アプリケーション開発 (NAD) を使用して管理されたアプリケーションの場合、このパラメータは NAD がシャットダウンするまでの遅延時間を制御します。デフォルトは 30 秒です。

- ViewNADRestartWaitPeriod=90

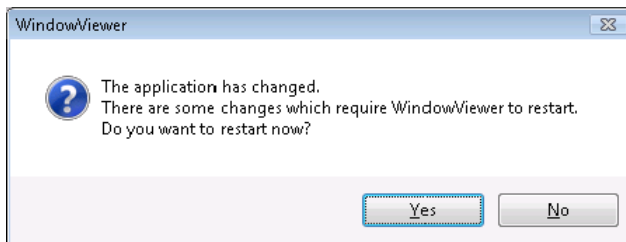
NAD を使用して管理されたアプリケーションの場合、このパラメータは NAD が再起動するまでの遅延時間を制御します。デフォルトは 90 秒です。

オペレータ ノードでの新しいアプリケーション バージョンの受け入れ

マネージド InTouch アプリケーションが変更され、その関連付けられている InTouchViewApp インスタンスが配置された場合、変更を受け入れるかまたは拒否するかをランタイム ノードで選択できます。

以下の場合に、マネージド InTouch アプリケーションへの変更を受け入れるかどうか確認するメッセージが表示されます。

- WindowViewer を InTouch アプリケーション マネージャから起動するとき。
- WindowViewer が実行しているとき。



変更の性質に応じて、WindowViewer アプリケーションを再起動するか、または再ロードのみを行うのが確認されます。また、アプリケーションの変更に対して、WindowViewer の動作を以下のように設定することもできます。

- これらの変更を WindowViewer が受け入れたり拒否する方法。
- 変更が保留中である間に、WindowViewer を再ロードするのか再起動するのかを WindowViewer が確認する頻度。

新しいアプリケーション バージョンをオペレータ ノードで受け入れるには

- [はい] をクリックします。マネージド InTouch アプリケーションへの変更がオペレータ ノードにコピーされ、WindowViewer が再起動または再ロードされます。

アプリケーションの変更に対する WindowViewer の動作を設定するには

1. マネージド InTouch アプリケーションを WindowMaker で開きます。
2. [ファイル] メニューで、[設定] をポイントし、[WindowViewer] をクリックします。
WindowViewer の設定画面が表示されます。
3. [マネージドアプリケーション] タブをクリックします。

4. [モードの変更] 領域で、変更が配置されたときに WindowViewer がどのように応答するのかを設定します。以下のいずれかを実行します。

- 配置された変更を WindowViewer が無視するように設定するには、[変更を無視] をクリックします。RestartWindowViewer() スクリプト関数および ReloadWindowViewer() スクリプト関数を手動で設定して、\$ApplicationChanged システム タグに応じて変更を受け入れることができます。
- WindowViewer を自動的に再起動するように設定するには、[WindowViewer の再起動] をクリックします。
- WindowViewer を再起動するように WindowViewer でプロンプトが表示されるように設定するには、[WindowViewer の再起動時にメッセージ表示] をクリックします。
- WindowViewer が変更を自動的にロードするように設定するには、[WindowViewer に変更をロード] をクリックします。
- WindowViewer への変更をロードするように WindowViewer でプロンプトが表示されるように設定するには、[WindowViewer へ変更ロード時にメッセージ表示] をクリックします。

注記: [WindowViewer に変更をロード] または [WindowViewer へ変更ロード時にメッセージ表示] オプションを選択した場合、WindowViewer を再起動して、マネージドアプリケーションに対する変更を認識させる必要があります。これらのオプションが選択されている場合でも、WindowViewer を再起動するよう求めるメッセージが表示されます。

1. [リマインダ間隔 (秒)] ボックスに、WindowViewer への変更のロードまたは WindowViewer の再起動をユーザーに確認する頻度を秒で入力します。このオプションは、該当する変更モードが設定された場合のみ使用できます。ユーザーに再度確認しない場合、間隔を 0 に設定します。
2. [OK] をクリックします。

WindowViewer のデフォルトの動作を設定するには

1. マネージド InTouch アプリケーションを WindowMaker で開きます。
2. [ファイル] メニューで、[設定] をポイントし、[WindowViewer] をクリックします。WindowViewer の設定画面が表示されます。
3. [マネージドアプリケーション] タブをクリックします。
4. [デフォルトの復元] をクリックします。設定がデフォルトの値にリセットされます。
5. [OK] をクリックします。

埋め込まれた産業用グラフィックでの ArchestrA スクリプトの実行

マネージド InTouch アプリケーションを WindowViewer で実行すると、要素またはシンボル スクリプト自体に関連付けられているすべての ArchestrA スクリプトが予期したとおりに実行されます。

ただし、シンボルに含まれているいくつかのスクリプトは長期に渡って実行され、その他の InTouch 要素の操作を妨げる場合があります。

これを回避するには、マネージド InTouch アプリケーション内のすべてのスクリプトに適用可能なスクリプト タイムアウトを設定します。スクリプト タイムアウトによりスクリプトの実行が停止され、オペレータに制御が戻ります。

デフォルトでは、スクリプトは 5 秒後にタイムアウトします。

スクリプト タイムアウトを設定するには

1. マネージド InTouch アプリケーションを WindowMaker で開きます。
2. [ファイル] メニューで、[設定] をポイントし、[WindowViewer] をクリックします。WindowViewer の設定画面が表示されます。
3. [マネージドアプリケーション] タブをクリックします。

The screenshot shows the 'Managed application' tab in the WindowViewer application. The 'Deployed updates' section includes a 'Local working directory' field with the value '%ITAPPDATA%\ArchestrA\ManagedApp'. Below this are two spinners: 'Reminder interval' set to 1800 seconds and 'Script timeout' set to 5000 milliseconds. There is an unchecked checkbox for 'Keep checked out'. The 'Change mode' section has five radio button options: 'Ignore changes', 'Restart WindowViewer', 'Prompt user to restart WindowViewer', 'Load changes into WindowViewer', and 'Prompt user to load changes into WindowViewer' (which is selected). A note at the bottom states: 'NOTE: WindowViewer must be restarted to recognize application changes made to a managed application. Even with this option selected users will be prompted to restart WindowViewer to recognize the application changes for a managed application.'

4. [スクリプトタイムアウト (ミリ秒)] ボックスに、値をミリ秒で入力します。
5. [OK] をクリックします。

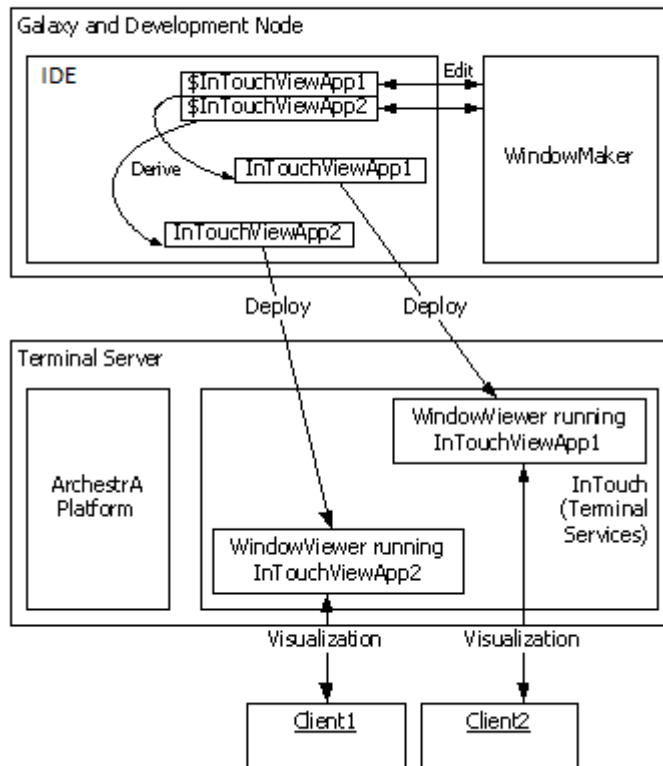
ターミナル サービス環境での InTouchViewApp オブジェクトの配置

ターミナル サービス環境でマネージド InTouch アプリケーションを実行できます。このアーキテクチャのおもな利点は、複数の InTouch アプリケーションを 1 つのコンピュータ上で同時に実行できる点です。

この処理を実行するには、以下を行う必要があります。

- ターミナル サービス エディションの InTouch を使用する。
- それぞれの ViewEngine ホストと共に InTouchViewApp の各インスタンスを配置した場合 (ターミナル サーバー上のいくつかの InTouchViewApplication インスタンスを選択した場合)。
- 各マネージド InTouch アプリケーションを、独自のターミナル サービス クライアント セッションで実行する。

Managed InTouch Applications in a Terminal Services Environment



注記: アプリケーションを複数のターミナルセッションクライアントで使用可能にする場合、ターミナルサーバーに配置する InTouchViewApp オブジェクトは1つだけです。アプリケーションを使用する各クライアントに対して個別の InTouchViewApp オブジェクトを配置する必要はありません。

AVEVA

Operate

章 18 ランタイムのアプリケーションの表示

WindowViewer を使用して、InTouch アプリケーションを実行できます。Application Server 環境専用設計されているアプリケーションは InTouchView アプリケーションと呼ばれます。InTouch Web Client を使用して、HTML5 対応の任意の Web ブラウザで産業用グラフィックを表示できます。このようなアプリケーションは WindowViewer で動作しますが、HMI のほとんどの機能は Application Server によって提供されます。

ランタイム時のアプリケーションの表示の概要

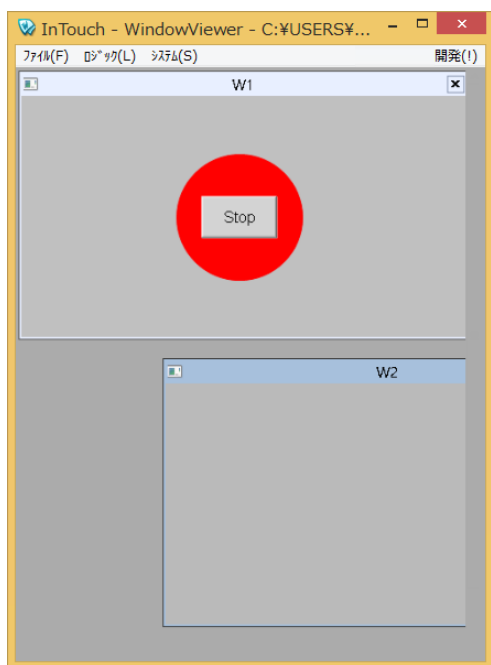
WindowViewer を使用して、InTouch アプリケーションを実行できます。Application Server 環境専用設計されているアプリケーションは InTouchView アプリケーションと呼ばれます。InTouch Web Client を使用して、HTML5 対応の任意の Web ブラウザで産業用グラフィックを表示できます。このようなアプリケーションは WindowViewer で動作しますが、HMI のほとんどの機能は Application Server によって提供されます。

別のターゲット解像度サイズでのランタイムのアプリケーションの表示

画面解像度と異なるアプリケーション ターゲット解像度を指定した場合、WindowViewer では、アプリケーションは指定されたターゲット解像度で表示されます。ランタイム時に表示されるのは、ターゲット解像度に収まるキャンバス境界内で開発されたウィンドウ、ウィンドウ コントロール、およびグラフィックだけです。ターゲット解像度サイズは、WindowViewer のメニューおよびタイトルバー コントロールに応じてランタイム時に自動的に調整されます。

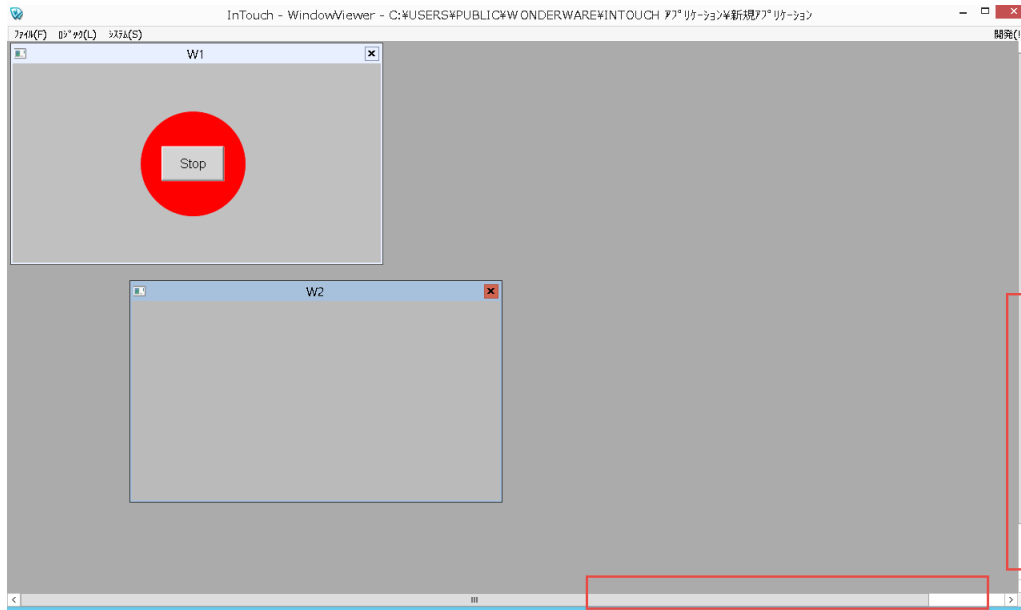
埋め込まれたコントロールのアスペクト比はランタイム時に維持されます。

以下に例を示します。



注意：指定したターゲット解像度が画面解像度よりも低い場合、ランタイム ウィンドウの幅と高さは、指定したターゲット解像度以上に調整することはできません。**WindowViewer** ウィンドウを最大化した場合、ターゲット解像度の最大解像度までしか拡大されません。

指定したターゲット解像度が画面解像度よりも高い場合、ランタイム時に垂直および水平スクロールバーが表示されます。ウィンドウはスクロールバーの操作に合わせてスクロールします。シンボルの表示アニメーションおよび **ShowGraphic** スクリプトのポップアップウィンドウとポップアップグラフィックはスクロールしません。



エラーメッセージとポップアップダイアログは、画面の中央ではなく、ターゲット解像度でのアプリケーションの中央に表示されます。キーボードとキーパッドの表示も同様です。

InTouch Web Client について

マウスジェスチャでズームするには、**InTouch Web Client** を使用すると、**InTouch HMI** アプリケーション内で使用されている産業用グラフィックを **HTML5** 対応 **Web** ブラウザで表示できます。組み込みの **Web Werver** によって、リモートデスクトッププロトコル (**RDP**) または **Microsoft Windows® Server** 用の **Internet Information Services (IIS)** を使用することなく、任意の **Microsoft Windows** クライアントまたはサーバーオペレーティングシステムからアプリケーショングラフィックへのアクセスが提供されます。スタンドアロンアプリケーションとマネージドアプリケーションの両方のアプリケーショングラフィックを **Web** ブラウザで表示できます。スタンドアロンアプリケーションウィンドウを **Web Client** で表示するには、産業用グラフィックに変換する必要があります。

InTouch Web Client の詳細については、「[Web ブラウザでのアプリケーショングラフィックの表示](#)」を参照してください。**Web Client** は、**System Platform** のインストールの一部としてインストールされます。**Web Client** の設定の詳細については、『**System Platform** インストールガイド』を参照してください。

元のアプリケーション解像度

元のアプリケーション解像度は、ターゲット解像度の設定に関係なく、アプリケーションが作成されたときの画面解像度です。

元のアプリケーション解像度は、以下の条件の下でのみ更新されます。

- アプリケーションが作成されたとき
- アプリケーションに変換が適用されたとき

後でアプリケーションが画面解像度に戻された場合、現在の画面解像度が元のアプリケーション解像度と異なっているとアプリケーション変換が発生します。それ以外の場合は、変換は発生しません。アプリケーションを作成するときにターゲット解像度を使用する場合、動的な解像度変換の動作に影響します。

ランタイム時のキーボード、マウス、およびタッチ ジェスチャの操作によるパンとズーム

枠ウィンドウを使用すると、ランタイム時に産業用グラフィックのパンとズームを行うことができます。この機能は、**WindowMaker** の **InteractionMode** プロパティで有効にします。

ランタイム時のズーム

ランタイム時にフレーム コンテンツをズーム インまたはズーム アウトすることができます。枠で正しいパンおよびズーム プロパティが有効化されていることを確認してください。ズーム インの上限は 5000% で、下限は 100% です。

シンボルの **ZoomPercent** プロパティを編集して、ランタイム時のシンボルまたは要素の可視性を変更できます。たとえば、次に示すプロパティを表示アニメーションに追加すると、ズーム パーセントレベルに応じてシンボルを動的に変更できます。

ZoomPercent => 200

注記: このプロパティへはランタイム時に書き込むことができます。

シンボルに対して **ZoomPercent** が設定されている場合、シンボルは設定されたパーセントで表示可能な領域の中央にズームされます。

シンボルの要素に対して **ZoomPercent** が設定されている場合、シンボルは設定されたパーセントでズームされますが、中央に表示されるのは要素になります。

ZoomPercent を要素に対して設定するスクリプトの例を以下に示します。

```
TextBox1.ZoomPercent = 500
```

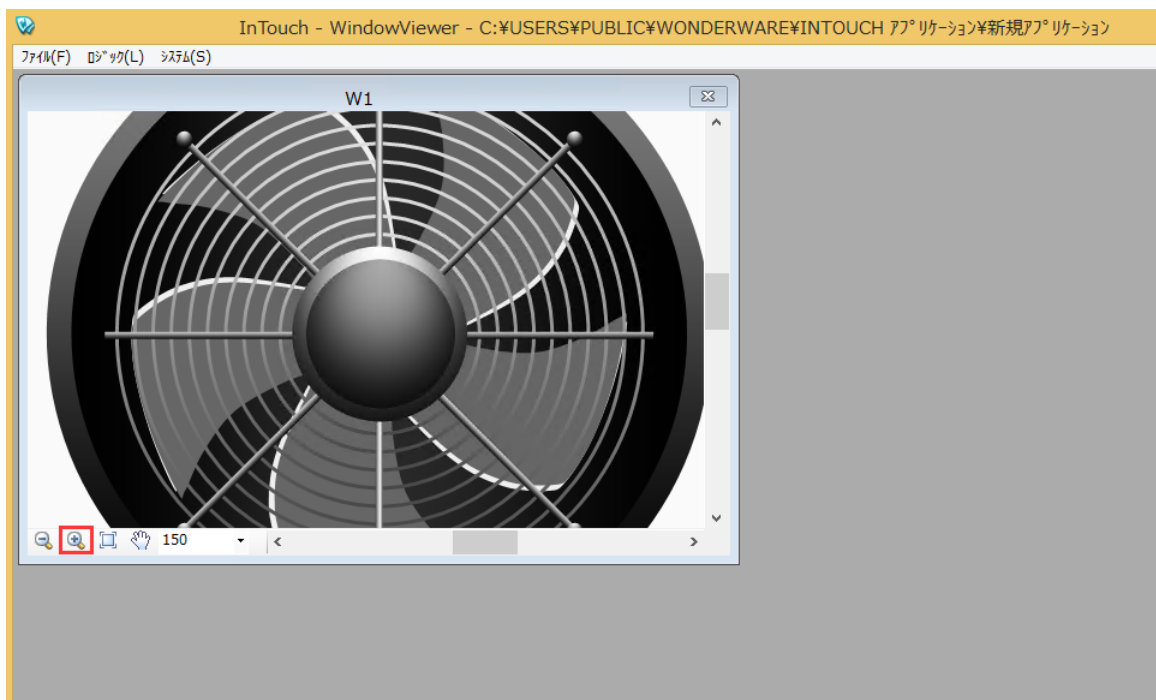
マウスでズームするには

以下の手順を実行します。

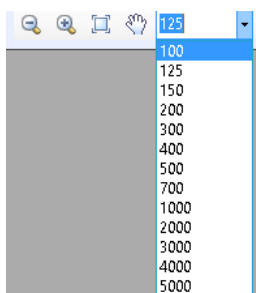
- 枠コンテンツをズーム インするには、"Ctrl" キーを押してマウス ホイールを上スクロールします。コンテンツがマウス ポインタの現在の位置からズーム インします。

注記: マウス ポインタが枠外にある場合、枠をズーム インできません。

- 枠コンテンツをズーム アウトするには、マウス ホイールを下スクロールします。
- [パンとズーム] コントロール ツールバーのズーム インおよびズーム アウト アイコンを選択します。その後、ズーム インする枠のコンテンツをクリックする必要があります。



- 枠コンテンツをダブルクリックすると、100% のズーム レベルに戻ります。
- [ズーム レベル] コンボボックスを使用して、事前定義済みのズーム レベルを選択します。



- [パンとズーム] コントロールツールバーで [ラバー バンド ズーム] アイコンを選択し、ズーム インする特定の領域を選択します。

キーボードでズームするには

以下の手順を実行します。

- ズーム インするには、Ctrl と + キーを同時に押します。
- ズーム アウトするには、Ctrl と - キーを同時に押します。

タッチ ジェスチャでズームするには

以下の手順を実行します。

- ズーム インするには、2 本の指を画面に置いて指の間の幅を広げます。
- ズーム アウトするには、2 本の指を画面に置いて指の間の幅を狭めます。
- ダブルタップすると、100% のズーム レベルに戻ります。

ズームの制限

ランタイム時のズーム機能には、以下の制限が適用されます。

- **Windows** の共通コントロールおよびクライアント コントロールのズームは **500%** までです。このようなコントロールには以下が含まれます。
 - カスタム コントロール: ラジオ ボタン グループ、チェックボックス、編集ボックス、コンボ ボックス、カレンダー、日付時刻ピッカー、およびリストボックス
 - 埋め込まれたアラーム クライアントおよびトレンド クライアント コントロール
 - サードパーティ製クライアント コントロール
- **Windows** コントロールは、マウス、キーボード、およびタッチによる入力を上書きすることがあります。
- **InTouch** は **Windows** コントロールのカスタム フォントを上書きしません。

ランタイム時のパン

該当するシンボルプロパティを設定して、ランタイム時の枠内のパン機能を有効化します。マウスおよびタッチ ジェスチャをしようしてランタイム時にパンできます。キーボードでのパン操作はサポートされていません。ランタイム時にパンするには、グラフィックのズーム レベルは **100%** 以上である必要があります。

マウスでパンするには

以下のいずれかを実行します。

- マウスの中ボタンを押します。
パン ハンドが表示されます。
- [パンとズーム] コントロール ツールバーのパン ハンドを選択します。マウスの左ボタンを押したまま、パンハンドを移動して画面をパンします。
画面は、マウス ボタンを離すまでパンされます。

タッチ ジェスチャでパンするには

1. 枠コンテンツ上に **1** 本の指を置きます。
2. 指を画面上で移動してパンします。

注記: どちらのパン方法でも、水平および垂直スクロール バーはパン方向に従って調整されます。

タッチ ジェスチャを使用してパンとズームを同時に行うには

1. 画面上に **2** 本の指を置き、指を下方向および左右に移動して、ズームしたコンテンツの中心を調整します。
2. 枠コンテンツ上でパンするには **1** 本の指を使用します。
3. 同時にズームするには、**2** 本目の指を枠コンテンツ上に置きます。

パンの制限

以下の制限が適用されます。

- キーボード ジェスチャを使用したパンはサポートされていません。

- **Windows** コントロールは、マウス、キーボード、およびタッチによる入力を上書きすることがあります。その結果、**Windows** コントロールを含む領域でのパンが無効化されることがあります。

タッチ ジェスチャのアニメーション サポート

タッチ サポートのすべてのアクション スクリプトは、ズーム レベルに関係なく同様に機能する必要があります。枠コンテンツをズーム インしたとき、すべてのインタラクションおよび表示アニメーションは、枠コンテンツが標準表示されている場合と同様に動作します。インタラクション アニメーションは、タッチで適切に機能します。

注記: **ShowSymbol** アニメーションまたは **ShowSymbol** スクリプトによって表示される産業用グラフィック ポップアップでは、デフォルトでパンとズームが有効化されています。しかし、この設定を無効にすることはできません。

以下の表には、タッチ サポート向けに一般的に設定されるアクション スクリプトが一覧表示されています。

アクション スクリプト	トリガされるタッチ
左クリック時	タッチ下げ
押している間	タッチしてスライド
左を離れた時	タッチ上げ
左ダブルクリック時	ダブルタップ
右クリック時	タッチして保持
右クリックを離れた時	タッチしてホールドし、離れたときに実行
右ダブルクリック時	サポートされていません。
右クリックを押している間	タッチしてホールドし、押したまま少しスライド
中クリック時	サポートされていません。
中クリックを押している間	サポートされていません。
中クリックを離れた時	サポートされていません。
中ダブルクリック時	サポートされていません。

注記: アニメーションを含む領域でタッチ インタラクションを使用すると、アニメーションがパン操作よりも優先され、パン操作は無視されます。1 本の指でのタッチ インタラクションを維持すると、その後のタッチ ポイントは無視されます。

このシナリオでパンを有効にするには、[パンおよびズーム] ツールバーの [パン] アイコンを選択します。

タッチ ジェスチャの制限

ランタイム時のパンとズームと同じ制限がタッチ ジェスチャに適用されます。以下の機能はサポートされていません。

- Windows 共通コントロールのフォントのスケール

注記: さらに、埋め込まれたウィンドウ コントロールを含むシンボルのスケール メカニズムは、コントロールを含まないシンボルのスケール メカニズムと異なります。埋め込まれたコントロールを含むシンボルの最大ズーム制限は **500%** ですが、コントロールを含まないシンボルは最大 **5000%** までズームできます。コントロールを含まないシンボルは、スムーズにスケールすることができます。ズームを行う際、埋め込まれたコントロールを含むシンボルを使用する場合のその他の制限は、コントロールが明滅することです。これは、タッチ ジェスチャでズームする場合に特に目に付きます。

枠ウィンドウと ShowGraphic() 関数の使用

ShowGraphic() スクリプト関数を実行して、枠ウィンドウに関連付けられたシンボルをランタイム時に変更できます。次の例に示すように、枠ウィンドウ名および関連付けられたシンボルを指定します。

```
Dim graphicInfo as aaGraphic.GraphicInfo;  
graphicInfo.Identity = "InTouch:FrameWindow01";  
graphicInfo.GraphicName = "Symbol_002";  
ShowGraphic( graphicInfo );
```

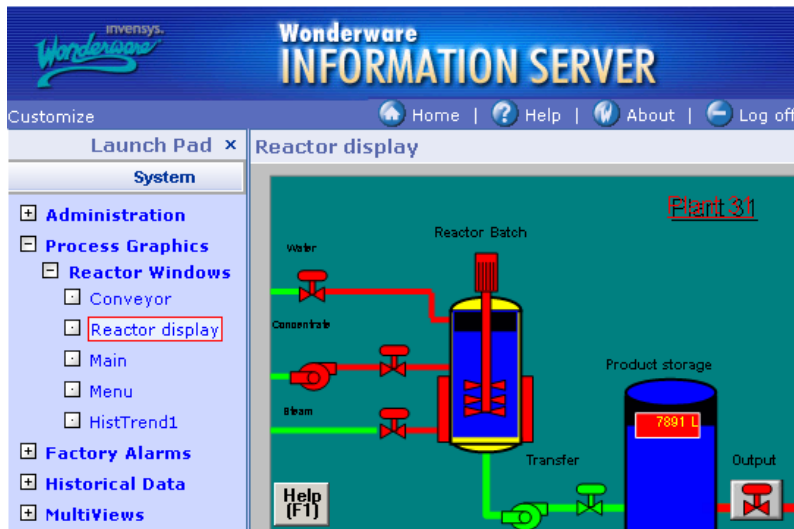
枠ウィンドウのデザイン時の設定に関係なく、「Symbol_002」に「FrameWindow01」がランタイム時に表示されます。ShowGraphic スクリプトを使用して、同じウィンドウで別のシンボルをホストできます。

制限事項:

グラフィックのキャッシュ処理は、ウィンドウが閉じてキャッシュされるときに枠ウィンドウに現在表示されているシンボルに対してのみ行われます。置換されたシンボルはキャッシュされません。

インターネット上での InTouch ウィンドウの実行

Information Server は、Web または社内のイントラネット上の工場での製造データを集計および表示することができる Web ポータル アプリケーションです。



以下の方法で Information Server で InTouch を使用できます。

- プロセスのビジュアル化

InTouch アプリケーションを Information Server ポータルにパブリッシュして、製造プロセスを表示することや、Web ブラウザを使用して制御することができます。

- データの操作

Information Server ポータルを使用して、InTouch タグ変数から値を読み取ったり、値を書き込むことができます。これにより、InTouch クライアントを使用しなくても工場のプロセスを操作できます。

- アラーム表示

Information Server ポータルを使用して、InTouch のリアルタイムデータおよび履歴アラームデータを表示できます。

- 履歴データ表示

Information Server ポータルを使用して、Historian データベースに保存された InTouch の履歴データを表示できます。

- Table Weaver 表示

InTouch を使用して、Table Weaver コンテンツ単位の表示を作成できます。

詳細については、Information Server のマニュアルを参照してください。

ランタイム時にウィンドウを表示する設定

「起動時」ウィンドウは、ユーザーがアイコンまたはメニュー コマンドから WindowViewer を直接起動したときに WindowViewer に表示されるウィンドウです。

WindowViewer を起動するために **【実行】** 高速切り替えを使用した場合、ホーム ウィンドウは表示されません。

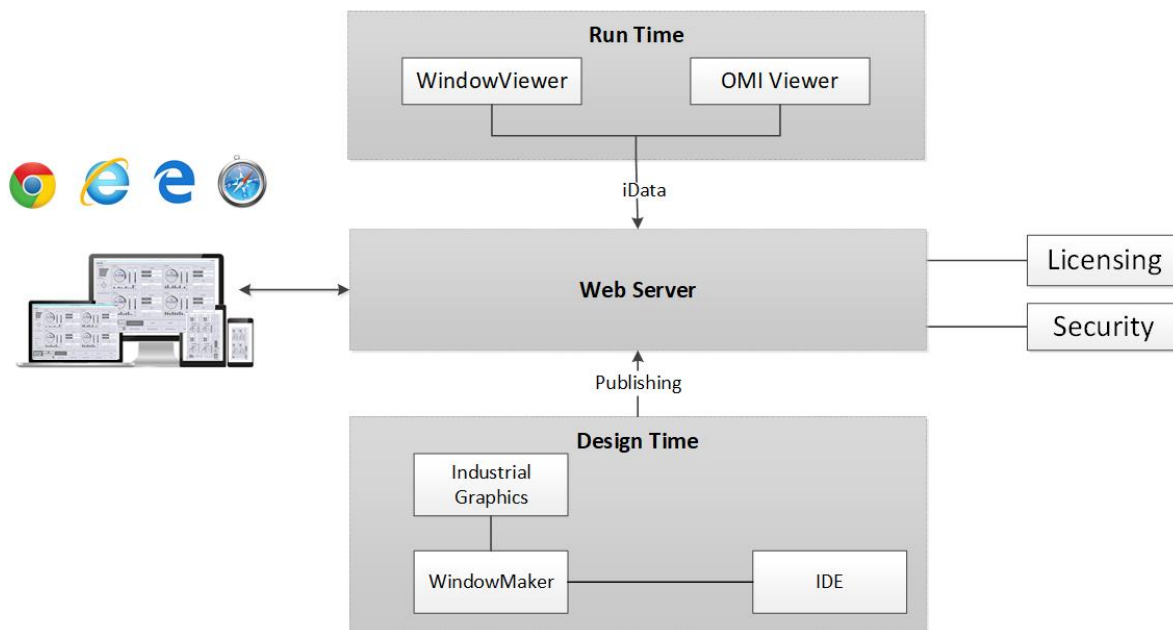
スクリプトで ShowHome() 関数を使用すると、ランタイム時にいつでもホーム ウィンドウを表示できます。

ホーム ウィンドウを設定するには

1. WindowMaker を開きます。
2. **【ファイル】** メニューで、**【設定】** をポイントし、**【WindowViewer】** をクリックします。WindowViewer の設定画面が表示されます。
3. **【起動】** タブをクリックします。
4. WindowViewer の起動時に開くウィンドウを選択します（複数可）。
5. **【OK】** をクリックします。

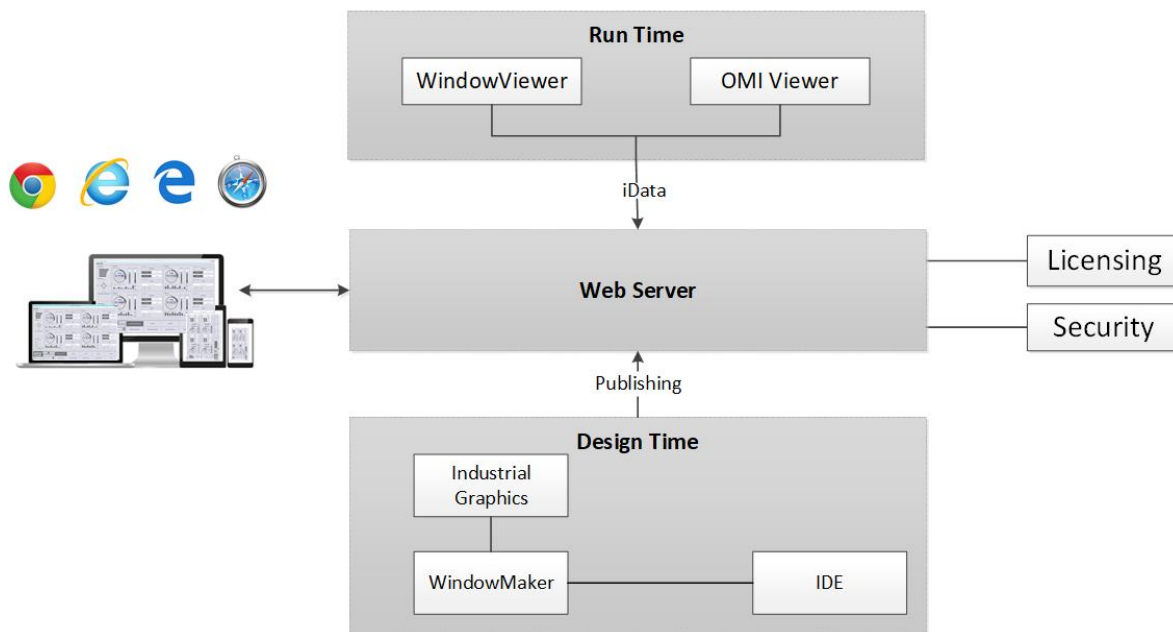
章 19 Web Client でのアプリケーション グラフィックの表示

Web Client では、HTML5 をサポートする任意の Web ブラウザで AVEVA InTouch HMI アプリケーションおよび AVEVA OMI ViewApp を表示できます。Web Client は、モバイルプラットフォームでも使用できます。組み込みの Web Server により、Microsoft Windows® Server のリモートデスクトッププロトコル (RDP) またはインターネットインフォメーションサービス (IIS) を使用することなく、任意の Microsoft Windows クライアントまたはサーバー オペレーティングシステム内の Web ブラウザからアプリケーションにアクセスできるようになります。



Web Client の使用

Web Client では、HTML5 をサポートする任意の Web ブラウザで AVEVA InTouch HMI アプリケーションおよび AVEVA OMI ViewApp を表示できます。Web Client は、モバイルプラットフォームでも使用できます。組み込みの Web Server により、Microsoft Windows® Server のリモートデスクトッププロトコル (RDP) またはインターネットインフォメーションサービス (IIS) を使用することなく、任意の Microsoft Windows クライアントまたはサーバー オペレーティングシステム内の Web ブラウザからアプリケーションにアクセスできるようになります。



InTouch Web Client アプリケーションの設計

InTouch Web Client アプリケーションの設計には以下のステージがあります。

設定（オプション）

1. セキュリティ プロトコルを設定します。
 - a. **System Platform** コンフィグレータを使用して **System Management Server** に接続します（リモート サーバーまたはローカル サーバーを選択します）。
2. AVEVA Identity Manager を設定します。
3. リバース プロキシ サーバーを設定します。

設計

1. アプリケーションを作成または編集します。
2. Web Client に表示するすべてのグラフィックを含むルート フォルダを識別します。
3. Web Client ルート フォルダを識別して設定します。
4. Web Client ホームのシンボルを識別して設定します。
5. InTouch WindowViewer への高速切り替えを行います。
 - InTouch タグにアクセスするには **WindowViewer** が実行している必要がありますが、アプリケーション サーバー オブジェクト属性参照にアクセスする場合は必要ありません。
6. Web Client の高速切り替えボタンを使用するか、ブラウザを起動して <http://localhost/InTouch> にアクセスします。

反復的開発

1. ホームのシンボルが Web ページに表示されます。
2. ナビゲーション オーバーレイを使用してシンボル/ウィンドウに移動します。

3. WindowMaker でグラフィック関連の変更を行うと、その変更はブラウザに自動的に反映されます。
クオリティとステータス スタイル、要素スタイル、および書式設定スタイルの変更は、グラフィックに対して以下の変更を行った場合または WindowViewer を再起動した後にのみ適用されます。
 - シンボルの内容の更新および保存
 - シンボルの作成、インポート、または削除
 - 別のツールセット フォルダへのシンボルの移動
 - ルート フォルダまたはホームのシンボルの割り当ての更新
4. アプリケーションの構築を続行して、出力をブラウザでテストできます。

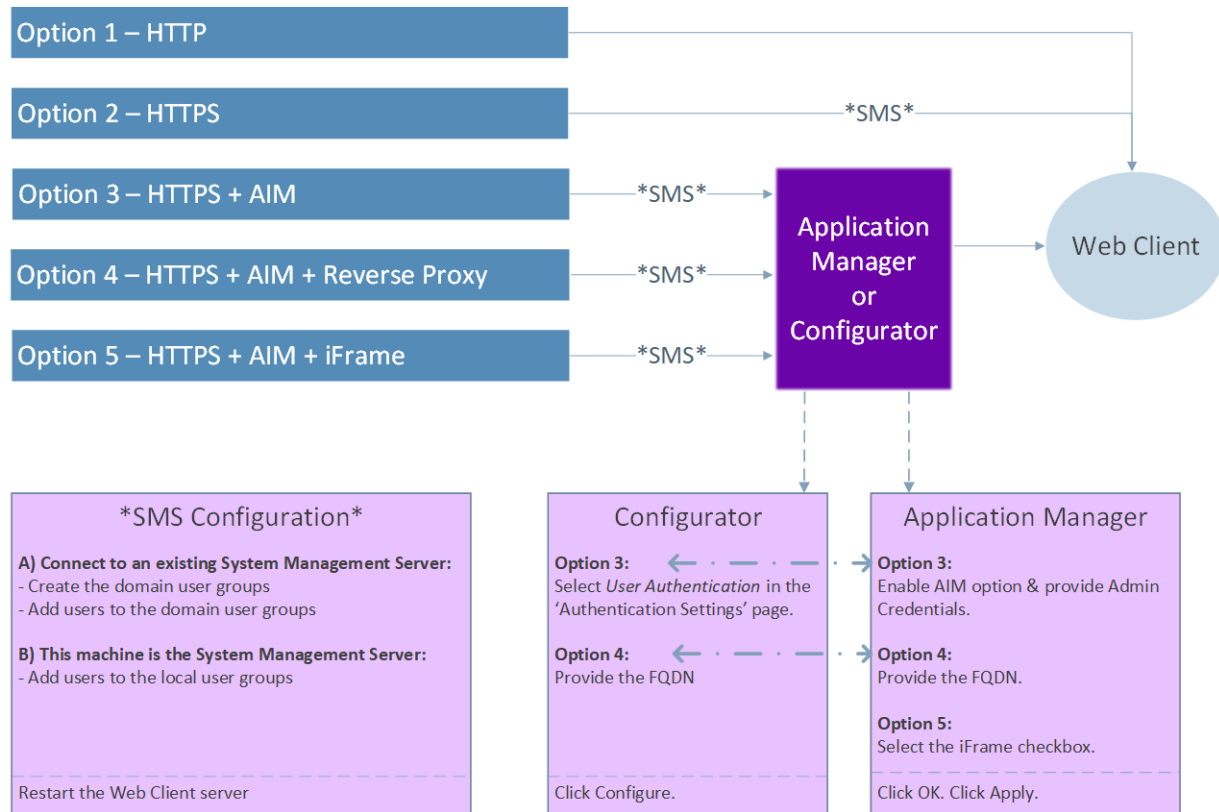
配置

1. ユーザーを Web Client 関連のユーザー グループ（ローカルまたはドメイン）に割り当てるか、匿名アクセスを設定します。
2. アプリケーションをパブリッシュまたは配置します。
3. アプリケーションを InTouch WindowViewer で実行します。InTouch タグからライブ データを受信するには、WindowViewer が実行している必要があります。
パブリッシュまたは配置したアプリケーションには、イントラネット上の任意のコンピュータからアクセスできます。

http://<IPAddress>/InTouch または http://<nodeName>/InTouch, をポイントできます。ここで <IPAddress> または <nodeName> はアプリケーションがパブリッシュまたは配置されたマシンを表します。

Web Client アクセスの保護

System Management Server (SMS) を設定すると、HTTPS プロトコルで Web Client を安全に使用できます。さらに、SMS を設定した後、Web Client を AVEVA Identity Manager (AIM) と共に使用して、Windows ベースでないセキュリティおよびシングルサインオンをサポートできます。管理者は、ユーザー認証およびセキュリティの複数の設定オプションを使用できます。セキュリティおよびユーザー管理は連動させる必要があります。詳細については、「[Web Client でのユーザー アクセスの設定](#)」を参照してください。



オプション 1: デフォルトでは、Web Client には http プロトコルおよび Windows ベースの認証を使用してアクセスできます。

オプション 2: コンフィグレータを使用してローカルまたはリモートの System Management Server に接続します。ここで、Web Client は https プロトコルと Windows ベースのセキュリティを使用します（デフォルト）。リモート サーバーに接続する場合、Web Client にアクセスするときにドメインユーザーの資格情報を入力する必要があります。

オプション 3: System Management Server を設定した後、AVEVA Identity Manager (AIM) を有効化できます（オプション）。AIM は、OpenID Connect エンドポイントを公開するスタンドアロン認証サーバーです。AIM を使用するには、クライアントを Identity Server に登録する必要があります。AIM は、System Management Server コンフィグレータまたは InTouch HMI アプリケーション マネージャを使用して設定できます。詳細については、「[AVEVA Identity Manager での登録](#)」を参照してください。

ランタイム時に Web Client ページがロードされると、有効なセキュリティ トークンがない場合、以下の処理が行われます。

- Web Client が AIM のログイン ページにリダイレクトされます。
- AIM が Active Directory のユーザー資格情報をチェックします。
- 資格情報が有効な場合、Active Directory からセキュリティ トークンが提供され、Web Client に返されます。
- Web Client がトークンのあるユーザーにアクセス権を付与します。

セキュリティ トークンが既に存在する場合、ユーザーにアクセス権が付与されます。AIM では、Active Directory で検証されたユーザーのみがサポートされます。

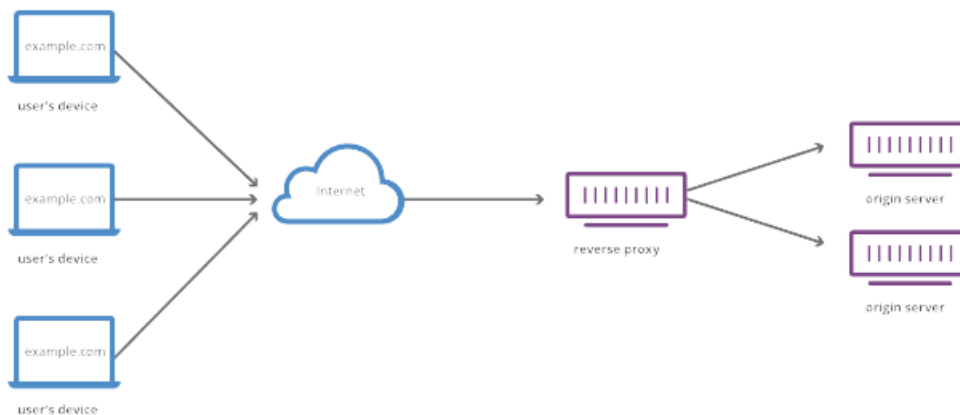
オプション 4: [リバース プロキシ] オプションを使用すると、OT ネットワーク外のユーザーが Web Client にアクセスできます。コンフィグレータのセキュア ゲートウェイ フィールドまたは InTouch HMI アプリケーションマネージャの [Web Client] タブにリバース プロキシ サーバーの FQDN を入力します。

オプション 5: ユーザー認証に AIM を使用する場合、グラフィックは iFrame に表示されません。[Web サイトへの Web Client の埋め込みを許可する] チェック ボックスをオンにすると、ランタイム時に iFrame 内にグラフィックを表示できます。ランタイム時に [共有] アイコンを使用して、iFrame に挿入するコード スニペットを選択します。

OT ネットワーク外からの Web Client へのアクセス

リバース プロキシは、クライアントからのリクエストを傍受するために Web Server の前に配置されているサーバーです。リバース プロキシを使用すると、クライアントから Web サイトのオリジン サーバーに送信されたリクエストはリバース プロキシ サーバーによってネットワーク エッジで傍受されます。その後、リバース プロキシ サーバーは、リクエストをオリジン サーバーに送信し、オリジン サーバーから応答を受け取ります。いずれかの Web Server がダウンしている場合、または Web Server のフェイルオーバーが行われている場合、リバース プロキシはクライアントからのリクエストを冗長 Web Server に送信します。

フォワード プロキシはオリジン サーバーがその特定のクライアントと直接通信することのないようにクライアントの前に配置されます。一方、リバース プロキシはオリジン サーバーの前に配置され、クライアントがオリジン サーバーと直接通信することを防止します。



OT ネットワーク外のデバイスで Web Client にアクセスするには、DMZ サーバー上でリバース プロキシを設定する必要があります。リバース プロキシが機能するには、AVEVA Identity Manager を有効にする必要があります。多くのプロバイダがリバース プロキシ ソリューションを提供しています。該当するリバース プロキシ ソリューションのマニュアル、冗長サポートの情報、およびインフラストラクチャでリバース プロキシ ソリューションを設定する手順を参照してください。

AVEVA Identity Manager での登録

AVEVA Identity Manager (AIM) を使用すると、デフォルトの Windows OS ベースの認証ではなく、シングルサインオンを使用するように Web Client を設定できます。

注記: Web Client は、AIM 設定を含む ArchestrA ベースのセキュリティのみをサポートします。

Identity Server を設定するには

1. System Platform コンフィグレータで [共通プラットフォーム] > [System Management Server] を設定します。

2. AVEVA アプリケーション マネージャでユーザー資格情報を使用して AIM サーバーを登録します。

[AIM 登録] ダイアログ ボックスを使用して、リバース プロキシ サーバーを設定することもできます。

1. リバース プロキシ サーバーをセットアップします。
2. System Platform コンフィグレータで **[Identity Server]** > **[System Management Server]** を設定します。
3. [AIM 登録] ダイアログ ボックスにセキュア ゲートウェイのアドレスを入力します。

リモートまたはローカルの System Management Server を選択できます。System Platform Server の設定の詳細については、『*System Platform インストールガイド*』を参照してください。

Identity Server に登録するには

1. アプリケーション マネージャで **[Web Client]** タブをクリックします。
2. [ツール]、[ID マネージャ] の順にクリックします。
[Identity Server 設定] 画面が表示されます。
3. [AIM サーバーを認証サーバーとして使用する] チェック ボックスをクリックして、AVEVA Identity Manager の使用を有効にします。
[Identity Server] フィールドにコンフィグレータで設定した Identity Server が表示されます。
4. 以下の設定を更新します。
 - a. ユーザー名: Identity Server に接続するユーザー名を入力します。ユーザーは Administrators ユーザー グループに属する必要があります。
 - b. パスワード: 入力したユーザー名のパスワードを入力します。
5. リバース プロキシのセットアップを完了するには、リバース プロキシまたは DMZ サーバーの URL を [セキュア ゲートウェイ] フィールドに入力します。
これはオプションの設定で、Web Client がリバース プロキシ サーバーの背後でホストされる場合にのみ設定する必要があります。
6. [Web サイトへの産業用グラフィックの埋め込みを許可する] チェック ボックスをクリックして、ランタイム時に HTML iframe 内に Web Client を表示することもできます (オプション)。
7. [OK] をクリックします。
8. [適用] をクリックします。

注記: Web サイトの名前またはアドレスが変更された場合、AVEVA Identity Manager を設定して新しい Web サイトを登録する必要があります。

シングルサインオンを使用した Web Client へのアクセス

コンフィグレータで [System Management Server] オプションが有効化されている場合、シングルサインオンを使用して Web Client にアクセスできます。これは、デフォルトの Windows ベースの認証とは異なります。この方法では HTTPS プロトコルが使用され、Microsoft Active Directory から検証できるドメイン ユーザーだけがサポートされます (リモート SMS を使用する場合)。HTTP プロトコルを使用する場合、Web Client は Windows ベースの認証のみを使用するようにフォールバックします。

System Management Server の設定の詳細については、『*AVEVA System Platform インストールガイド*』を参照してください。

1. Web Client を起動します。

ログイン ページが表示されます。

2. ユーザー名およびパスワードを入力します。
3. [ログイン] をクリックします。

[Windows 統合] をクリックすると、現在ログインしているユーザーの資格情報を使用できます（オプション）。

ユーザーは、Identity Manager の資格情報でログに記録されます。

4. セッションをログアウトするには、[サインアウト] を使用します。

AIM でのローカル タグの使用

Web Client のローカル タグの動作は、WindowViewer に依存しません。AIM が有効化されている場合、ローカル タグの動作は異なります。AIM が有効化されている場合、各ユーザーの各セッションで一意的なトークンが使用されます。

	ユーザー 1 – セッション 1	ユーザー 1 – セッション 2	ユーザー 2 – セッション 1
AIM が無効化されている場合	同じタグ値	同じタグ値	別のタグ値
AIM が有効化されている場合	別のタグ値	別のタグ値	別のタグ値

Web Client でのユーザー アクセスの設定

匿名での Web Client へのアクセス

Web Client では、匿名ユーザー アクセスおよび認証ユーザー アクセスの両方を使用できます。デフォルトでは、匿名アクセスは無効化されています。このオプションを有効にするには、InTouch HMI アプリケーションマネージャの [Web Client] タブの [匿名アクセスを許可] チェック ボックスをオンにします。ランタイム時にユーザー資格情報を入力する必要はありません。ユーザーは「ゲスト」ユーザーとしてログインされます。ゲスト ユーザーは読み取り専用で Web Client にアクセスできます。

Web Client の認証ユーザーの設定

認証およびライセンスの両方のワークフローでは、ユーザー グループを使用してアクセス権を付与するユーザーおよび Web セッション用に取得するライセンスのタイプを決定します。ユーザー グループは、SMS の設定に応じてローカル ノードまたはリモート ノードで設定できます。アクセスが必要なすべてのユーザーは、アクセス レベルに応じていずれかのグループに含める必要があります。

リモート System Management Server を使用するように Web Client が設定されている場合、リモートサーバーに aalInTouchUsers および aalInTouchRWUsers というドメインユーザー グループを作成します。アプリケーションを配置する前に、これらのグループに該当するすべてのユーザーを追加します。

InTouch Web Client の仮想アカウントの使用

Web Client 仮想サービス アカウントの特権は、セキュリティを向上させる目的で削減されています。InTouch アプリケーションに対して、Web Client を継続して使用するには、InTouch Web Client 仮想サービ

ス アカウントに InTouch アプリケーション フォルダへの読み取り/書き込みアクセス権限が必要です。InTouch Web サービスの仮想サービス アカウントは、"NT SERVICE\AIGWebServer" と呼ばれます。

InTouch Web Client の仮想サービス アカウントには、次のアプリケーションに対する読み取り/書き込みアクセス権限が付与されます。

- 既存のすべての InTouch アプリケーション
- アプリケーション マネージャから作成/インポートされたすべての InTouch アプリケーション
- [検索] 機能で見つかったすべての InTouch アプリケーション（見つかったアプリケーションに対する読み取り/書き込みアクセス権限を Web Client 仮想アカウントに付与するかどうかを確認するダイアログで [はい] を選択する必要があります）

InTouch アプリケーションへのアクセス権限を InTouch Web Client 仮想アカウントに手動で付与するには

1. ファイルエクスプローラを使用して InTouch アプリケーション フォルダに移動します。
2. フォルダを右クリックして [プロパティ] を選択します。
3. [プロパティ] ダイアログの [セキュリティ] タブで [編集] をクリックします。
4. [グループ名またはユーザー名] セクションで [追加] をクリックします。
[ユーザー、コンピュータ、サービス アカウント、またはグループの選択] ダイアログが表示されます。
5. [場所] を選択します。
[場所] ダイアログが表示されます。
6. コンピュータ名を選択し、[OK] をクリックします。
7. [選択するオブジェクト名を入力してください] フィールドに「NT SERVICE\AIGWebServer」と入力します。
8. [OK] をクリックします。
9. [認証されたユーザーの権限] セクションの [許可] カラムの次の 4 つのチェック ボックスをオンにします。[読み取りと実行]、[フォルダの内容の一覧表示]、[読み取り]、および [書き込み]。
10. [OK] をクリックします。

注記: アプリケーションのパスが "\\" で始まる場合、ファイル共有のパスを示します。InTouch Web Client 仮想サービス アカウントには、共有場所にあるアプリケーション フォルダへのアクセス権限は付与されません。

Web Client 機能の有効化

デフォルトでは、インストール時に Web Client 機能は無効化されます。Web Client 機能を使用するには、この機能を有効にする必要があります。コンフィグレータで以下のオプションのいずれかを設定すると、Web Client の機能は自動的に有効化されます。

- 共通プラットフォーム > System Management Server
- アプリケーション マネージャ > Web Client

コンフィグレータの詳細については、『System Platform インストールガイド』を参照してください。Web Client の機能を手動で有効化または無効化するには、管理特権が必要です。Web Client の機能は、アプリケーション マネージャまたは InTouch WindowMaker から有効化できます。

アプリケーションマネージャからの Web Client 機能の有効化

1. アプリケーションマネージャを起動します。
2. [Web Client] タブを選択します。
3. [ファイル] メニューの [メイン] グループで [Web Client] をクリックします。
4. 無効化するには、[Web Client] を再度クリックします。

WindowMaker からの Web Client 機能の有効化

管理特権のあるユーザーは、WindowMaker から Web Client 機能を有効にすることができます。

1. WindowMaker でアプリケーションを起動します。
2. [Web Client] 高速切り替えボタンをクリックします。

ここで、「このシステムでは、InTouch Web Client の機能は現在、無効化されています。この機能を有効にしますか?」というメッセージを含むダイアログ ボックスが表示されます。

3. [はい] をクリックします。

InTouch Web Client ページが表示されます。

無効化されている Web Client を起動しようとする、「HTTP エラー 404.0 – 見つかりません」というエラー メッセージが表示されます。

管理者以外のユーザーが WindowMaker から Web Client 機能を有効にしようとする、アプリケーションマネージャから機能を有効にするようリダイレクトされます。

Web Client のカスタマイズ

AVEVA アプリケーション マネージャの [Web Client] タブには、Web Client 関連の設定を構成するオプションがあります。設定は InTouch HMI と OMI アプリケーションの両方に適用できます。

Web Client 設定の更新:

1. アプリケーションマネージャを起動し、[Web Client] タブをクリックします。
[Web Client の設定] 画面が表示されます。
 2. 以下の設定を構成します。
 - **現在のアプリケーション: Web Client** の設定を変更するアプリケーションをリストから選択します。
 - **グラフィックのリフレッシュ レート (ms)** : Web ブラウザが Web Server にグラフィック データをクエリーする頻度を設定します。デフォルトは 1 秒です。詳細については、『System Platform インストールガイド』を参照してください。
 - **アラームのリフレッシュ レート (ms)** : Web ブラウザが Web Server にアラーム データをクエリーする頻度を設定します。デフォルトは 1 秒です。詳細については、『System Platform インストールガイド』を参照してください。
 - **ヘッダーを表示**: チェック ボックスをオンにするとタイトル バーが表示されます。
 - **ナビゲーションバーを有効化**: チェック ボックスをオンにするとナビゲーションバーが表示されます。
- [ヘッダーを表示] 設定が選択されていない場合、この設定は無効化されます。

- **匿名アクセスを許可:** ユーザーが認証なしで **Web Client** にアクセスすることを許可する場合はチェック ボックスをオンにします。
 - **作業領域を有効化:** 作業領域を有効にするには、このチェック ボックスをオンにします。
3. [詳細設定] セクションで **[カスタマイズ]** をクリックして以下の設定を構成します。
- **Web サイト名:** 標準の URL を置き換える文字列を入力します。
 - **アプリケーション タイトル:** アプリ バーのアプリケーションのタイトルとして表示する文字列を入力します。
 - **Web サイト タイトル:** ブラウザのタイトルバーのタイトルとして表示する文字列を入力します。
 - **Web サイト アイコン:** ブラウザのタイトルバーのアイコンを置換する画像ファイルを入力します。
 - i. **[ファイルを選択...]** をクリックしてアイコン画像を選択します。
 - ii. ファイルを選択します。
 - iii. **[開く]** をクリックします。
4. **[保存]** をクリックして詳細設定を保存します。
5. 設定を変更したら **[適用]** をクリックします。
6. **Web Client** を表示するには、**[起動]** をクリックします。

注記: Web サイトの名前またはアドレスが変更された場合、**AVEVA Identity Manager** を設定して新しい Web サイトを登録してください。詳細については、「[AVEVA Identity Manager での登録](#)、[AVEVA Identity Manager での登録](#)」を参照してください。

これらの設定の詳細については、「[Web ブラウザでのアプリケーション グラフィックの表示](#)」を参照してください。

Web ブラウザで表示するためのグラフィックの設定

グラフィック ツールボックスを使用して、**Web ブラウザ**に表示されるアプリケーション グラフィックを含むフォルダを設定できます。**Web ブラウザ**に表示されるのは、**Web Client** のルート フォルダとして設定したフォルダ内に格納されているグラフィックだけです。



注: インストール時に **aaInTouchUsers** および **aaInTouchRWUsers** グループが InTouch ランタイム ノードに作成されます。**Web ブラウザ**でアプリケーションのグラフィックにアクセスできるのは、**aaInTouchUsers** または **aaInTouchRWUsers** ユーザー グループに含まれるユーザーだけです。アプリケーションを構成する前に、関連するユーザーをグループに追加します。デフォルトでは、インストールを行ったユーザーがグループに追加されます。

1. InTouch WindowMaker を起動します。
2. アプリケーションを開きます。

産業用グラフィック ツールボックスで、**Web ブラウザ**に表示するフォルダの階層およびグラフィックを含めるフォルダを作成または選択します。
3. フォルダを右クリックして、**[Web Client ルート フォルダの設定]** を選択します。

アイコンのサムネイルが変更され、設定が反映されます。

アイコン	説明
------	----

	ルート フォルダが Web Client のルート フォルダとして設定された場合
	ルート フォルダ以外のフォルダが Web Client のルート フォルダとして設定された場合

デフォルトでは、グラフィック ツールボックスのルートが **Web Client** のルート フォルダとして設定されます。

Web Client ホームのシンボルの設定

Web Client ホームのシンボルを設定して、起動時に **Web Client** ページに表示するデフォルトのシンボルを設定できます。

1. **Web Client** のルート フォルダを設定します。「[Web ブラウザで表示するための InTouch アプリケーション グラフィックの設定](#)」を参照してください。

ホームのシンボルは、**Web Client** ルート フォルダの子である必要があります。

2. **Web Client** ルート フォルダの下でホームのシンボルを見つけます。
3. シンボルを右クリックして、**[Web Client ホームのシンボルの作成]** をクリックします。

アイコンのサムネイル (📁) が変更され、設定が反映されます。ホームのシンボルが設定されていない場合、起動時の **Web Client** ページには空白のページが表示されます。ブラウザで **Web Client** を表示しているときにアプリ バーの **[ホーム]** アイコンをクリックしてホームのシンボルに移動します。

Web Client ホーム アイコンの動作の理解

スタンドアロンアプリケーションは **Web Client** 上で表示できます。スタンドアロンアプリケーション ウィンドウを **Web Client** で表示するには、産業用グラフィックに変換する必要があります。**Web Client** で表示できるのは、枠ウィンドウと産業用グラフィックだけです。

InTouch アプリケーションの場合、**WindowMaker** で **[ホーム ウィンドウ]** タブを使用して、**WindowViewer** でデフォルトで表示されるウィンドウを設定できます。詳細については、「[ランタイム時にウィンドウを表示する設定](#)」を参照してください。同様に、**Web Client** を起動したときに表示されるデフォルト シンボルとしてグラフィックを設定できます。「[Web Client ホームのシンボルの設定](#)」を参照してください。

InTouch ShowHome スクリプト関数を使用した場合、ウィンドウをグラフィックに変換すると、スクリプトは **ShowHome()** 関数に変換され、対応するグラフィックがホーム シンボルとして **WindowViewer** に表示されます。

これらの設定は、**Web Client** の起動時にデフォルトで表示されるウィンドウまたはグラフィックに影響します。InTouch HMI アプリケーションの場合、2 つの可能性あります。

1. ホーム ウィンドウが設定されている場合

ユーザーがホーム シンボルを設定している場合、すべての状況で、デフォルトでホーム ウィンドウだけが表示されます。**[ホーム]** アイコンをクリックすると、ホーム ウィンドウが表示されます。

2. ホーム ウィンドウが設定されていない場合

ホーム シンボルが設定されている場合、デフォルトでホーム シンボルが表示されます。

ホーム シンボルが設定されていない場合、デフォルトではシンボルやホーム ウィンドウは表示されません。ホーム シンボルをクリックしてもグラフィックやウィンドウは表示されません。

Web ブラウザでのアプリケーションの表示

Web Client アプリケーションを設定した後、Web ブラウザでアプリケーションを表示できます。テスト済みのサポートされるブラウザのリストについては、「[ブラウザとモバイルのサポート](#)」セクションを参照してください。

1. HTML5 をサポートするブラウザを起動して、URL (<http://<hostname>/InTouch> または <http://<IPAddress>/InTouch>) を入力します。

資格情報が認証され、有効な Web Server ライセンスが取得されると、Web Client ページが表示されます。デフォルトで、ホームのシンボルが表示されます。

<https://<hostname>/InTouch> または <https://<IPAddress>/InTouch> の URL を使用して Web Client にアクセスできます。サーバーとクライアント間の暗号化された通信用に共通プラットフォームの System Management Server が証明書 (ローカルまたはリモート サーバー) で設定されている場合、Web Client では HTTPS プロトコルが自動的に使用されます。Web Client ページが表示されない場合は、「Web ブラウザでのグラフィックの表示に関するトラブルシューティング」を参照してください。

Web Client の URL のカスタマイズについては、「[Web Client のカスタマイズ](#)」を参照してください。

2. ハンバーガー アイコンをクリックします。



左側のオーバーレイで、実行中のアプリケーション内の選択した Web Client のルート フォルダの下にグラフィックの階層構造が表示されます。

3. フォルダ名をクリックします。

フォルダ内のグラフィックが表示されます。

4. グラフィックをクリックします。

選択したグラフィックが InTouch Web Client に表示されます。

Web ブラウザに表示されるのは、Web Client のルート フォルダとして設定したフォルダ内に格納されているグラフィックだけです。匿名アクセスが無効な場合、グラフィックにアクセスするユーザー アカウントは認証および承認が必要です。詳細については、「[ライセンスの取得](#)」を参照してください。

注記: 向上したグラフィック レンダリングと全体的なパフォーマンス改善を活用するために、GPU ハードウェア アクセラレーションを有効にすることが推奨されます。

Web Client 内のシンボルで InTouch タグからのライブ データを受信するには、読み取り/書き込み InTouch ライセンスで InTouch WindowViewer がホスト マシン上で実行している必要があります。WindowViewer をサービスとして実行することもできます。アプリケーション オブジェクトからライブ データを受信するには Application Server が実行している必要があります。

注記: 読み取り専用 InTouch ライセンスはリモート データ接続をサポートしないので、InTouch Web Client のデータ ソースとして機能するための十分な機能を提供しません。InTouch Web Client にデータをフィードするすべての WindowViewer アプリケーションは、読み取り/書き込み InTouch ライセンスで実行する必要があります。

Web Client の高速切り替え

アプリケーションの開発時に Web Client の高速切り替え機能を使用して、WindowMaker と Web Client を切り替えることができます。

- Web Client の高速切り替え機能を使用するには、InTouch WindowMaker の **[Web Client]** ボタンをクリックします。

同じアプリケーションを WindowMaker で編集し、Web Client で表示している場合、WindowMaker で行った変更が Web Client に動的に反映されます。自動更新が機能するのは、一度に 1 つのアプリケーションを編集および表示している場合だけです。InTouch Web ページは以前に表示されていたシンボルに更新されます。別のアプリケーションに対して WindowViewer を実行している場合、Web Client ページは、そのアプリケーションに自動的に切り替わります。

- **[グラフィックのリフレッシュ レート]** は、コンフィグレータから指定できます。デフォルトでは、ポーリング レートは 1000 ミリ秒です。
- **[アラームのリフレッシュ レート]** は、コンフィグレータから指定できます。デフォルトでは、ポーリング レートは 1000 ミリ秒です。

アラーム リフレッシュ レートにはグラフィック リフレッシュ レートよりも高い値を設定してください。また、グラフィック リフレッシュ レートの値は、180 秒未満に設定してください。

注記: リモート IDE からマネージド InTouch アプリケーションを編集する際に Web Client を高速切り替えモードで実行するには、GR マシンとリモート IDE マシンの両方の **aaConfigTools** ユーザー グループのメンバーであるユーザー アカウントで **InTouch Web Windows** サービスを実行するように設定します。詳細については、『Application Server ユーザー ガイド』を参照してください。

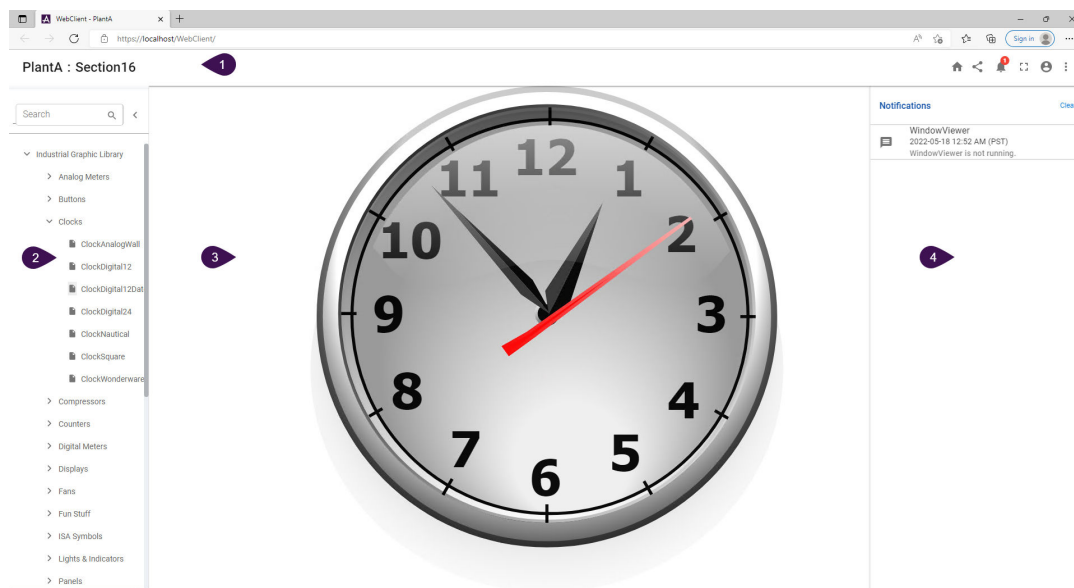
InTouch アプリケーションに対して、WindowMaker の Web Client の高速切り替えボタンはデフォルトで無効化されています。アプリケーションを WindowViewer で実行すると、このボタンは自動的に有効になります。


注記: Web Client の高速切り替え機能は、次の場合に使用できます。








- 開発時のみ
 - InTouch WindowMaker および InTouch WindowViewer がインストールされているコンピュータ上のみ
 - URL で <http://localhost/InTouch> をポイントして Web Client にアクセスする Web ブラウザ上のみ
-


Web Client ページの理解

Web Client ページには、ページ上のアイコンを使用してユーザー エクスペリエンスに関する情報の提供、およびユーザー エクスペリエンスの編成を行うことのできるさまざまなオプションがあります。これらのオプションの説明を次の表に示します。



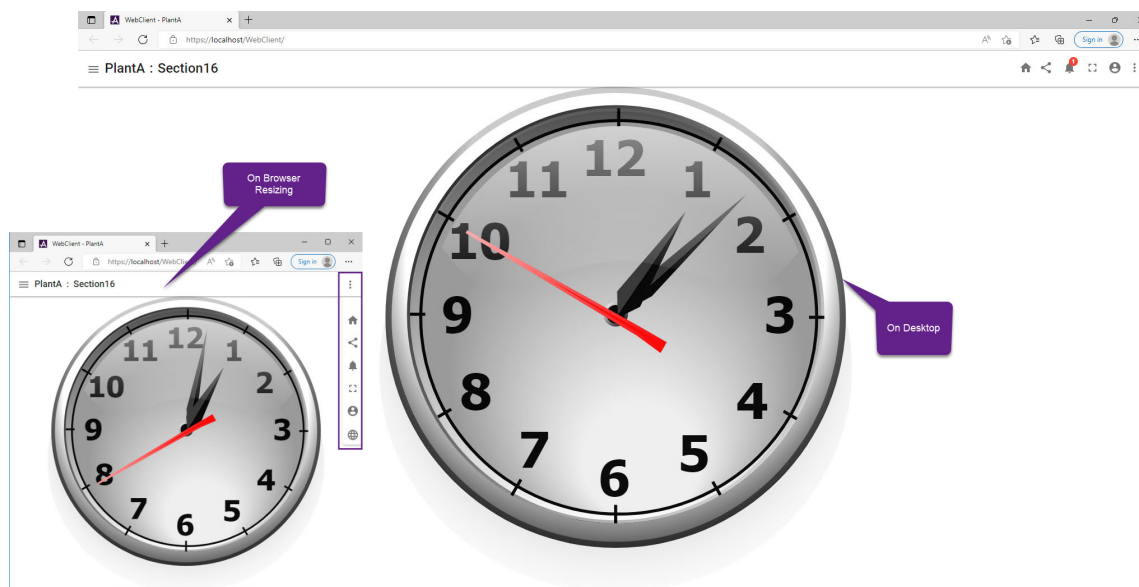
要素	説明
1	アプリ バーには、InTouch アプリケーションの名前、シンボル名、ホームアイコン、通知アイコン、全画面アイコン、およびプロフィールアイコンが表示されます。
アプリケーション名: シンボル名	<p>アプリ バーにはアプリケーションとシンボルの名前が表示されます。</p> <p>例 –<i>PlantA: Section16</i></p> <p>- この例では、PlantA がアプリケーションの名前で、Section16 がシンボルの名前です。</p>
	クリックすると、 Web Client ホームのシンボルとして設定されているシンボルが表示されます。
	選択したグラフィックをクリックすると、iFrame コードスニペットが表示されます。
	ライセンスのステータスに関連する通知アイコン メッセージをクリックすると、 WindowViewer のステータスの変化に関するメッセージが表示されます。通知の数がアイコンの上に表示されます。

	<p>クリックするとキャンバス領域が最大化され、Web ページが全画面モードで表示されます。F11 キーを押しても同じ操作を行うことができます。全画面モードではアプリ バーとナビゲーション オーバーレイは表示されません。</p> <p>タイトルバーを表示するには</p> <ul style="list-style-type: none"> カーソルをページの上部に移動するとアプリ バーが下にスライドして表示されます。 下向き矢印をクリックすると、アプリ バーがページにピン留めされます。 上向き矢印をクリックすると、アプリ バーが非表示になります。
	<p>クリックするとキャンバス領域が最小化され、全画面モードが終了します。</p>
	<p>プロフィールアイコンには、ログインしているユーザーが表示されます。</p>
 Sign Out	<p>プロフィールアイコンをクリックして、[サインアウト] アイコンをクリックします。[サインアウト] は、System Management Server オプションが設定されている場合にのみ使用可能です。詳細については、「シグナルサインオンを使用した Web Client へのアクセス」を参照してください。</p>
	<p>アプリ バーで  をクリックして、[言語] アイコンをクリックします。言語を選択します。選択した言語でツールヒント、翻訳された静的テキスト、およびカスタム プロパティが表示されます。表示される言語は WindowMaker で設定されている言語だけです。</p> <p>注記: 言語を変更すると、新しいセッションが作成され、ローカル タグのデータは失われます。</p>
<p>2</p>	<p>フォルダナビゲーション オーバーレイを表示するには、 アイコンをクリックします。選択した Web Client ルート フォルダの下にあるすべてのフォルダとシンボルが表示されます。</p> <p>オーバーレイ上部の検索ボックスを使用して、シンボルを検索できます。検索ボックスではオートコンプリート機能が有効になり、検索用語を入力すると入力した検索用語に一致するシンボル名が一覧表示されます。シンボル名を選択すると、シンボルがキャンバス領域にシンボルが表示されます。</p>
<p>3</p>	<p>キャンバス領域には、ナビゲーション オーバーレイで選択したシンボルが表示されます。シンボルのサイズは、アスペクト比を維持した状態でブラウザのビューポート サイズに合わせて調整されます。ナビゲーションおよび通知オーバーレイはキャンバスのスペースを使用しません。また、グラフィックのサイズにも影響しません。</p>

4	通知アイコンをクリックして。接続とライセンスの問題に関する通知を表示します。
	警告アイコンをクリックすると、InTouch Web Client でサポートされないアニメーションまたはプロパティのリストが表示されます。このアイコンは、アプリケーション開発時およびブラウザが http://localhost/InTouch という URL をポイントしている場合にのみ使用できます。

アプリケーション グラフィックおよびブラウザのサイズ調整

Web Client ページは、さまざまな画面サイズで表示できます。ブラウザを調整すると、新しいブラウザのビューポート寸法に合わせてグラフィックのサイズが変更されます。グラフィックのサイズは、モバイルデバイスを縦向きまたは横向きに回転させたときにも変更されます。

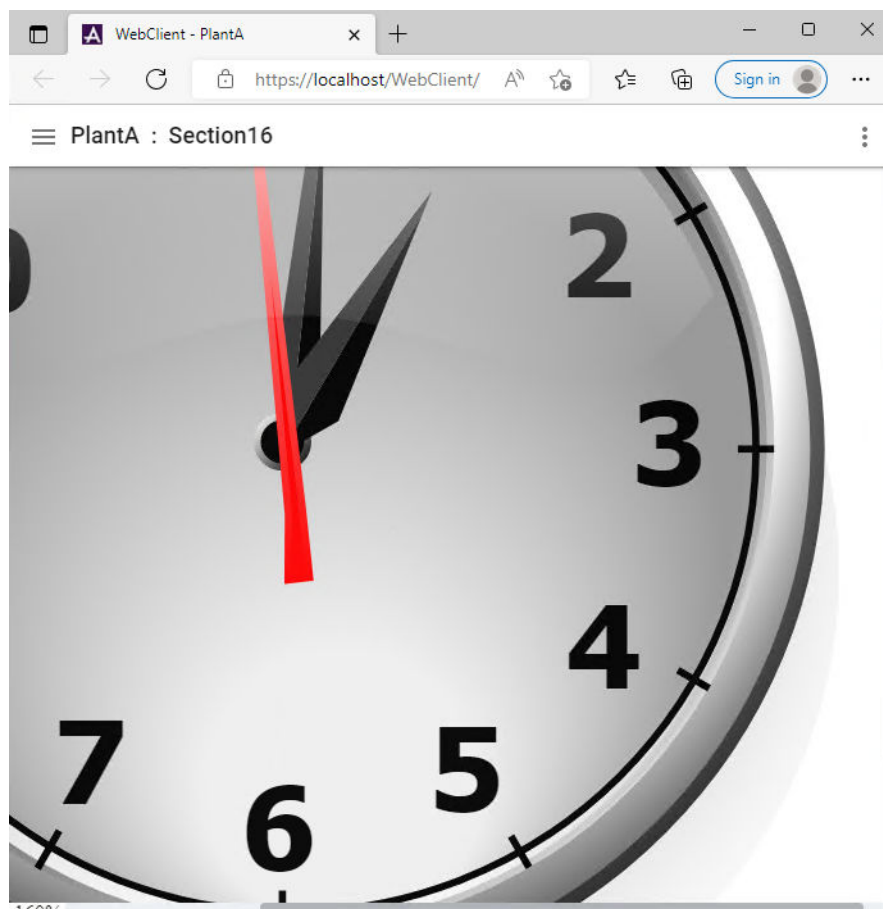


アプリ バーと小さい画面

画面が非常に小さい場合、右側のアイコンは、垂直バーに格納されます。画面のスペースを節約するためにアイコンは非表示になります。産業用グラフィックは任意のモバイルデバイスで表示して、パンとズームのジェスチャを使用してグラフィックを拡大表示することが可能です。

パンとズームのサポート

Web Client は、サポートされるすべてのブラウザおよびデバイスでのパンとズームのジェスチャをサポートします。モバイルデバイス以外のデバイスで Web ブラウザに表示されているグラフィックをズームインすると、水平スクロールバーの左下にズーム率が表示されます。ズーム率は最大 500% に制限されています。



タッチ操作によるパンとズーム

タッチ対応ディスプレイでは、2本の指を使用してズームインおよびズームアウトを行うことができます。

- ズームインするには、2本の指を画面に置いて指の間の幅を広げます。ズームアウトするには、2本の指の間を狭くします。
- ダブルタップすると 100% ズームの表示に戻ります。

表示をパンするには、1本の指で画面を押したまま指を移動します。

マウス操作によるパンとズーム

マウスを使用してズームインおよびズームアウトを行うことができます。

- ズームインまたはズームアウトするには、**Ctrl** キーを押したままホイールを上または下にスクロールします。マウスポインタがペインの外にある場合、ズームは行われません。
- マウスの左ボタンをダブルクリックすると、100% ズームの表示に戻ります。

ディスプレイをパンするには、マウスホイールを押した状態でマウスを目的の領域に移動してホイールを離します。

キーボード操作によるズーム

キーボードを使用してズームインおよびズームアウトを行うことができます。

ズームインするには、**Ctrl** キーと **+** キーを同時に押します。ズームアウトするには、**Ctrl** キーと **-** キーを同時に押します。

場所によるズーム

クリックした場所からグラフィックをズームインすることもできます。

パンおよびズーム ツールバーはサポートされていません。パンおよびズームのジェスチャは、**ShowSymbol** アニメーションまたは **ShowGraphic** スクリプト、ボタン、クリック、およびスライダーを使用して表示されるグラフィックに対してサポートされません。

外部 Web サイトでの InTouch アプリケーション グラフィックのホスティング

InTouch WindowMaker でグラフィックを設定し、グラフィックへのダイレクト URL を使用して外部 Web サイトでグラフィックをホストすることができます。

1. 産業用グラフィック ツールボックスを使用してシンボルを作成します。
2. Web Client でシンボルをテストします。
3. 外部 Web サイトで HTML **<iframe>** タグを作成します。
4. URL 属性には **http://<hostname>/InTouch/?iframe=true&symbol=<graphicname>** または **http://<IPAddress>/InTouch/?iframe=true&symbol=<graphicname>** へのリンクを含めます。ここで、**<hostname>** または **<IPAddress>** は、アプリケーションがパブリッシュまたは配置されているマシンを表し、**<graphicname>** は表示するグラフィックの名前を表します。
5. 英語以外の言語でシンボルを表示する場合、URL の形式は異なります。たとえば、フランス語の URL は以下ようになります。
http://<hostname>/InTouch/?localeid=1036&iframe=true&symbol=<graphicname>
ランタイム時の言語変更が可能な新しいバージョンの Web Client では URL の形式が変更されます。最新バージョンの Web Client にアップグレードすると、以前の URL 形式では英語以外の言語向けに設定されたシンボルは表示されません。
6. アプリケーションをパブリッシュまたは配置します。
7. アプリケーションを InTouch WindowViewer で実行します。InTouch タグからライブ データを受信するには、WindowViewer が実行している必要があります。

構文:

```
<iframe src="http://localhost/InTouch/?iframe=true&symbol=YourGraphicName"></iframe>  
または  
<iframe src="http://NodeName/InTouch/?iframe=true&symbol=YourGraphicName"></iframe>
```

例: ローカル マシンの Web Client アプリケーションに格納されているシンボル **ClockAnalogWall** を例に説明します。

```
<iframe src="http://localhost/InTouch/?iframe=true&symbol=ClockAnalogWall"></iframe>  
<iframe src="http://localhost/InTouch/?localeid=1036&iframe=true&symbol=ClockAnalogWall"></iframe>
```

URL を使用してシンボルに直接アクセスする場合、同じ InTouch Web Client のセキュリティとライセンスが適用されます。InTouch Web Client のエラー メッセージと同様に、該当するエラー メッセージが **<iframe>** セクションに返されます。

注記: Google Chrome 80 以降のバージョンでのポリシーの変更により、**http** プロトコルを使用する場合、**<iframe>** タグ内に配置されたグラフィックは表示されません。**http** プロトコルを使用してグラフィックを表示するには、**chrome://flags** にアクセスして以下の cookie 設定を無効にします。

- SameSite by default cookies
- Enable removing SameSite=None cookies
- Cookies without SameSite must be secure

詳細については、<https://www.chromestatus.com/feature/5633521622188032> を参照してください。セキュリティの問題を防止するために、Web Client では https プロトコルを使用することが推奨されます。

iFrame コードブロックの生成

シンボルの <iframe> ブロックを Web ブラウザから直接生成できます。その後、このコードブロックを外部の Web サイトで使用して、ランタイム時に産業用グラフィックを表示できます。

1. グラフィックに移動します。

2. [共有] アイコン  をクリックします。

3. [所有オブジェクト] フィールドで、このグラフィックの所有オブジェクトを指定します。

4. サポートされるカスタムプロパティのリストが表示されます。カスタムプロパティを選択して値を指定します。

指定したすべての値は、データ型に対して検証されます。たとえば、カスタムプロパティで整数値が要求される場合、文字値を入力することはできません。

5. Web Client (ローカルまたはリモートマシン) 上のスクリプトを使用する場合、[クロスオリジンをサポート] をクリックします。

埋め込まれた HTML コードは、選択したオプションに基づいて変更されます。

6. [コピー] をクリックします。

7. 生成されたコードを Web サイトの .html ファイルに貼り付けることができます。

```
<iFrame src="http://<hostname>/InTouch/?iframe=true&symbol=Sym1?crossOrigin=1"
id="symbol" width="640px" height="480px" customProperty='[{"n":"CP1","v":"True","t":1},
{"n":"CP2","v":"24.5","t":6},{"n":"CP3","v":"Orange","t":7}]' onload="postMsg()"></iFrame>
<script type="text/javascript">
function postMsg(){
  var iframe=document.getElementById('symbol');
  var obj={};
  for (var i=0;i<iframe.attributes.length;i++) {
    obj[iframe.attributes[i].nodeName]=iframe.attributes[i].nodeValue;
  }
  var win=iframe.contentWindow;
  win.postMessage(obj,obj['src']);
};
</script>
```

言語がフランス語 (France) の場合

```
<iFrame src="http://<hostname>/InTouch/?localeid=1036&iframe=true&symbol=Sym1"
id="symbol" width="640px" height="480px" onload="postMsg()"></iFrame>
<script type="text/javascript">
function postMsg(){
  var iframe=document.getElementById('symbol');
  var obj={};
  for (var i=0;i<iframe.attributes.length;i++) {
```

```
obj[iframe.attributes[i].nodeName]=iframe.attributes[i].nodeValue;  
}  
var win=iframe.contentWindow;  
win.postMessage(obj,'*');  
};  
</script>
```

サポートされているグラフィック要素と既知の制限事項

このセクションでは、サポートされているグラフィック要素と既知の問題について説明します。

既知の制限事項

Web Client は、製品リリースごとに強化される予定です。以下のリストに現在のリリースの主な制限事項を示します。

- **Web Server** とのデータ通信では、RDS セッションで実行している **InTouch WindowViewer** はサポートされません。
- **System Platform 2017** 以前のバージョンからパブリッシュされたアプリケーションは **Web Client** でサポートされません。 **Web Client** で表示するには、元のアプリケーションを現在のバージョンに移行して、アプリケーションを再度パブリッシュする必要があります。
- ウィンドウ名に特殊文字（サポートされていない文字）を含む枠ウィンドウはサポートされません。
- 間接型タグはサポートされていません。
- **PlaySound()** スクリプト関数はサポートされません。
- **InTouch HMI** マネージドアプリケーションでは、**Galaxy** ベースのセキュリティはサポートされません。
- システム タグは、ユーザー名および各ユーザーの関連詳細を返しません。
- スクリプトを使用したグラフィック プロパティの設定: スクリプトを使用したグラフィック要素プロパティの読み取りまたはグラフィック要素プロパティへの書き込みでは、以下のプロパティのみがサポートされています。
 - **Position (X,Y)**
 - **Size(Height,Width)**
 - **Angle**
 - **StartAngle**
 - **SweepAngle**
- シンボル ライブラリのいくつかのシンボルには、現在のリリースの **Web Client** でサポートされていないグラフィック要素または色設定が組み込まれています。
- **Windows** 共通コントロール (**WCC**) とは、ラジオボタン グループ、チェック ボックス、編集ボックス、コンボ ボックス、カレンダー、日付時刻ピッカー、およびリスト ボックスを指します。
- マルチペン トレンド グラフィック要素はサポートされません。
- ビット状態アニメーションはサポートされません。
- 真理値表アニメーション（無効化、表示オン/オフ、点滅、値の表示）はサポートされません。

- 履歴サマリ データ型を含むカスタム プロパティはサポートされません。

以下のセクションでは、これらの制限事項について詳細に説明します。

重要: Web Client にグラフィックをロードすると、サポートされていない機能は、ロガーに警告として記録されます。生成されるレポートには警告パスが含まれます。

すべてのサポート対象グラフィック要素でサポートされるプロパティ

ブラウザで表示する場合、以下のプロパティは、シンボル内のサポートされるすべてのグラフィック要素でサポートされます。

- Angle
- X
- Y
- Width
- Height
- Enable
- Visible
- AbsoluteAnchor
- RelativeAnchor
- Transparency
- ElementStyle
- OwningObject
- Start
- End
- Radius
- Tension
- カスタム プロパティの上書き

注記: ElementStyle プロパティでは、定義済みの要素スタイルを選択して、グラフィック要素に適用できます。要素のスタイルには、色プロパティ（FillColor、LineColor、TextColor、OutlineColor など）を定義するプロパティが含まれています。

すべてのグラフィック要素でサポートされるプロパティ（いくつかの制限あり）

ブラウザで表示する際、以下のグラフィック要素にはレンダリング制限があります。

グラフィック要素プロパティ	プロパティの制限
FillColor	テクスチャはサポートされていません。FillColor 上の一部の制限が個々の要素に適用されることがあります。

グラフィック要素プロパティ	プロパティの制限
UnFilledColor	FillColor プロパティと同じ制限が UnFilledColor プロパティに適用されます。
LineColor	テクスチャはサポートされていません。
Font	Font プロパティのフォント サイズ、フォント タイプ、およびフォント スタイル（太字または標準） オプションのみがサポートされます。
TextColor	TextColor プロパティの純色オプションのみがサポートされます。
FillOrientation	RelativeToScreen は、色が 'Solid'、'Fill'、または 'UnFill' に設定されている場合にのみサポートされます。
FillBehavior	常に Both に設定されます。
Runtime Behavior	TabOrder、TabStop、ZoomPercent は、いずれのグラフィック要素でもサポートされていません。
Relative references	MyPlatform、MyEngine、MyHost、MyArea、MyContainer、および Me がサポートされています。 MyViewApp はサポートされていません。
AutoScale	サポートされません。
ShowGraphic	サポートされます。
Line	シンボルの縁が 1px よりも大きい LineWeight プロパティ値で設定される線グラフィックは切り詰められます。
Group/Embed Graphic	<p>表示オン/オフ、点滅、無効化、押しボタン、ユーザー入力/値の表示、ShowSymbol、およびアクション スクリプト アニメーションがサポートされます。</p> <p>スクリプトによる 'Treat as Icon' および 'Enabled' プロパティの変更はサポートされません。</p> <p>%FillHorizontal および %FillVertical アニメーション:</p> <ul style="list-style-type: none"> - グループにのみ適用されます。サブ要素ではサポートされません。 - ボタンと画像はサポートされません。 - グループとサブ要素の「画面を基準」はサポートされません。 - サブ要素の角度が変更された場合はサポートされません。

サポートされているグラフィック要素と追加のプロパティ

ブラウザ ベースのクライアントで表示されるグラフィック要素を以下の一覧に示します。この表に含まれないグラフィック要素はブラウザ ベースのクライアントでサポートされません。

表には、各グラフィック要素でサポートされるプロパティ、およびそれらのプロパティの制限が示されています。色プロパティを組み込むグラフィックを使用する場合、色には、FillColor、LineColor およびその他の関連するグラフィック プロパティと同じ制限が適用されます。

注記: ここに示すプロパティに加えて、「[すべてのサポート対象グラフィック要素でサポートされるプロパティ](#)」および「[すべてのグラフィック要素でサポートされるプロパティ \(いくつかの制限あり\)](#)」に一覧表示されているすべてのプロパティは、サポートされているグラフィック要素で使用できます。

グラフィック要素	サポートされるプロパティ	制限事項
直線 折れ線 曲線 2 ポイントの円弧型 3 ポイントの円弧型 コネクタ	線スタイル: LineWeight LinePattern StartCap EndCap	StartCap、EndCap: サポートされますが、レンダリングが若干異なることがあります。
ボタン	ButtonStyle Text WordWrap FillOrientation FillColor Alignment	ButtonStyle: 標準のボタン スタイルのみがサポートされます。 WordWrap: ボタンの幅を超えるキャプションは切り詰められます。 FillOrientation: 'RelativeToScreen' は、色が設定されている場合にのみサポートされます。 FillColor: 純色のみがサポートされます。複数の色のグラデーションを選択した場合、最初の色を使用して単一の色に変換されます。 Alignment: 中央揃えのみがサポートされます。
テキスト	Alignment	Alignment: 左上のみがサポートされます。
画像	HasTransparentColor TransparentColor ImageStyle	FillColor、FillPercent、および UnFilledColor はサポートされません。
テキスト ボックス	Text TextFormat WordWrap LineWeight LinePattern Alignment Font	Flip はサポートされません。

グラフィック要素	サポートされるプロパティ	制限事項
ステータス	Graphics Expression	Status Style: デフォルト設定のみがサポートされます。
Web アラーム クライアント (EAC) (この要素は一部の HMI/SCADA ソフトウェアでのみサポートされます。)	アラーム モード Colors Column Details Queries and Filters Time Settings Run-Time Behavior	<p>アラーム モード</p> <ul style="list-style-type: none"> • Client Mode: [現在のアラーム]、および [最近のアラームとイベント]のみがサポートされます。 • Alarm Query: InTouch および Application Server アラームがサポートされます。 • Use Default Ack Comment: アラームとイベントを示すコメントのみがサポートされます。 <p>アラームとイベント:</p> <p>Colors: [シェルフ] および [点滅未確認アラーム] はサポートされません。</p> <p>Column Details: ソートはサポートされません。</p> <p>Queries and Filters: [デフォルト] クエリーのみがサポートされます。カスタムクエリーおよびフィルタはサポートされません。</p> <p>Data Binding: サポートされていません。</p> <p>Events: サポートされていません。</p> <p>Time Settings: サポートされていません。</p> <p>Run-Time Behavior: [見出しの表示] および [グリッドの表示] のみがサポートされ、列のサイズを変更できます。選択したレコードのユーザー ACK またはすべてのアラーム レコードの ACK のみがサポートされます。</p>

グラフィック要素	サポートされるプロパティ	制限事項
Web トレンドクライアント (この要素は一部の HMI/SCADA ソフトウェアでのみサポートされます。)	ペン 外観 オプション 履歴ソース	ペン 表示、ペン名 (説明)、および式または参照のみがサポートされます。 Pen Details: 最小、最大 Pen Options: カラープロットタイプ (すべて線として処理されます)。 外観 PlotArea: 単一タグモード、グリッド (垂直グリッドを表示、水平グリッドを表示、色)。 X time axis: カーソルを表示 (Cursor1: 色)、値の数。 Y value axis: 値の数、値軸ラベル (複数のスケール、単一のスケール)。 オプション [取得: トレンド期間] のみがサポートされます。 Data Binding: サポートされていません Event: サポートされていません 履歴ソース: 接続タイムアウト (秒)、クエリタイムアウト (秒)、および HTTP の使用はサポートされません。 注記: トレンドクライアントのメソッドおよびプロパティはサポートされません。

グラフィック要素	サポートされるプロパティ	制限事項
<p>Web SQL データ グリッド</p> <p>(この要素は一部の HMI/SCADA ソフトウェアでのみサポートされます。)</p>		<p>データ バインディング アニメーション</p> <p>イベント アニメーション</p> <p>カスタム プロパティ</p> <ul style="list-style-type: none"> • CmdCopy_dg • CmdPaste_dg • CmdWrite_dg • RowsChanged_dg • WriteButtonHide_dg • ColumnAggregateEnable_dg • SupportThemes_dg <p>- SQLDataGrid コントロールはサポートされません。</p> <p>- selectedRowNumber プロパティ値はカスタム プロパティを使用して取得できません。</p> <p>- データ バインディングはサポートされないため、プロパティがカスタム プロパティで参照されていて、デフォルト値が空の場合、プロパティのデフォルト値は取得できません。</p> <p>- kendo グリッドの制限により RowCount プロパティではグループ化のカウントを表示できません。レコードの総数が常に表示されます。</p> <p>- ローカル データベースで Windows 認証を使用する場合、仮想ユーザー NT Service\AIGWebServer を追加してデータベースへのアクセスを許可します。'Windows' 認証でのリモート データベースへのアクセス、および 'SQLServer' モードはサポートされません。</p> <p>- カルーセル ウィジェットで使用する場合、および WindowViewer で表示する場合、SQLGrid はサポートされません。</p> <p>- 2000 以上のレコードと 5 つ以上の列を含む SQLGrid にグループを適用する場合、行のスクロールと更新でパフォーマンスの問題が発生することがあります。</p>

グラフィック要素	サポートされるプロパティ	制限事項
トレンド ペン (この要素は一部の HMI/SCADA ソフトウェアでのみサポートされます。)	Data Source	Trend time Period: 固定タイプはサポートされません。
チェック ボックス		<p>Selected Value Changes: ユーザー設定に関係なく、直ちに送信された値のみがサポートされます。</p> <p>Checked: Checked プロパティはサポートされません。</p>
ラジオ ボタン グループ	Static Values and Captions States	Selected Value Changes: ユーザー設定に関係なく、直ちに送信された値のみがサポートされます。
編集ボックス		Selected Value Changes: ユーザー設定に関係なく、直ちに送信された値のみがサポートされます。
コンボ ボックス		<p>Selected Value Changes: ユーザー設定に関係なく、直ちに送信された値のみがサポートされます。</p> <p>Type: MaximumLength はサポートされません。</p>
カレンダー		<p>Selected Value Changes: ユーザー設定に関係なく、直ちに送信された値のみがサポートされます。</p> <p>Bold Dates: サポートされません。</p> <p>ShowToday: チェックアウトされている場合でも常に表示されます。</p> <p>Calendar Colors: サポートされません。</p>
日付時刻ピッカー		<p>Selected Value Changes: ユーザー設定に関係なく、直ちに送信された値のみがサポートされます。</p> <p>Configuration: サポートされません。</p> <p>Calendar DropDown Colors: サポートされていません。</p>
リスト ボックス		Selected Value Changes: ユーザー設定に関係なく、直ちに送信された値のみがサポートされます。

グラフィック要素	サポートされるプロパティ	制限事項
ConnectorPoint		ConnectorPoint の移動はサポートされません。

サポートされるアニメーション

グラフィックを Web ブラウザで表示する際にランタイム時にサポートされるアニメーションを以下の表に示します。色プロパティを組み込むアニメーションを使用する場合、選択したアニメーションの色には、FillColor、LineColor、およびその他の関連するグラフィック要素プロパティと同じ制限が適用されます。また、ブラウザの表示では値の書き込みがサポートされません。属性値を更新するアニメーションのタイプはサポートされません。

注記: サポートされるすべてのアニメーションを以下の表に示します。一覧に含まれないアニメーションは Web ブラウザで表示する際にサポートされません。

アニメーション	制限事項
アラーム輪郭	アラーム輪郭アニメーションは、表示される外観がグラフィック要素のアラーム タイプと確認ステータスを表すグラフィック要素の周囲にアラーム輪郭を表示します。各アラーム輪郭の外観は、関連付けられた要素スタイルによって定義されます。一部の要素スタイルプロパティはサポートされません。たとえば、アラーム輪郭要素スタイルの「線の色の上書き」オプションの値として「グラデーション」が指定されている場合、このオプションはサポートされません。
要素スタイル	色、線、およびフォントに関するグラフィック要素の制限を参照してください。グラフィック要素に適用されるものと同じ制限が要素スタイルアニメーションにも適用されます。 式または参照: [時間] および [経過時間] はサポートされません。 要素スタイルの定義の変更は、アプリケーションを再配置またはパブリッシュした後にのみ有効になります。 Windows 共通コントロールはサポートされません。
塗りつぶしスタイル 線スタイル テキストスタイル	色、線、およびフォントに関するグラフィック要素の制限を参照してください。グラフィック要素に適用されるものと同じ制限が要素スタイルアニメーションにも適用されます。 式または参照: [時間] および [経過時間] 状態はサポートされません。 Windows 共通コントロールはサポートされません。 TextColor: 純色のみがサポートされます。

アニメーション	制限事項
水平塗りつぶしパーセント 垂直塗りつぶしパーセント	RelativeToGraphic のみがサポートされます。
水平スライダー 垂直スライダー	スライダーはタグにデータを書き込むことはできませんが、カスタムプロパティとのデータバインディングをサポートできます。 カスタムプロパティへの書き込み時、スライダーは「マウスボタンを離したとき」のみをサポートします。 スライダーは「カーソルアンカー」をサポートしません。 スライダーは「ツールヒントを表示する」をサポートしません。
幅 高さ	Windows 共通コントロールはサポートされません。
向き	向きアニメーションを設定する際、RelativeAnchor および RelativeOrigin はサポートされません。(サポートされる唯一のアンカーは中央です)。
無効化	ラジオボタン グラフィック要素は無効化アニメーションをサポートしません。
ポイント	曲線および閉曲線のポイント アニメーションはサポートされません。
ツールヒント	ツールヒント アニメーションは、画像、ラジオボタン、ボタン、およびテキスト グラフィック要素以外でサポートされます。 ツールヒント アニメーションはグループおよび埋め込まれたシンボルではサポートされません。 Windows 共通コントロールはサポートされません。
アクション スクリプト	アクションスクリプトは、ダイアログを呼び出す関数以外でサポートされます。ShowGraphic アニメーションは枠ウィンドウでのみサポートされます。
ShowSymbol	タイトル: デフォルト オプションのみがサポートされます。 タイプ: 「閉じるボタン」オプションが選択されている場合、「モーダル」と「モードレス」がサポートされます。 位置: クライアント領域 x,y の値が 0,0 の中心のみがサポートされます。 サイズ: 「シンボルを基準」オプションのみがサポートされます。 ショートカット: サポートされません。

アニメーション	制限事項
ShowGraphic	タイトル: デフォルト オプションのみがサポートされます。 タイプ: [閉じるボタン] オプションが選択されている場合、[モーダル] および [モードレス] がサポートされます。 位置: クライアント領域 x,y の値が 0,0 の中心のみがサポートされます。 サイズ: [シンボルを基準] オプションのみがサポートされます。 ショートカット: サポートされません。
ハイパーリンク	Windows 共通コントロール、ボタン、テキスト、およびテキスト ボックス上でサポートされません。

Web Client モバイル アプリ

オペレーティング システムの要件

Apple App Store または Google Play Store からモバイル アプリケーションをダウンロードします。AVEVA Mobile Operations を検索して、[インストール] をタップします。

Android の最低バージョン: Android 6.0 (API レベル 23 – Marshmallow)

Android の推奨バージョン: Android 9.0 (API レベル 28 – Pie)

iOS の最低バージョン: iOS 9.0 以降

デバイスのサポートについては、「[ブラウザとモバイルのサポート](#)」を参照してください。

モバイル アプリへのログイン

アプリのインストール時にユーザー資格情報を入力してアプリにログインし、グラフィックを表示します。

1. モバイル アプリを起動します。
2. アプリケーションが実行するノードの**サーバー名**または **IP アドレス**を入力します。
3. **Web サイト名**を選択または入力します。提供される Web サイト名は、InTouch HMI アプリケーション マネージャで指定した [カスタム URL] に一致する必要があります。
4. **ユーザー名**と**パスワード**を入力します。必要に応じて、ユーザー名のドメイン名を入力します。
5. アプリを全画面モードで表示するには、[**全画面モード**] チェック ボックスをオンにします。
この設定は、ナビゲーション バーとヘッダー バーがデフォルトで非表示になっているときに TV などの大画面ディスプレイでフルサイズのグラフィックを表示する場合に便利です。
6. [**ログイン**] をクリックします。

正常にログインすると、Web Client のホーム ページが表示されます。

初回起動時にユーザー資格情報を入力する必要があります。その後の使用では、ユーザー資格情報がアプリケーションに記憶されます。ログイン画面から情報をクリアするには [**クリア**] ボタンを使用しま

す。モバイルアプリケーションにアクセスできるのは、**Web Client** ユーザー グループに含まれるユーザーだけです。ログオフ ボタンを使用してアプリを終了します。

注記: Web Client モバイル アプリは、匿名ログインに加えて、AVEVA Identity Manager (AIM) 経由のログインもサポートします。

Web Client モバイル アプリの使用

資格情報が認証され、有効な **Web Server** ライセンスが取得されるとアプリケーションが表示されます。デフォルトでホーム シンボルが表示されます (設定されている場合)。



1. この アイコンをクリックします。

左側のオーバーレイに実行中のアプリケーション内のグラフィックの階層構造が表示されます。

2. フォルダ名をクリックします。

フォルダ内のグラフィックが表示されます。

3. グラフィックをクリックします。

選択したグラフィックが表示されます。

[パン] および [ズーム] を使用して、グラフィックをさまざまなスケールで表示することもできます。

Web Client モバイル アプリはランタイムの言語切り替えをサポートします。

章 20 ランタイムでの言語の切り替え

ランタイム時に別の言語に切り替えることができるアプリケーションを開発できます。

ランタイム時の言語の切り替えを有効にするには、次のタスクを完了する必要があります。

- アプリケーションに対して複数の言語を設定します。
- オフライン翻訳用にアプリケーション テキストをエクスポートします。
- エクスポートされた 1 つまたは複数のディクショナリ ファイルを翻訳します。
- 翻訳された 1 つまたは複数のディクショナリ ファイルをインポートします。

ランタイム時の言語の切り替えの概要

ランタイム時に別の言語に切り替えることができるアプリケーションを開発できます。

ランタイム時の言語の切り替えを有効にするには、次のタスクを完了する必要があります。

- アプリケーションに対して複数の言語を設定します。
- オフライン翻訳用にアプリケーション テキストをエクスポートします。
- エクスポートされた 1 つまたは複数のディクショナリ ファイルを翻訳します。
- 翻訳された 1 つまたは複数のディクショナリ ファイルをインポートします。

ランタイム時の言語の切り替えのための言語の設定

すべての InTouch アプリケーションは、アプリケーションを開発するために使用されるベース言語に関連しています。サポートする追加の言語を設定する必要があります。

注記: 言語の切り替えとネットワーク アプリケーション開発 (NAD) を組み合わせて使用している場合、NAD クライアント ノードに対しては、変更モードは [WindowViewer に変更をロード] または [WindowViewer へ変更ロード時にメッセージ表示] ではなく、[WindowViewer の再起動] または [WindowViewer の再起動時にメッセージ表示] に設定することをお勧めします。

ランタイム時の言語の切り替え用に言語を設定するには

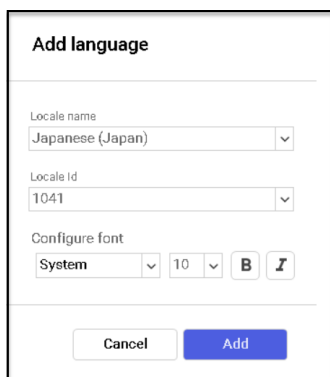
1. WindowMaker で、言語を設定するアプリケーションを開きます。
2. [ファイル] メニューの [設定] グループで [言語] をクリックします。[言語] ウィンドウが表示されます。



Language Name	Locale Id	Font Name	Font Size	Bold	Italic
English (United...)	1033		0	<input type="checkbox"/>	<input type="checkbox"/>

[言語の設定] ダイアログ ボックスに、アプリケーションのベース言語が表示されます。

3. [追加] をクリックします。[言語の追加] ダイアログ ボックスが表示されます。



4. 言語およびフォント設定を指定します。フォントを設定すると、翻訳されたテキストのデフォルトのフォントプロパティが定義されます。
 - **[名前で指定]** リストまたは **[ロケール ID]** リストで、追加する言語をクリックします。名前で言語を選択した場合、対応するロケール ID が **[ロケール ID]** リストに表示され、ロケール ID で言語を選択した場合、対応する名前が表示されます。
 - **[フォント]** をクリックします。**[フォント]** ダイアログ ボックスが表示されます。フォントを設定して、**[OK]** をクリックします。
5. **[OK]** をクリックして、**[言語の追加]** ダイアログ ボックスを閉じます。設定した言語が **[言語の設定]** ダイアログ ボックスに一覧表示されます。
6. 他にも言語を追加する場合は、手順 3 ～ 5 を繰り返します。
7. 終了したら、**[閉じる]** をクリックします。

言語を削除するには

- a. 削除する言語を **[言語の設定]** ダイアログ ボックスで択します。
- b. **[削除]** をクリックします。
[削除の確認] ダイアログ ボックスが表示され、アプリケーションからの言語の削除の確認を求めるメッセージが表示されます。
- c. **[はい]** をクリックします。
[言語の設定] ダイアログ ボックスが更新され、言語が削除されたことが示されます。

デフォルト言語を設定するには

- a. アプリケーションのデフォルトとして設定する言語を **[言語の設定]** ダイアログ ボックスで選択します。
- b. **[デフォルトに設定]** をクリックします。
[言語の設定] ダイアログ ボックスが更新され、左下隅にアプリケーションのデフォルト言語が表示されます。

設定した言語のフォント設定の変更

すべての言語のデフォルトのフォントは **Tahoma** です。フォント スタイルおよびサイズは、ベース言語内の個々の語句の対応する設定によって異なります。既に設定されている言語のフォント設定は変更できます。異なる言語間ではテキスト表示が異なるため、翻訳されたテキストがボタンやその他のオブジェクトに正しく表示されるように適切なフォントを指定できます。

設定言語のフォント設定を変更するには

1. WindowMaker で、設定言語のフォント設定を変更するアプリケーションを開きます。
2. [ファイル] メニューで、[設定] をポイントし、[言語] をクリックします。
[言語] 画面が表示されます。



3. [全般] タブの言語のリストから目的の言語を選択します。
4. [フォント名] と [フォントサイズ] エントリをダブルクリックして値を編集します。
5. 必要に応じて、[太字] と [斜体] チェック ボックスをオンまたはオフにします。
6. 編集を行い、[OK] をクリックします。

ランタイム時の言語の切り替え機能の追加

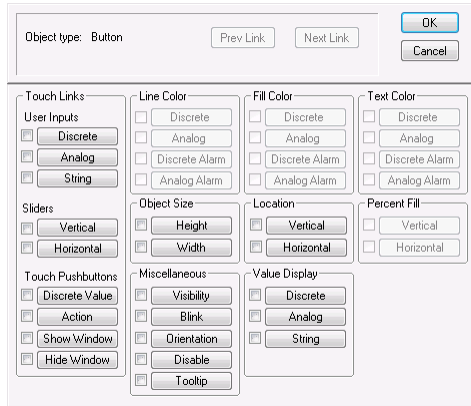
ランタイムユーザーは、WindowViewer の [システム] メニューの [言語] コマンドを使用してアプリケーションインターフェイスの言語を切り替えることができます。

ランタイムユーザーが言語を切り替えることができるように、アプリケーションにボタンを追加することもできます。開始する前に、アプリケーションで追加の言語を設定し、言語のロケール ID を知っておくようにしてください。アプリケーションの言語設定の詳細については、[「ランタイム時の言語の切り替えのための言語の設定」](#)を参照してください。

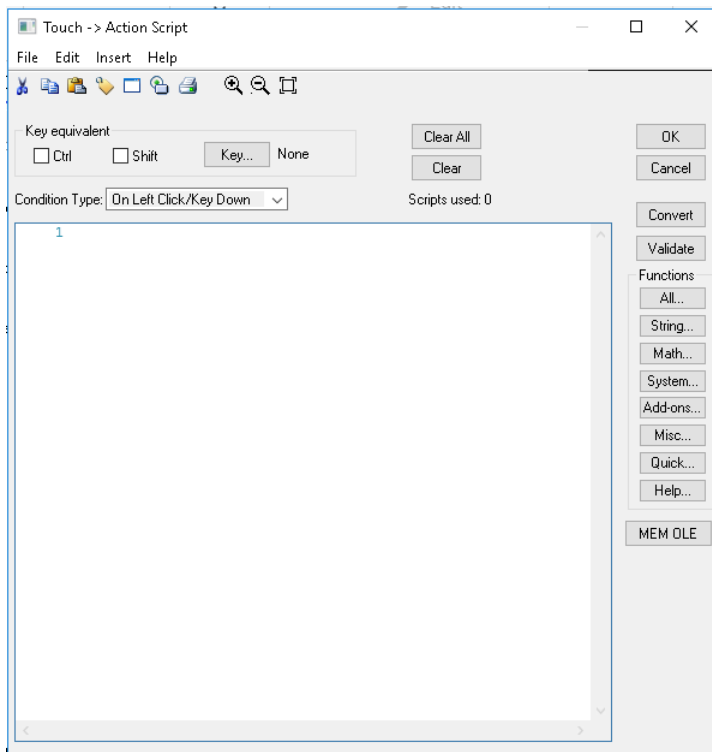
ランタイム時に言語を切り替えるためのボタンを追加するには

1. WindowMaker で、言語の切り替えボタンを追加するアプリケーション ウィンドウを開きます。
2. ボタンを描画し、選択したときに切り替えられる言語を示すテキスト ラベルをボタンに割り当てます。
3. ボタンをダブルクリックします。

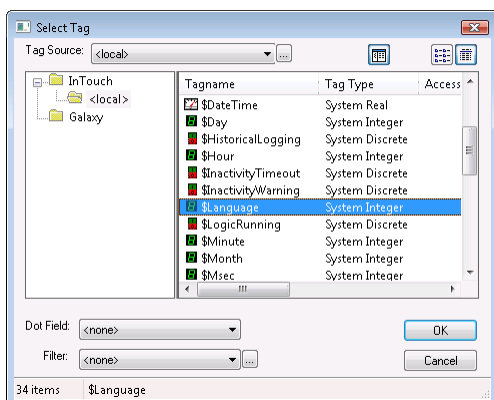
アニメーションを選択するダイアログ ボックスが表示されます。



4. [タッチ押しボタン] 領域で、[アクション] をクリックします。
[タッチ->アクションスクリプト] ダイアログ ボックスが表示されます。

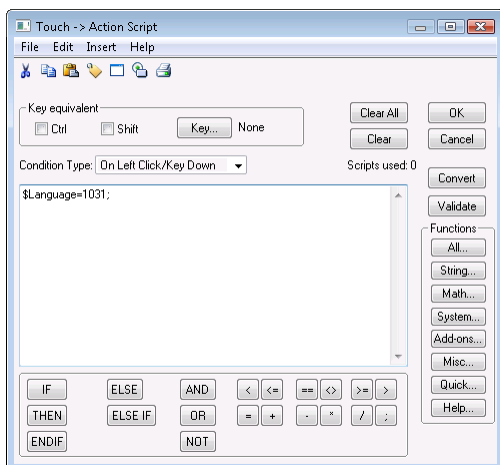


5. [タッチ->アクションスクリプト] ダイアログ ボックスのスクリプト領域内をダブルクリックします。
[タグの選択] ダイアログ ボックスが表示されます。



1. **[\$Language]** システム タグをクリックし、**[OK]** をクリックします。

\$Language システム タグを、ボタンに割り当てる言語のロケール ID に等しく設定し、**[OK]** をクリックします。



注記: **\$Language** タグの代わりに、スクリプト関数 **SwitchDisplayLanguage(LocaleID)** を使用することもできます。

1. **[OK]** をクリックして、ダイアログ ボックスを閉じます。

SwitchDisplayLanguage() 関数

翻訳された文字列とアラーム フィールドが表示される目的の言語で表示される、静的テキストの表示を切り替えます。

カテゴリ

その他

構文

```
SwitchDisplayLanguage(LocaleID);
```

パラメータ

LocaleID

ランタイムに静的テキストの文字列とアラーム フィールドが表示される言語

例

この例では、ランタイムに表示される言語はドイツ語です。

```
SwitchDisplayLanguage(1031);
```

参照項目

\$Language システム タグ変数

\$Language システム タグ

InTouch アプリケーションで複数の言語が定義されている場合、\$Language システム タグには、現在表示されている言語の言語 ID の値が反映されます。デフォルトでは、これは、ベースの InTouch システム (E/F/G/J/SC) の言語 ID (ロケール ID) です。これを他の ID に設定すると、新しい言語で定義された値を使用して文字列とアラーム フィールドが切り替わります。

注記: \$Language タグをローカル タグとして設定すると、Web Client で独立した言語変更が可能になります。複数のユーザー セッションで Web Client を異なる言語で表示できます。Web Client で言語を変更しても、WindowViewer で選択されている言語には影響しません (その逆も同様です)。

カテゴリ

システム

データ タイプ

整数型 (読み取り/書き込み)

オフライン翻訳用のアプリケーション テキストのエクスポート

InTouch アプリケーションに多くの文字列が含まれている場合、通常バルク翻訳用にテキスト文字列を送信します。アプリケーションの文字列を翻訳用にエクスポートし、テキストエディタ、XML エディタ、または Microsoft Excel などのスプレッドシート プログラムを使用して整理できます。

静的テキストは、次のアイテムからエクスポートできます。

- テキスト オブジェクト
- ボタン テキスト
- SmartSymbol 内テキスト
- ツールヒント静的テキスト
- ユーザー メッセージ
- 入力リンク内のメッセージのオン/オフ
- 出力リンク内のメッセージのオン/オフ
- ウィザード上のテキスト

WindowMaker 内のすべてのウィンドウを閉じるまでは、ディクショナリをエクスポートできません。ディクショナリ ファイルをエクスポートした後にアプリケーションを変更する場合、ディクショナリ ファイルを再度エクスポートする必要があります。詳細については、「[既存のディクショナリ ファイルへのテキストのエクスポート](#)」を参照してください。

テキスト文字列は、一度に 1 つの言語に対してのみエクスポートできます。デフォルトでは、[マイ InTouch アプリケーション] フォルダが InTouch HMI で開かれます。他のフォルダを選択した場合、

InTouch HMI ではそのパスがデフォルトになります。各言語の語句をエクスポートするために新しいフォルダを作成すると、ディクショナリ ファイルを管理しやすくなります（たとえば、...\マイ InTouch アプリケーション\My German Files\ など）。

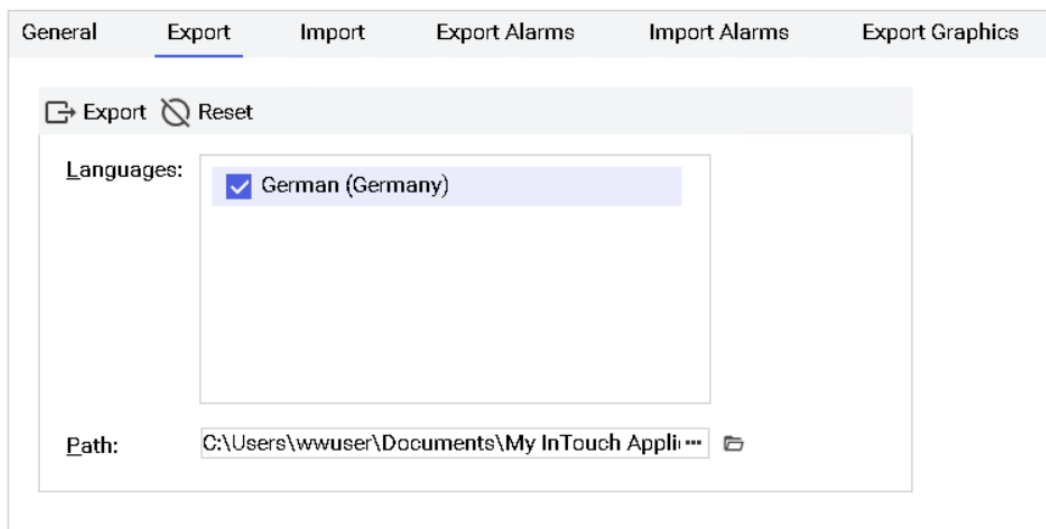
InTouch HMI によって、アプリケーションのディクショナリ ファイルが作成され、アプリケーション内の各 SmartSymbol 用に別のディクショナリ ファイルが作成されます。アプリケーションディクショナリ名は、アプリケーション名_localeID の形式を持ち、SmartSymbol ディクショナリ ファイルは Symbol_localeID_GUID の SSD_Name の形式を持ちます。

アプリケーションのディクショナリをエクスポートすると、ファイルは Microsoft Excel 2003 以降を使用して編集できる .xml ファイルになります。

オフライン翻訳用にアプリケーションテキストをエクスポートするには

1. WindowMaker を起動し、オフライン翻訳のためにテキスト文字列をエクスポートするアプリケーションを開きます。
2. [ファイル] メニューで、[設定] をポイントし、[言語] をクリックします。
[言語] 画面が表示されます。
3. [エクスポート] タブをクリックします。[ディクショナリのエクスポート] ダイアログボックスが表示されます。

Languages



4. エクスポートの設定を行います。
 - [定義された言語] リストで、エクスポートする言語ディクショナリをクリックします。
 - [パス] ボックスで、ディクショナリをエクスポートするフォルダを入力します。[参照] をクリックして、既存のフォルダを選択するか、新しいフォルダを作成します。
5. [エクスポート] をクリックします。エクスポートの進行が表示されます。エクスポートが正常に終了すると、[エクスポートに成功しました] ダイアログボックスが表示されます。
6. [閉じる] をクリックして WindowMaker ウィンドウに戻るか、[閉じてからエクスプローラを起動する] をクリックしてディクショナリ ファイルを含むフォルダを開きます。

既存のディクショナリ ファイルへのテキストのエクスポート

オフライン翻訳用にアプリケーション テキストをエクスポートした後、アプリケーションを変更する場合があります。アプリケーションを変更した場合は、テキストを再度エクスポートする必要があります。詳細については、「[オフライン翻訳用のアプリケーション テキストのエクスポート](#)」を参照してください。

同じディレクトリに 2 回以上エクスポートすると、[ファイル置換の確認] メッセージが表示されます。

[はい] をクリックすると、新しい文字列や最後にエクスポートしてから追加した言語情報で、既存の .xml ファイルが更新されます。既存のディクショナリ ファイルに語句の翻訳が含まれ、それを InTouch HMI に以前インポートしていた場合、これらの翻訳は維持されます。最後のエクスポート以来、アプリケーションから任意の語句を削除した場合、その語句はディクショナリ ファイルから削除されます。

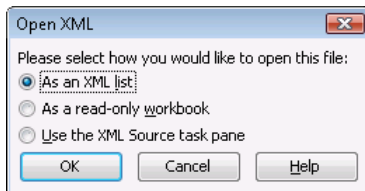
エクスポートされたディクショナリ ファイルの翻訳

アプリケーション テキストが含まれているディクショナリ ファイルをエクスポートした後に、Microsoft Excel 2003 以降を使用してテキストを編集します。

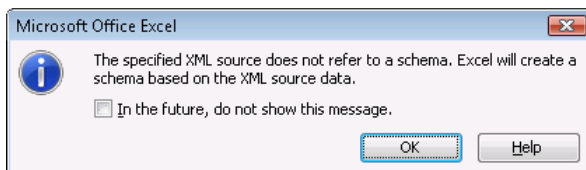
エクスポートした各言語に対する個別のディクショナリ ファイルが InTouch HMI によって作成されます。また、アプリケーションの各 SmartSymbol に対しても、個別のディクショナリ ファイルが InTouch HMI によって作成されます。すべての言語および SmartSymbol に対して、すべてのディクショナリ ファイルを翻訳してください。

エクスポートされたディクショナリ ファイルを翻訳するには

1. Excel で XML ファイルを開きます。[XML を開く] ダイアログ ボックスが表示されます。



2. [XML リストとして開く] リストをクリックし、[OK] をクリックします。メッセージが表示されます。

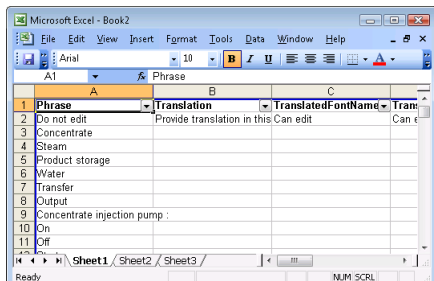


3. [OK] をクリックします。

Excel で XML ファイルが開き、次のカラムが表示されます。

- アプリケーション内のフレーズ
- 翻訳されたフレーズ
- 翻訳されたフォント名
- 翻訳されたフォント プロパティ

- 翻訳されたフォント サイズ
- ベース フォント プロパティ
- ベース フォント サイズ
- コンテキスト、フレーズ ID、言語 ID および 外国語 ID



重要: [Translation] カラム、[TranslatedFontSize] カラム、[TranslatedFontName] カラムおよび [TranslatedFontProperty] カラムのデータのみを修正してください。カラム ヘッダーは変更しないでください。行を挿入または削除しないでください。

1. [語句] カラムのベース言語の文字列に対応する行の [翻訳] カラムに、言語固有のテキストを入力します。
2. 必要に応じて、WindowViewer でのスペースにテキストを表示できるように、翻訳された文字列のフォント パラメータを変更します。

- [TranslatedFontName] カラムで、フォント名を入力します。
- [TranslatedFontProperty] カラムで、フォント プロパティの表記法を入力します。

B = 太字

I = 斜体

U = 下線

たとえば、テキストを太字にする場合、[TranslatedFontProperty] カラムに **B** と入力します。テキストを太字および下線付きにするには、[TranslatedFontProperty] カラムに **BU** と入力します。

3. XML データのファイルのタイプを使用して、ファイルを保存します。

重要: XML スプレッドシートなどの他のファイルのタイプで保存すると、Excel によってスキーマが変更され、InTouch HMI でファイルをロードできません。XML ファイルの名前を変更すると、ランタイム時の言語の切り替えが機能しません。

翻訳されたディクショナリ ファイルのインポート

エクスポートした各言語に対するディクショナリ ファイルが InTouch HMI によって作成されます。また、アプリケーションの各 SmartSymbol に対しても、個別のディクショナリ ファイルが InTouch HMI によって作成されます。翻訳後、各言語に対してディクショナリ ファイルをインポートし、これらの言語に対してランタイム時の言語の切り替えを有効にする必要があります。1つの言語のすべてのディクショナリ ファイルは同じフォルダに配置する必要があります。

翻訳されたディクショナリ ファイルをインポートするには

1. WindowMaker を起動し、翻訳されたディクショナリ ファイルをインポートするアプリケーションを開きます。
2. [ファイル] メニューで、[設定] をポイントし、[言語] をクリックします。
1. [言語] 画面の [インポート] タブをクリックします。



1. インポート設定を構成します。
 - [定義された言語] リストで、インポートする言語ディクショナリをクリックします。
 - [パス] ボックスで、インポートするディクショナリ ファイルへのパスを入力します。[参照] をクリックし、ファイルを参照して選択します。
2. [インポート] をクリックします。
3. SmartSymbol ディクショナリ ファイルを再度インポートしている場合、既存のファイルを置換するよう求めるメッセージが表示されます。

インポートが正常に終了すると、[インポートに成功しました] ダイアログ ボックスが表示されます。

アラームのランタイム時の言語切り替え

ランタイム時の言語の切り替えのセットアップの一部として、アラーム グループ名、アラーム コメント、アラーム フィールド、内部ステータス メッセージもローカライズできます。テキスト文字列のランタイム時の言語の切り替えのほかに、Alarm Viewer および Alarm DB View コントロールのアラーム コメント、アラーム状態、アラーム タイプ、およびアラーム クラスのランタイム時の言語の切り替えを設定することもできます。

翻訳されたアラーム グループ名を使用してスクリプト記述、クエリー、またはフィルタも行うことができます。

翻訳のためのアラーム コメントのエクスポート

翻訳するために、アラーム コメントをエクスポートできます。

以下の項目のアラーム状態、アラーム タイプ、およびアラーム クラス フィールドをエクスポートできます。

- アラーム コメントを持つすべてのタグ
- タグ コメントを持つすべてのタグ
- システム タグによりイベントが発生したとき、クライアントに示されるコメントをローカライズできるようにするシステム タグ

2 文字のアプリケーション ID の理解

ローカライズ用にアラームおよびタグ変数コメントをエクスポートするとき、2 文字のアプリケーション ID を指定する必要があります。ID は、同じ名前のアプリケーションによって複数のアラームが生成された場合、個々のアラームを識別するためにシステムによって内部で使用されます。

タグ変数にはタグ変数コメントとアラーム コメントの両方を含めることができるため、2 文字のアプリケーション ID の最後に 1 および 2 が追加され、これらの 2 つのフィールド間が区別されます。タグ変数コメントでは、ID とタグ変数名の間に 1 が挿入されます。アラーム コメントでは、ID とタグ変数名の間に 2 が挿入されます。たとえば、AA1TankLevel はタグ変数コメントで、AA2TankLevel はアラーム コメントです。

アプリケーションをエクスポートすると、アプリケーション ID 情報は削除されます。

アラーム データベースに 2 文字のアプリケーション ID を持たない古いデータが含まれていて、新しいレコードの先頭には ID が付いている場合、Alarm DB View コントロール内のアラーム コメント クエリーでは、<、<=、>、および >= の演算子を使用することはできません。

アラーム コメントのエクスポート

翻訳するために、アラーム コメントをエクスポートできます。

注意: アラームとタグ コメントをエクスポートする前に、データの破損やエラーの可能性に備えて、ターゲットディレクトリにファイルをバックアップしてください。

オフライン翻訳のためにアラーム コメントをエクスポートするには

1. WindowMaker を起動し、オフライン翻訳のためにアラーム コメントをエクスポートするアプリケーションを開きます。
2. [ファイル] メニューで、[設定] をポイントし、[言語] をクリックします。
[言語] 画面が表示されます。
3. [言語] 画面の [アラームのエクスポート] タブをクリックします。

Languages

The screenshot shows the 'Languages' dialog box with the 'Export Alarms' tab selected. At the top, there are tabs for 'General', 'Export', 'Import', 'Export Alarms', 'Import Alarms', and 'Export Graphics'. Below the tabs, there are buttons for 'Export' (with a folder icon) and 'Reset'. The 'Languages' section contains a list box with 'German (Germany)' selected. Below this, the 'Path' field shows 'C:\Users\wwuser\Documents\My InTouch Appli...'. At the bottom, there is a section for 'Two characters' with a text box containing 'DE' and the label 'for Unique applications'.

4. [パス] ボックスで、ディクショナリをエクスポートするフォルダを入力します。[参照] をクリックして、既存のフォルダを選択するか、新しいフォルダを作成します。
5. [2 文字の文字は一意のアプリケーションを表します] ボックスで、2 文字を入力します。ID には英数字のみが含まれ、大文字小文字が区別されます。

注意: このアプリケーションからアラームやタグ コメントを以前エクスポートしたことがある場合は、次回エクスポートする際にも同じ 2 文字のアプリケーション ID を使用する必要があります。新しい 2 文字のアプリケーション ID を入力すると、InTouch HMI ではすべてのアラームやタグに ID が再生成されるため、すべての既存の翻訳が失われる原因となります。

1. [エクスポート] をクリックして、XML ディクショナリ ファイルに情報をエクスポートします。

InTouch HMI によって、それぞれの設定言語に対して個々のエクスポート ファイルが作成されます。異なる言語のすべてのディクショナリ ファイルが、指定する単一のディクショナリにエクスポートされます。

エクスポートされる言語に対して重複するファイルが存在する場合、ファイルの名前でメッセージが表示されます。エクスポートをキャンセルするか、エクスポート操作を続行できます。

エクスポートが正常に終了すると、[エクスポートに成功しました] ダイアログ ボックスが表示されます。

注記: 設定されたタグ名ディクショナリのアラーム コメントのサイズが 127 文字を超える場合、またはタグ コメントが 46 文字を超える場合、アラームまたはタグ コメントはエクスポートされません。エクスポート プロセスの最後に、ディクショナリ ファイルにコメントがエクスポートされなかったことが通知され、エクスポート ディクショナリに AlarmComment.log ファイルまたは TagComment.log ファイルが作成されます。

2. [閉じる] をクリックして WindowMaker ウィンドウに戻るか、[閉じてからエクスプローラを起動する] をクリックしてディクショナリ ファイルを含むフォルダを開きます。

既存のアラーム コメント ファイルへのエクスポート

オフライン翻訳のためにアラームやタグ コメントをエクスポートした後で、アラームおよびタグ コメントを再びエクスポートする必要があるアプリケーションに変更を加える場合があります。詳細については、「[翻訳のためのアラーム コメントのエクスポート](#)」を参照してください。

同じディレクトリに 2 回以上エクスポートすると、[ファイル置換の確認] ダイアログ ボックスが表示されます。

既存のディクショナリ ファイルを、最後にエクスポートしてから追加された新しい文字列と言語情報で更新するには、[はい] をクリックします。既存のディクショナリ ファイルに語句の翻訳が含まれ、それを InTouch に以前インポートしていた場合、これらの翻訳は維持されます。最後のエクスポート以来、アプリケーションから任意の語句を削除した場合、その語句はディクショナリ ファイルから削除されます。

InTouch HMI で設定されたすべての言語に対する既存のディクショナリ ファイルを更新するには、[すべてにいいえ] をクリックします。

既存ファイルまたはすべての言語に対する複数の既存ファイルが上書きされることを防ぐには、[いいえ] または [すべていいえ] をクリックします。

再びエクスポートされる場合、アラーム コメント、アラーム フィールド、およびタグ コメントの既存の翻訳は保持されます。

ディクショナリ ファイルの編集

ディクショナリ ファイルを作成した後で、文字列を編集する必要があります。

ディクショナリ ファイルの名前は 2 文字のアプリケーション ID とエクスポートされる言語から作成されます。たとえば、設定言語が Chinese(PCR)-2052 で、2 文字のアプリケーション ID が AA の場合、ファイル名は AA_2052_AlarmComment.xml となります。このファイルは、ランタイム言語の切り替えファイルにより使用されるものと同じ XML スキーマを使用して記述されます。

ディクショナリ ファイルの一般的な構造は、以下のとおりです。

```
<?xml version="1.0" encoding="utf-8" ?>
- <ArrayOfAlarmCommentPhraseItem xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
- <AlarmCommentPhraseItem Phrase="Do not edit">
  <Translation>Provide translation in this column</Translation>
  <Context>Do not edit</Context>
  <PhraseID>0</PhraseID>
  <LanguageID>0</LanguageID>
  <ForeignLanguageID>0</ForeignLanguageID>
</AlarmCommentPhraseItem>
- <AlarmCommentPhraseItem Phrase="System">
  <Translation />
  <Context>$System : System Tag Comment</Context>
  <PhraseID>AA1$System</PhraseID>
  <LanguageID>1033</LanguageID>
  <ForeignLanguageID>2052</ForeignLanguageID>
</AlarmCommentPhraseItem>
</ArrayOfAlarmCommentPhraseItem>
```

翻訳文字列の翻訳を入力します。その他の情報は変更しないでください。

アラーム状況、アラーム タイプ、およびアラーム クラスの値の一部を上書きできます。これらの値に使用できる最大長は 50 文字です。

以下のアラーム状況の値は、InTouch が生成するアラームに対して上書きできます。

上書きする値	デフォルトで表示される文字列
UNACK_RTN	UNACK_RTN
ACK_RTN	ACK_RTN
UNACK_ALM	UNACK_ALM
ACK_ALM	ACK_ALM

以下のアラームのタイプの値は、InTouch が生成するアラームに対して上書きできます。

上書きする値	デフォルトで表示される文字列
SPC	SPC
HiHi	HiHi
HI	HI
LO	LO
LOLO	LOLO
MINDEV	MINDEV
MAJDEV	MAJDEV
ROC	ROC
DSC	DSC
OPR	OPR
LGC	LGC
DDE	DDE
SYST	SYST
USER	USER
PRO	PRO
LOGON_FAILED	LOGON_FAILED

以下のアラーム クラスの値は、InTouch が生成するアラームに対して上書きできます。

上書きする値	デフォルトで表示される文字列
DEV	DEV
ROC	ROC

上書きする値	デフォルトで表示される文字列
DSC	DSC
EVENT	EVENT
VALUE	VALUE

翻訳されたアラーム コメントのインポート

文字列を翻訳した後で、各言語に対してディクショナリ ファイルをインポートし、これらの言語に対してランタイム時の言語の切り替えを有効にする必要があります。

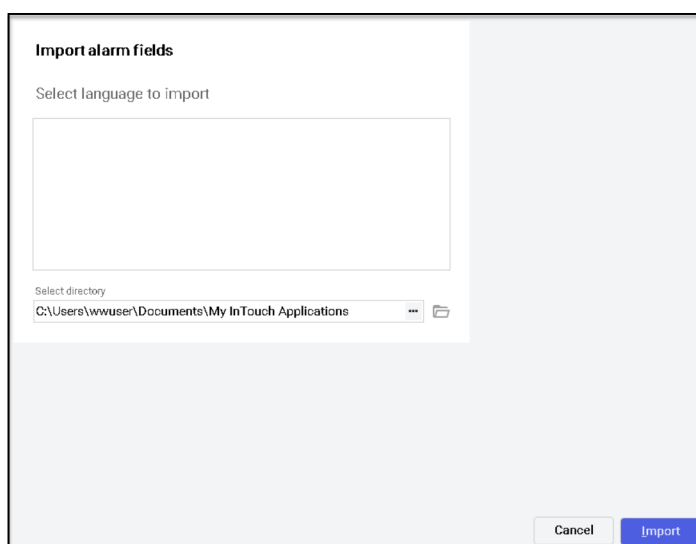
翻訳されたアラーム コメント ディクショナリ ファイルをインポートすると、それらはアプリケーション ディレクトリ内の該当する言語フォルダにコピーされます。

既存のアプリケーションに対して翻訳されたアラーム コメントは、インポートされたファイルの内容で上書きされ、アプリケーションバージョン（\$AppVersion）は1ずつ増大します。

他のノードから複数のディクショナリ ファイルをインポートして、複数ノードからのアラーム フィールドのローカリゼーションをサポートするには、翻訳されたディクショナリ ファイルを他のノードから1つのディレクトリにコピーします。このディレクトリをインポートパスとして選択します。複数のディクショナリ ファイルが、1回のインポート操作でインポートされます。InTouch HMI では、インポートされる言語に基づいて自動的にファイルパスが作成されます。

翻訳されたアラーム コメント ファイルをインポートするには

1. WindowMaker を起動し、翻訳されたディクショナリ ファイルをインポートするアプリケーションを開きます。
1. [ファイル] メニューの [設定] をクリックし、[言語] をクリックします。
[言語] ダイアログ ボックスが表示されます。
1. [言語] 画面の [アラームのインポート] タブをクリックします。



2. [パス] ボックスで、インポートするディクショナリ ファイルへのパスを入力するか、[参照] をクリックしてファイルを選択します。
3. [インポート] をクリックします。ディクショナリ ファイルで翻訳が行われていない場合、インポートする翻訳済みディクショナリ ファイルが存在しないことを示すダイアログ ボックスが表示されます。
4. [OK] をクリックします。インポートが正常に終了すると、[インポートに成功しました] ダイアログ ボックスが表示されます。

翻訳のためのアラーム グループ名のエクスポート

翻訳するためにアラーム グループ名をエクスポートできます。

注意: アラームとタグ コメントをエクスポートする前に、データの破損やエラーの可能性に備えて、ターゲット ディレクトリにファイルをバックアップしてください。

オフライン翻訳のためにアラーム コメントをエクスポートするには

1. WindowMaker を起動し、オフライン翻訳のためにアラーム コメントをエクスポートするアプリケーションを開きます。
2. [ファイル] メニューで、[エクスポート] をポイントします。
3. [ローカリゼーション] をクリックし、[アラーム メッセージ] をクリックします。
4. [アラーム フィールドのエクスポート] 画面が表示されます。
5. [エクスポートする言語の選択] リストで、翻訳用にエクスポートする言語を選択します。
6. [ディレクトリの選択] ボックスで、ファイルをエクスポートする場所を選択します。
7. [2つの文字は一意のアプリケーションを表します] ボックスで、2文字を入力します。ID には英数字のみが含まれ、大文字小文字が区別されます。

注意: このアプリケーションからアラームやタグ コメントを以前エクスポートしたことがある場合は、次回エクスポートする際にも同じ 2 文字のアプリケーション ID を使用する必要があります。新しい 2 文字のアプリケーション ID を入力すると、InTouch HMI ではすべてのアラームやタグに ID が再生成されるため、すべての既存の翻訳が失われる原因となります。

8. [エクスポート] をクリックして、XML ディクショナリ ファイルに情報をエクスポートします。

InTouch HMI によって、それぞれの設定言語に対して個々のエクスポート ファイルが作成されます。異なる言語のすべてのディクショナリ ファイルが、指定する単一のディクショナリにエクスポートされます。

エクスポートされる言語に対して重複するファイルが存在する場合、ファイルの名前でメッセージが表示されます。エクスポートをキャンセルするか、エクスポート操作を続行できます。

エクスポートが正常に終了すると、[エクスポートに成功しました] ダイアログ ボックスが表示されます。

注記: 設定されたタグ名ディクショナリのアラーム コメントのサイズが 127 文字を超える場合、またはタグ コメントが 46 文字を超える場合、アラームまたはタグ コメントはエクスポートされません。エクスポート プロセスの最後に、ディクショナリ ファイルにコメントがエクスポートされなかったことが通知され、エクスポート ディクショナリに AlarmComment.log ファイルまたは TagComment.log ファイルが作成されます。

9. [閉じる] をクリックして WindowMaker ウィンドウに戻るか、[閉じてからエクスプローラを起動する] をクリックしてディクショナリ ファイルを含むフォルダを開きます。

翻訳されたアラーム グループ名のインポート

文字列を翻訳した後で、各言語に対してディクショナリ ファイルをインポートし、これらの言語に対してランタイム時の言語の切り替えを有効にする必要があります。

翻訳されたアラーム コメント ディクショナリ ファイルをインポートすると、それらはアプリケーション ディレクトリ内の該当する言語フォルダにコピーされます。

既存のアプリケーションに対して翻訳されたアラーム コメントは、インポートされたファイルの内容で上書きされ、アプリケーション バージョン (\$AppVersion) は 1 ずつ増大します。

他のノードから複数のディクショナリ ファイルをインポートして、複数ノードからのアラーム フィールドのローカリゼーションをサポートするには、翻訳されたディクショナリ ファイルを他のノードから 1 つのディレクトリにコピーします。このディレクトリをインポートパスとして選択します。複数のディクショナリ ファイルが、1 回のインポート操作でインポートされます。InTouch HMI では、インポートされる言語に基づいて自動的にファイルパスが作成されます。

翻訳されたアラーム グループ ファイルをインポートするには

1. WindowMaker を起動し、翻訳されたディクショナリ ファイルをインポートするアプリケーションを開きます。
1. [ファイル] メニューで、[インポート] をクリックします。
2. [ローカリゼーション] をクリックし、[アラーム メッセージ] をクリックします。
[アラーム フィールドのインポート] ダイアログ ボックスが表示されます。
1. [インポートする言語の選択] リストで目的の言語を選択します。
2. [パス] ボックスで、インポートするディクショナリ ファイルへのパスを入力するか、[参照] をクリックしてファイルを選択します。
3. [インポート] をクリックします。ディクショナリ ファイルで翻訳が行われていない場合、インポートする翻訳済みディクショナリ ファイルが存在しないことを示すダイアログ ボックスが表示されます。
4. [OK] をクリックします。インポートが正常に終了すると、[インポートに成功しました] ダイアログ ボックスが表示されます。

ランタイム時の言語の切り替え機能のテスト

アプリケーションでランタイム時の言語の切り替えを有効にした後で、言語の切り替え機能をテストします。アラームとタグ コメント、およびアラーム フィールドの言語の切り替えは、Alarm Viewer コントロールおよび Alarm DB View コントロールでのみ表示できます。

ローカライズされたアラームおよびタグ コメントを操作する場合は、以下の点に注意してください。

- アラームまたはタグ コメントが \$Language によって指定されている言語に翻訳されていない場合、アラーム クライアントによりデフォルトのコメントが表示されます。
- Alarm Viewer コントロールが複数のプロバイダからクエリーを実行している場合、アプリケーションにリモート ノードアプリケーションの翻訳されたディクショナリ ファイルが存在すると、リモー

ト ノードからのアラーム コメント、タグ コメント、およびアラーム フィールドも翻訳された状態で表示されます。

- アラームを確認し、コメントを提供すると、ローカライズされたアラーム コメントの代わりに、アラーム クライアントによってこのコメントが表示されます。
- **Alarm Viewer** コントロールが停止モードの場合、言語を切り替えても、言語は切り替わった状態で表示されません。コントロールの停止モードを解除したときに、コントロールは翻訳された文字列で更新されます。
- **Alarm Viewer** コントロールのローカリゼーションは、コントロールの表示用のみを目的として行われています。言語が切り替えられても、すべてのスクリプト関数では通常どおりデフォルトの文字列が返されます。
- **Alarm DB Logger** は、データベースのデータのデフォルト言語文字列のみを保存します。データベースには、ローカライズされた文字列は保存されません。
- **EVENT** および **ACK** などのアラーム フィールドの固有 ID は事前定義され、異なるノードの複数のディクショナリ ファイルにわたり同じ ID を持ちます。アラーム クライアントは最初にロードされたディクショナリ ファイルから翻訳を選択し、他のディクショナリ ファイルの翻訳は無視されます。理想的には、1つの言語のすべてのディクショナリ ファイルのアラーム フィールドの翻訳が同じである必要があります。指定された言語の同じアラーム状態に対しては、複数のアラーム クライアント (**Alarm DB View** コントロールおよび **Alarm Viewer** コントロール) で同じ翻訳が使用されます。
- 翻訳されたテキストは、アラーム コメントの場合は 131 文字、タグ コメントの場合は 160 文字に切り捨てられます。

言語の切り替え機能をテストするには

1. **WindowViewer** でアプリケーションを開きます。
2. **[ファイル]** メニューで、**[設定]** をポイントし、**[言語]** をクリックします。次に、切り替える言語の名前を選択します。
対応する翻訳されたディクショナリ ファイル（存在する場合）からの情報がロードされ、表示されます。
3. 言語を切り替えるためのボタンを追加している場合は、ボタンをクリックして、スクリプトをテストします。

ネットワーク アプリケーション開発 (NAD) クライアントへのローカライズされたファイルの配布

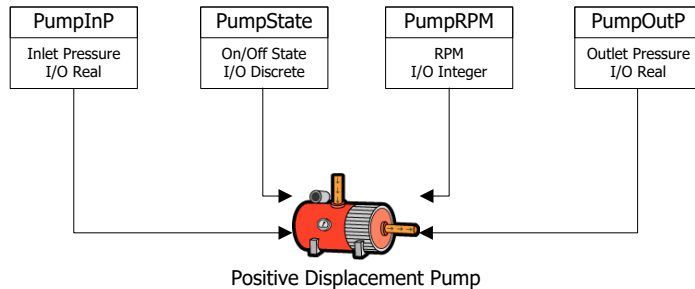
ローカライズされたアラーム コメント、タグ変数コメント、およびアラーム フィールドを含むファイルは、**InTouch** アプリケーションの一部としてネットワーク アプリケーション開発 (NAD) クライアントに配布されます。アラーム コメントを含む更新ファイルを受け取った場合、サポートされているアラーム クライアントで翻訳されたアラーム コメントを表示するには、**WindowViewer** を再起動する必要があります。

ネットワーク アプリケーション開発 (NAD) と組み合わせた言語の切り替え機能を使用する場合は、**NAD** クライアント ノードの変更モードを、「**WindowViewer** の再起動」または「**WindowViewer** の再起動時にメッセージ表示」に設定します。

章 21 タグ

InTouch ヒューマン マシン インターフェイス (HMI) は、製造環境におけるコンポーネントをグラフィック表示するアプリケーションです。このグラフィカル インターフェイスを使用して、工場のオペレータは製造工程を監視および管理することができます。

以下の図は、製造工程のコンポーネントであるポンプの例を示しています。ポンプには、値に関連付けられているプロパティがあります。圧力、RPM、およびステータスはポンプのプロパティであり、その値は HMI から監視されます。



タグは InTouch HMI アプリケーションのデータ アイテムを表します。タグを使用すると、製造環境からの特定のデータ アイテムとして、特定のコンポーネント プロパティをアクセス可能にできます。上の図では、**PumpState** タグはポンプがオンであるかオフであることを示しています。製造環境において、InTouch アプリケーションでプロパティを監視または制御するコンポーネントに対してタグを作成します。

製造コンポーネントから収集された異なるタイプのデータに対しては異なるタイプのタグを使用できます。たとえば、**PumpState** タグは、ポンプが実行中であるか停止中であることを示すブール型のオン/オフ値を返します。アプリケーションの一部とするデータ型に対しては、適切なタイプの InTouch タグを割り当てます。

Tag Viewer の概要

Tag Viewer はタグを監視し、ランタイム時にタグの値を変更できる外部アプリケーションです。**Tag Viewer** では、アラームグループに基づいて、アプリケーションで使用可能なすべてのタグの階層の一覧を表示できます。**Tag Viewer** は **WindowViewer** が実行しているときにのみ実行できます。**Tag Viewer** にはローカルの InTouch アプリケーションで使用可能なタグだけが表示されます。リモート リファレンスはサポートされていません。

Tag Viewer を使用するには、最初に **WindowMaker** で **Tag Viewer** を有効にしておく必要があります。その後、ランタイム時にアプリケーションを起動します。一度に開くことのできる **Tag Viewer** のインスタンスは 1 つだけです。

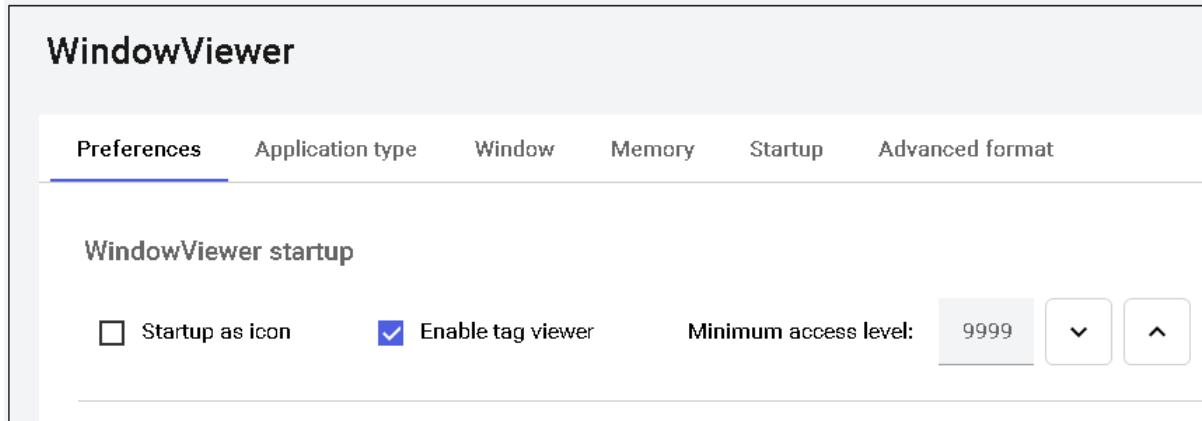
Tag Viewer の有効化

デザイン時に **WindowViewer** のプロパティを設定して **Tag Viewer** を有効にすることができます。また **WindowViewer** のメニューを設定して、**[システム]** メニューに **Tag Viewer** オプションを表示することもできます。

WindowViewer が開いている状態でこれらのプロパティを変更した場合、変更を適用するには **WindowViewer** を再起動する必要があります。

Tag Viewer を有効化するには

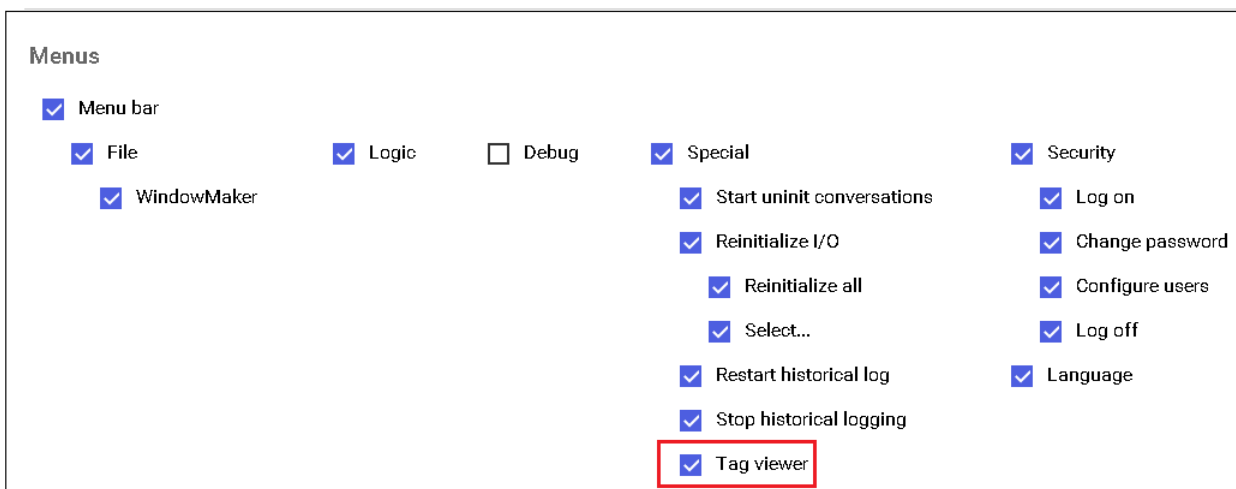
1. WindowMaker を開きます。
2. [ファイル] メニューで、[設定] をポイントし、[WindowViewer] をクリックします。
WindowViewer の設定画面が表示されます。
3. [システム設定] タブをクリックします。



4. [WindowViewer 起動時の設定] 領域で、[Tag Viewer の有効化] チェック ボックスをオンにして、Tag Viewer を起動できるようにします。デフォルトでは、このチェック ボックスはオンになっていません。
5. [最小アクセス レベル] ボックスで、Tag Viewer アプリケーションを実行するために必要なセキュリティ アクセス レベルを設定します。デフォルトの値は、9999 ですが、0 ～ 9999 の値を入力できます。

注記: ユーザーのアクセス レベルが最低限のアクセス レベルよりも低い場合、ランタイム時に Tag Viewer を起動することはできません。

6. [ウィンドウ] タブをクリックします。



1. [メニュー] 領域の [Tag Viewer] チェック ボックスをオンにして、WindowViewer の [システム] メニューの [Tag Viewer] オプションを有効にします。

2. [OK] をクリックします。
3. これらのパラメータのいずれかを変更した場合、変更を適用するには WindowViewer を再起動する必要があります。

Tag Viewer の起動

Tag Viewer は、WindowViewer が実行していて、デザイン時に WindowViewer が有効に設定されている場合にのみ開始することができます。Tag Viewer を起動するには、WindowViewer のメニューから選択するか、LaunchTagViewer() スクリプト関数を呼び出します。この関数はどのスクリプトタイプからでも実行することができます。ただし、OnStartup と OnShutdown は除きます。Tag Viewer が有効化されていない場合、スクリプト関数を呼び出しても Tag Viewer は起動せず、警告メニューがログに記録されます。

セキュリティ

Tag Viewer を実行するには、適切なセキュリティ権限が必要です。使用可能な InTouch ユーザー セキュリティ レベルを次の表に示します。

セキュリティ レベル	説明
Archestra	このオプションを選択した場合、以下の手順を実行する必要があります。 <ol style="list-style-type: none">1. [WindowViewer のプロパティ] 画面で [最小アクセス レベル] 以上のアクセス レベルを設定します。2. WindowViewer にログオンして、Tag Viewer を起動します。 この手順は、マネージド InTouch アプリケーションを実行している場合に実行します。
InTouch	このオプションを選択した場合、以下の手順を実行する必要があります。 <ol style="list-style-type: none">1. [WindowViewer のプロパティ] 画面で [最小アクセス レベル] 以上のアクセス レベルを設定します。2. WindowViewer にログオンして、Tag Viewer を起動します。 この手順は、スタンドアロンの InTouch アプリケーションを実行している場合に実行します。
なし	このオプションを選択した場合、ユーザーは WindowViewer にログオンせずにランタイム時に Tag Viewer を起動することができます。
OS	このオプションを選択した場合、以下の手順を実行する必要があります。 <ol style="list-style-type: none">1. [WindowViewer のプロパティ] 画面で [最小アクセス レベル] 以上のアクセス レベルを設定します。2. WindowViewer にログオンして、Tag Viewer を起動します。 この手順は、スタンドアロンの InTouch アプリケーションを実行している場合に実行します。

InTouch アプリケーションを開いている場合、Tag Viewer を実行するには、再度ログオンして再認証を受ける必要があります。ユーザーの設定の詳細については、『AVEVA™ InTouch HMI 管理ガイド』の「[ユーザーの管理と承認レベルの設定](#)」を参照してください。

注: InTouch のセキュリティ レベルが [なし] の場合、再度ログオンする必要はありません。

ログオンするには

1. WindowViewer を開きます。
2. [システム] メニューで [セキュリティ] をクリックして、[ログオン] をクリックします。
3. 有効なユーザー ID とパスワード、および必要最低限のアクセス レベルを使用してログオンします。

注: 管理者としてログオンするか、適切な権限を持つユーザー ID を作成することができます。

Tag Viewer を開始するには

1. WindowViewer を開きます。
2. [システム] メニューで、[Tag Viewer] をクリックします。Tag Viewer が表示されます。

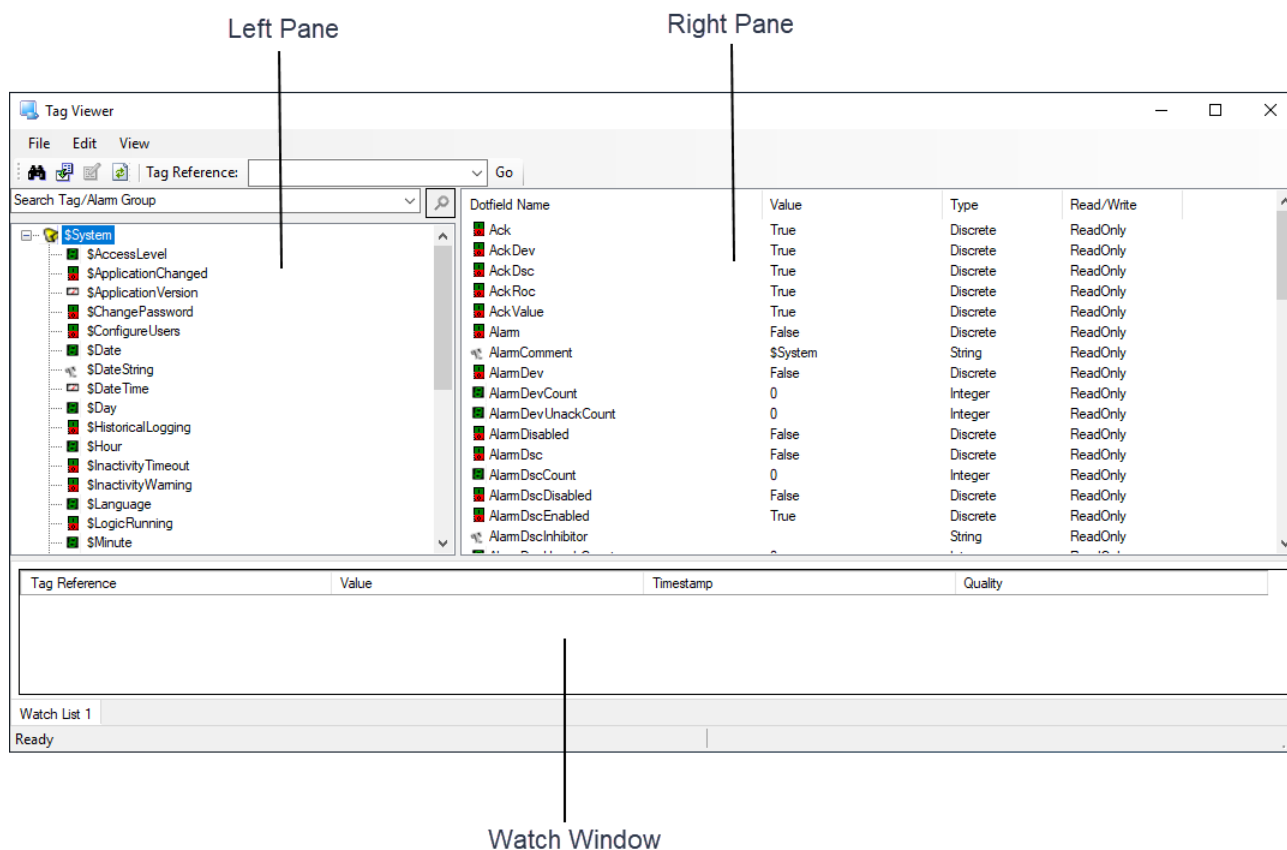
Tag Viewer をスクリプトで開始するには

1. WindowMaker を開きます。
2. 「On Startup」 および 「On Shutdown」 アプリケーション スクリプト以外の LaunchTagViewer() スクリプトを設定します。
3. WindowViewer を開きます。
4. スクリプトを実行して Tag Viewer を起動します。

Tag Viewer でのナビゲート

Tag Viewer ウィンドウは 3 つの部分で構成されています。

- 左ペインには、アラーム グループ別の階層形式でタグが表示されます。
- 右ペインには、左ペインで選択したタグまたはアラーム グループの使用可能なドットフィールドのリストが表示されます。
- 下部の監視ウィンドウには、監視するタグのランタイム時の値が表示されます。



Tag Viewer を閉じる

Tag Viewer は次の方法で閉じることができます。

- Tag Viewer を閉じる
- WindowViewer を閉じる
- ログオンしているユーザーを変更する

Tag Viewer の使用

Tag Viewer をランタイム時に使用して、以下のことを行うことができます。

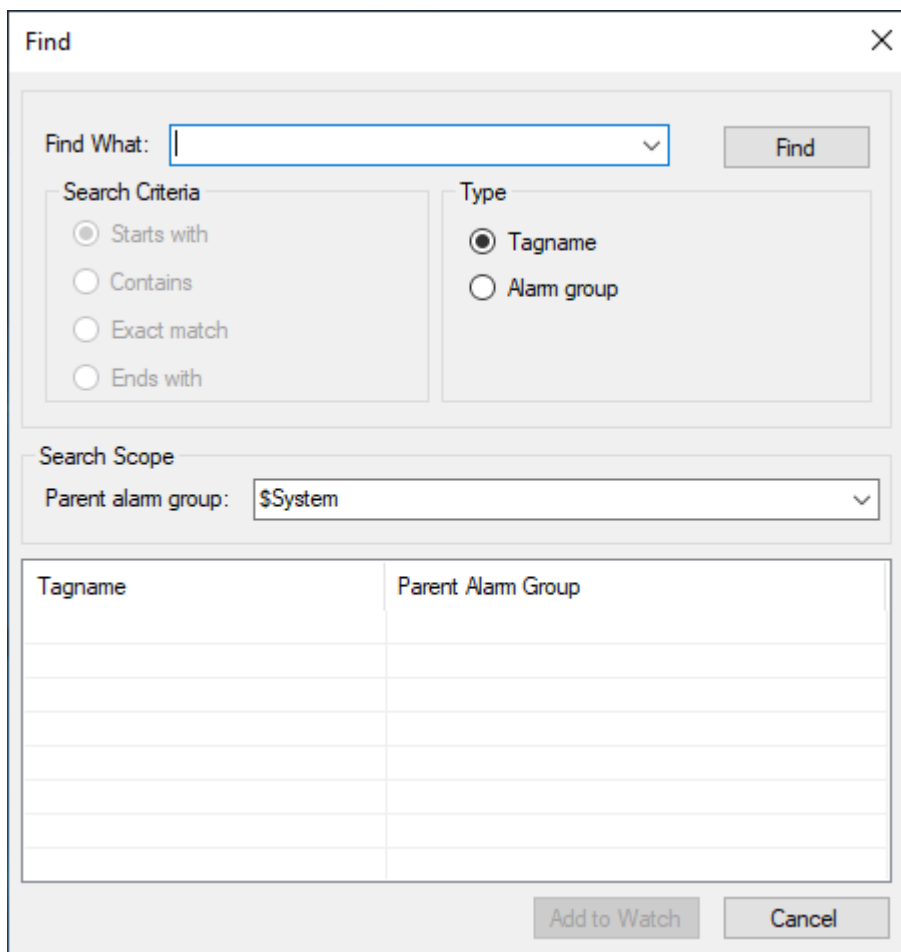
- タグの検索
- タグの表示
- タグのプロパティの変更
- 監視ウィンドウの追加と監視ウィンドウへの情報入力
- 監視ウィンドウの管理

タグの検索

WindowViewer で特定のタグを検索するには、**[検索]** ダイアログ ボックスを使用します。

タグを検索するには

1. [編集] メニューで、[タグを検索/アラーム グループ] をクリックします。[検索] ダイアログ ボックスが表示されます。



The image shows a 'Find' dialog box with a title bar containing a close button (X). Inside the dialog, there is a 'Find What:' text box followed by a dropdown arrow and a 'Find' button. Below this, there are two sections: 'Search Criteria' and 'Type'. The 'Search Criteria' section has four radio buttons: 'Starts with' (selected), 'Contains', 'Exact match', and 'Ends with'. The 'Type' section has two radio buttons: 'Tagname' (selected) and 'Alarm group'. Below these sections is a 'Search Scope' section with a label 'Parent alarm group:' and a dropdown menu showing '\$System'. At the bottom of the dialog is a table with two columns: 'Tagname' and 'Parent Alarm Group'. The table has several empty rows. At the very bottom of the dialog are two buttons: 'Add to Watch' and 'Cancel'.

Tagname	Parent Alarm Group

2. 検索するタグまたはアラーム グループの名前を [検索する文字列] ボックスに入力します。デフォルトでは、最後に検索したタグまたはアラーム グループの名前が表示されます。[検索する文字列] ボックスには検索履歴も保持されています。タグまたはアラーム グループを初めて検索する場合、[検索する文字列] ボックスは空白です。

注: 検索履歴は Tag Viewer を閉じると削除されます。

3. [検索条件] 領域で、以下のいずれかを選択します。
 - 次で開始: [検索する文字列] ボックスに入力したテキストで始まるタグまたはアラーム グループを検索する場合、このオプションを選択します。デフォルトでは、このオプションが選択されています。
 - 含む: [検索する文字列] ボックスに入力したテキストを含むタグまたはアラーム グループを検索する場合、このオプションを選択します。
 - 完全一致: [検索する文字列] ボックスに入力したテキストと一致するタグまたはアラーム グループを検索する場合、このオプションを選択します。

- 次で終了: [検索する文字列] ボックスに入力したテキストで終わるタグまたはアラーム グループを検索する場合、このオプションを選択します。
4. [タイプ] 領域で、以下のいずれかを選択します。
 - タグ名: タグを検索する場合、このオプションを選択します。デフォルトでは、このオプションが選択されています。
 - アラーム グループ: アラーム グループを検索する場合、このオプションを選択します。
 5. タグの所属するアラーム グループの名前を [親のアラーム グループ] ボックスに入力します。デフォルト値の **\$System** を保持する場合、フィルタされていない検索結果がアラーム グループ別に表示されます。
 6. [検索] をクリックします。タグまたはアラーム グループのリストが表示されます。
 7. 検索結果に表示されたタグの名前をダブルクリックすると、そのタグが **Tag Viewer** で選択されます。

注: タグを監視ウィンドウに追加するには、[監視に追加] をクリックします。

クイック検索を行う

必要に応じて、**Tag Viewer** ではタグをすばやく検索することができます。

クイック検索を行うには

1. 左ペインの上にある検索ボックスに、タグの完全な名前またはその一部を入力します。

注: **Tag Viewer** を閉じるまで、検索ボックスは検索履歴を保持します。



2. **Enter** キーを押すか、検索アイコンをクリックします。入力したテキストで始まるタグが選択されます。

注: 入力したテキストがワイルドカード (*) で始まる場合、テキストを含むすべてのタグが検索されます。

3. 検索アイコンをもう一度クリックして、検索を続行します。

注: クイック検索を行うには、テキストを入力してから **F3** キーを押します。

タグの管理

Tag Viewer では、タグの監視やランタイム時のタグの値の監視を行うことができます。アプリケーションで使用可能なすべてのタグを表示することができます。タグはアラーム グループ別にツリー形式で表示されます。

タグの表示

Tag Viewer でタグの詳細を表示することができます。

タグを表示するには

1. **Tag Viewer** を開きます。
2. 左ペインには、アプリケーションで使用可能なタグのリストが表示されます。

注: Tag Viewer には InTouch アプリケーションで使用可能なタグだけが表示されます。リモート リファレンスはサポートされません。

3. 左ペインでタグをクリックすると、右ペインにそのタグの詳細が表示されます。次の情報が表示されます。
 - ドットフィールド名: 選択したタグまたはアラーム グループのドットフィールドが表示されます。
 - 値: ドットフィールドの値を表示します。ドットフィールドをクリックすると、その値が更新されます。
 - タイプ: 整数型や論理型など、ドットフィールドのタイプを表示します。
 - 読み取り/書き込み: タグが読み取り専用または書き込み可能かどうかを示します。
4. ランタイム時にドットフィールド値を表示するには、[監視] ウィンドウにタグを追加します。[監視] ウィンドウへのタグの追加の詳細については、この付録の「[監視ウィンドウへのタグまたはドットフィールドの追加](#)」を参照してください。

タグのプロパティの変更

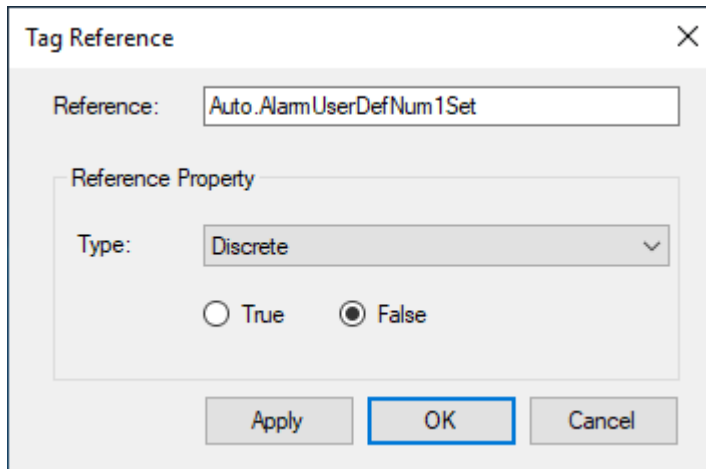
Tag Viewer でタグのプロパティを変更することができます。タグのプロパティは、ドットフィールドが [読み取り/書き込み] または [書き込み専用] としてマーク付けされている場合にのみ変更できます。WindowViewer を読み取り専用ライセンスで実行している場合、タグは変更できません。

以下のタグを変更しようとするすると再認証が求められます。

- セキュリティ書き込みの設定された Galaxy 属性をポイントしている間接型タグ
- 確認書き込みの設定された Galaxy 属性をポイントしている間接型タグこの場合、確認者であるユーザーのログオン認証も求められます。

タグのプロパティを変更するには

1. Tag Viewer を開きます。
2. 左ペインで、プロパティを変更するタグをクリックします。タグの詳細が右ペインに表示されます。
3. 以下のいずれかを実行します。
 - 右ペインでプロパティをクリックして、ツールバーの [Go] をクリックします。[タグ リファレンス] ダイアログ ボックスが表示されます。

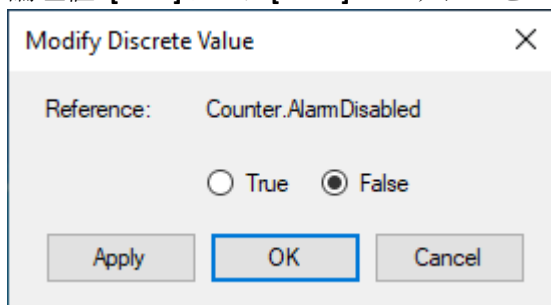


The image shows a 'Tag Reference' dialog box. It has a title bar with a close button (X). Inside, there is a 'Reference:' label followed by a text box containing 'Auto.AlarmUserDefNum1Set'. Below this is a 'Reference Property' section. It contains a 'Type:' label followed by a dropdown menu showing 'Discrete'. Underneath the dropdown are two radio buttons: 'True' and 'False', with 'False' being selected. At the bottom of the dialog are three buttons: 'Apply', 'OK', and 'Cancel'.

- [タイプ] リストで、タグのタイプを選択して、値を変更します。

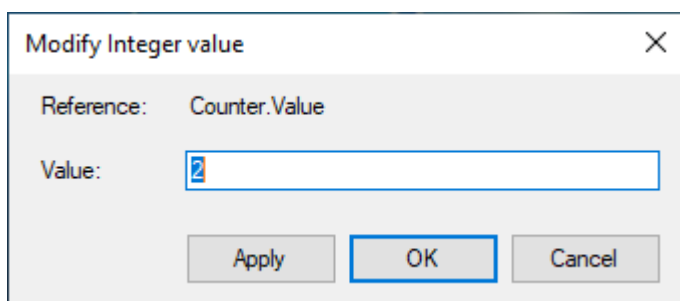
ドットフィールドのデータ タイプを変更する場合は、右ペインの値をダブルクリックします。
[変更] ダイアログ ボックスが表示されます。ドットフィールドのデータ タイプによって、表示されるダイアログ ボックスが異なります。以下のデータ タイプの値を変更することができます。

論理値: [True]または[False]のいずれかをクリックできます。



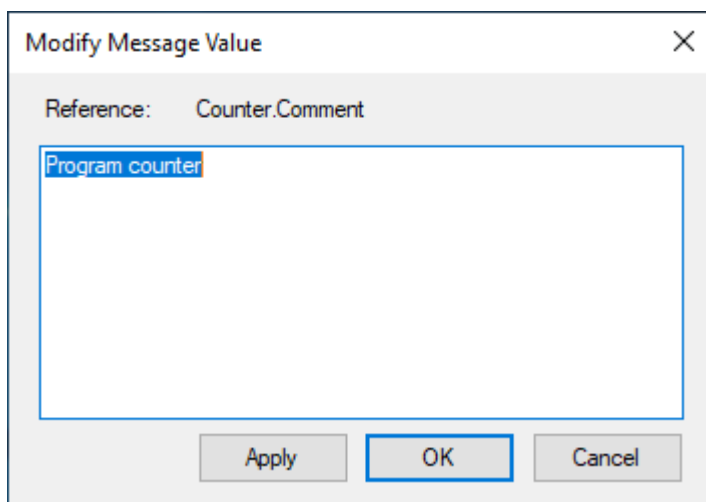
The dialog box is titled "Modify Discrete Value" with a close button (X) in the top right corner. It contains a "Reference:" label followed by the text "Counter.AlarmDisabled". Below this, there are two radio buttons: "True" and "False". The "False" radio button is selected, indicated by a filled circle. At the bottom, there are three buttons: "Apply", "OK", and "Cancel". The "OK" button is highlighted with a blue border.

数値: -2147483648 から 2147483647 の間の任意の数値を入力できます。



The dialog box is titled "Modify Integer value" with a close button (X) in the top right corner. It contains a "Reference:" label followed by the text "Counter.Value". Below this, there is a "Value:" label followed by a text input field containing the number "2". At the bottom, there are three buttons: "Apply", "OK", and "Cancel". The "OK" button is highlighted with a blue border.

メッセージ: 最大 131 文字のテキスト メッセージを入力できます。



The dialog box is titled "Modify Message Value" with a close button (X) in the top right corner. It contains a "Reference:" label followed by the text "Counter.Comment". Below this, there is a large text input field containing the text "Program counter". At the bottom, there are three buttons: "Apply", "OK", and "Cancel". The "OK" button is highlighted with a blue border.

- アイテムを右クリックし、[変更] をクリックします。[変更] ダイアログ ボックスが表示されます。値を変更します。

4. [適用] をクリックします。値が変更されます。

監視ウィンドウの管理

デフォルトでは、監視ウィンドウは **Tag Viewer** の下部に表示されます。このウィンドウを使用して、ランタイム時にタグの値を表示することができます。追加の監視ウィンドウを作成して、関連性のあるタグをグループ化できます。必要に応じて、監視ウィンドウを削除することもできます。

監視ウィンドウの追加

追加の監視ウィンドウを作成して、グループ化したタグを監視することができます。新規に作成した監視ウィンドウは、デフォルトの監視ウィンドウの横にタブとして追加されます。監視ウィンドウは最大 50 個まで作成できます。

監視ウィンドウを追加するには

1. 監視ウィンドウを右クリックして、**[監視ウィンドウを追加]** をクリックします。新しい監視ウィンドウが追加されます。デフォルトでは、このウィンドウには「監視リスト <n>」という名前が付けられます。

注: 監視ウィンドウの名前を変更するには、監視ウィンドウのタブを右クリックして **[タブ名を変更]** をクリックします。

2. 必要に応じて、タグを監視ウィンドウに追加します。
3. タグを別の監視ウィンドウに移動するには、移動元と移動先のタブをクリックします。

監視ウィンドウへのタグまたはドットフィールドの追加

タグまたはドットフィールドを監視ウィンドウに追加して、ランタイム時の値を監視することができます。1 つの監視ウィンドウには、最大 2000 個のタグまたはドットフィールドを追加できます。

注: アラーム グループを監視ウィンドウに追加した場合、そのアラーム グループに属するすべてのタグが監視リストに追加されます。

タグを監視ウィンドウに追加するには

1. 以下のいずれかを実行します。
 - 左ペインでタグを右クリックして、**[監視に追加]** をクリックします。タグが監視ウィンドウに追加されます。
 - 左ペインでタグをクリックしてドラッグし、監視ウィンドウにドロップします。
 - 監視ウィンドウを右クリックして、**[タグ リファレンスを追加]** をクリックします。
 - 右ペインでドットフィールドを右クリックして、**[監視に追加]** をクリックします。ドットフィールドが監視ウィンドウに追加されます。

注: 複数のタグまたはドットフィールドを選択して、監視ウィンドウに追加することができます。

2. 昇順または降順でテーブルをソートするには、カラム ヘッダーをクリックします。

グループ タグへのセパレータの追加

複数のタグの後にセパレータ（区切り文字）を追加して、同類のタグをグループ化することができます。監視ウィンドウを保存すると、監視ウィンドウのセパレータも自動的に保存されます。

グループ化したタグにセパレータを追加するには

1. 監視ウィンドウでセパレータを追加する行をクリックします。

2. 監視ウィンドウを右クリックして、**[セパレータを追加]** をクリックします。選択した行の下にセパレータが追加されます。
3. タグ リファレンスと一緒にセパレータをドラッグして別の場所にドロップすると、監視ウィンドウに表示されるタグの順番を変更できます。

注: 監視ウィンドウの情報をソートした場合、セパレータは削除されます。

監視ウィンドウのバックアップとリストア

監視ウィンドウをバックアップとして保存しておき、必要に応じて再度読み込むことができます。

監視ウィンドウの保存

必要に応じて、開いているすべての監視ウィンドウを保存することができます。これらの監視ウィンドウを後で読み込んでタグを監視することができます。

監視ウィンドウを保存するには

- **[ファイル]** メニューの **[監視リストを上書き保存]** をクリックして、監視ウィンドウを保存します。デフォルトでは、監視ウィンドウを現在のアプリケーションディレクトリに **監視リストフォルダ** に保存します。

注: 監視ウィンドウを別の場所に保存するには、**[ファイル]** メニューの **[監視リストに名前を付けて保存]** をクリックします。

監視ウィンドウを読み込む

保存した監視ウィンドウを開くことができます。

監視ウィンドウを読み込むには

1. **[ファイル]** メニューで、**[監視リストをロード]** をクリックします。**[ファイルの選択]** ダイアログボックスが表示されます。

注: 監視ウィンドウを右クリックして、**[監視リストをロード]** をクリックすることもできます。

2. ファイルの保存されている場所を参照して、ファイルを開きます。

注: 監視ウィンドウを開くと、既存のウィンドウが新しいウィンドウに置き換えられます。現在のアプリケーションで有効なリファレンスのみが、監視ウィンドウに追加されます。

I/O リファレンス

Tag Viewer に表示される I/O リファレンスまたは間接型タグは、WindowViewer のウィンドウのように動作します。

- I/O または間接型タグを監視ウィンドウで削除した場合、I/O リファレンスは要求されません。
- I/O または間接型タグを監視ウィンドウに追加していない状態で、ドットフィールドが更新された場合、または左ペインのタグをクリックした場合は、Tag Viewer が I/O リファレンスをサブスクライブします。そのタグが選択されている間は、サブスクライブが持続します。
- Tag Viewer を閉じた場合、Tag Viewer が要求した I/O または間接型タグのサブスクライブが解除されます。
- Tag Viewer を閉じた場合や監視ウィンドウでタグを削除した場合、WindowViewer のタグには影響しません。

Tag Viewer の持続性

次の点において、Tag Viewer には InTouch アプリケーションの持続性があります。

- メイン ウィンドウのサイズと場所
- 左ペインに表示されるタグのツリー構造のサイズ、左ペインに表示されるドットフィールドのリスト、および監視ウィンドウ

次の点において、Tag Viewer には WindowViewer セッションの持続性があります。

- 左ペインで選択したタグ
- 監視ウィンドウ
- ステータス バーおよびツール バー

注: WindowViewer を閉じると、WindowViewer の持続性は失われます。

監視ウィンドウのファイル形式

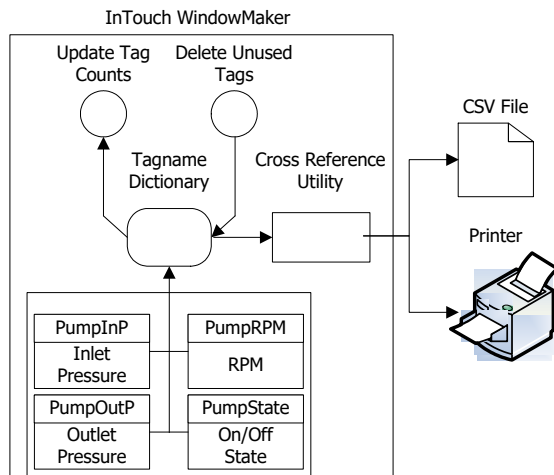
保存した監視ウィンドウのファイルは、XML ファイル形式で保存されます。XML スキーマを以下に示します。

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="InTouchTagViewer">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Watch" minOccurs="1" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="ReferenceString" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/></xs:element>
              <xs:element name="Separator" minOccurs="0" maxOccurs="unbounded"/></
xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

タグの使用数の削減

InTouch アプリケーションで使用可能なタグの最大数は、所有するライセンスによって異なります。実行中のアプリケーションで最大制限を超える数のタグを同時に使用することはできません。

InTouch アプリケーションでは、タグの使用数をトラッキングできます。以下の図は、クロス リファレンス ユーティリティがタグ名ディクショナリの内容を分析して、タグ使用数レポートを生成する方法を示しています。



InTouch アプリケーションを実行するためには、最低限のタグを保持している必要があります。タグの数を最小限にする方法の 1 つは、未使用のタグを削除することです。未使用のタグを削除する前に、タグの数を更新する必要があります。

InTouch と Historian

Historian は、工場のプロセス データを保存するリアルタイムな履歴データベースです。リレーショナルデータベースの記憶容量と、リアルタイム システムのスピードを兼ね備えています。Historian は、Microsoft SQL Server の機能強化として工場のデータ取得を大幅に高速化します。また、InTouch HMI およびその他の関連製品のイベント、サマリ、生産、および履歴データが工場のデータと統合されます。

InTouch HMI では、履歴タグ変数データをリモートの履歴リポジトリに保存できます。この場合、InTouch HMI アプリケーションディレクトリを共有し、InTouch アプリケーションにアクセスするよう Historian を設定する必要があります。Historian を使用して InTouch HMI 履歴データを保存する場合、WindowMaker から分散名前マネージャを使用して、データベースへの接続を指定する必要があります。InTouch HMI では、Historian データベースに保存された履歴データをトレンドウィザードに表示できます。履歴タグ変数データにアクセスするには、SQL Server クライアント コンポーネントをインストールし、Historian コンピュータを参照する必要があります。

タグ変数の使用数の確認

InTouch HMI は、ローカル タグ変数ディクショナリで定義されているすべてのタグ変数の使用数を管理します。タグ変数の数に、内部システム タグ変数は含まれません。

リモート タグは、タグ変数ディクショナリでは定義しません。InTouch では、アプリケーション内でリモート タグが参照されるたびにタグ変数の数が増えます。InTouch でのリモート タグ変数リファレンスのカウント方法の詳細については、「[ライセンスに基づくリモート タグの最大数の確認](#)」を参照してください。

未使用のタグ変数を削除する前に、以下のタスクを完了してください。

- WindowViewer を閉じます。
- ローカル タグ変数およびリモート タグへの参照の数を検索します。
- クロス リファレンスのタグ変数レポートを生成します。
- クロス リファレンス ユーティリティでアプリケーション内のタグ変数の使用数を調べます。

- アプリケーションを WindowMaker に保存します。

タグの数の確認

アプリケーションのタグ名ディクショナリで現在定義されているローカル タグの数を更新できます。また、別のノードに置かれたリモート タグに対するアプリケーションの参照数も検索可能です。

ローカル タグの数を表示するには

1. WindowMaker で InTouch アプリケーションを開きます。
2. [ファイル] メニューの [設定] をクリックし、[WindowMaker] をクリックします。

WindowMaker の設定画面が表示されます。

3. [タグの数の表示] を選択します。

このオプションを選択すると、タグ名ディクショナリ全体が読み込まれ、表示されているタグの数が更新されます。タグ名ディクショナリの更新には時間がかかることがあります。

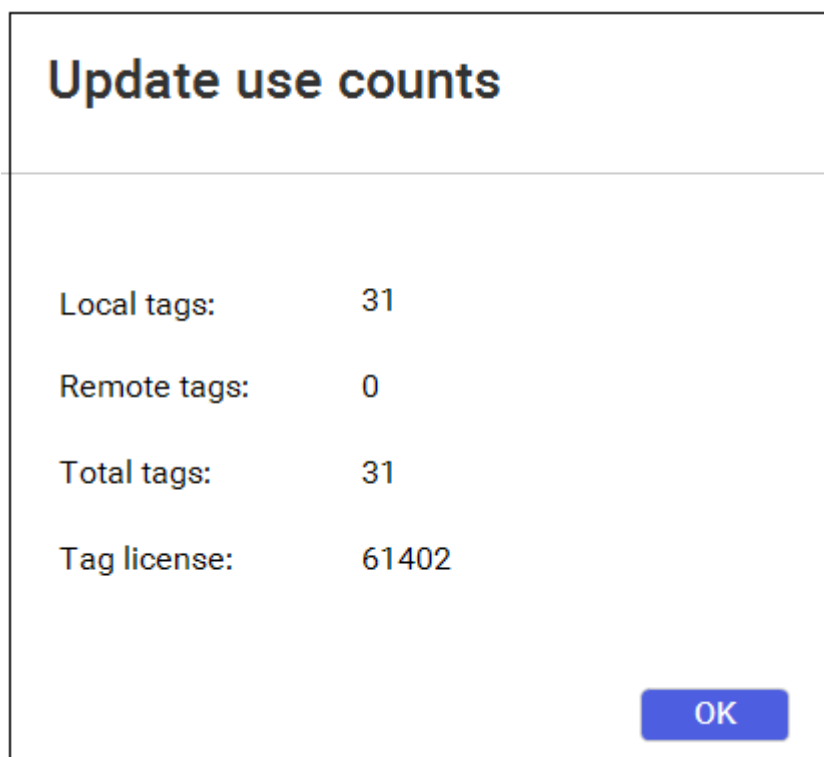
4. [保存] をクリックします。設定の変更を有効にするために WindowMaker を再起動するようメッセージが表示されます。
5. WindowMaker を閉じます。
6. WindowMaker でアプリケーションを再起動します。

アプリケーションのタグ名ディクショナリで定義されているローカル タグの数が、WindowMaker のメニューバーの右側に表示されます。

アプリケーション内でリモート参照を含むタグの合計数を更新するには

1. WindowMaker で InTouch アプリケーションを開きます。
2. 開いているすべてのウィンドウを閉じます。
3. [ホーム] メニューの [タグ] グループで [使用カウントを更新] をクリックします。

計算が終了すると、ダイアログ ボックスにアプリケーションのローカル タグおよびリモート タグへの参照の数が表示されます。



このダイアログ ボックスには、タグの合計数と InTouch のライセンスに基づいたタグの最大使用数も表示されます。

ライセンスに基づくリモート タグの最大数の確認

製品のライセンスによって、InTouch アプリケーションで利用できるタグ変数の数が決定されます。ローカルタグ変数の数だけでなく、リモート ノードにあるタグ ソースを参照するタグ変数の数もライセンスによって施行されます。

アプリケーションで利用できるリモート リファレンスのタグ変数の数が InTouch ライセンスによってどのように施行されるのかについては、「InTouch ライセンス」を参照してください。

タグの使用箇所の特定

クロス リファレンス ユーティリティを使用すると、InTouch アプリケーション内のタグの使用数を示すオンライン レポートを生成できます。

クロス リファレンス ユーティリティのレポートには、次の中に含まれるローカル タグ、リモート タグ、スーパータグが表示されます。

- アニメーション リンク
- ウィザード
- すべてのタイプの InTouch スクリプト
- クイック関数
- ActiveX コントロール
- 最適な InTouch コンポーネント (SQL アクセス マネージャやレシピ マネージャなど)
- 産業用グラフィック参照

1. WindowMaker で InTouch アプリケーションを開きます。
2. [ホーム] メニューの [タグ] グループで [クロスリファレンス] をクリックします。

クロスリファレンスグリッドには、現在のアプリケーションのタグ名ディクショナリで定義されているローカルおよびリモートタグに関連するすべてのレコードが一覧表示されます。

グリッド形式では、レポートには、参照タイプの名前、タグリファレンスのタイプ、使用場所、ウィンドウ上の位置、ウィンドウ名、グラフィック名、タグリファレンスのある産業用グラフィックの階層のフルパス、および条件タイプ（スクリプトでタグが使用されている場合）が一覧表示されます。

以下に例を示します。


レポートの下部にあるステータス バーには、以下の内容が表示されます。

- **レコード数:** アプリケーション内のタグに関連するレコードの合計数。この値は、適用したフィルタに基づいて動的に変更されます。
- **ローカル タグ:** このノード上で作成されたタグの合計数。
- **リモート タグ:** リモート タグを参照するタグの合計数。
- **タグの合計数:** InTouch アプリケーション内のタグの合計数。
- **タグ ライセンス:** ライセンスごとに使用可能なタグの合計数。

InTouch のクロス リファレンス ユーティリティのレポートには、ステータスおよび使用数を示すアイコンが表示されます。

アイコン	説明
	タグまたはスーパータグはアプリケーションのタグ名ディクショナリで定義されていますが、オブジェクトには割り当てられていません。
	タグまたはスーパータグは、アニメーション リンク、InTouch QuickScript、または産業用グラフィックで使用されています。
	産業用グラフィックは、タグまたはスーパータグを参照しています。
	タグまたはスーパータグはアニメーション リンクに割り当てられています。
	タグまたはスーパータグは、アプリケーション スクリプトで使用されています。
	選択されているすべてのスクリプトに表示されます。スクリプトを読み取り専用モードで表示するには、スクリプト名をダブルクリックします。
	タグまたはスーパータグはウィンドウ スクリプトで使用されています。
	タグまたはスーパータグはデータ変更スクリプトで使用されています。
	タグまたはスーパータグは条件スクリプトで使用されています。
	タグまたはスーパータグはキー スクリプトで使用されています。
	タグまたはスーパータグはクイック関数で使用されています。
	タグまたはスーパータグは、ActiveX イベント スクリプトで使用されています。
	ウィンドウによるクロス リファレンスを行うと、表示されているタグまたはスーパータグが使用されているウィンドウ名の前にこのアイコンが付きます。アイコンをダブルクリックすると、このウィンドウで使用されているすべてのタグが表示されます。
	表示されているタグまたはスーパータグは SQL アプリケーションで使用されています。
	表示されているタグまたはスーパータグはレシピ マネージャ アプリケーションで使用されています。

アイコン	説明
------	----

- | | |
|---|----------------------|
|  | タグは、アラーム抑止に使用されています。 |
|---|----------------------|
-

グラフィック ツールボックスからのグラフィックの包含

- [更新] をクリックして、最新のタグ情報を表示します。クロス リファレンス ウィンドウが開いているときにタグ変数情報を編集した場合、またはウィンドウに埋め込まれたシンボルを編集した場合、ウィンドウ [更新] ボタンが有効になります。
- [グラフィック ツールボックスのすべてのグラフィックを含める] チェックボックスを選択すると、グラフィック ツールボックスに存在していてもいずれのウィンドウでも使用されていないタグが表示されます。デフォルトでは、このチェックボックスは選択されていません。

以下に例を示します。タグディクショナリに 10 個のタグが含まれていて、4 つのタグがグラフィック ツールボックスのグラフィックで参照されているとします（ウィンドウには埋め込まれていません）。その他の 4 つのタグは 1 つのウィンドウに配置されたグラフィックで参照されているとします。デフォルトでは、4 つのタグ（ウィンドウ上に配置されたタグ）が表示されます。チェックボックスを選択すると、8 つのタグ（ウィンドウの 4 つのタグ + グラフィックの 4 つのタグ）が表示されます。

表示されるグラフィックには、以下の条件が適用されます。

- InTouch ウィンドウで使用されていないシンボル内のリモート タグ リファレンスは表示されません。
- InTouch ウィンドウで使用されていないオブジェクト シンボルは表示されません。
- このチェックボックスがクロス リファレンス ユーティリティで選択されている場合、未使用タグ変数の検索や未使用タグの削除などのその他の操作にも同じ条件が設定されます。

Archestra または Situational Awareness Library シンボルでの InTouch タグのクロス リファレンス タグの使用

InTouch クロス リファレンス ユーティリティが強化され、産業用グラフィックがアプリケーションの標準の InTouch タグ クロス リファレンスのリストに含まれるようになりました。以下に示す条件を使用して産業用グラフィックのクロス リファレンス検索を実行できます。これらの条件は、クロス リファレンス ユーティリティの InTouch タグの使用の一部として含まれます。

- InTouch タグへの参照を含むカスタム プロパティ
- InTouch タグへの直接参照を含むアニメーション
- InTouch タグへの直接参照を含むクライアント スクリプト
- InTouch タグへの埋め込まれたシンボルの参照

クロス リファレンス ユーティリティでは、オブジェクト属性の一部である属性参照を検索する際に産業用グラフィックも検索されます。タグ使用数レポートの例外を以下に示します。

1. オブジェクト属性参照:

- 相対参照 (me.l1 など) を使用するオブジェクト属性はサポートされず、クロス リファレンス ユーティリティ タグ使用数レポートに表示されません。

- <InTouchViewApp instance name>.<Tagname> の構文を使用する参照は InTouch タグ リファレンス と見なされません。このリファレンスはオブジェクト属性とみなされます。

例: InTouchViewApp1.Tag1

- 「InTouch:<Tag>」の構文ではないリファレンスはオブジェクト属性参照と見なされます。

2. スクリプト関数パラメータとして使用されるクライアント スクリプト リファレンスは InTouch タグ およびオブジェクト属性参照とは見なされません。

例: 文字列パラメータとしてのリファレンス

```
SetCustomPropertyValue( "CP1", "InTouch:Tag1", false);
```

例: 属性パラメータとしてのリファレンス

```
SetBad(InTouch:Tag1);
```

3. 現在の InTouch アプリケーションで定義されていない InTouch タグを参照する産業用グラフィック は「未定義」タイプとして表示されます。

例: オブジェクト シンボル

ArchestrA オブジェクト "UD_001" にはインスタンス シンボル "S1"、"S2"、および "S3" があります。インスタンス シンボル "UD_001.S1" には InTouch タグ "Tag51" への参照があります。

UD_001.S1 を UD_001.S2 に埋め込みます。

UD_001.S2 を UD_001.S3 に埋め込みます。

UD_001.S3 を InTouch ウィンドウ "Win1" に埋め込みます。


また、UD_001.S2 をウィンドウ "Win1" に埋め込みます。

インスタンス シンボルは、インスタンス内またはその派生テンプレート内で定義できます。インスタンスは、絶対または相対属性を使用して埋め込むことができます。







クロス リファレンス ユーティリティでのタグ使用数の表示

フィルタ、ソート、およびグループ化オプションを使用して、クロス リファレンス タグ使用数レポートの表示を変更できます。フィルタを適用すると、表示されるレコードの数が更新されます。ソートおよびグループ化では、グリッドでのレコードの表示方法だけが変更されます。

フィルタ オプションを使用したタグ使用数レポートの絞り込み

1. [含む]、[含まない]、[次で開始]、[次で終了]、[等しい]、[等しくない]、[Is Null]、[IsNot Null]、および [カスタム] などのオプションのリストからフィルタ アイコン () を使用します。
2. 選択したフィルタ オプションに対する検索条件を入力します。

フィルタ オプションおよび検索条件に一致するすべてのレコードがグリッドに表示されます。

Canvas		Cross reference	
	Name ▲	Type	
	Contains: Operator	Contains:	Contains: Cont
	\$Operator	InTouch tag	
	\$OperatorDomain	InTouch tag	
	\$OperatorDomainEnt...	InTouch tag	
	\$OperatorEntered	InTouch tag	
	\$OperatorName	InTouch tag	

カスタム フィルタ オプションの使用

カスタム フィルタ オプションを使用すると、複数のタグを追加して複雑な検索式を作成できます。

RadGridView Filter Dialog [Name] ×

Show rows where:

⌵
⋮
All
of the following are true
×

⋮
Name
Contains

×

⋮
Name
Does not contain

×

Add ▼

OK

Cancel

タグ使用数レポートのソート

タグ使用数レポートの各列は、昇順または降順でソートできます。

1. 列見出しをクリックします。

グリッド内のすべてのレコードが、その列を基準にソートされます。矢印の向きはソート順を示します。

2. 列見出しを再度クリックすると、ソート順が変更されます。

タグ使用数レポートのグループ化

列の組み合わせを使用してタグ使用数レポートをパターンで整理します。

1. 列名を選択して、列名の上の空白スペースにドロップします。
列の名前を含むボックスが表示されます。
2. 別の列名を選択して、最初のボックスの上または下にドロップします。
2つの列の間に階層関係が自動的に作成されます。
3. 同じレベルに2つの列を作成するには、2番目の列名を最初のボックスにドロップします。
4. 個々の列を昇順または降順にソートするには、パターン内の列名ボックスをクリックします。

Canvas Cross reference

Group by: Name ▲ ×

Window name ▲ ×

Position ▲ ×

	Name ▲	Graphic name	Type
▶	No filter: ▼	Contains: ▼	Contains: ▼
▼	Name: \$AccessLevel		
▼	Name: \$ApplicationChanged		
▼	Name: \$ApplicationVersion		
▼	Name: \$ChangePassword		
▼	Name: \$ConfigureUsers		

タグのクロス リファレンス リストの保存と印刷

クロス リファレンス タグのリストをファイルに保存し、.csv ファイル形式をサポートしているアプリケーションで表示できます。

クロス リファレンス タグのリスト ファイルには、名前ごとのすべてのタグ、アプリケーション内でのタグの使用方法、タグの使用位置のウィンドウ名が一覧表示されます。クロス リファレンス タグのリスト ファイルは、Excel または .csv 形式ファイルをサポートしているその他のプログラムで開くことができます。

また、タグ名ディクショナリの内容を印刷することも可能です。タグ名ディクショナリの内容を印刷すると、アプリケーションで使用されているデータベース エントリを参照できます。表示される詳細のレベルを指定できます。

このレポートは、プリンタに送信したり、ファイルに保存したりできます。

クロス リファレンス ファイルを保存するには

1. [クロス リファレンス ユーティリティ] ダイアログ ボックスの [名前を付けて保存] をクリックします。[名前を付けて保存] ダイアログ ボックスが表示されます。
ファイルの名前と場所を指定します。
2. [保存] をクリックします。タグの使用数を含むファイルが指定のフォルダに保存されます。

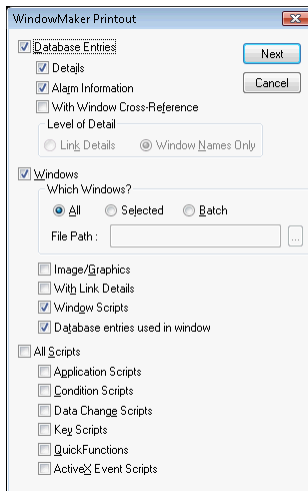
.csv ファイルに保存されるタグ使用数レポートは、以下の条件に基づきます。

- 検索条件を選択しない場合、すべてのレコードが .csv ファイルに保存されます。たとえば、クロスリファレンス グリッド内のレコードの総数が **1000** で、検索を絞り込むとレコード数は **320** に削減されます。[名前を付けて保存] を使用すると、選択された **320** のレコードだけが .csv ファイルに保存されます。
- グループ化パターンをレコードに適用した場合、.csv は、そのグループ化パターンで保存されます。

タグ名ディクショナリの内容を印刷するには

- WindowMaker で InTouch アプリケーションを開きます。
- クイック アクセス ツールバーの **[印刷]** をクリックします。

[アプリケーション情報の印刷] ダイアログ ボックスが表示されます。



- タグ名ディクショナリからタグの情報を印刷する場合は、**[データベース エントリ]** を選択します。

[データベース エントリ] を選択すると、以下のオプションが選択できるようになります。

- [詳細]** を選択すると、タグの詳細が印刷されます。
- [アラーム設定]** を選択すると、タグのアラーム設定情報が印刷されます。
- [ウィンドウとの関連]** を選択すると、ウィンドウのクロス リファレンスを持つタグ名ディクショナリの内容がすべて印刷されます。このオプションを選択した場合は、印刷する詳細レベルを指定します。

[リンク詳細] を選択すると、タグが使用されている場所とアニメーション リンクの詳細が印刷されます。

[ウィンドウ名のみ] を選択すると、クロス リファレンス先のウィンドウの名前だけが印刷されます。

- [次へ]** をクリックします。**[出力の送信先の選択]** ダイアログ ボックスが表示されます。
- タグ名ディクショナリの内容を印刷するか、または出力結果をテキスト ファイルや .html ファイルに送信するかのオプションを選択します。
- [印刷]** をクリックします。

クロス リファレンス タグのリストをファイルに保存し、.csv ファイル形式をサポートしているアプリケーションで表示できます。

未使用タグの削除

使用回数を更新した後で、InTouch アプリケーションから未使用のタグを削除できます。タグ名ディクショナリから一度に 1 個のタグを削除するか、または一度に複数のタグを削除することが可能です。

タグを含むウィンドウがアプリケーションから削除されたり、またはリンク スクリプトでタグが変更された場合でも、タグの数は自動更新されません。InTouch はタグが使用中であると見なすため、削除されないようにします。

注記: タグ名ディクショナリからの単一のタグの削除: 産業用グラフィック参照で **【削除】** ボタンが無効化されている場合、**【使用カウントを更新】** を実行して **【削除】** ボタンを有効にします。

未使用のタグを削除する前に、タグの数を更新して、合計数から未使用のタグを取り除く必要があります。タグの数の更新については、「[タグの数の確認](#)」を参照してください。

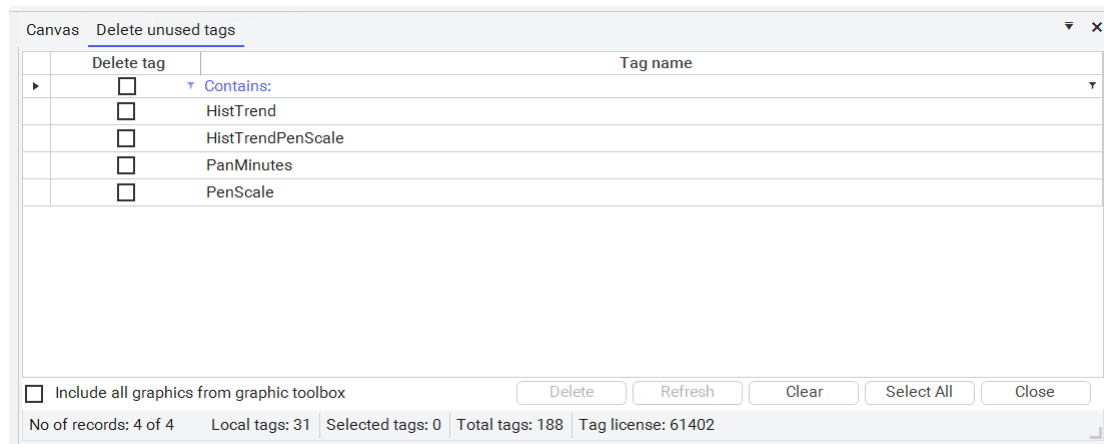
注意: アラーム状態になっているだけのタグは使用回数に含まれないため、誤って削除される可能性があります。このようなタグを使用回数に含めるには、ウィンドウまたは QuickScript でこれらのタグを使用する必要があります。


複数の未使用のタグを削除するには

1. WindowViewer が起動されている場合は閉じます。
2. WindowMaker で InTouch アプリケーションを開きます。
3. **【ホーム】** メニューの **【タグ】** グループで **【未使用タグの削除】** をクリックします。

未使用のタグが一覧表示された **【未使用タグの削除】** ダイアログ ボックスが表示されます。

ダイアログ ボックスの下部にあるステータス バーには、レコードの数、ローカルタグの数、選択されたタグの数、タグの合計数、およびタグ ライセンス カウントが一覧表示されます。



4. フィルタ () オプションを使用して、未使用タグのリストを絞り込みます。フィルタ オプションを選択して、検索条件を入力します。
条件に一致するタグが表示されます。
5. 列をソートするには、列見出しをクリックします。
6. 削除するタグを選択します。現在のフィルタ条件に一致するすべてのタグを選択するには、**【すべて選択】** をクリックします。

注記: ステータス バーには、選択されたタグの数（「レコード数: 1/14」など）が表示されます。タグの選択は、列の条件が変更された場合でも維持されます。「選択したタグ」の値には、ユーザーが選択したタグの合計数が反映されます。フィルタ条件には依存しません。選択されたタグの完全なリストを表示するには、すべてのフィルタ条件をクリアします。

7. **[クリア]** をクリックして、現在のフィルタのタグ選択をクリアします。
8. **[グラフィック ツールボックスのすべてのグラフィックを含める]** チェック ボックスをクリックすると、いずれのグラフィックまたはウィンドウでも使用されていないタグだけが表示されます。これらのタグはグラフィックやグラフィックを含むウィンドウで参照されていないので、安全に削除できます。

たとえば、タグディクショナリに 10 個のタグが含まれていて、4 つのタグがグラフィック ツールボックスのグラフィックで参照されているとします（ウィンドウには埋め込まれていません）。その他の 4 つのタグはウィンドウに配置されたグラフィックで参照されているとします。デフォルトでは、6 つのタグ（2 つの未使用タグ + ウィンドウに埋め込まれていないグラフィック ツールボックスのグラフィックの 4 つのタグ）が表示されます。チェック ボックスをオンにすると、未使用の 2 つのタグだけが表示されます。
9. **[削除]** をクリックして、すべてのタグ選択を削除します。フィルタ条件には依存しません。

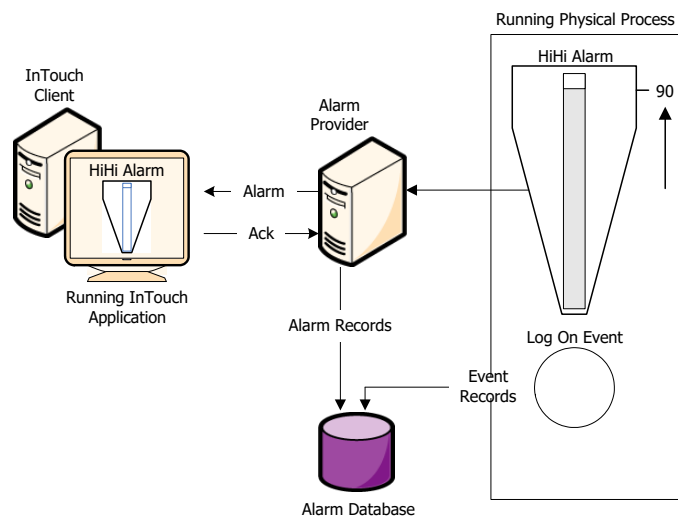
確認メッセージが表示されます。メッセージには、削除されるタグの数が示されます。
10. **[OK]** をクリックして確定します。

章 22 アラーム

アラームとイベントを生成してオペレータにプロセスの動作状況を通知する InTouch アプリケーションを作成できます。

- 問題を引き起こす可能性のあるプロセス状態に関する警告が、アラームによってランタイム オペレータに警告されます。通常は、プロセス値が定義済みのしきい値を超えた場合にトリガするようにアラームを設定します。オペレータは、アラームを確認する必要があります。
- イベントは、通常のシステム ステータス メッセージを表します。オペレータが InTouch アプリケーションにログオンするなど、システム条件が発生するときに典型的なイベントとなります。オペレータは、イベントを確認する必要はありません。

以下の図は、アプリケーションが実行されている間、InTouch HMI でアラームとイベントが処理される方法を示しています。アラームとイベントのデータは、アラーム データベースに保存されます。



イベント監視用のタグを設定できます。イベントメッセージは、タグの値が変更するたびにアラーム システムにログ記録されます。イベントメッセージには、値がどのように変更されたかに加えて、オペレータ、I/O、スクリプト、またはシステムのいずれによってその変更が開始されたのかという情報が含まれます。

現在のアラームの表示

InTouch Alarm Viewer ActiveX コントロールを使用して、アラームを表示します。**Alarm Viewer** コントロールには、スクロールバー、サイズ変更可能なカラム、複数アラーム選択、更新ステータス バー、動的表示タイプ、アラームのタイプに基づいたカラー表示などの機能があります。

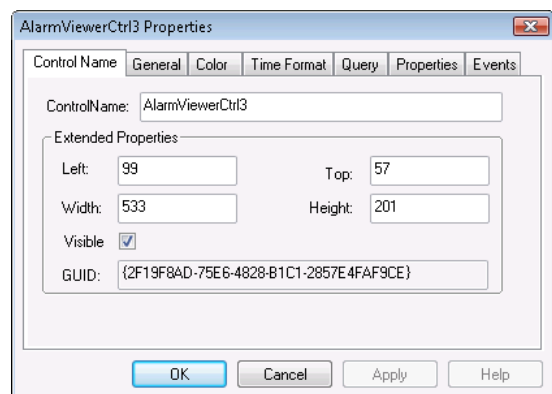
Time /	State	Class	Type	Priority
12/14/2006 10:15:11 AM	UNACK_RTN	VALUE	HI	1
12/14/2006 10:15:13 AM	UNACK	VALUE	HI	1

Displaying 1 to 2 of 2 alarms. Default Query 100 % Complete

InTouch アラームを表示するには、**Alarm Viewer** コントロールを使用することが推奨されます。ただし、7.1 よりも古いバージョンの InTouch で作成されたアプリケーションのアラームを表示するには、分散アラーム オブジェクトの使用を続行できます。

Alarm Viewer コントロールの設定

WindowMaker から **Alarm Viewer** コントロール オプションを設定できます。また、**Alarm Viewer** コントロールの実行中にユーザーが変更できるオプションも設定できます。これらのオプションは、**[AlarmViewerCtrl のプロパティ]** ダイアログ ボックスで設定します。



グリッドの外観の設定

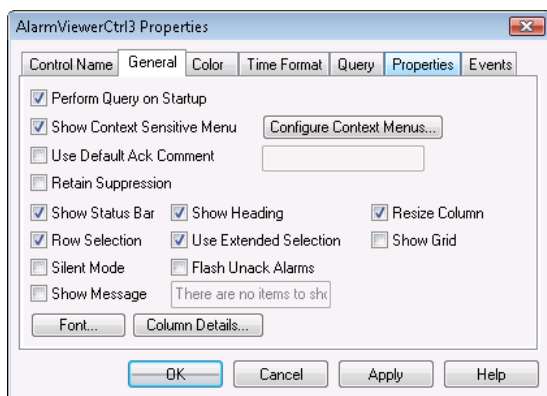
Alarm Viewer コントロールの表示外観を設定する場合、以下の操作を行うことができます。

- ステータス バーを含めます。
- カラム ヘッダーを含めます。
- 行とカラムを表示する水平と垂直のグリッド線を含めます。
- ユーザーのカラム幅を調整することを許可するランタイム オプションを含めます。
- 表示要素の色を設定します。

以下の図は、すべての表示プロパティがアクティブである場合に **Alarm Viewer** コントロールが表示される様子を示しています。

表示外観を設定するには

1. Alarm Viewer コントロールを右クリックして、[プロパティ] をクリックします。[AlarmViewerCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [全般] タブをクリックします。



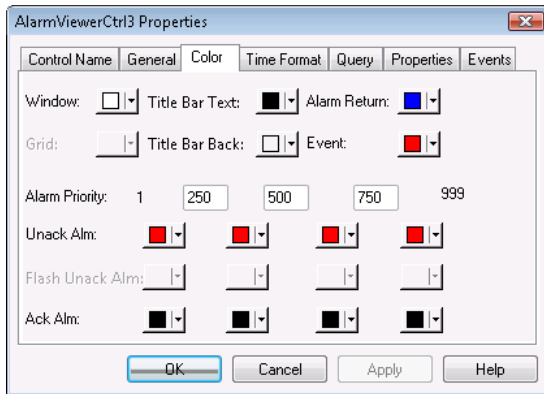
3. 表示外観を設定します。以下のいずれかを設定します。

オプション	説明
起動時にクエリー実行	デフォルトのクエリー プロパティを使用して、コントロールの更新を自動的に開始します。アプリケーションの起動時にクエリーが実行されない場合は、 Requery() 関数を使用したスクリプトを実行して、グリッドを更新する必要があります。ランタイム中には、グリッドのショートカットメニューでも [再クエリー] オプションを利用できます。
コンテキストメニューを表示	ランタイム中に右クリックのショートカットメニューが有効になります。
デフォルトの確認コメントを使用	オペレータがアラームを確認する際にデフォルトのコメントを表示するかどうかを制御します。このオプションをオンにして文字列を入力すると、その文字列がランタイム中にデフォルトのコメントとして使用されます。 このチェック ボックスをオフにすると、オペレータがアラームを確認する場合に、オプションのコメントを入力するためのダイアログボックスが表示されます。このダイアログ ボックスはテキストを入力することも、空白のまま残すこともできます。

オプション	説明
非表示を保持	アラーム クエリーが変更された場合に、アラーム クエリー間のアラーム非表示を保持します。
ステータス バーの表示	Alarm Viewer コントロールの一番下にステータス バーを表示するかどうかを切り替えます。
行の選択	ランタイム中にユーザーが個々の行を選択できるようにします。各行はアラーム レコードを表しています。ユーザーは複数アラームを選択できます。
サイレント モード	[サイレント モード] チェック ボックスをオンにすると、Alarm Viewer コントロールではランタイム中にエラー メッセージは表示されません。このチェック ボックスをオフにすると、アラーム表示にエラー メッセージが表示されます。 いずれの場合も、エラー メッセージは、常に Log Viewer に送信されます。
メッセージの表示	テスト ボックスに入力したメッセージが表示されます。このメッセージは、アラームがない場合に表示されるメッセージです。
見出しの表示	Alarm Viewer コントロールの一番上に見出しを表示するかどうかを切り替えます。
拡張アラーム選択の使用	CTRL キーまたは SHIFT キーを押しながらマウス ボタンと共に使用することにより、ユーザーは複数のアラームを同時に選択できるようになります。[行の選択] チェック ボックスがオンの場合のみ選択できます。
点滅未確認アラーム	未確認アラームが確認されるまで、そのアラームが 1 秒間隔で点滅します。 WindowViewer でアラーム表示を停止しても、未確認アラームの点滅は停止しません。
カラムのサイズ変更	[カラムのサイズ変更] をオンにすると、ユーザーはランタイム中にカラム幅を変更できるようになります。それ以外の場合は、カラム幅は静的となり、WindowMaker でのみ設定できます。
グリッドの表示	[グリッドの表示] をオンにすると、アラーム表示の各行と各カラムを区切る縦線と横線が Alarm Viewer コントロールに表示されます。このチェック ボックスをオフにすると、グリッドは表示されません。

4. [適用] をクリックします。

5. [色] タブをクリックします。



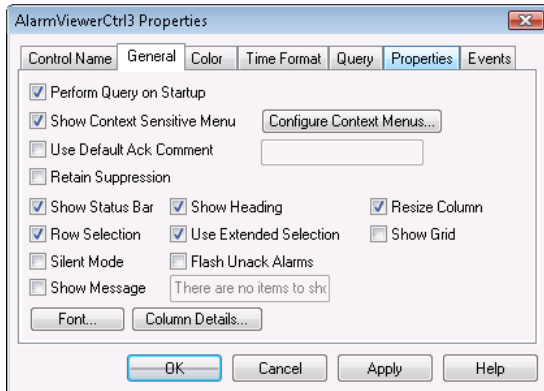
6. パレットのボタンをクリックして、Alarm Viewer コントロールの表示要素に色を割り当てます。
7. [適用] をクリックして、色の選択を保存します。
8. [OK] をクリックします。

フォント表示の設定

Alarm Viewer コントロールでテキストが表示される方法を設定できます。

フォントのプロパティを設定するには

1. Alarm Viewer コントロールを右クリックして、[プロパティ] をクリックします。[AlarmViewerCtrl プロパティ] ダイアログ ボックスが表示されます。
2. [一般] タブをクリックします。



3. [フォント] をクリックします。Windows 標準の [フォント] ダイアログ ボックスが表示されます。フォントを設定して、[OK] をクリックします。
4. [OK] をクリックします。

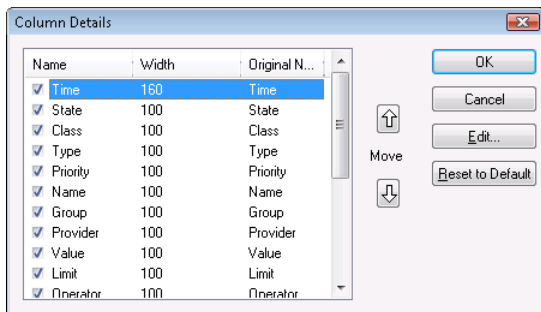
表示カラムの詳細の設定

Alarm Viewer コントロールの場合、以下の操作を行えます。

- カラムを選択して、順序を設定します。
- カラム幅をピクセル単位で設定します。
- カラムの名前を変更します。

表示カラムの詳細を設定するには

1. Alarm Viewer コントロールを右クリックして、[プロパティ] をクリックします。[AlarmViewerCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [全般] タブをクリックします。
3. [カラムの詳細] をクリックします。[カラムの詳細] ダイアログ ボックスが表示されます。

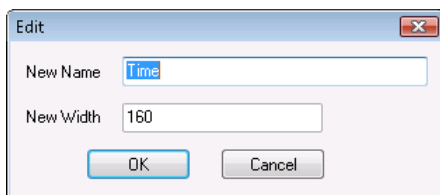


4. [アラーム名] カラムで、表示するカラム名の横にあるチェック ボックスをオンにします。リストから少なくとも 1 つ以上のカラムを選択する必要があります。

カラム	内容
時間	[時間の形式] タブで指定された時間です。
状態	アラームの状態です。
クラス	アラーム カテゴリです。
タイプ	アラーム タイプです。
優先度	アラームの優先度です。
名前	タグ名です。
グループ	アラーム グループの名前です。
プロバイダ	アラーム プロバイダの名前です。
値	アラーム発生時のタグの値です。カラムの幅は、希望する正確なレベルを提供するために十分な大きさである必要があります。
しきい値	タグのアラームしきい値です。カラムの幅は、希望する正確なレベルを提供するために十分な大きさである必要があります。
オペレータ	アラーム状況に関連付けられているログオンしたオペレータの ID です。
オペレータ フルネーム	ログオンしたオペレータのフルネームです。

オペレータ ノード	アラーム状況に関連付けられているログオンしたオペレータのノードです。 ターミナル サービス環境で、オペレータ ノードはオペレータがターミナル サービス セッションを確立したクライアント コンピュータの名前です。ノード名を取得できない場合、ノードの IP アドレスが代わりに使用されます。
オペレータ ドメイン	アラーム状況に関連付けられているログオンしたオペレータのドメインです。
タグ コメント	タグのコメントです。
アラーム コメント	タグのアラームに関連付けられているコメントです。これらのコメントは、タグのアラームが定義された際に [アラーム コメント] ボックスに入力されたものです。アラームに対して確認コメントが指定されると、新しいコメントはこのコメント カラムで更新されます。
ユーザー 1	アラームの AlarmUserDefNum1 プロパティの数値です。
ユーザー 2	アラームの AlarmUserDefNum2 プロパティの数値です。
ユーザー 3	アラームの AlarmUserDefStr プロパティの文字列値です。

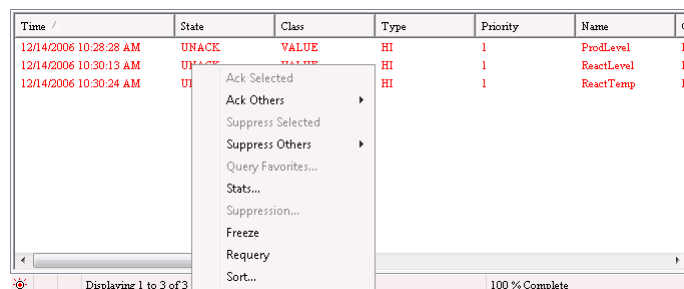
5. カラム名を選択し、上下の矢印を使用して、カラムを並べ替えます。[カラムの詳細] ダイアログ ボックスの一番上に表示されるカラム名は、アラーム コントロールの一番左のカラムの名前です。
6. カラムの名前または幅を変更するには、カラムを選択して、[編集] をクリックします。[編集] ダイアログ ボックスが表示されます。



- a. [新しい名前] ボックスに、新しいカラム名を入力します。
 - b. [新しい幅] ボックスに、カラム幅を入力します。カラム幅は、1 ～ 999 ピクセルの範囲で設定できます。
 - c. [OK] をクリックします。
7. [カラムの詳細] ダイアログ ボックスの [OK] をクリックします。
 8. [適用] をクリックします。

ランタイム時における機能へのアクセスの制御

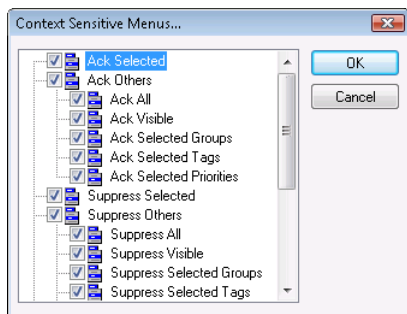
ランタイム中に **Alarm Viewer** コントロールを右クリックすると、コンテキスト（ショートカット）メニューが表示されます。



このショートカットメニューに、どのメニュー コマンドを表示するかを管理できます。

ショートカットメニューを設定するには

1. **Alarm Viewer** コントロールを右クリックして、**[プロパティ]** をクリックします。
[AlarmViewerCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. **[全般]** タブで **[コンテキストメニューを表示]** チェック ボックスをオンにします。
3. **[コンテキストメニューの設定]** をクリックします。
[コンテキストメニュー] ダイアログ ボックスが表示されます。



このダイアログ ボックスには、**Alarm Viewer** コントロールのショートカットメニューに表示できるコマンドのリストが階層表示されます。

4. ショートカットメニューのオプションを設定します。ショートカットメニューのコマンドは少なくとも 1 つ選択する必要があります。

コマンド名	ランタイム ユーザーに許可される内容
選択アラームを確認	選択したアラームを確認できます。 [選択アラームを確認] および [その他を確認] のどちらのメニュー項目も選択しないと、 [デフォルトの確認コメントを使用] チェック ボックスとテキスト ボックスは無効になります。
その他を確認	その他の方法によってアラームを確認します。どのアラームを確認するかは、ユーザーが選択できます。 [その他を確認] が選択されている場合、サブメニュー項目を少なくとも 1 つ選択する必要があります。

全てを確認	すべてのアクティブなアラームを確認します。
表示部分を確認	表示されているアラームを確認します。
選択グループを確認	選択したグループと同じグループ名および同じプロバイダ名を持つアラームをすべて確認します。
選択タグを確認	選択したタグとグループ名が同じで、同じプロバイダ、グループ、および優先度を持つアラームをすべて確認します。
選択優先度を確認	選択した優先度と同じ優先度を持ち、同じプロバイダおよびグループを持つアラームをすべて確認します。
選択アラームを非表示	選択したアラームを非表示にします。
その他を非表示	ショートカットメニューに表示されるその他の方法で、アラームを非表示にします。
全て非表示	すべてのアラームを非表示にします。
表示部分を非表示	表示されているすべてのアラームを非表示にします。
選択グループを非表示	選択したグループと同じグループ名を持つすべてのアラームを非表示にします。
選択タグを非表示	選択したタグと同じタグ名を持つすべてのアラームを非表示にします。
選択優先度を非表示	選択した優先度と同じ優先度を持つすべてのアラームを非表示にします。
全て再表示	非表示のアラームをすべて再表示にします。
クエリー設定	[アラーム クエリー] ダイアログ ボックスを開きます。
統計	[アラームの統計] ダイアログ ボックスを開きます。
非表示	[非表示中のアラーム] ダイアログ ボックスを開きます。
停止	Alarm Viewer コントロールの停止/停止解除モードを切り替えます。
再クエリー	アラーム クエリーを再実行します。
ソート	[ソートのカスタマイズ] ダイアログ ボックスを開きます。

5. [OK] をクリックします。
6. [行の選択] をクリックすると、ユーザーはランタイム中に Alarm Viewer コントロールから行を選択できます。
7. [拡張アラーム選択の使用] をクリックすると、ユーザーは **Shift** キーまたは **Ctrl** キーを使用して、Alarm Viewer コントロールから複数のアラーム レコードを同時に選択できます。
8. [適用] をクリックします。

表示するアラームの選択

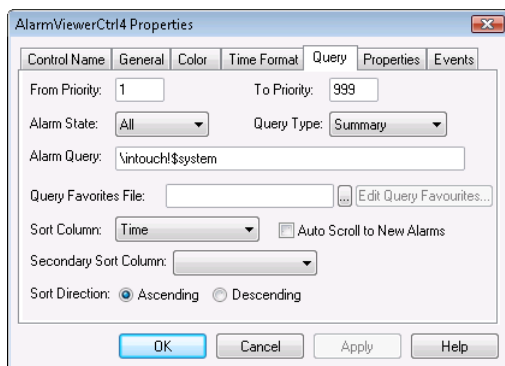
Alarm Viewer コントロールには、アクティブなアラームのサマリや履歴アラームのリストを表示できます。

一般のアラーム クエリー プロパティを設定するには

1. Alarm Viewer コントロールを右クリックして、**[プロパティ]** をクリックします。**[AlarmViewerCtrl のプロパティ]** ダイアログ ボックスが表示されます。
2. **[一般]** タブをクリックします。
3. **[起動時にクエリー実行]** チェック ボックスをオンにすると、アプリケーションの起動時にデフォルトのクエリー プロパティを使用して Alarm Viewer コントロールが自動的に更新されます。
4. **[メッセージの表示]** チェック ボックスをオンにすると、アラームがない場合にデフォルトのメッセージが表示されます。テキスト ボックスには、表示するメッセージを入力します。
5. **[適用]** をクリックします。

クエリーのデフォルトを設定するには

1. Alarm Viewer コントロールを右クリックして、**[プロパティ]** をクリックします。**[AlarmViewerCtrl のプロパティ]** ダイアログ ボックスが表示されます。
2. **[クエリー]** タブをクリックします。



3. **[優先度始点]** ボックスに、最低アラーム優先度の値 (1 ~ 999) を入力します。
4. **[優先度終点]** ボックスに、最高アラーム優先度の値 (1 ~ 999) を入力します。
5. **[クエリー タイプ]** の矢印をクリックして、デフォルトのランタイム アラーム表示として **[Historical]** または **[Summary]** のいずれかを選択します。

表示のデフォルト タイプは、クエリー関数を含む QuickScript を実行することによって、ランタイム中に変更できます。たとえば、スクリプトに Type パラメータが "Summary" に設定されている ApplyQuery() メソッドが含まれている場合、グリッドには現在のアラームのサマリが表示されます。反対に、同じグリッドに対して、Type パラメータが "Historical" に設定された ApplyQuery() メソッドが実行されている場合、履歴アラームが表示されます。QueryType プロパティは、アラーム表示の現在の状態を反映しています。

6. **[アラーム クエリー]** ボックスに、有効なアラーム クエリーを入力します。たとえば、\InTouch! \$System と入力すると、デフォルトの \$System アラーム グループに属するすべてのアラームに対してクエリーされます。
7. **[OK]** をクリックします。

クエリー設定を使用したカスタム保存クエリーの作成

オペレータがショートカットメニューから選択できるクエリー設定のリストを設定できます。

Windows Vista または新しい OS の標準ユーザーがアクセスできるフォルダにクエリー設定ファイルを配置してください。WindowViewer が常に同じユーザー アカウントで実行される場合、以下を使用できます。

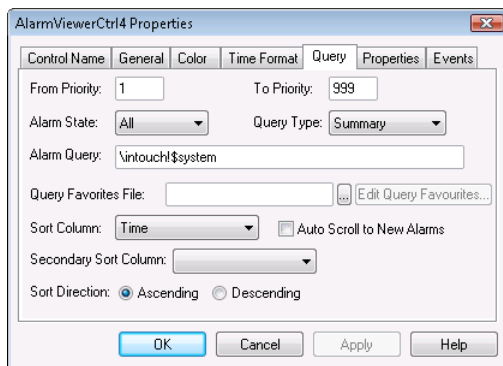
C:\Users\<ユーザー名>\AppData\Local\

異なるユーザーが WindowViewer を実行する場合、以下を使用できます。

C:\Users\Public\Documents\

クエリー設定ファイルを設定するには

1. Alarm Viewer コントロールを右クリックして、[プロパティ] をクリックします。[AlarmViewerCtrl のプロパティ] ダイアログ ボックスが表示されます。



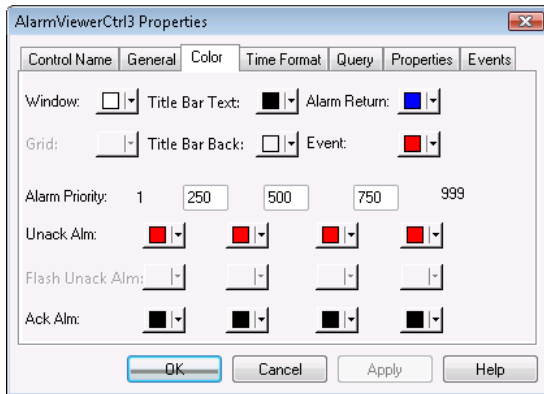
2. [クエリー] タブをクリックします。
3. クエリー設定ファイルを設定します。
 - a. [クエリー設定ファイル] ボックスに、ネットワーク パスとファイル名を入力するか、省略記号 ボタンをクリックしてファイルを参照します。
 - b. フィルタ設定ファイルを編集するには、[クエリー設定の編集] ボタンをクリックします。[アラーム クエリー] ウィンドウが開き、設定ファイルのフィルタを追加、変更、または削除できます。完了したら、[OK] をクリックして変更を保存し、ウィンドウを閉じます。
4. [OK] をクリックします。

さまざまなタイプのアラーム レコードの色の使用

Alarm Viewer コントロールに表示されるさまざまなアラーム状態に対して、色オプションを設定できます。

アラーム表示の色を設定するには

1. Alarm Viewer コントロールを右クリックして、[プロパティ] をクリックします。[AlarmViewerCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [色] タブをクリックします。



3. 各色ボックスをクリックして、カラーパレットを開きます。以下の各項目に対応するパレットで、使用する色をクリックします。

プロパティ	説明
ウィンドウ	背景色を設定します。
タイトルバーのテキスト	タイトルバーのテキストの色を設定します（[見出しの表示] オプションがオンである場合のみ）。
アラーム復帰	平常状態に復帰したアラーム（確認されずに平常に戻ったアラーム）の色を設定します。
グリッド	グリッドの色を設定します。デフォルトでは、グリッドは表示されません。デフォルトのグリッド色は薄いグレーです。アラームオブジェクトのグリッドの色は、選択したウィンドウの色に対してコントラストのはっきりした色に自動設定されます。
タイトルバーの背景	タイトルバーの背景色を設定します（[見出しの表示] オプションがオンである場合のみ使用可能）。
イベント	イベントの色を設定します。

4. [アラーム優先度] ボックスに、未確認アラーム、確認アラーム、および点滅未確認アラームを識別するために使用される異なる色に対する境界として使用されるアラーム優先度の数値を入力します。
5. [未確認アラーム] と [確認アラーム] の色ボックスをクリックして、カラーパレットを開きます。使用する色をパレットでクリックします。
6. 点滅未確認アラームに対してアラームクエリーを設定するには、[全般] タブをクリックして [点滅未確認アラーム] チェックボックスをオンにし、[色] タブをクリックして [点滅未確認アラーム] の色ボックスを選択します。各アラーム優先度範囲に使用する色を選択します。

注記: Alarm Viewer コントロールには、1 秒未満の間に発生する変化は表示できません。1 秒間に 2 回アラーム状態が変化する場合、Alarm Viewer コントロールではその変化を認識できません。

7. [適用] をクリックします。

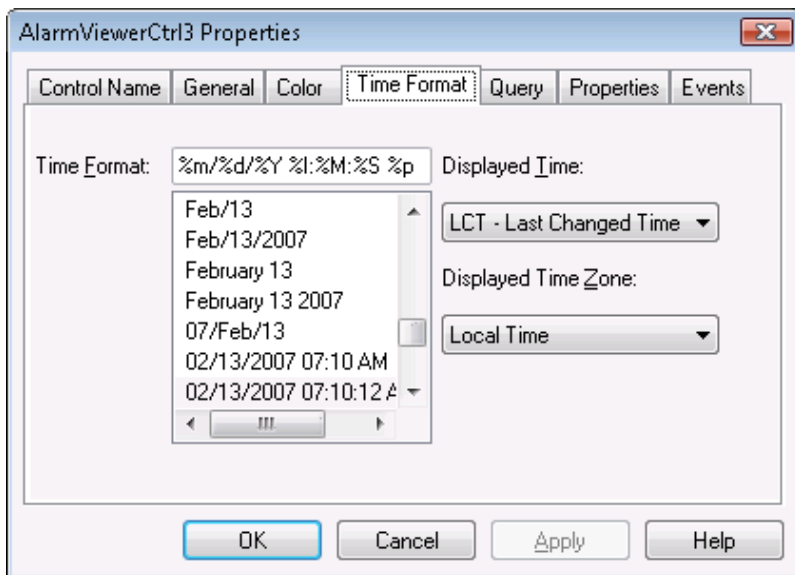
アラーム レコードの表示時間形式の設定

表示されているアラーム レコードの時間形式を設定できます。

Alarm Viewer コントロールの場合、元のアラーム時刻は、アラームが開始した日付/時刻スタンプです。タグが I/O 型タグであり、そのサーバーがタイムスタンプを渡すことができる場合は、I/O サーバーからのタイムスタンプです。

時間形式を設定するには

1. **Alarm Viewer** コントロールを右クリックして、[プロパティ] をクリックします。[AlarmViewerCtrl のプロパティ] ダイアログ ボックスが表示されます。
2. [時間の形式] タブをクリックします。



3. [時間の形式] リストで、使用する時間の形式をクリックします。[時間の形式] ボックスには、選択した形式に対して、% 記号で区切られた文字で構成される一組の文字列が表示されます。

文字列	説明
d	2 桁の日付
b	月を示す 3 文字の略語
y	年を表す 4 桁の数
m	月を表す 2 桁の数
y	年を表す 2 桁の数
#x	日付と曜日。次に例を示します。2007 年 8 月 10 日 金曜日
B	月の名前
H	24 時間形式の時間
M	分

文字列	説明
p	午前または午後（12 時間形式で使用）
S	秒
s	秒の端数
l	12 時間形式の時間

4. [表示される時刻] リストで表示時間を選択します。

OAT	元のアラーム時刻です。アラームが開始した日付/時刻スタンプです。
LCT	最終変更時刻です。アラームのインスタンスに対して発生した最新の状態変化（アラームの開始、サブステートの変化、平常への復帰、または確認）の日付/時刻スタンプです。
LCT（確認時には OAT）	最終変更時刻ですが、確認時の元のアラーム時刻です。アラームが未確認である間は最終変更時刻、アラームが確認された後は元のアラーム時刻が使用されます。

5. [表示されるタイムゾーン] リストで、タイムゾーンを選択します。

GMT	協定世界時（UTC または Zulu）とも呼ばれるグリニッジ標準時です。
ローカル時間	ローカルタイムゾーンに対して調整されるアラーム時間です。
標準時間	アラームソースのタイムゾーンに対して調整されるアラーム時間です。

6. [適用] をクリックします。

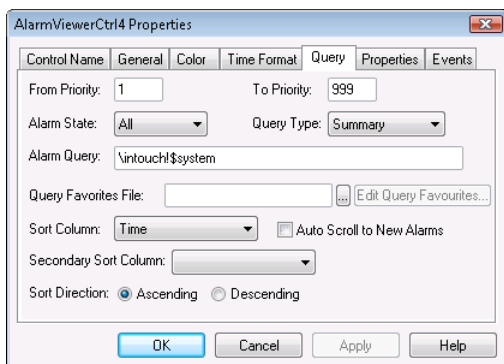
アラームレコードのソート順の設定

リストのアラームレコードをソートできます。デフォルトで、Alarm Viewer コントロールにはアラームレコードが時刻で昇順に一覧表示されます。

プライマリ カラムやオプションのセカンダリ ソート カラムに基づいて、アラームレコードを昇順または降順にソートできます。

アラームレコードのソート順を設定するには

1. Alarm Viewer コントロールを右クリックして、[プロパティ] をクリックします。[AlarmViewerCtrl のプロパティ] ダイアログボックスが表示されます。
2. [クエリー] タブをクリックします。



3. 以下の手順に従って、ソート オプションを選択します。
 - a. **【カラムのソート】** リストから、プライマリ ソート カラムを選択します。**【カラムのソート】** リストに表示されるのは、表示されるカラムのみです。目的のカラムが見つからない場合は、**【全般】** タブに移動し、**【カラムの詳細】** からカラムを選択します。
 - b. **【セカンダリ ソート カラム】** リストから、セカンダリ ソート カラムを選択します。
 - c. プライマリ ソート カラムとして時間を選択すると、**【新規アラームに自動スクロール】** チェックボックスが使用できるようになります。新しいアラームが発生した場合に、自動的にスクロールして、それらのアラームを表示する場合は、このオプションをオンにします。
 - d. **【昇順】** または **【降順】** を選択して、ソート 順を設定します。
4. **【適用】** をクリックします。

ランタイムでの Alarm Viewer コントロールの使用

Alarm Viewer コントロールには、1 つまたは複数の選択したアラーム、アラーム グループ、タグ変数、および優先度の表示オブジェクトに適用できるコマンドへの簡単アクセスをオペレータに提供するショートカットメニューが含まれています。


以下のリストは、Alarm Viewer コントロールのショートカット メニューから使用できるコマンドを一覧表示しています。

- **選択アラームを確認** - 選択したアラームを確認します。
- **その他の確認** - その他の確認コマンドを一覧するサブメニューが表示されます。
 - **全てを確認** - 現在のアラーム クエリーのすべてのアラームを確認します。アラーム グリッドの表示領域は制限されているため、**【全てを確認】** コマンドではグリッドで表示されていないアラームも確認されます。
 - **表示部分を確認** - アラーム グリッドに現在表示されているアラームのみを確認します。
 - **選択グループを確認** - 1 つまたは複数の選択したアラームと同じプロバイダからの同じグループ名を持つすべてのアラームを確認します。
 - **選択タグ変数を確認** - 1 つまたは複数の選択したアラームと同じ優先度を持つ、同じプロバイダおよびグループ名からの同じタグ変数を持つすべてのアラームを確認します。
 - **選択優先度を確認** - 1 つまたは複数の選択したアラームと同じプロバイダおよびグループ名からの同じ優先度を持つすべてのアラームを確認します。
- **選択アラームを非表示** - 選択したアラームを非表示にします。

- **その他を非表示** - 非表示コマンドが含まれるサブメニューが表示されます。
 - 全て非表示 - すべてのアラームの現在および今後の発生を非表示にします。
 - 表示部分を非表示 - 表示されているアラームの現在および今後の発生を非表示にします。
 - 選択グループを非表示 - 同じプロバイダ名を持つ1つまたは複数の選択したアラームの同じグループに属す任意のアラームの現在および今後の発生を非表示にします。
 - 選択タグ変数を非表示 - 同じプロバイダ名、グループ名、および優先度範囲を持つ1つまたは複数の選択したアラームの同じタグ変数に属す任意のアラームの現在および今後の発生を非表示にします。
 - 選択優先度を非表示 - 同じプロバイダ名およびグループ名を持つ1つまたは複数の選択したアラームの同じ優先度に属す任意のアラームの現在および今後の発生を非表示にします。
 - 全て再表示 - 非表示の設定をクリアします。
- **クエリー設定** - 以前保存されたアラーム クエリーを選択するための [アラーム クエリー] ダイアログ ボックスを表示します。アラーム クエリーを追加、変更、および削除できます。
- **統計** - [アラームの統計] ダイアログ ボックスを表示します。
- **非表示中** - [非表示中のアラーム] ダイアログ ボックスを表示します。
- **停止** - 現在の表示をフリーズします。
- **再クエリー** - アラーム プロバイダを再びクエリーします。
- **ソート** - [ソートのカスタマイズ] ダイアログ ボックスを表示します。

ステータス バー情報の表示

[一般] プロパティ ページから [ステータス バーの表示] オプションを選択すると、ランタイム中に Alarm Viewer コントロールの一番下にステータス バーが表示されます。

	Displaying 1 to 12 of 451 alarms.	Default Query	100 % Complete
---	-----------------------------------	---------------	----------------

ステータス メッセージ、現在のアラーム クエリー、および進行状況バーの3つのインジケータが含まれているステータス バーです。このインジケータでは、表示クエリーの現在の状況の概要と Alarm Viewer コントロールで利用できる非表示機能に関する詳細が提供されます。コントロールが停止の状態である場合は、ステータス バーの右ペインが赤となり、1つまたは複数のアラームが非表示の状態である場合は、ステータス バーの左ペインが赤となります。非表示の状態である場合、「非表示中」という単語が左ペインに表示されます。

ランタイム時のクエリー設定の使用

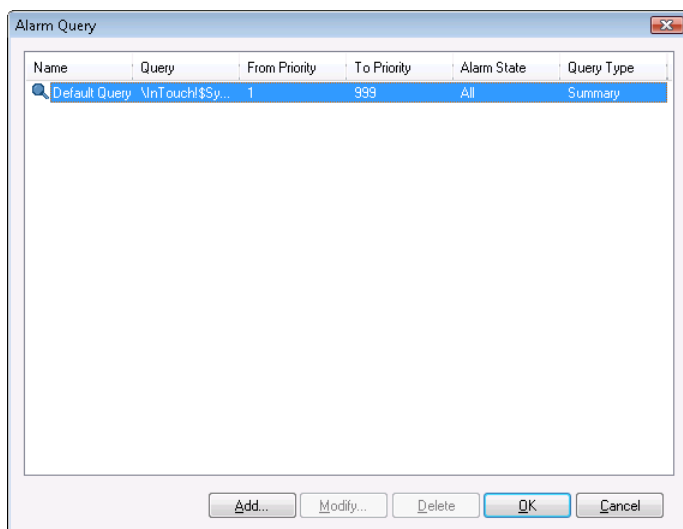
以前定義されたアラーム クエリーのリストから素早くアラーム クエリーを選択するには、Alarm Viewer コントロールのショートカット メニューの [クエリー設定] コマンドを使用します。また、新しい名前付きクエリーを作成したり、既存のクエリーを編集したり、既存のクエリーを削除したりすることもできます。

アラーム クエリーへの変更は、同じクエリーを使用する別の Alarm Viewer コントロールには、自動的に適用されません。アラーム クエリーを削除しても、同じクエリーを使用する他の Alarm Viewer コントロールからクエリーは自動的に削除されません。

注記: Alarm Viewer コントロールに複数行アラーム クエリーが表示される場合、改行部分が文字化けします。ただし、機能には影響ありません。

ランタイム時にアラーム クエリーを選択するには

1. **Alarm Viewer** コントロールを右クリックして、**[クエリー設定]** をクリックします。**[アラーム クエリー]** ダイアログ ボックスが表示されます。



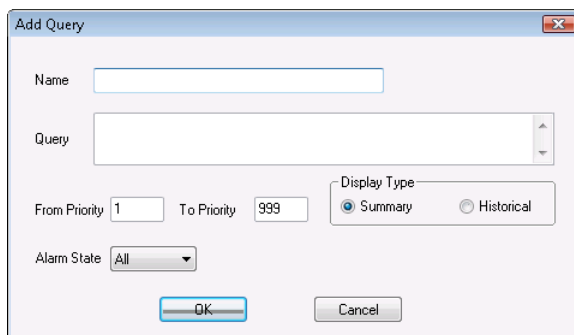
2. 現在定義されているクエリーのリストから表示する名前付きクエリーを選択します。
3. **[OK]** をクリックします。これで、クエリーによって取得されるアラーム情報が **Alarm Viewer** コントロールで表示されるようになります。

ランタイム時に新しい名前付きクエリーを追加するには

1. **Alarm Viewer** コントロールを右クリックして、**[クエリー設定]** をクリックします。**[アラーム クエリー]** ダイアログ ボックスが表示されます。

2. **[追加]** をクリックします。

[クエリーの追加] ダイアログ ボックスが表示されます。



3. クエリーを設定します。以下の手順を実行します。
 - a. **[名前]** ボックスに、クエリーを識別するために使用する名前を入力します。
 - b. **[クエリー]** ボックスに、実行する InTouch アラーム クエリーのセットを入力します。1 つまたは複数のアラーム プロバイダとグループを指定できます。
 - c. **[優先度始点]** ボックスに、最低アラーム優先度の値 (1 ~ 999) を入力します。
 - d. **[優先度終点]** ボックスに、最高アラーム優先度の値 (1 ~ 999) を入力します。

- e. [アラーム状態] の矢印をクリックして、アラーム クエリーで使用するアラーム状態（すべて、確認、未確認）を選択します。
 - f. [表示タイプ] 領域で、クエリーするレコードのタイプとして [サマリ] または [履歴] を選択します。
4. [OK] をクリックして、[クエリーの追加] ダイアログ ボックスを閉じます。
 5. [アラーム クエリー] ダイアログ ボックスで [OK] をクリックして、クエリーをクエリー設定に追加します。

ランタイム時に既存の名前付きクエリーを変更するには

1. **Alarm Viewer** コントロールを右クリックして、[クエリー設定] をクリックします。
[アラーム クエリー] ダイアログ ボックスが表示されます。
2. 現在定義されているクエリーのリストで、変更する名前付きクエリーを選択します。
3. [修正] をクリックします。
[クエリーの修正] ダイアログ ボックスが表示されます。
4. 必要な変更を行い、[OK] をクリックします。
5. [アラーム クエリー] ダイアログ ボックスで、[OK] をクリックします。

ランタイム時に既存の名前付きクエリーを削除するには

1. **Alarm Viewer** コントロールを右クリックして、[クエリー設定] をクリックします。
[アラーム クエリー] ダイアログ ボックスが表示されます。
2. 現在定義されているクエリーのリストで、削除する名前付きクエリーを選択します。
3. [削除] をクリックします。メッセージが表示されたら、[はい] をクリックします。
4. [アラーム クエリー] ダイアログ ボックスで、[OK] をクリックします。

Alarm Viewer コントロールの ActiveX プロパティの使用

スクリプトを使用して **Alarm Viewer** コントロールのプロパティの値を直接設定するか、InTouch タグまたは I/O 参照に割り当てることができます。プロパティの設定の詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[ActiveX コントロールのスクリプト](#)」を参照してください。

以下の表には、Alarm Viewer コントロール プロパティが一覧表示されています。

プロパティ	タイプ	説明
AckAllMenu	論理型	[全てを確認] メニュー項目を有効または無効にします。
AckAlmColorRange1	整数型	優先度範囲が 1 から ColorPriorityRange1 までの確認済みアラームを表示するために使用される色を設定します。デフォルトの優先度範囲は 1～250 です。

プロパティ	タイプ	説明
AckAlmColorRange2	整数型	優先度範囲が ColorPriorityRange1 から ColorPriorityRange2 までの確認済みアラームに対して使用される色を設定します。デフォルトの優先度範囲は 250～500 です。
AckAlmColorRange3	整数型	優先度範囲が ColorPriorityRange2 から ColorPriorityRange3 までの確認済みアラームに対して使用される色を設定します。デフォルトの優先度範囲は 500～750 です。
AckAlmColorRange4	整数型	優先度範囲が ColorPriorityRange3 から 999 までの確認済みアラームに対して使用される色を設定します。デフォルトの優先度範囲は 750～999 です。
AckOthersMenu	論理型	[その他を確認] メニュー項目を有効または無効にします。
AckSelectedGroupsMenu	論理型	[選択グループを確認] メニュー項目を有効または無効にします。
AckSelectedMenu	論理型	[選択アラームを確認] メニュー項目を有効または無効にします。
AckSelectedPrioritiesMenu	論理型	[選択優先度を確認] メニュー項目を有効または無効にします。
AckSelectedTagsMenu	論理型	[選択タグを確認] メニュー項目を有効または無効にします。
AckVisibleMenu	論理型	[表示部分を確認] メニュー項目を有効または無効にします。
AlarmQuery	メッセージ型	<p>初期のアラーム クエリーを設定します。このフィールドはテキストのみを受け入れ、タグは受け入れられません。以下の例は、アラーム グループへフルパスを使用しています。</p> <p>\\ノード\InTouch!Group</p> <p>以下の例では、ローカル アラーム グループへのフルパスを使用しています。</p> <p>\InTouch!Group</p> <p>この例では、別のグループ リストを使用しています。</p> <p>GroupList</p>
AlarmState	メッセージ型	クエリーに対するデフォルトのアラーム状態（すべて、未確認、確認）です。

プロパティ	タイプ	説明
AlmRtnColor	整数型	平常状態に復帰し、確認されていないアラームの色を設定します。この色は、確認状態から平常状態に復帰したものの、確認状態への移行が確認されていないアラームにも使用されます。
AutoScroll	論理型	ユーザーがリストを先頭からスクロールする場合、自動的に新しいアラームにジャンプします新しいアラームは、表示オブジェクト内に現在表示されていないアラームです。
ColorPriorityRange1	整数型	アラームが表示される優先度範囲の境界を設定します。このプロパティの値は1より大きく、 ColorPriorityRange2 の値よりも小さい値を指定する必要があります。
ColorPriorityRange2	整数型	アラームが表示される優先度範囲の境界を設定します。このプロパティには、 ColorPriorityRange1 より大きく、 ColorPriorityRange3 よりも小さい値を指定する必要があります。
ColorPriorityRange3	整数型	アラームが表示される優先度範囲の境界を設定します。このプロパティの値には、 ColorPriorityRange2 より大きく、999 よりも小さい値を指定する必要があります。
ColumnResize	論理型	ランタイム時にカラムのサイズを変更ができるかどうかを決定する値を返すか設定します。
CustomMessage	メッセージ型	アラームがない場合も表示されるデフォルトメッセージです。
DefaultAckComment	メッセージ型	アラームが確認され、"UseDefaultAckComment" が True の場合に、コメントとして使用されます。それ以外の場合には、ユーザーはコメントを入力することを要求されます。
DisplayedTime	メッセージ型	アラーム メッセージの時間を表示します。値は "OAT" (元のアラーム時刻)、"LCT" (最終変更時刻)、または "LCT But OAT on ACK" (最終変更時刻 (確認時には元のアラーム時刻)) になります。
DisplayedTimeZone	メッセージ型	現在のタイム ゾーンの文字列を取得または設定します。

プロパティ	タイプ	説明
		値は "GMT"、"Origin Time"（標準時間）、または "Local Time"（ローカル時間）です。
EventColor	整数型	イベントの色を設定します。
ExtendedSelection	論理型	Ctrl キーまたは Shift キーを押しながらマウス ボタンと共に使用することにより、ユーザーは複数のアラームを選択できるようになります。 デフォルトでは、簡単にクリックすることによってアラームの選択を切り替えることができます（[行の選択] チェック ボックスがオンである場合のみ使用できます）。
FlashUnAckAlarms	論理型	未確認アラームの点滅を有効または無効にします。1 または 0 の論理型入力値を取ります。このプロパティが 1 に設定されると、未確認アラームは 1 秒ごとに点滅します。このプロパティが 0 に設定されると、未確認アラームは点滅しません。 このプロパティは、Alarm Viewer コントロールの [全般] タブにある [点滅未確認アラーム] チェック ボックスに対応します。
FlashUnackAlmColorRange1	整数型	アラーム優先度範囲 1 に属す未確認アラームの点滅カラーを設定します。
FlashUnackAlmColorRange2	整数型	アラーム優先度範囲 2 に属す未確認アラームの点滅カラーを設定します。
FlashUnackAlmColorRange3	整数型	アラーム優先度範囲 3 に属す未確認アラームの点滅カラーを設定します。
FlashUnackAlmColorRange4	整数型	アラーム優先度範囲 4 に属す未確認アラームの点滅カラーを設定します。
Font	なし	コントロールのレコードと見出しのフォントを設定します。
FreezeMenu	論理型	[停止] メニュー項目を有効または無効にします。
FromPriority	整数型	デフォルトのクエリーの低い優先度値を設定します。
GridColor	整数型	背景グリッドの色を設定します。

プロパティ	タイプ	説明
NewAlarmEventMode	整数型	NewAlarm イベントのトリガを制御します。 0 = NewAlarm イベントはトリガできません (デフォルト)。 1 = NewAlarm イベントがアクティブになります。 2 = NewAlarm イベントがアクティブになり、新しい未確認アラームが少なくとも 1 つ取得されると常にトリガされます。
QueryFavoritesFile	メッセージ型	クエリー設定ファイルの名前を返すか、設定します。
QueryFavoritesMenu	論理型	[クエリー設定] メニュー項目を有効または無効にします。
QueryName	文字列型	現在のクエリー名を返します。
QueryStartup	論理型	設定された場合、デフォルトのクエリー プロパティを使用して、グリッドの更新を自動的に開始します。 選択しない場合、スクリプトで ApplyDefaultQuery メソッドまたは ApplyQuery メソッドを実行して、グリッドを更新する必要があります。
QueryType	メッセージ型	表示タイプを [サマリ] または [履歴] のいずれかに設定します。
RequeryMenu	論理型	[再クエリー] ショートカットメニュー項目を有効または無効にします。
RetainSuppression	論理型	アラーム クエリーが変更された場合に、アラーム クエリー間のアラーム非表示を保持します。
RowSelection	論理型	ランタイム中にユーザーがアラームを選択できるようになります。
SecondarySortColumn	メッセージ型	現在のセカンダリ ソート カラムを返すか、設定します。
SelectedCount	整数型	選択したアラームの合計数を返します。
ShowContextMenu	論理型	ショートカット メニューの起動を有効にします。
ShowGrid	論理型	コントロールでグリッド ラインが表示されるかどうかを決定する値を返すか、設定します。

プロパティ	タイプ	説明
ShowHeading	論理型	コントロールのタイトルバーを表示します。
ShowMessage	論理型	コントロールでエラーメッセージを表示されるかどうかを決定する値を返すか、設定します。
ShowStatusBar	論理型	ステータスバーが表示されるかどうか決定する値を取得するか、設定します。
SilentMode	論理型	コントロールがサイレントモードであるかどうかを決定する値を取得または設定します。
SortColumn	メッセージ型	現在のソートカラムを取得または設定します。
SortMenu	論理型	[ソート] メニュー項目を有効または無効にします。
SortOrder	論理型	ソート順を取得または設定します。可能な値は「昇順」および「降順」です。それぞれ、0 および 1 で示されます。
StatsMenu	論理型	[統計] メニュー項目を有効または無効にします。
SuppressAllMenu	論理型	[全て非表示] メニュー項目を有効または無効にします。
SuppressedAlarms	整数型	非表示アラームの合計数を取得します。
SuppressionMenu	論理型	[非表示中] メニュー項目を有効または無効にします。
SuppressOthersMenu	論理型	[その他を非表示] メニュー項目を有効または無効にします。
SuppressSelectedGroupsMenu	論理型	[選択グループを非表示] メニュー項目を有効または無効にします。
SuppressSelectedMenu	論理型	[選択アラームを非表示] メニュー項目を有効または無効にします。
SuppressSelectedPrioritiesMenu	論理型	[選択優先度を非表示] メニュー項目を有効または無効にします。
SuppressSelectedTagsMenu	論理型	[選択タグを非表示] メニュー項目を有効または無効にします。
SuppressVisibleMenu	論理型	[表示部分を非表示] メニュー項目を有効または無効にします。

プロパティ	タイプ	説明
TimeFormat	メッセージ型	アラームのタイムスタンプの形式を設定します。
TitleBackColor	整数型	タイトルバーの背景色を設定します。
TitleForeColor	整数型	タイトルバーの前景色を設定します。
ToPriority	整数型	アラーム クエリーの最大優先度を設定します。
TotalAlarms	整数型	アラームの数を取得します。
UnackAlarms	整数型	未確認アラームの合計数を取得します。
UnAckAlmColorRange1	整数型	優先度範囲が 1 から ColorPriorityRange1 までの未確認アラームを表示するために使用される色を設定します。
UnAckAlmColorRange2	整数型	優先度範囲が ColorPriorityRange1 から ColorPriorityRange2 までの未確認アラームを表示するために使用される色を設定します。
UnAckAlmColorRange3	整数型	優先度範囲が ColorPriorityRange2 から ColorPriorityRange3 の未確認アラームを表示するために使用される色を設定します。
UnAckAlmColorRange4	整数型	優先度範囲が ColorPriorityRange3 から 999 までの未確認アラームを表示するために使用される色を設定します。
UnsuppressAllMenu	論理型	[全て再表示] メニュー項目を有効または無効にします。
UseDefaultAckComment	論理型	True に設定すると、アラームが確認されるときにデフォルトの確認コメントが使用されます。それ以外の場合には、オペレータはコメントを入力することを要求されます。
WindowColor	整数型	グリッドの背景色を設定します。

ActiveX コントロールの色の設定

整数値として色を指定します。アラーム ActiveX コントロールでは、ABGR モデルを使用して 32 ビットの整数値として色を指定しています。

A = 透明度

B = 青

G = 緑

R = 赤

透明度はアラーム ActiveX コントロールではサポートされていません。上位 8 ビットの値はすべて無視されます。

たとえば、色を青に設定するには、以下のような ABGR 値を使用します。

A = 0

B = 255

G = 0

R = 0

この色の 16 進数値は 0x00FF0000 です。10 進数値は 16711680 です。

次の表は、使用できる色の例を示しています。

色	16 進数値	10 進数値
白	0x00FFFFFF	16777215
黒	0x00000000	0
青	0x00FF0000	16711680
赤	0x000000E1	225
緑	0x0000FF00	65280

Alarm Viewer コントロールの ActiveX メソッドの使用

Alarm Viewer コントロールの ActiveX メソッドをスクリプトで使用すると、以下の操作を行えます。

- アラームを確認する
- アラームを非表示にする
- アラームに関する情報を取得する
- アラーム クエリーを実行する
- 表示を移動および停止する
- アラーム レコードをソートする
- 特定のアラームを選択する
- ショートカットメニュー、[バージョン情報] ダイアログ ボックス、および [アラームの統計] ダイアログ ボックスを表示します。

メソッドの呼び出しの詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[ActiveX コントロールのスクリプト](#)」を参照してください。

リアルタイムでのアラームの確認

以下のメソッドを使用して、ランタイム中にアラームを確認します。

- [AckSelected\(\) メソッド](#)
- [AckAll\(\) メソッド](#)

- [AckVisible\(\) メソッド](#)
- [AckSelectedGroup\(\) メソッド](#)
- [AckSelectedTag\(\) メソッド](#)
- [AckSelectedPriority\(\) メソッド](#)
- [AckGroup\(\) メソッド](#)
- [AckPriority\(\) メソッド](#)
- [AckTag\(\) メソッド](#)

AckSelected() メソッド

ランタイムで Alarm Viewer コントロールで選択されたアラームを確認します。

構文

```
Object.AckSelected (Comment)
```

パラメータ

Comment

アラーム確認コメントです。

例

Tag1 がメッセージ型タグ変数として定義され、コントロールの名前は AlarmViewerCtrl1 です。

```
Tag1 = "Alarm Comment";  
#AlarmViewerCtrl1.AckSelected (Tag1);
```

AckAll() メソッド

現在のアラーム クエリーにあるすべてのアラームを確認します。Alarm Viewer コントロールの表示領域は制限されているため、AckAll() メソッドでは画面に表示されていないアラームも確認します。

構文

```
Object.AckAll (Comment)
```

パラメータ

Comment

アラーム確認コメントです。

例

Tag1 がメッセージ型タグ変数として定義され、コントロールの名前は AlarmViewerCtrl1 です。

```
Tag1 = "Alarm Comment";  
#AlarmViewerCtrl1.AckAll (Tag1);
```

AckVisible() メソッド

Alarm Viewer コントロールに現在表示されているアラームのみを確認します。

構文

```
Object.AckVisible (Comment)
```

パラメータ

Comment

アラーム確認コメントです。

例

Tag1 がメッセージ型タグ変数として定義され、コントロールの名前は AlarmViewerCtrl1 です。

```
Tag1 = "Alarm Comment";  
#AlarmViewerCtrl1.AckVisible (Tag1);
```

AckSelectedGroup() メソッド

選択した 1 つまたは複数のアラームと同じグループ名を持つすべてのアラームを確認します。

構文

```
Object.AckSelectedGroup (Comment)
```

パラメータ

Comment

アラーム確認コメントです。

例

Tag1 がメッセージ型タグ変数として定義され、コントロールの名前は AlarmViewerCtrl1 です。

```
Tag1 = "Alarm Comment";  
#AlarmViewerCtrl1.AckSelectedGroup (Tag1);
```

AckSelectedTag() メソッド

選択した 1 つまたは複数のアラームと同じタグ変数、グループ名、および優先度を持つすべてのアラームを確認します。

構文

```
Object.AckSelectedTag (Comment)
```

パラメータ

Comment

アラーム確認コメントです。

例

Tag1 がメッセージ型タグ変数として定義され、コントロールの名前は AlarmViewerCtrl1 です。

```
Tag1 = "Alarm Comment";  
#AlarmViewerCtrl1.AckSelectedTag (Tag1);
```

AckSelectedPriority() メソッド

選択した 1 つまたは複数のアラームと同じ優先度範囲を持つすべてのアラームを確認します。

構文

```
Object.AckSelectedPriority (Comment)
```

パラメータ

Comment

アラーム確認コメントです。

例

Tag1 がメッセージ型タグ変数として定義され、コントロールの名前は AlarmViewerCtrl1 です。

```
Tag1 = "Alarm Comment";  
#AlarmViewerCtrl1.AckSelectedPriority (Tag1);
```

AckGroup() メソッド

指定のグループ名およびプロバイダに対するすべてのアラームを確認します。

構文

```
Object.AckGroup(ApplicationName, GroupName, Comment)
```

パラメータ

ApplicationName

たとえば、\\node1\Intouch などアプリケーション名です。

GroupName

グループの名前です。たとえば、Turbine です。

Comment

アラーム確認コメントです。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.AckGroup ("\\Intouch", "Turbine", "Turbine acknowledgement Comment");
```

AckPriority() メソッド

同じプロバイダ名とグループ名を持つ指定した優先度範囲のすべてのアラームを確認します。

構文

```
Object.AckPriority(ApplicationName, GroupName, FromPriority, ToPriority, Comment)
```

パラメータ

ApplicationName

アプリケーションの名前です。たとえば、\\node1\Intouch です。

GroupName

グループの名前です。たとえば、Turbine です。

FromPriority

アラームの開始優先度です。たとえば、100 です。

ToPriority

アラームの終点優先度です。たとえば、900 です。

Comment

アラーム確認コメントです。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.AckPriority ("\\Intouch", "Turbine", 100, 900, "Turbine acknowledgement Comment");
```

AckTag() メソッド

指定した優先度範囲で同じプロバイダ名およびグループ名を持つ指定のタグ変数のアラームを確認します。

構文

```
Object.AckTag(ApplicationName, GroupName, tag, FromPriority, ToPriority, Comment)
```

パラメータ

ApplicationName

たとえば、\\node1\Intouch などアプリケーション名です。

GroupName

グループの名前です。たとえば、Turbine です。

tag

アラーム タグ変数の名前です。たとえば、Valve1 です。

FromPriority

アラームの開始優先度です。たとえば、100 です。

ToPriority

アラームの終点優先度です。たとえば、900 です。

Comment

アラーム確認コメントです。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.AckTag ("\\Intouch", "Turbine", "Valve1", 100, 900, "Turbine  
acknowledgement Comment");
```

リアルタイムでのアラームの非表示

以下のメソッドを使用して、ランタイム中にアラームを非表示にします。

- [ShowSuppression\(\) メソッド](#)
- [SuppressSelected\(\) メソッド](#)
- [SuppressAll\(\) メソッド](#)
- [SuppressVisible\(\) メソッド](#)
- [SuppressSelectedGroup\(\) メソッド](#)
- [SuppressSelectedTag\(\) メソッド](#)
- [SuppressSelectedPriority\(\) メソッド](#)
- [UnSuppressAll\(\) メソッド](#)
- [SuppressGroup\(\) メソッド](#)
- [SuppressPriority\(\) メソッド](#)
- [SuppressTag\(\) メソッド](#)

ShowSuppression() メソッド

非表示にされているすべてのアラームが含まれる [非表示中のアラーム] ダイアログ ボックスを表示します。

構文

```
Object.ShowSuppression()
```

例

コントロールの名前は AlarmViewerCtrl1 です。


```
#AlarmViewerCtrl1.ShowSuppression();
```

SuppressSelected() メソッド

選択したアラームの現在および今後の発生を非表示にします。

構文

```
Object.SuppressSelected()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SuppressSelected();
```

SuppressAll() メソッド

すべてのアクティブなアラームの現在および今後の発生を非表示にします。

構文

```
Object.SuppressAll()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SuppressAll();
```

SuppressVisible() メソッド

表示されているアラームの現在および今後の発生を非表示にします。

構文

```
Object.SuppressVisible()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SuppressVisible();
```

SuppressSelectedGroup() メソッド

選択した 1 つまたは複数のアラームと同じグループおよびプロバイダに属すアラームの現在および今後の発生を非表示にします。

構文

```
Object.SuppressSelectedGroup()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SuppressSelectedGroup();
```

SuppressSelectedTag() メソッド

同じグループ名、プロバイダ名、および優先度範囲を持つ選択した 1 つまたは複数のアラームの同じタグ変数名に属す任意のアラームの現在および今後の発生を非表示にします。

構文

```
Object.SuppressSelectedTag()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SuppressSelectedTag();
```

SuppressSelectedPriority() メソッド

1 つまたは複数の選択したアラームの同じ優先度範囲に属す任意のアラームの現在および今後の発生を非表示にします。

構文

```
Object.SuppressSelectedPriority()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SuppressSelectedPriority();
```

UnSuppressAll() メソッド

アラームの非表示をクリアします。

構文

```
Object.UnSuppressAll()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.UnSuppressAll();
```

SuppressGroup() メソッド

指定のグループ名に属す任意のアラームの現在および今後の発生を非表示にします。

構文

```
Object.SuppressGroup(ApplicationName, GroupName)
```

パラメータ

ApplicationName

たとえば、\\node1\Intouch などアプリケーション名です。

GroupName

グループの名前です。たとえば、Turbine です。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SuppressGroup ("\\Intouch", "Turbine");
```

SuppressPriority() メソッド

同じプロバイダ名およびグループ名を持つ指定した優先度範囲の任意のアラームの現在および今後の発生を非表示にします。

構文

```
Object.SuppressPriority(ApplicationName, GroupName, FromPriority, ToPriority)
```

パラメータ

ApplicationName

たとえば、\\node1\Intouch などアプリケーション名です。

GroupName

グループの名前です。たとえば、Turbine です。

FromPriority

アラームの開始優先度です。たとえば、100 です。

ToPriority

アラームの終点優先度です。たとえば、900 です。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SuppressPriority ("\\Intouch", "Turbine", 100, 900);
```

SuppressTag() メソッド

指定した優先度範囲の指定したタグ変数名またはグループ名によって発行される任意のアラームの現在および今後の発生を非表示にします。

構文

```
Object.SuppressTag(ApplicationName, GroupName, tag, FromPriority, ToPriority)
```

パラメータ**ApplicationName**

たとえば、\\node1\\Intouch などのアプリケーション名です。

GroupName

グループの名前です。たとえば、Turbine です。

tag

アラーム タグ変数の名前です。たとえば、valve 1 です。

FromPriority

アラームの開始優先度です。たとえば、100 です。

ToPriority

アラームの終点優先度です。たとえば、900 です。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SuppressTag ("\\Intouch", "Turbine", "Valve1", 100, 900);
```

特定のアラームに関する情報の取得

アラームに関する情報を取得するには、GetItem() 関数を使用します。

GetItem() メソッド

指定された行とカラムのデータを文字列として返します。

構文

```
Object.GetItem(Integer, Message)
```

パラメータ**Integer**

コントロールの特定の行を評価する整数式です。

Message

コントロールのカラム名を評価する文字列式です。

例

コントロール名は `AlmDbView1` です。タグはメッセージ型タグとして定義されます。

```
tag = #AlmDbView1.GetItem(1, "Group");
```

アラーム クエリーの実行

クエリーを実行するには、以下のメソッドを使用します。

- [ShowQueryFavorites\(\) メソッド](#)
- [Requery\(\) メソッド](#)
- [ApplyQuery\(\) メソッド](#)
- [ApplyDefaultQuery\(\) メソッド](#)
- [SetQueryByName\(\) メソッド](#), [SetQueryByName\(\) メソッド](#)

ShowQueryFavorites() メソッド

QueryFavoritesFile プロパティに .xml 形式で有効なクエリー設定ファイルが含まれている場合に、[クエリー設定] ダイアログ ボックスを表示します。

構文

```
Object.ShowQueryFavorites()
```

例

コントロールの名前は `AlarmViewerCtrl1` です。

```
#AlarmViewerCtrl1.ShowQueryFavorites();
```

Requery() メソッド

アラーム プロバイダを再びクエリーします。

構文

```
Object.Requery()
```

例

コントロールの名前は `AlarmViewerCtrl1` です。

```
#AlarmViewerCtrl1.Requery();
```

ApplyQuery() メソッド

アラーム クエリー、優先度始点、優先度終点、クエリー対象のアラームの状況、および取得するアラームのタイプなど、パラメータによって指定されているようにクエリーを実行します。

構文

```
Object.ApplyQuery(AlarmQuery, FromPriority, ToPriority, State, Type)
```

パラメータ

AlarmQuery

アラーム クエリーです。次に例を示します。 `\InTouch!$System`

FromPriority

アラームの開始優先度です。たとえば、100 です。

ToPriority

アラームの終点優先度です。たとえば、900 です。

状態

表示するアラームのタイプを指定します。たとえば、"UnAck" またはメッセージ型タグ変数です。有効な状態は、All、UnAck、および Ack です。

タイプ

クエリーのタイプを指定します。たとえば、Historical（履歴）（履歴アラーム）または Summary（サマリ）（サマリ アラーム）です。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.ApplyQuery ("\\InTouch!$System",100,900,"All", "Historical");
```

ApplyDefaultQuery() メソッド

デザイン時に指定されているように、優先度始点、優先度終点、アラーム状況、クエリータイプ、およびアラームクエリーのプロパティを使用してクエリーを実行します。デフォルトのプロパティは開発時でのみ変更でき、その他のアラームクエリーによって上書きされることはありません。

構文

```
Object.ApplyDefaultQuery()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.ApplyDefaultQuery();
```

SetQueryByName() メソッド

渡されるクエリー名によって指定されている通りに現在のクエリーを設定します。クエリーはクエリー設定ファイルに存在する必要があります。

構文

```
Object.SetQueryByName(QueryName)
```

パラメータ

QueryName

クエリー設定を使用して作成されたクエリーの名前です。たとえば、Turbine Queries です。

例

コントロールの名前は AlarmTreeViewCtrl1 です。

```
#AlarmTreeViewCtrl1.SetQueryByName("Turbine Queries");
```

表示の移動と停止

表示を移動または停止するには、以下の関数を使用します。

- [MoveWindow\(\) メソッド](#)
- [FreezeDisplay\(\) メソッド](#)

MoveWindow() メソッド

コントロールのアラームを指定した方法でスクロールします。

構文

```
Object.MoveWindow(Option, Repeat)
```

パラメータ

オプション

実行する動作のタイプです。

タイプ	説明
LineDn	次の行へ移動。Repeat パラメータによって、スクロールされる行数が制御されます。
LineUp	前の行へ移動。Repeat パラメータによって、スクロールされる行数が制御されます。
PageDn	次ページへ移動。Repeat パラメータによって、スクロールされるページ数が制御されます。
PageUp	前ページへ移動。Repeat パラメータによって、スクロールされるページ数が制御されます。
Top	コントロールの一番上へ移動。
Bottom	コントロールの一番下へ移動。
PageRt	右ページへ移動。Repeat パラメータによって、スクロールされるページ数が制御されます。
PageLf	左ページへ移動。Repeat パラメータによって、スクロールされるページ数が制御されます。
Right	右へスクロール。Repeat パラメータによって、スクロールされるカラム数が制御されます。
Left	左へスクロール。Repeat パラメータによって、スクロールされるカラム数が制御されます。
Home	コントロールの一番上の行の最も左のカラムまでスクロールします。

Repeat

操作が繰り返される回数です。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.MoveWindow ("Bottom", 0);
#AlarmViewerCtrl1.MoveWindow ("LineUp", 3);
#AlarmViewerCtrl1.MoveWindow ("PageLf", 7);
```

FreezeDisplay() メソッド

表示を停止します。

構文

```
Object.FreezeDisplay(Freeze)
```

パラメータ

Freeze

True = 表示を停止します。

False = 表示の停止を解除します。

例

Tag1 はメモリ論理型タグ変数として定義されており、コントロールの名前は AlarmViewerCtrl1 です。

```
Tag1 = 1;  
#AlarmViewerCtrl1.FreezeDisplay(Tag1);
```

アラーム レコードのソート

アラーム レコードをソートするには、以下の関数を使用します。

- [ShowSort\(\) メソッド](#), [ShowSort\(\) メソッド](#)
- [SetSort\(\) メソッド](#)

ShowSort() メソッド

SortMenu プロパティが有効である場合、[セカンダリ ソート] ダイアログ ボックスを表示します。

構文

```
Object.ShowSort()
```

例

コントロールの名前は AlmDbView1 です。

```
#AlmDbView1.ShowSort();
```

SetSort() メソッド

SortColumn プロパティおよび SortOrder プロパティで指定されているソート条件を設定します。

構文

```
Object.SetSort()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SetSort();
```

追加情報の表示

[バージョン情報] ダイアログ ボックスと [アラームの統計] ダイアログ ボックスを表示するには、以下の関数を使用します。

- [AboutBox\(\) メソッド](#), [AboutBox\(\) メソッド](#), [AboutBox\(\) メソッド](#)
- [ShowStatistics\(\) メソッド](#)

AboutBox() メソッド

[バージョン情報] ダイアログ ボックスを表示します。

構文

```
Object.AboutBox()
```

例

コントロールの名前は AlarmPareto1 です。

```
#AlarmPareto1.AboutBox();
```

ShowStatistics() メソッド

[アラームの統計] ダイアログ ボックスを表示します。

構文

```
Object.ShowStatistics()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.ShowStatistics();
```

特定のアラームの選択

特定のアラームを選択するには、以下の関数を使用します。

- [SelectGroup\(\) メソッド](#)
- [SelectPriority\(\) メソッド](#)
- [SelectTag\(\) メソッド](#)
- [SelectAll\(\) メソッド](#)
- [SelectItem\(\) メソッド](#)
- [UnSelectAll\(\) メソッド](#)

SelectGroup() メソッド

同じアラーム グループ名とプロバイダ名が含まれるすべてのアラームを選択します。

構文

```
Object.SelectGroup(ApplicationName, GroupName)
```

パラメータ

ApplicationName

たとえば、\\node1\Intouch などアプリケーション名です。

GroupName

グループの名前です。たとえば、Turbine です。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SelectGroup ("\\Intouch", "Turbine");
```

SelectPriority() メソッド

同じプロバイダ名とグループ名を持つ指定した優先度範囲のアラームをすべて選択します。

構文

```
Object.SelectPriority(ApplicationName, GroupName, FromPriority, ToPriority)
```

パラメータ

ApplicationName

たとえば、\\node1\Intouch などアプリケーション名です。

GroupName

グループの名前です。たとえば、Turbine です。

FromPriority

アラームの開始優先度です。たとえば、100 です。

ToPriority

アラームの終点優先度です。たとえば、900 です。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SelectPriority ("\\Intouch", "Turbine", 100, 900);
```

SelectTag() メソッド

特定のプロバイダ、グループ、またはタグ変数からのすべてのアラームを選択します。優先度範囲または 1～999 の値を指定できます。

構文

```
Object.SelectTag(ApplicationName, GroupName, tag, FromPriority, ToPriority)
```

パラメータ**ApplicationName**

たとえば、\\node1\\Intouch などアプリケーション名です。

GroupName

グループの名前です。たとえば、Turbine です。

tag

アラーム タグ変数の名前です。たとえば、valve 1 です。

FromPriority

アラームの開始優先度です。たとえば、100 です。

ToPriority

アラームの終点優先度です。たとえば、900 です。

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.SelectTag ("\\Intouch", "Turbine", "Valve1",100, 900);
```

SelectAll() メソッド

表示のすべてのアラームの選択を切り替えます。アラーム表示の表示領域は制限されているため、SelectAll() 関数では画面に表示されていないアラームも選択します。

構文

```
Object.SelectAll()
```

例

```
#AlarmViewerCtrl1.SelectAll();
```

SelectItem() メソッド

指定した行にあるアラーム レコードの選択を切り替えます。

構文

```
Object.SelectItem(LineNumber)
```

パラメータ

RowNumber

選択するアラーム レコードの行番号である整数値です。コントロールの最初の行は 0 です。

例

コントロールの名前は AlarmViewerCtrl1 で、Tag1 はメモリ整数型として定義されています。Alarm Viewer コントロールの 10 行目のアラーム レコードの選択を切り替えます。

```
Tag1 = 9;  
#AlarmViewerCtrl1.SelectItem (Tag1);
```

UnSelectAll() メソッド

選択したすべてのレコードを選択解除します。

構文

```
Object.UnSelectAll()
```

例

コントロールの名前は AlarmViewerCtrl1 です。

```
#AlarmViewerCtrl1.UnSelectAll();
```

ランタイム時のコンテキスト メニューの表示

ランタイムでショートカット メニューを表示するには、ShowContext() メソッドを使用します。

ShowContext() メソッド

RefreshMenu、ResetMenu、SortMenu の 3 つのプロパティのうち 1 つを有効にすると、ショートカット メニューが表示されます。

構文

```
Object.ShowContext()
```

例

コントロールの名前は AlmDbView1 です。

```
#AlmDbView1.ShowContext();
```

メソッドとプロパティを使用するときのエラー処理

SilentMode プロパティを使用して、ランタイム中にエラーを非表示にできます。SilentMode プロパティを 1 に設定すると、Alarm Viewer コントロールではランタイム中にエラー メッセージは表示されません。0 に設定すると、Alarm Viewer コントロールではエラー メッセージが表示されます。エラー メッセージは、常に Operations Control Log Viewer に送信されます。

ActiveX イベントを使用したスクリプトのトリガ

マウス クリックやダブルクリックなどの Alarm Viewer コントロール イベントに、QuickScript を割り当てることができます。イベントが発生すると、QuickScript が実行されます。

Alarm Viewer コントロールは、以下のイベントをサポートしています。

- クリック
- DoubleClick
- NewAlarm

- ShutDown
- StartUp

Click イベントは ClicknRow というパラメータを 1 つ持ち、これはランタイム時にクリックされた行を識別します。

DoubleClick イベントは DoubleClicknRow というパラメータを 1 つ持ち、これはランタイム時にダブルクリックされた行を識別します。

Click イベントおよび DoubleClick イベントはゼロを基準としています。Click イベントおよび DoubleClick イベントがユーザーに対して発行されると、表示される行数は 0 から始まります。

注: ユーザー インターフェイス メソッドが StartUp イベントから呼び出される場合、コントロールはまだ表示されていないため、Alarm Viewer コントロールでは無視されます。このようなメソッドには、ShowSort()、ShowContext()、GetSelectedItem()、GetNext()、GetPrevious()、および AboutBox() があります。

ActiveX イベントのスクリプトの詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[ActiveX コントロールのスクリプト](#)」を参照してください。

新しいアラームが検出されたときのスクリプトの実行

Alarm Viewer コントロールで新しい未確認アラーム（つまり、平常状態から未確認状態に移行し、コントロールのクエリーや優先度のフィルタ条件に一致するアラーム）が検出された場合に、ActiveX イベント スクリプトを実行するように Alarm Viewer コントロールを設定できます。

NewAlarmEventMode プロパティは、NewAlarm イベントのトリガを制御します。

- NewAlarmEventMode プロパティを 0 に設定すると、NewAlarm イベントはトリガされません。これがデフォルトです。
- NewAlarmEventMode プロパティを 1 に設定すると、新しいアラームが発生します。
 - イベントがトリガされます。
 - NewAlarm イベントに関連付けられた ActiveX イベント スクリプトが実行されます。
 - NewAlarmEventMode プロパティが 0 に設定されます。

後続のイベントを処理するには、NewAlarmEventMode プロパティを 1 に戻す必要があります。

この設定は、状況が改善されて確認されるまでは実行するべきではない動作をアプリケーションで実行する場合に使用されます。たとえば、イベントがトリガされた場合に、アラームが確認されるまで ActiveX スクリプトでアラーム音を鳴らすことができます。新しいアラームを次回受け取ったときに、再びアラーム音を再生できます。

- NewAlarmEventMode プロパティを 2 に設定すると、NewAlarm イベントがアクティブとなり、新しい未確認アラームが少なくとも 1 つ受け取られると続けてトリガされます。後続のイベントを処理するために値を変更する必要はありません。1 秒間に受け取る新しい追加のアラームの数に関わらず、1 秒間にトリガされる新しいイベントは最大でも 1 回のみです。

リアルタイムでのアラームの確認

タグ データが通常状態からアラーム状態に移行すると、アラームの新しいインスタンスが生成されます。InTouch の分散アラーム システムでは、各アラーム インスタンスの以下の状態を追跡しています。

- タグが初めてアラーム状態になったとき
- アラームがマルチステートのアラームである場合、アラームがサブステートへの移行を行ったとき

- アラームが通常状態に戻ったとき
- アラームが確認を待機しているかどうか
- アラームが確認されたとき

アラーム インスタンスのライフサイクルは、アラームに関連付けられているタグ値が通常状態、つまりアラーム解除の状態に戻った時点で終了します。その後でアラーム状態に移行すると、新しいアラーム インスタンスが発生します。

アラーム確認モデルの理解

InTouch HMI では、3 つのアラーム確認モデルをサポートしています。

- 条件指向のアラームの場合、確認の時点までのアラーム状況へのすべてのエントリに対して確認はカウントされます。
- 拡張サマリ のアラームの場合、アラーム状況やサブステートへの移行、または平常状態への復帰などの場合でも、確認は特定の移行に対してのみ行われます。アラーム全体が確認されたとみなされるためには、異なるアラーム サブステートへのすべての移行が確認される必要があります。
- イベント指向のアラーム（OPC など）の場合、最後のアクティベーション イベントを参照している場合のみ確認が受け入れられます。

条件確認アラーム モデル

条件指向のアラームの場合、確認の時点までのアラーム状況へのすべてのエントリに対して確認はカウントされます。

確認はアラームのインスタンスに対して行われます。アラーム インスタンスは初めてアラーム状況になると、確認を待機します。アラームが確認され、新しいアラーム サブステートにその後で移行する場合（たとえば「Hi」から「HiHi」に移行）、別の確認の待機を開始します。確認は受け取られるたびに、受け入れられ、これまでに発生したアラームのすべての移行に適用されます。

最後のインスタンスが確認されたときに、アラームは確認されたとみなされます。

拡張サマリ アラーム モデル

拡張サマリ のアラームの場合、アラーム状況やサブステートへの移行、または平常状態への復帰などの場合でも、確認は特定の移行に対してのみ行われます。アラーム全体が確認されたとみなされるためには、異なるアラーム サブステートへのすべての移行が確認される必要があります。

アラーム状況への最初のエントリ、および平常状態への復帰は個別に確認される必要があります。

新しいアラーム サブステートへの移行はすべて確認される必要のある新しい発生と見なされ、平常への復帰も確認する必要があります。サブステート間の移行は、アイテムが以前平常であったときのアラーム状況への最初のエントリで始まる「平常への復帰グループ」に属すると見なされます。

アイテムが平常に戻った後で再びアラーム状況に入ると、新しい平常への復帰グループが作成されます。

それぞれの移行は個別に明示的に確認される必要があり、アラームはアイテムが平常に戻り、保留状態にある平常への復帰グループがすべて確認されて初めて確認されたと見なされます。

注意：「サマリ」という単語は、「確認の待機状態」を意味します。このモデルのアラームは、「リングバック」アラームとも呼ばれます。

拡張サマリ アラーム レコード

拡張サマリ アラームの場合、アラームが発生すると、アラーム表示オブジェクトにレコードが生成され、アラーム状況が発生したことが表示されます。レコードには、アラームの日付／時間スタンプが表示されます。このレコードは、オペレータがアラームを確認し、平常への復帰状態になるまで表示されたまま残ります。アラームが確認される前に平常への復帰状態が発生した場合、アラーム オブジェクトに2つのレコードが表示されます。

たとえば、ボイラーの温度が上限値を超えて、アラームがトリガされるとします。その後で、オペレータがアラームを確認する前にボイラーの温度が通常の範囲内に戻ります。アラーム システムは上限アラームの発生を示すレコードと、アラームが未確認であることを示す別のレコードを生成します。

拡張サマリ アラームの使用

タグ変数を定義する際に確認モデルとして**【拡張サマリ】**を選択した場合、アラームをトリガする状況がすでに平常状態に戻っていたとしても、オペレータはそのアラームが発生したことを確認する必要があります。アラームを確認すると、アラーム エントリの色は変更されますが、時刻スタンプは変更されません。アラームが確認され、平常状態に戻ると、アラームの表示だけがクリアされます。

注意：拡張サマリの確認モードを使用してタグ変数を定義する場合、**【アラームのプロパティ】** ダイアログ ボックスの**【アラーム復旧時に確認する】** オプションはタグ変数には適用されません。

イベント指定アラーム モデル

イベント指向のアラーム（OPC など）の場合、最後のアクティベーション イベントを参照している場合のみ確認が受け入れられます。

アラーム インスタンスは初めてアラーム状況になると、確認の待機を開始します。インスタンスが確認され、その後で新しいアラーム サブステートに移行する場合、別の確認の待機を開始します。それ以降の移行には連番が割り当てられます。確認は応答している移行の連番を付ける必要があります。

確認は最後の移行に対してのみ受け入れられます。受け入れられる場合、これまでに発生したアラームのすべての移行に対して適用されます。最後のインスタンスが確認されたときに、アラームは確認されたとみなされます。

拒否された確認は、診断のためにログ記録されますが、システムでは追跡されません。

イベント指向のモデルでは、アラームが2つの状態の間で保留となっている場合、確認は現在の情報に応答します。遅延時間の短いシステムでは、このモデルは条件指向のアラームのように見えます。インターネットなどその他の環境では、このモデルの特徴が重要となります。

ランタイムでのタグ変数の確認モデルの確認

タグ変数に関連付けられた確認モデルの監視には、**.AlarmAckModel** ドットフィールドを使用します。

.AlarmAckModel ドットフィールド

タグ変数に関連付けられている確認モデルを以下のように監視します。

0 = 条件（デフォルト）

1 = イベント指向

2 = 拡張サマリ

カテゴリ

アラーム

使用法

```
Tagname.AlarmAckModel
```

パラメータ

Tagname

論理型、整数型、実数型、間接論理型、およびアナログ型の任意のタグ変数

備考

このドットフィールドのデフォルト値は0です（条件確認モデル）。

データタイプ

アナログ型（読み取り専用）

有効値

0、1、または2

例

PumpStation タグ変数がイベント アラームに関連付けられている場合に、この IF-THEN ステートメントの本文が処理されます。

```
IF (PumpStation.AlarmAckModel == 1) THEN  
    MyAlarmMessage="PumpStation はイベント アラームです";  
ENDIF;
```

参照項目

.Alarm、.Ack、.UnAck、.AckDev、.AckDSC、.AckROC

アラームを確認するためのドットフィールドの使用

現在のすべてのアラーム、選択したアラーム、または指定タイプのアラームのみを確認するドットフィールドを使用するスクリプトを作成できます。

アラームまたはアラーム グループの確認

指定したローカル タグ変数のアラームか、指定したアラーム グループ内のアラームを確認する以下のドットフィールドを使用するスクリプトを作成できます。

- [.Ack ドットフィールド](#)
- [.UnAck ドットフィールド](#)

これらのドットフィールドを使用して、Application Server から生成されるアラームは確認できません。

実行している InTouch アプリケーションのすべてのローカル アラームを確認するには、\$System アラーム グループを適切な .Ack ドットフィールドとの組み合わせで使用します。

.Ack ドットフィールド

すべてのタイプのローカル アラームのアラーム確認状況を監視または制御します。

カテゴリ

アラーム

使用法

```
TagName.Ack=1;
```


パラメータ

TagName

論理型、整数型、実数型、間接論理型、およびアナログ型のタグ変数、またはアラーム グループ

備考

指定したタグ変数またはアラーム グループに関連付けられるすべての未処理のアラームを確認するには、このドットフィールドの値を **1** に設定します。指定したタグ変数がアラーム グループである場合、指定したグループ内のタグ変数に関連付けられるすべての未確認アラームが確認されます。指定したタグ変数がアラーム グループ以外の場合、そのタグ変数に関連付けられている未確認アラームだけが確認されます。**.Ack** ドットフィールドに **1** 以外の値を設定しても意味がありません。

データタイプ

論理型（読み取り／書き込み）

有効値

1

例

以下のステートメントでは、**Tag1** のタグ変数に関連付けられているアラームが確認されます。

```
Tag1.Ack=1;
```

この次の例は、**PumpStation** アラーム グループ内のすべての未確認アラームを確認するために使用されます。

```
PumpStation.Ack=1;
```

注意：**.Ack** ドットフィールドには、**.UnAck** と呼ばれる逆のドットフィールドがあります。未確認アラームが発生すると、**.UnAck** は **1** に設定されます。**.UnAck** はその後でアニメーション リンクや条件スクリプトで使用して、未確認アラームのアナランシエーターをトリガできます。

参照項目

Alarm、UnAck、AckDev、AckROC、AckDSC、AckValue、AlarmAckModel

.UnAck ドットフィールド

ローカルアラームのアラーム確認状況を監視または制御します。

カテゴリ

アラーム

使用法

```
TagName.UnAck=0;
```

パラメータ

TagName

論理型、整数型、実数型、間接論理型、およびアナログ型のタグ変数、またはアラーム グループ

備考

指定したタグ変数またはアラーム グループに関連付けられるすべての未処理のアラームを確認するには、このドットフィールドの値を **0** に設定します。指定したタグ変数がアラーム グループである場合、指定したグループ内のタグ変数に関連付けられるすべての未確認アラームが確認されます。指定した

タグ変数がそれ以外のタイプの場合、そのタグ変数に関連付けられる未確認のアラームだけが確認されます。このドットフィールドに 0 以外の値を設定しても意味がありません。

データ タイプ

論理型（読み取り／再設定）専用

有効値

0

例

以下のステートメントでは、Tag1 タグ変数に関連付けられているアラームが確認されます。

```
Tag1.UnAck=0;
```

以下のステートメントは、PumpStation という名前のアラーム グループ内のすべての未確認アラームを確認します。

```
PumpStation.UnAck=0;
```

.UnAck には .Ack と呼ばれる逆ドットフィールドがあります。アラームが確認されている場合、.Ack ドットフィールドの値は 1 に設定されます。

参照項目

.Ack、Ack()、.Alarm、.AlarmAckModel

値型アラームの確認

.AckValue ドットフィールドを使用して、指定したローカル タグ変数のすべての値型アラームか、指定したアラーム グループ内のすべての値型アラームを確認するスクリプトを作成できます。

.AckValue ドットフィールド

ローカルの値型アラームの確認を監視および制御します。

使用法

```
TagName.AckValue=1;
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

指定したタグ変数／グループに関連付けられる未処理の値型アラームをすべて確認するには、.AckValue ドットフィールドを 1 に設定します。指定したタグ変数がアラーム グループである場合、指定したグループ内のタグ変数に関連付けられるすべての未確認の値型アラームが確認されます。指定したタグ変数が任意のタイプの場合、そのタグ変数に関連付けられる未確認の値型アラームだけが確認されます。

このドットフィールドに 1 以外の値を設定しても意味がありません。

データタイプ

論理型（読み取り／書き込み）

有効値

1

例

以下のステートメントでは、Tag1 タグ変数に関連付けられている値型アラームが確認されます。

```
Tag1.AckValue=1;
```

以下の例では、PumpStation という名前のアラーム グループ内のすべての未確認の値型アラームが確認されます。

```
PumpStation.AckValue=1;
```

間接アラーム グループ（GroupVar の使用）を使用して、値型アラームを確認できます。たとえば、以下のような割り当てを使用します。

```
StationAlarms.Name = "PumpStation";
```

ここで StationAlarms は、アラーム グループ型タグ変数として定義され、PumpStation に関連付けられます。このように、以下のステートメントは、StationAlarms アラーム グループ変数タグに現在関連付けられているアラーム グループ PumpStation 内の未確認の値型アラームを確認するために使用される以外は、上の例と類似しています。

```
StationAlarms.AckValue=1;
```

参照項目

.Alarm、.AlarmValue、.Ack、.UnAck、.AckDev、.AckDSC、.AckROC、.AlarmAckModel

論理値アラームの確認

指定したタグ変数の論理値アラームまたは指定したアラーム グループのすべての論理値アラームを確認するために .AckDsc ドットフィールドを使用するスクリプトを作成できます。

.AckDsc ドットフィールド

指定したタグ変数の論理値アラームか、指定したアラーム グループのすべての論理値アラームを確認します。

使用法

```
TagName.AckDsc=1;
```

パラメータ

TagName

論理型タグ変数に割り当てられている名前か、アラーム グループの名前です。

備考

指定したタグ変数またはアラーム グループに関連付けられるアクティブな論理値アラームを確認するには、1 に設定します。指定したタグ変数がアラーム グループである場合、指定したグループ内のタグ変数に関連付けられるすべての未確認の論理値アラームが確認されます。指定したタグ変数がアラーム グループ以外の場合、そのタグ変数に関連付けられている未確認の論理値アラームだけが確認されます。このドットフィールドに 1 以外の値を設定しても意味がありません。

データタイプ

論理型（読み取り／書き込み）

有効値

0 または 1

例

以下のステートメントでは、Tag1 にアクティブなアラームが関連付けられているかどうかを検証します。

```
IF (Tag1.AlarmDsc == 1) THEN  
    MyAlarmMessage="現在ポンプ ステーションにアラームが発生しています!";  
ENDIF;
```

このドットフィールドは、.Ack ドットフィールドや .UnAck ドットフィールドにはリンクされていません。このため、アクティブなアラームが確認された場合も、このドットフィールドの値は 1 のまま維持されます。

参照項目

.Alarm、.AlarmDSC、.Ack、.UnAck、.AckDev、.AckDSC、.AckROC、.AckValue、.AlarmAckModel

偏差アラームの確認

指定したローカル タグ変数の小偏差アラームまたは大偏差アラーム、または指定したアラーム グループの偏差アラームを確認するために .AckDev ドットフィールドを使用するスクリプトを作成できます。

.AckDev ドットフィールド

指定したローカル タグ変数の小偏差アラームまたは大偏差アラーム、または指定したアラーム グループのすべての偏差アラームを確認します。

カテゴリ

Alarm

使用法

```
TagName.AckDev=1;
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

指定したタグ変数またはアラーム グループに関連付けられるすべての未処理の偏差アラームを確認するには、このドットフィールドの値を 1 に設定します。指定したタグ変数がアラーム グループである場合、指定したグループ内のタグ変数に関連付けられるすべての未確認の偏差アラームが確認されます。このドットフィールドに 1 以外の値を設定しても意味がありません。

データタイプ

論理型（読み取り／書き込み）

有効値

1

例

以下のステートメントでは、Tag1 タグ変数に関連付けられている偏差アラームが確認されます。

```
Tag1.AckDev=1;
```

次の例は、PumpStation という名前のアラーム グループ内のすべての未確認の偏差アラームを確認するために使用されます。

```
PumpStation.AckDev=1;
```

参照項目

.Alarm、.AlarmDev、.Ack、.UnAck、.AckDSC、.AckROC、.AckValue、.AlarmAckModel

変化率アラームの確認

指定したローカルタグ変数の変更率アラーム、または指定したアラームグループの変更率アラームを確認するために **.AckROC** ドットフィールドを使用するスクリプトを作成できます。

.AckROC ドットフィールド

指定したローカルタグ変数の変化率アラームまたは指定したアラームグループのすべての変化率アラームを確認します。

使用法

```
TagName.AckROC=1;
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数に割り当てられた名前、またはアラームグループの名前

備考

指定したタグ変数またはアラームグループに関連付けられるすべての未処理の変化率アラームを確認するには、このドットフィールドの値を **1** に設定します。指定したタグ変数がアラームグループの場合、指定したグループ内のタグ変数に関連付けられるすべての未確認の変化率アラームが確認されます。指定したタグ変数がアラームグループ以外の場合は、そのタグ変数に関連付けられている未確認の変化率アラームだけが確認されます。このドットフィールドに **1** 以外の値を設定しても意味がありません。

データタイプ

論理型（読み取り／書き込み）

有効値

1

例

以下のステートメントでは、**Tag1** という名前のタグ変数に関連付けられている変化率アラームが確認されます。

```
Tag1.AckROC=1;
```

この次の例では、**PumpStation** という名前のアラームグループ内のすべての未確認の変化率アラームが確認されます。

```
PumpStation.AckROC=1;
```

参照項目

.Alarm、.AlarmROC、.Ack、.UnAck、.AckDev、.AckDSC、.AckValue、.AlarmAckModel

アラームを確認するためのスクリプト関数の使用

Ack() スクリプト関数を使用して、タグ変数またはグループのすべてのアラームを確認できます。

Alarm Viewer コントロールを使用している場合は、メソッドを使用してアラームを確認できます。詳細については、「[アラームの確認](#)」を参照してください。

分散アラーム表示オブジェクトを使用している場合は、スクリプト関数を使用してアラームを確認します。詳細については、「[アラームの確認](#)」を参照してください。

Ack() 関数

未確認の InTouch アラームを確認します。

カテゴリ

アラーム

構文

```
Ack TagName;
```

引数

TagName

任意の InTouch タグ変数、アラーム グループ、またはグループ変数です。

備考

この関数は、タグ変数またはアラーム グループに適用できます。

例

以下のステートメントは押しボタンで使用して、未確認のアラームを確認するために使用されます。

```
Ack $System; {すべてのアラーム}
```

```
Ack Tagname;
```

```
Ack GroupName;
```

参照項目

almAckAll()、**almAckGroup()**、**almAckTag()**、**almAckDisplay()**、**almAckRecent()**、**almAckPriority()**、**almAckSelect()**、**almAckSelectedGroup()**、**almAckSelectedPriority()**、**almAckSelectedTag()**

タグ値が平常に復帰するときの自動確認の使用

InTouch HMI では、アラーム状態のタグ値が平常状態に復帰する際に、アラームを自動的に確認できます。このオプションは、拡張サマリ アラームには適用されません。

復帰時のアラーム確認を設定するには

1. WindowMaker でアプリケーションを開きます。
2. [ファイル] メニューで、[設定] をポイントし、[アラーム] をクリックします。
アラーム設定画面が表示されます。
3. [全般] 領域で、[アラーム復旧時に確認する] をクリックすると、値が平常状態に復帰 (RTN) したアラームが InTouch HMI によって自動的に確認されます。

Alarms

General

Alarm buffer size: 500 entries

☒ RTN implies ACK

☒ Events enabled

☐ Alarm event retentive

☐ Retain ACK comment as alarm comment

1. [アラーム復帰時に確認する] チェック ボックスをオンにします。
2. [OK] をクリックします。

アラームを確認するためのアラーム クライアントの使用

オペレータは Alarm Viewer コントロールに表示されるアラームを WindowViewer から確認します。

表示の [状況] カラムには、選択したアラーム レコードの現在の確認状況が表示されます。また、レコードのテキストには色が付けられており、確認状況を示しています。

Time	State	Class	Type	Priority	Name	Group
11/17/2006 04:07:19 PM	UNACK_RTN	VALUE	HI	1	ProdLevel	Reactor
11/17/2006 04:10:31 PM	UNACK_RTN	VALUE	HI	1	ReactLevel	Reactor
11/17/2006 04:10:31 PM	UNACK	VALUE	HI	1	ReactTemp	Reactor

Context menu options: Ack Selected, Ack Others, Suppress Selected, Suppress Others, Query Favorites..., Stats..., Suppression..., Freeze, Requery, Sort...

Displaying 1 to 3 of 3 alarms. Default Query 100 % Complete

確認するアラームは、表示から削除されます。

すべてのアラームを確認するには

1. 表示を右クリックし、[その他を確認] をポイントし、適切なコマンドをクリックします。
 - 現在のアラームをすべて確認するには、[全てを確認] をクリックします。
 - 表示されているすべてのアラームを確認するには、[表示部分を確認] をクリックします。

[確認コメント] ダイアログ ボックスが表示されます。

2. オプションの確認コメントを入力して、[OK] をクリックします。

選択したアラームを確認するには

1. 1 つまたは複数のアラームを選択します。
2. 右クリックして、[選択アラームを確認] をクリックします。[確認コメント] ダイアログ ボックスが表示されます。
3. オプションの確認コメントを入力して、[OK] をクリックします。

グループ、タグ変数、または優先度を確認するには

1. アラームを選択します。
2. 表示を右クリックし、[その他を確認] をポイントし、適切なコマンドをクリックします。
 - 選択したアラーム グループに属するすべてのアラームを確認するには、[選択グループを確認] をクリックします。

- 選択したアラームと同じ名前を持つすべてのタグ変数のアラームを確認するには、[選択タグ変数を確認] をクリックします。
- 選択したアラームと同じ優先度を持つすべてのアラームを確認するには、[選択優先度を確認] をクリックします。

[確認コメント] ダイアログ ボックスが表示されます。

3. オプションの確認コメントを入力して、[OK] をクリックします。

アラーム コメントと確認コメントの使用

コメントには、アラーム コメントと確認コメントという 2 種類のコメントがあります。

- アラーム コメントは、新しいアラーム インスタンスが発生したときに設定されます。アラーム コメントに使用される `.AlarmComment` ドットフィールドは、InTouc スクリプトで設定したり、読み取ったりできます。このコメントのデフォルト値は、タグ名ディクショナリのタグ定義で指定します。アラーム コメントの最大文字数は 131 文字です。
- 確認コメントは、アラームを確認するときにオペレータによって提供されます。

確認コメントを使用すると、タグ データベースのアラーム コメントを更新できます。

アラーム確認コメントで `.AlarmComment` ドットフィールドを更新することを許可するには

1. WindowMaker でアプリケーションを開きます。
2. [ファイル] メニューで、[設定] をポイントし、[アラーム] をクリックします。
[アラーム] 画面が表示されます。

General

Alarm buffer size: 500 entries

☒ RTN implies ACK ☐ Alarm Enable retentive

☒ Events enabled ☒ Retain ACK comment as alarm comment

☐ Alarm Latch enabled

3. アラーム確認で入力されたコメントでタグの `.AlarmComment` ドットフィールドとタグ名ディクショナリを更新するには、[確認コメントをアラーム コメントとして保持] チェック ボックスをオンにします。

このチェック ボックスをオンにしない場合、確認コメントは確認したアラームと共に表示されますが（データベース、印刷、および画面）、`.AlarmComment` ドットフィールドは変更されません。

4. [OK] をクリックします。

ランタイム時のアラームの解除

アラームが発生すると、ランタイム オペレータ（またはシステム）がアラームを確認して、アラームを認識したことを示します。Galaxy に対してラッチ機能が有効になっている場合、標準値に戻った確認済みアラームは引き続き表示されます。ラッチが有効な場合、通常状態に戻った確認済みアラームは、LATCHED アラーム状態になります。

LATCHED アラームは、アラームが発生したことを示すために現在のアラーム モードで表示されることがあります。次の場合、アラームは LATCHED 状態になります。

- アラームが UNACK_RTN 状態から確認された場合。

または

- アラームが **ACK** 状態から通常状態に戻った場合。

LATCHED 状態を表示するには、グローバル設定で **LATCHED** 状態を有効にする必要があります。詳細については、「」トピックを参照してください。

LATCHED アラームを解除して、アラーム クライアント コントロール グリッドの現在のモードから **LATCHED** アラームを削除できます。解除された **LATCHED** アラームは、アラーム クライアント コントロールの最近のモードで表示されます。

選択したアラームを解除するには

1. **LATCHED** 状態のアラームを 1 つまたは複数選択します。
2. グリッドの任意の場所を右クリックして、[選択項目を閉じる] をクリックします。

選択したアラームがグリッドから削除されます。

その他のアラームを解除するには

1. **LATCHED** 状態のアラームを 1 つまたは複数選択します。
2. グリッドの任意の場所を右クリックし、[その他を閉じる] をクリックして次のいずれかのオプションをクリックします。
 - [すべて閉じる] をクリックするとアラーム状態のすべてのアラームが解除されます。
 - [表示項目を閉じる] をクリックすると表示されているすべてのアラームが解除されます。
 - [選択したグループを閉じる] をクリックすると、選択した 1 つまたは複数のアクティブなアラームと同じプロバイダ名とグループ名のアラームが解除されます。
 - [選択したタグを閉じる] をクリックすると、選択した 1 つまたは複数のアクティブなアラームと同じプロバイダ名、グループ名、およびタグ名のアラームが解除されます。

関連するアラームがグリッドから削除されます。

ドットフィールドを使用したアラームの解除

ドットフィールドを使用して **LATCHED** 状態のすべてのアラームを解除するスクリプトを作成できます。

カテゴリ

アラーム

使用法

```
TagName.Dismiss=1;
```

パラメータ

TagName

論理型、整数型、実数型、間接論理型、およびアナログ型のタグ、またはアラーム グループ

備考

指定したタグまたはアラーム グループに関連付けられるすべての未処理のアラームを解除するには、このドットフィールドの値を 1 に設定します。指定したタグがアラーム グループである場合、指定したグループ内のタグに関連付けられたすべての **LATCHED** アラームが解除されます。指定したタグがアラーム グループ以外のタイプの場合、そのタグに関連付けられている **LATCHED** アラームだけが解除されます。Dismiss ドットフィールドを 1 以外の値にしても何の意味もありません。

データ型

論理型（読み取り/書き込み）

有効値

1

例

以下のステートメントでは、Tag1 タグに関連付けられているアラームが解除されます。

```
Tag1.Dismiss=1;
```

次の例は、PumpStation アラーム グループ内のすべての LATCHED アラームを解除するために使用されます。

```
PumpStation.Dismiss=1;
```

ランタイム時のタグとグループのアラーム プロパティの制御

アラーム ドットフィールドを使用して、アラーム条件を動的に管理できます。これらの多くのドットフィールドは、I/O、式、およびスクリプトを使用してアクセスできます。I/O アクセスを使用して、Excel やリモート ノードで実行している WindowViewer などの他の Windows アプリケーションを使用して、特定のタグのアラーム情報を監視したり、制御したりできます。

タグに関連付けられているドットフィールドにアクセスするには、以下の構文を使用します。

tag.dotfield

たとえば、Analog_tag という名前のタグの HiHi アラームしきい値のランタイム変更を許可するには、ボタンへのアナログ ユーザー入力タッチ リンクを作成して、そのリンクのダイアログ ボックスで Analog_tag.HiHiLimit を式として入力します。ランタイム時、オペレータはそのボタンをクリックするだけで、Analog_tag に割り当てられた HiHi アラームしきい値に新しい値を入力できます。

以下の表では、各アラーム ドットフィールドの概要を説明しています。

ドットフィールド	説明
.Ack	タグとアラーム グループのアラーム確認状況を監視または制御します。 .Ack には、.UnAck と呼ばれる逆タグドットフィールドがあります。未確認アラームが発生すると、.UnAck は 1 に設定されます。その後、UnAck ドットフィールドをアニメーション リンクや条件スクリプトで使用して、未確認アラームのアナシエーターをトリガできます。
.AckDev	アナログ型タグまたはアラーム グループでアクティブな偏差型アラームのアラーム確認状況を監視または制御します。
.AckDsc	論理型タグの現在の確認状況を監視または制御します。
.AckROC	タグでアクティブな変化率タイプのアラームのアラーム確認状況を監視または制御します。
.AckValue	タグでアクティブな値型アラームのアラーム確認状況を監視または制御します。

ドットフィールド	説明
.Alarm	アラーム条件が存在する場合に通知します。
.AlarmAckModel	<p>タグに関連付けられている確認モデルを次のように監視または制御します。</p> <p>0 = 条件（デフォルト）</p> <p>1 = イベント指向</p> <p>2 = 拡張サマリ</p> <p>アラームを持つ論理型またはアナログ型タグに適用されます。読み取り専用ですが、WindowMaker で設定できます。</p>
.AlarmDev	偏差アラームの存在を通知します。
.AlarmDevCount	指定されたタグまたはアラーム グループでアクティブな偏差アラームの合計数を追跡します。
.AlarmDevDeadband	大偏差アラームおよび小偏差アラームの偏差率デッドバンドを監視または制御します。
.AlarmDevUnAckCount	指定されたタグまたはアラーム グループでアクティブな未確認の偏差アラームの合計数を追跡します。
.AlarmDisabled	イベントやアラームを有効または無効にします。アラームを持つ論理型タグとアナログ型タグ、またはアラーム グループに適用されます。
.AlarmDsc	論理型アラーム条件が現在アクティブであることを示します。
.AlarmDscCount	指定されたタグまたはアラーム グループでアクティブな論理型アラームの合計数を追跡します。
.AlarmDscDisabled	<p>タグが論理型アラームを生成できるかどうかを示します。</p> <p>注記: このドットフィールドは論理型タグに対しては .AlarmDisabled ドットフィールドと同じです。</p>
.AlarmDscEnabled	<p>タグが論理型アラームを生成できるかどうかを示します。</p> <p>注記: このドットフィールドは論理型タグに対しては .AlarmEnabled ドットフィールドと同じです。</p>
.AlarmDscInhibitor	このタグの論理型アラーム（存在する場合）に割り当てられている抑止タグ名を返します。
.AlarmDscUnAckCount	指定されたタグまたはアラーム グループでアクティブな未確認の論理型アラームの合計数を追跡します。
.AlarmEnabled	イベントやアラームを有効または無効にします。
.AlarmHiDisabled	アラームを持つアナログ型タグの Hi 限界値を有効または無効にします。

ドットフィールド	説明
.AlarmHiHiDisabled	アラームを持つアナログ型タグの HiHi 限界値を有効または無効にします。
.AlarmHiHiEnabled	アラームを持つアナログ型タグの HiHi 限界値を有効または無効にします。
.AlarmHiHiInhibitor	HiHi 限界値に対する抑止タグ リファレンスを返します。アラームを持つアナログ型タグに適用されます。読み取り専用ですが、 WindowMaker で設定できます。
.AlarmHiInhibitor	Hi 限界値に対する抑止タグ リファレンスを返します。アラームを持つアナログ型タグに適用されます。 読み取り専用ですが、 WindowMaker で設定できます。
.AlarmLoDisabled	アラームを持つアナログ型タグの Lo 限界値を有効または無効にします。
.AlarmLoEnabled	アラームを持つアナログ型タグの Lo 限界値を有効または無効にします。
.AlarmLoInhibitor	Lo 限界値に対する抑止タグ リファレンスを返します。アラームを持つアナログ型タグに適用されます。 読み取り専用ですが、 WindowMaker で設定できます。
.AlarmLoLoDisabled	アラームを持つアナログ型タグの LoLo 限界値を有効または無効にします。
.AlarmLoLoEnabled	アラームを持つアナログ型タグの LoLo 限界値を有効または無効にします。
.AlarmLoLoInhibitor	LoLo 限界値に対する抑止タグ リファレンスを返します。アラームを持つアナログ型タグに適用されます。 読み取り専用ですが、 WindowMaker で設定できます。
.AlarmMajDevDisabled	アラームを持つアナログ型タグの大偏差限界を有効または無効にします。
.AlarmMajDevEnabled	アラームを持つアナログ型タグの大偏差限界を有効または無効にします。
.AlarmMajDevInhibitor	大偏差限界に対する抑止タグ リファレンスを返します。アラームを持つアナログ型タグに適用されます。 読み取り専用ですが、 WindowMaker で設定できます。
.AlarmMinDevDisabled	アラームを持つアナログ型タグの小偏差限界を有効または無効にします。
.AlarmMinDevEnabled	アラームを持つアナログ型タグの小偏差限界を有効または無効にします。

ドットフィールド	説明
.AlarmMinDevInhibitor	小偏差限界に対する抑止タグ リファレンスを返します。アラームを持つアナログ型タグに適用されます。 読み取り専用ですが、 WindowMaker で設定できます。
.AlarmROC	変化率アラームが存在することを通知します。
.AlarmROCCount	指定されたタグまたはアラーム グループでアクティブな変化率アラームの合計数を追跡します。
.AlarmROCDisabled	アラームを持つアナログ型タグの変化率の限界を有効または無効にします。
.AlarmROCEnabled	アラームを持つアナログ型タグの変化率の限界を有効または無効にします。
.AlarmROCIInhibitor	変化率の限界に対する抑止タグ リファレンスを返します。アラームを持つアナログ型タグに適用されます。 読み取り専用ですが、 WindowMaker で設定できます。
.AlarmROCUAckCount	指定されたタグまたはアラーム グループで未確認の変化率アラームの合計数を追跡します。
.AlarmTotalCount	指定されたタグまたはアラーム グループでアクティブなアラームの合計数を追跡します。
.AlarmUnAckCount	任意のタグまたはアラーム グループでアクティブな未確認アラームの合計数を追跡します。
.AlarmUserDefNum1Set	読み取り/書き込みの実数値（浮動小数点）です。デフォルトは 0 で、値は設定されていません。アラームを持つ論理型タグやアナログ型タグ、またはアラーム グループに適用されます。 注記: このドットフィールドの値は、スクリプトや POKE などによって設定済みの場合のみ、アラームに付加されます。
.AlarmUserDefNum2	読み取り/書き込みの実数値（浮動小数点）です。デフォルトは 0 で、値は設定されていません。アラームを持つ論理型タグやアナログ型タグ、またはアラーム グループに適用されます。 注記: このドットフィールドの値は、スクリプトや POKE などによって設定済みの場合のみ、アラームに付加されます。
.AlarmUserDefNum2Set	読み取り/書き込みの論理型。対応するタグの .AlarmUserDefNum2 がスクリプトで定義されている場合に、 TRUE に設定されます。タグの .AlarmUserDefNum2 の値の関連付けを解除するには、このドットフィールドを FALSE に設定します。デフォルトは FALSE です。

ドットフィールド	説明
.AlarmUserDefStr	<p>読み取り/書き込みのテキスト文字列です。デフォルトは "" で、値は設定されていません。</p> <p>アラームを持つ論理型タグやアナログ型タグ、またはアラーム グループに適用されます。</p> <p>注記: このドットフィールドの値は、スクリプトや POKE などによって設定済みの場合のみ、アラームに付加されます。</p>
.AlarmUserDefStrSet	<p>読み取り/書き込みの論理型。対応するタグの .AlarmUserDefStr がスクリプトで定義されている場合に、TRUE に設定されます。タグの .AlarmUserDefStr の値の関連付けを解除するには、このドットフィールドを FALSE に設定します。デフォルトは FALSE です。</p>
.AlarmValDeadband	<p>アラーム値のデッドバンドの値を監視または制御します。</p>
.AlarmValueCount	<p>指定されたタグまたはアラーム グループでアクティブな値型アラームの合計数を追跡します。</p>
.AlarmValueUnAckCount	<p>指定されたタグまたはアラーム グループでアクティブな未確認の値型アラームの合計数を追跡します。</p>
.DevTarget	<p>大偏差アラームおよび小偏差アラームのターゲットを監視または制御します。</p>
.HiLimit、.HiHiLimit、.LoLimit、.LoLoLimit	<p>値型アラーム チェックのしきい値を監視または制御する読み取り/書き込みのアナログ型タグ名ドットフィールドです。これらのフィールドは、整数型および実数型タグに対してのみ有効です。</p>
.HiStatus、.HiHiStatus、.LoStatus、.LoLoStatus	<p>指定したタイプのアラームが存在するかどうかを決定する読み取り専用の論理型ドットフィールドです。これらのフィールドは、整数型および実数型のタグに対してのみ有効です。</p>
.MajorDevPct	<p>アラーム チェックのための大偏差率を監視または制御する読み取り/書き込みの整数型ドットフィールドです。</p>
.MajorDevStatus	<p>指定したタグに対して大偏差アラームが存在するかどうかを判断する読み取り専用の論理型ドットフィールドです。</p>
.MinorDevPct	<p>アラーム チェックのための小偏差率を監視または制御する読み取り/書き込みの整数型ドットフィールドです。</p>
.MinorDevStatus	<p>指定したタグに対して小偏差アラームが存在するかどうかを判断する読み取り専用の論理型ドットフィールドです。</p>
.Name	<p>タグの実際の名前を表示する読み取り/書き込みのメッセージ型ドットフィールドです。たとえば、グループ変数がポイントしているアラーム グループの名前、または TagID タグの名前を判断する場合に使用できます。また、グループ変数がポイントしているアラームグループを変更するために書き込むこともできます。</p>

ドットフィールド	説明
.Normal	指定したタグに対してアラームが存在しない場合、読み取り専用の論理型ドットフィールドは 1 に等しくなります。このドットフィールドは、普通のタグだけでなく、アラーム グループやグループ変数に対しても有効です。
.ROCPct	アラーム チェックの変化率を監視または制御する読み取り/書き込みのドットフィールドです。
.ROCStatus	指定したタグに対して変化率アラームが存在するかどうかを判断する読み取り専用の論理型ドットフィールドです。

タグ変数またはアラーム グループがアラーム状況にあるかどうかの確認

実行中のアプリケーションのアラーム状況を確認するには、以下のドットフィールドとシステム タグ変数を使用します。新しいアラームが発生したか、タグ変数またはアラーム グループがアラーム状況と正常状況のいずれにあるかを確認できます。また、論理値、偏差、変化率の各アラーム状況を確認することもできます。

- [\\$NewAlarm システム タグ変数](#)
- [\\$System システム タグ変数](#)
- [.Alarm ドットフィールド](#)
- [.Normal ドットフィールド](#)
- [.AlarmDsc ドットフィールド](#)
- [.AlarmDev ドットフィールド](#)
- [.AlarmROC ドットフィールド](#)
- [.LoStatus ドットフィールド](#)
- [.LoLoStatus ドットフィールド](#)
- [.HiStatus ドットフィールド](#)
- [.HiHiStatus ドットフィールド](#)
- [.MinorDevStatus ドットフィールド](#)
- [.MajorDevStatus ドットフィールド](#)
- [.ROCStatus ドットフィールド](#)

\$NewAlarm システム タグ変数

実行中のアプリケーションで新しいアラームが発生した場合 **1** に設定します。**\$NewAlarm** システム タグ変数は、リモート アラームに対しては設定されません。

構文

```
$NewAlarm=value;
```

データタイプ

論理型（読み取り／書き込み）

有効値

0 または 1

備考

\$NewAlarm システム タグ変数をアプリケーション ウィンドウのアニメーション オブジェクトに関連付けます。たとえば、タグ変数の値を 0 にリセットして、アラームを確認するためにオペレータがクリックする確認ボタンにタグ変数を関連付けます。また、**\$NewAlarm** システム タグ変数を **PlaySound** ロジック関数にリンク付けて、アラームが発生したときに警告音を鳴らすこともできます。

例

ボタンをアラーム確認ウィンドウに追加して、オペレータがこのボタンをクリックすると実行される以下のアクション スクリプトを添付します。

```
Ack $System;  
$NewAlarm=0;  
HideSelf;
```

オペレータがボタンをクリックすると、すべてのアラームが確認され、**\$NewAlarm** システム タグ変数は 0 にリセットされ、アラーム確認ウィンドウがビューから非表示となります。

\$System システム タグ変数

デフォルトのアラーム グループです。

カテゴリ

アラーム

使用法

\$System

備考

デフォルトでは、タグ変数はこのルート アラーム グループに割り当てられます。定義されたすべてのアラーム グループは、**\$System** の下位に作成されます。

データタイプ

システム アラーム グループ

例

```
$System.Ack = 1;{すべてのアラームを確認します}
```

.Alarm ドットフィールド

指定したタグ変数またはアラーム グループが現在アラーム状況でない場合に 0 を返します。アラームが発生すると **Alarm** ドットフィールドは 1 を返します。アラーム状況が存在しなくなるまで、この値は 1 のまま維持されます。**.Alarm** ドットフィールドには、**.Normal** という逆ドットフィールドがあります。

タグ変数にアラーム グループの名前を指定すると、そのグループに属する何らかのタグ変数がアラーム状況にある場合に **.Alarm** ドットフィールドは 1 を返します。

カテゴリ

アラーム

使用法

TagName.Alarm

パラメータ

TagName

論理型、整数型、実数型のタグ変数、間接論理型、アナログ型のタグ変数、またはアラーム グループタグ変数

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下のステートメントでは、タグ変数 **Tag1** に関連付けられたアラームがアクティブかどうかを検証します。

```
IF (Tag1.Alarm == 1) THEN
```

以下の例では、**PumpStation** アラーム グループにアクティブなアラームが存在する場合に、IF-THEN ステートメントの本文が処理されます。

```
IF (PumpStation.Alarm == 1) THEN
```

```
    MyAlarmMessage="現在ポンプ ステーションにアラームが発生しています!";
```

```
ENDIF;
```

このドットフィールドは、**.Ack** や **.UnAck** ドットフィールドにはリンクされていません。そのため、アクティブなアラームが確認された後も、**.Alarm** の値は 1 のまま維持されます。

.Normal ドットフィールド

指定したタグ変数がアラーム状況でない場合に 1 を返します。アラームが発生すると **Normal** ドットフィールドは 0 を返します。**.Normal** ドットフィールドには、**.Alarm** という逆ドットフィールドがあります。

カテゴリ

アラーム

構文

```
TagName.Normal
```

パラメータ

TagName

論理型、整数型、実数型のタグ変数、間接論理型、アナログ型のタグ変数、またはアラーム グループタグ変数

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下の IF-THEN ステートメントは、**Tag1** タグ変数に関連付けられたアラームが存在しない場合に実行されます。**Tag1** に対して 1 つまたは複数のアラームがアクティブになると、「ELSE」の本文が実行されます。

```
IF (Tag1.Normal==1) THEN
    MyOperatorMessage="Tag1 に関連付けられているアラームはありません。";
ELSE
    MyOperatorMessage="Tag1 に 1 つまたは複数のアクティブなアラームがあります!";
ENDIF;
```

.AlarmDsc ドットフィールド

指定したタグ変数またはアラーム グループにアラーム状況が存在するかどうかを示します。デフォルト値は 0 です。指定したタグ変数に論理値アラーム状況が存在する場合は 1 の値が設定されます。アラーム状況が存在しなくなるまでこの値は 1 のまま維持されます。

タグ変数にアラーム グループの名前を指定すると、そのグループ内の何らかのタグ変数がアクティブな論理型アラームである場合に **.AlarmDsc** ドットフィールドに 1 が設定されます。

カテゴリ

アラーム

使用法

TagName.AlarmDsc

パラメータ

TagName

任意の論理型タグ変数、間接論理型タグ変数、またはアラーム グループ

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下のステートメントでは、**Tag1** にアクティブなアラームに関連付けられているかどうかを検証します。

```
IF (Tag1.AlarmDsc == 1) THEN
    MyAlarmMessage="現在ポンプ ステーションにアラームが発生しています!";
ENDIF;
```

このドットフィールドは、**.Ack** や **.UnAck** ドットフィールドにはリンクされていません。このため、アクティブなアラームが確認された後も、**.AlarmDsc** ドットフィールドの値は 1 のまま維持されます。

参照項目

.Ack、.UnAck、.Alarm、.AlarmDsc、.AckDsc

.AlarmDev ドットフィールド

指定したタグ変数またはアラーム グループの偏差アラームがアクティブになったことを示します。デフォルト値は **0** です。指定したタグ変数に偏差アラーム状況が存在する場合は **1** の値が設定されます。アラーム状況が存在しなくなるまでこの値は **1** のまま維持されます。

タグ変数にアラーム グループの名前を指定すると、そのグループ内の何らかのタグ変数がアクティブなアラーム状況にある場合に **.AlarmDev** ドットフィールドに **1** が設定されます。

カテゴリ

アラーム

使用法

`TagName.AlarmDev`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下のステートメントでは、タグ変数 **Tag1** に関連付けられた偏差アラームがアクティブかどうかを検証します。

```
IF (Tag1.AlarmDev == 1) THEN
```

以下の例では、**PumpStation** アラーム グループにアクティブな偏差アラームが存在する場合に、**IF-THEN** ステートメントの本文が処理されます。

```
IF (PumpStation.AlarmDev == 1) THEN  
    MyAlarmMessage="現在ポンプ ステーションにアラームが発生しています!";  
ENDIF;
```

このドットフィールドは、**.Ack** や **.UnAck** ドットフィールドにはリンクされていません。このため、アクティブなアラームが確認された場合も、このドットフィールドの値は **1** のまま維持されます。

参照項目

.Ack、**.UnAck**、**.Alarm**、**.AckDev**

.AlarmROC ドットフィールド

指定したタグ変数またはアラーム グループの変化率アラーム状況がアクティブになったことを示します。デフォルト値は **0** です。指定したタグ変数に変化率アラーム状況が存在する場合は **1** の値が設定されます。変化率アラーム状況が存在しなくなるまでこの値は **1** のまま維持されます。

タグ変数にアラーム グループの名前を指定すると、そのグループ内の何らかのタグ変数に変化率アラーム状況にある場合に **.AlarmROC** ドットフィールドに **1** が設定されます。

カテゴリ

アラーム

使用法

`TagName.AlarmROC`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

データ タイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下のステートメントは、タグ変数 **Tag1** に関連付けられている変化率アラームがアクティブかどうかを検証します。

```
IF (Tag1.AlarmROC == 1) THEN
```

PumpStation というアラーム グループ内にアクティブな変化率アラームが存在する場合、この IF-THEN ステートメントの本文が処理されます。

```
IF (PumpStation.AlarmROC == 1) THEN
```

```
    MyAlarmMessage="The pumping station currently has an ALARM!";
```

```
ENDIF;
```

このドットフィールドは、**.Ack** ドットフィールドや **.UnAck** ドットフィールドにはリンクされていません。このため、アクティブな変化率アラームが確認された後も、このドットフィールドの値は **1** のまま維持されます。

参照項目

.Ack、**.AckROC**、**.Alarm**、**.AlarmROCEnabled**、**.AlarmROCDisabled**

.LoStatus ドットフィールド

指定したタグ変数またはアラーム グループの **Lo** アラーム状況がアクティブになったことを示します。デフォルト値は **0** です。指定したタグ変数に **Lo** アラーム状況が存在する場合は **1** の値が設定されます。**Lo** アラーム状況が存在しなくなるまでこの値は **1** のまま維持されます。

このドットフィールドは、特定のタグ変数が指定のアラーム状況にあるかを調べるために、**.Alarm** や **.Ack** ドットフィールドと組み合わせてよく使用されます。

カテゴリ

アラーム

使用法

`TagName.LoStatus`

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下の IF-THEN ステートメントは、タグ変数 **MyTag** の **.LoStatus**（Lo アラーム状況）が **1** の場合に実行されます。

```
IF (MyTag.LoStatus == 1) THEN
    OperatorMessage=" MyTag で Lo アラームが発生しました";
ENDIF;
```

参照項目

.Alarm、.AlarmValue、.Ack、.LoLimit、.LoSet、.AlarmDisabled、.AlarmEnabled、.AlarmLoDisabled、.AlarmLoEnabled、.AlarmLoInhibitor

.LoLoStatus ドットフィールド

指定したタグ変数またはアラーム グループの **LoLo** アラーム状況がアクティブになったことを示します。デフォルト値は **0** です。指定したタグ変数に **LoLo** アラーム状況が存在する場合は **1** の値が設定されます。**LoLo** アラーム状況が存在しなくなるまでこの値は **1** のまま維持されます。

このドットフィールドは、システム内の特定のタグ変数で発生したアラーム状況の正確な内容を調べるために、**.Alarm** や **.Ack** フィールドと組み合わせてよく使用されます。

カテゴリ

アラーム

使用法

TagName.LoLoStatus

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下の IF-THEN ステートメントは、タグ変数 **MyTag** の **.LoLoStatus**（LoLo アラームしきい値）が **1** の場合に実行されます。

```
IF (MyTag.LoLoStatus == 1) THEN
    OperatorMessage=" MyTag で LoLo アラームが発生しました";
ENDIF;
```

参照項目

.Alarm、.AlarmValue、.Ack、.LoLoLimit、.LoLoSet、.AlarmDisabled、.AlarmEnabled、.AlarmLoLoDisabled、.AlarmLoLoEnabled、.AlarmLoLoInhibitor

.HiStatus ドットフィールド

指定したタグ変数またはアラーム グループの Hi アラーム状況がアクティブになったことを示します。デフォルト値は 0 です。指定したタグ変数に Hi アラーム状況が存在する場合は 1 の値が設定されます。Hi アラーム状況が存在しなくなるまでこの値は 1 のまま維持されます。

このドットフィールドは、タグ変数が指定のアラーム状況にあるかを調べるために、**.Alarm** や **.Ack** ドットフィールドと組み合わせてよく使用されます。

カテゴリ

アラーム

使用法

TagName.HiStatus

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下のスクリプトは、**MotorAmps** タグ変数が上限アラーム状況になった場合に別のスクリプトを呼び出して、ポンプ モータの出力を停止します。

```
IF (MotorAmps.HiStatus == 1) THEN  
    CALL PumpShutdown( );  
ENDIF;
```

参照項目

.Alarm、**.AlarmValue**、**.Ack**、**.HiLimit**、**.HiSet**、**.AlarmDisabled**、**.AlarmEnabled**、**.AlarmHiDisabled**、**.AlarmHiEnabled**、**.AlarmHiInhibitor**

.HiHiStatus ドットフィールド

指定したタグ変数またはアラーム グループの HiHi アラーム状況がアクティブになったことを示します。デフォルト値は 0 です。指定したタグ変数に HiHi アラーム状況が存在する場合は 1 の値が設定されます。HiHi アラーム状況が存在しなくなるまでこの値は 1 のまま維持されます。

このドットフィールドは、タグ変数が指定のアラーム状況にあるかを調べるために、**.Alarm** や **.Ack** ドットフィールドと組み合わせてよく使用されます。

カテゴリ

アラーム

使用法

TagName.HiHiStatus

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下の IF-THEN ステートメントは、タグ変数 **MyTag** の **.HiHiStatus**（HiHi アラーム）が **1** の場合に実行されます。

```
IF (MyTag.HiHiStatus == 1) THEN
    OperatorMessage=" MyTag で HiHi アラームが発生しました";
ENDIF;
```

参照項目

.Alarm、.AlarmValue、.Ack、.HiHiLimit、.HiHiSet、.AlarmDisabled、.AlarmEnabled、.AlarmHiHiDisabled、.AlarmHiHiEnabled、.AlarmHiHiInhibitor

.MinorDevStatus ドットフィールド

指定したタグ変数またはアラーム グループの小偏差アラームがアクティブになったことを示します。デフォルト値は **0** です。指定したタグ変数に小偏差アラーム状況が存在する場合は **1** の値が設定されます。小偏差アラーム状況が存在しなくなるまでこの値は **1** のまま維持されます。

このドットフィールドは、タグ変数が指定のアラーム状況にあるかを調べるために、**.Alarm** や **.Ack** ドットフィールドと組み合わせてよく使用されます。

カテゴリ

アラーム

使用法

TagName.MinorDevStatus

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下の IF-THEN ステートメントは、タグ変数 **MyTag** の **.MinorDevStatus**（小偏差アラーム）が **1** の場合に実行されます。

```
IF (MyTag.MinorDevStatus == 1) THEN
    OperatorMessage=" MyTag で小偏差アラームが発生しました";
ENDIF;
```

参照項目

.AckDev、**.AlarmDev**、**.AlarmMinDevDisabled**、**.AlarmMinDevEnabled**、**.AlarmMinDevInhibitor**、**.MinorDevPct**、**.MajorDevStatus**

.MajorDevStatus ドットフィールド

指定したタグ変数またはアラーム グループの大偏差アラームがアクティブになったことを示します。デフォルト値は **0** です。指定したタグ変数に大偏差アラーム状況が存在する場合は **1** の値が設定されます。大偏差アラーム状況が存在しなくなるまでこの値は **1** のまま維持されます。

このドットフィールドは、タグ変数が指定のアラーム状況にあるかを調べるために、**.Alarm** や **.Ack** ドットフィールドと組み合わせてよく使用されます。

カテゴリ

アラーム

使用法

`TagName.MajorDevStatus`

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下の IF-THEN ステートメントは、タグ変数 **MyTag** の **.MajorDevStatus**（大偏差アラーム）が **1** の場合に実行されます。

```
IF (MyTag.MajorDevStatus == 1) THEN
    OperatorMessage=" MyTag で大偏差アラームが発生しました";
ENDIF;
```

参照項目

.AckDev、**.AlarmDev**、**.AlarmMajDevDisabled**、**.AlarmMajDevEnabled**、**.AlarmMajDevInhibitor**、**.MajorDevPct**、**.MajorDevSet**、**.MinorDevStatus**

.ROCStatus ドットフィールド

指定したタグ変数またはアラーム グループの変化率アラームがアクティブになったことを示します。デフォルト値は **0** です。指定したタグ変数に変化率アラーム状況が存在する場合は **1** の値が設定されます。変化率アラーム状況が存在しなくなるまでこの値は **1** のまま維持されます。

このドットフィールドは、タグ変数が指定のアラーム状況にあるかを調べるために、**.Alarm** や **.Ack** ドットフィールドと組み合わせてよく使用されます。

カテゴリ

アラーム

使用法

`TagName.ROCStatus`

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

以下の IF-THEN ステートメントは、タグ変数 `MyTag` の `.ROCStatus`（変化率アラーム）が 1 の場合に実行されます。

```
IF (MyTag.ROCStatus == 1) THEN
    OperatorMessage=" MyTag で変化率アラームが発生しました";
ENDIF;
```

参照項目

`.ROCPct`、`.ROCSet`

InTouch 7.1 の動作にアラーム状況に戻す

InTouch 7.11 移行では、HiHi、LoLo、MinDev（小偏差）、または MajDev（大偏差）アラームの発生は、それぞれ Hi、Lo、MajDev、または MinDev アラームの In Alarm、ACK、および \$NewAlarm 状態にリセットされます。InTouch 7.1 およびそれ以前のバージョンの場合、これらのアラーム状況は HiHi、LoLo、MinDev、および MajDev アラームの発生による影響を受けません。

その他のアラームの発生に関係なく In Alarm、ACK、および \$NewAlarm 状態を維持するために Hi、Lo、MajDev、または MinDev アラームを予期する従来のアプリケーションをサポートするには、InTouch.ini ファイルに IT71StatusFlag パラメータを含めることができます。

InTouch 7.1 およびそれ以前のバージョンと同様に動作するようにアラーム処理を強制するには

- 次の行を InTouch.ini ファイルに入力します。
`IT71StatusFlags=1`

InTouch 7.11 およびそれ以降のデフォルト動作に戻すには

- 次の行を InTouch.ini ファイルに入力します。
`IT71StatusFlags=0`

InTouch.ini ファイルを編集して、このパラメータおよびその他の InTouch の動作パラメータを設定する方法の詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』を参照してください。

タグ変数のアラームしきい値設定の確認

アプリケーションの実行中に、タグ変数にアラームしきい値が設定されているかどうかを調べるには、以下のドットフィールドを使用します。

- [`.LoLoSet` ドットフィールド](#)

- [.LoSet ドットフィールド](#)
- [.HiSet ドットフィールド](#)
- [.HiHiSet ドットフィールド](#)
- [.MinorDevSet ドットフィールド](#)
- [.MajorDevSet ドットフィールド](#)
- [.ROCSet ドットフィールド](#)

.LoLoSet ドットフィールド

整数型または実数型タグ変数に LoLo アラームしきい値が設定されているかどうかを示します。

カテゴリ

アラーム

使用法

TagName.LoLoSet

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

THEN ブロックは、タグ変数 **MyTag** に対して LoLo アラームしきい値が設定されている場合に実行されます。

```
IF (MyTag.LoLoSet== 1) THEN  
    MsgTag="MyTag には LoLo アラームしきい値が設定されています";  
ENDIF;
```

参照項目

.Alarm、.AlarmValue、.Ack、.LoLoStatus、.LoLoLimit、.AlarmDisabled、.AlarmEnabled、.AlarmLoLoDisabled、.AlarmLoLoEnabled、.AlarmLoLoInhibitor

.LoSet ドットフィールド

整数型または実数型タグ変数に Lo アラームしきい値が設定されているかどうかを示します。

カテゴリ

アラーム

使用法

TagName.LoSet

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

THEN ブロックは、タグ変数 **MyTag** に対して Lo アラームしきい値が設定されている場合に実行されます。

```
IF (MyTag.LoSet== 1) THEN  
    MsgTag="MyTag には Lo アラームしきい値が設定されています";  
ENDIF;
```

参照項目

.Alarm、.AlarmValue、.Ack、.LoStatus、.LoLimit、.AlarmDisabled、.AlarmEnabled、.AlarmLoDisabled、.AlarmLoEnabled、.AlarmLoInhibitor

.HiSet ドットフィールド

整数型または実数型タグ変数に Hi アラームしきい値が設定されているかどうかを示します。

カテゴリ

アラーム

使用法

TagName.HiSet

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

THEN ブロックは、タグ変数 **MyTag** に対して Hi アラームしきい値が設定されている場合に実行されます。

```
IF (MyTag.HiSet== 1) THEN  
    MsgTag="MyTag には Hi アラームしきい値が設定されています";  
ENDIF;
```

参照項目

.Alarm、.AlarmValue、.Ack、.HiHiStatus、.HiHiLimit、.AlarmDisabled、.AlarmEnabled、.AlarmHiHiDisabled、.AlarmHiHiEnabled、.AlarmHiHiInhibitor

.HiHiSet ドットフィールド

整数型または実数型タグ変数に HiHi アラームしきい値が設定されているかどうかを示します。

カテゴリ

アラーム

使用法

`TagName.HiHiSet`

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

THEN ブロックは、タグ変数 `MyTag` に対して HiHi アラームしきい値が設定されている場合に実行されます。

```
IF (MyTag.HiHiSet== 1) THEN
    MsgTag="MyTag には HiHi しきい値が設定されています";
ENDIF;
```

参照項目

.Alarm、.AlarmValue、.Ack、.HiHiStatus、.HiHiLimit、.AlarmDisabled、.AlarmEnabled、.AlarmHiHiDisabled、.AlarmHiHiEnabled、.AlarmHiHiInhibitor

.MinorDevSet ドットフィールド

整数型または実数型タグ変数に小偏差アラームしきい値が設定されているかどうかを示します。

カテゴリ

アラーム

使用法

`TagName.MinorDevSet`

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

THEN ブロックは、タグ変数 **MyTag** に対して小偏差パーセント値のアラームしきい値が設定されている場合に実行されます。

```
IF (MyTag.MinorDevSet== 1) THEN  
    MsgTag="MyTag には小偏差しきい値が設定されています";  
ENDIF;
```

参照項目

.AckDev、.AlarmDev、.AlarmMinDevDisabled、.AlarmMinDevEnabled、.AlarmMinDevInhibitor、.MinorDevPct、.MinorDevStatus

.MajorDevSet ドットフィールド

整数型または実数型タグ変数に大偏差アラームしきい値が設定されているかどうかを示します。

カテゴリ

アラーム

使用法

TagName.MajorDevSet

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

THEN ブロックは、タグ変数 **MyTag** に対して大偏差パーセント値のアラームしきい値が設定されている場合に実行されます。

```
IF (MyTag.MajorDevSet== 1) THEN  
    MsgTag="MyTag には大偏差しきい値が設定されています";  
ENDIF;
```

参照項目

.AckDev、.AlarmDev、.AlarmMajDevDisabled、.AlarmMajDevEnabled、.AlarmMajDevInhibitor、.MajorDevPct、.MajorDevStatus

.ROCSet ドットフィールド

整数型または実数型タグ変数に変化率アラームしきい値が設定されているかどうかを示します。

カテゴリ

アラーム

使用法

TagName.ROCSet

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

データタイプ

論理型（読み取り専用）

有効値

0 または 1

例

THEN ブロックは、タグ変数 **MyTag** に対して変化率アラームしきい値が設定されている場合に実行されます。

```
IF (MyTag.ROCSet == 1) THEN  
    MsgTag="MyTag には変化率しきい値が設定されています";  
ENDIF;
```

参照項目

.Alarm、.Ack、.LoLimit、.LoLoLimit、.HiHiLimit、.HiLimit、.HiSet、.LoSet、LoLoSet、.HiStatus、.HiHiStatus、ROCPct、ROCStatus

タグ変数またはアラームグループのアラームの有効化と無効化

InTouch HMI ではドットフィールドを使用して、タグ変数に設定されているアラームをアプリケーションの実行中に有効または無効にできます。

すべてのアラームの有効化または無効化

.AlarmEnabled および **.AlarmDisabled** ドットフィールドにより、タグ変数またはアラームグループのアラームを設定値に応じて有効または無効にできます。この 2 つのドットフィールドには、相反するアラーム状況が関連付けられます。**.AlarmEnabled** の値を 1 に設定すると、タグ変数またはアラームグループのアラームを有効にします。**.AlarmDisabled** の値を 1 にすると、タグ変数またはアラームグループのアラームを無効にします。

いずれかのドットフィールドによってアラームグループのアラームを有効化すると、そのグループに属するすべてのタグ変数のアラームが有効になります。いずれかのドットフィールドによってアラームを無効にすると、すべてのイベントとアラームは無視されます。この場合、アラームはアラームメモリに保持されないだけでなく、ディスクにも書き込まれません。

.AlarmEnabled ドットフィールド

タグ変数またはアラームグループに対するアラームを有効または無効にします。

カテゴリ

アラーム

使用法

TagName.AlarmEnabled

パラメータ

TagName

論理型、整数型、実数型、間接論理型、間接アナログ型のタグ変数、またはアラームグループ

備考

.AlarmEnabled を 0 に設定すると、イベントとアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

指定されたタグ変数がアラーム グループの場合、指定アラーム グループ内のタグ変数に関連付けられているすべてのアラームが有効になります。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

1 = アラームを有効化（デフォルト）

例

以下のステートメントは、タグ変数 **Tag1** のアラームを無効にします。

```
Tag1.AlarmEnabled=0;
```

参照項目

.AlarmDisabled

.AlarmDisabled ドットフィールド

タグ変数またはアラーム グループに対するアラームを有効または無効にします。

カテゴリ

アラーム

使用法

```
TagName.AlarmDisabled
```

パラメータ

TagName

論理型、整数型、実数型、間接論理型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmDisabled を 1 に設定すると、イベントとアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

指定されたタグ変数がアラーム グループの場合、指定アラーム グループ内のタグ変数に関連付けられているすべてのアラームが無効になります。

.AlarmEnabled ドットフィールドとは逆のドットフィールドです。

例

以下のステートメントは、タグ変数 **Tag1** のアラームを有効にします。

```
Tag1.AlarmDisabled=0;
```

参照項目

.AlarmEnabled

LoLo アラームの有効化または無効化

.AlarmLoLoEnabled ドットフィールドおよび **.AlarmLoLoDisabled** ドットフィールドにより、タグ変数またはアラーム グループのアラームを設定値に応じて有効または無効にできます。この 2 つのドットフィールドには、相反する LoLo アラーム状況が関連付けられます。**.AlarmLoLoEnabled** の値を 1 に設定すると、タグ変数またはアラーム グループの LoLo アラームを有効にします。**.AlarmLoLoDisabled** の値を 1 にすると、タグ変数またはアラーム グループの LoLo アラームを無効にします。

いずれかのドットフィールドによってアラーム グループの LoLo アラームを有効化すると、そのグループに属するすべてのタグ変数のアラームが有効になります。いずれかのドットフィールドによって LoLo アラームを無効にすると、すべての LoLo アラームは無視されます。この場合、アラームはアラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。

.AlarmLoLoEnabled ドットフィールド

LoLo 条件のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

`TagName.AlarmLoLoEnabled`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmLoLoEnabled を 0 に設定すると、LoLo 条件のイベントとアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

1 = アラームを有効化（デフォルト）

例

以下のステートメントは、タグ変数 **Tag1** の LoLo アラームを無効にします。

`Tag1.AlarmLoLoEnabled=0;`

参照項目

.AlarmDisabled、.AlarmEnabled、.AlarmLoLoDisabled

.AlarmLoLoDisabled ドットフィールド

LoLo 条件のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

`TagName.AlarmLoLoDisabled`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmLoLoDisabled を 1 に設定すると、LoLo 条件のイベントとアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

データ タイプ

論理型（読み取り／書き込み）

有効値

1 = アラームを無効化

0 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 Tag2 の LoLo アラームを有効にします。

```
Tag2.AlarmLoLoDisabled=0;
```

参照項目

.AlarmDisabled、.AlarmEnabled、.AlarmLoLoEnabled

Lo アラームの有効化または無効化

.AlarmLoEnabled ドットフィールドおよび **.AlarmLoDisabled** ドットフィールドにより、タグ変数またはアラーム グループの Lo アラームを設定値に応じて有効または無効にできます。この 2 つのドットフィールドには、相反する Lo アラーム状況が関連付けられます。**.AlarmLoEnabled** の値を 1 に設定すると、タグ変数またはアラーム グループの Lo アラームを有効にします。**.AlarmLoDisabled** の値を 1 にすると、タグ変数またはアラーム グループの Lo アラームを無効にします。

いずれかのドットフィールドによってアラーム グループの Lo アラームを有効化すると、そのグループに属するすべてのタグ変数の Lo アラームが有効になります。いずれかのドットフィールドによって Lo アラームを無効にすると、すべての Lo アラームは無視されます。この場合、アラームはアラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。

.AlarmLoEnabled ドットフィールド

Lo 条件のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

`TagName.AlarmLoEnabled`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmLoEnabled を 0 に設定すると、Lo 条件のイベントとアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

1 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 Tag1 の Lo アラームを無効にします。

```
Tag1.AlarmLoEnabled=0;
```

参照項目

.AlarmDisabled、.AlarmEnabled、.AlarmLoDisabled

.AlarmLoDisabled ドットフィールド

Lo 条件のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

```
TagName.AlarmLoDisabled
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmLoDisabled を 1 に設定すると、Lo 条件のイベントとアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

データ タイプ

論理型（読み取り／書き込み）

有効値

1 = アラームを無効化

0 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 **Tag2** の Lo アラームを有効にします。

```
Tag2.AlarmLoDisabled=0;
```

参照項目

.AlarmDisabled、.AlarmEnabled、AlarmLoEnabled

Hi アラームの有効化または無効化

.AlarmHiEnabled ドットフィールドおよび **.AlarmHiDisabled** ドットフィールドにより、タグ変数またはアラーム グループの Hi アラームを設定値に応じて有効または無効にできます。この 2 つのドットフィールドには、相反する Hi アラーム状況が関連付けられます。**.AlarmHiEnabled** の値を 1 に設定すると、タグ変数またはアラーム グループの Hi アラームを有効にします。**.AlarmHiDisabled** の値を 1 にすると、タグ変数またはアラーム グループのアラームを無効にします。

いずれかのドットフィールドによってアラーム グループの Hi アラームを有効化すると、そのグループに属するすべてのタグ変数の Hi アラームが有効になります。いずれかのドットフィールドによって Hi アラームを無効にすると、すべての Hi アラームは無視されます。この場合、Hi アラームはアラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。

.AlarmHiEnabled ドットフィールド

Hi 条件のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

```
TagName.AlarmHiEnabled
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmHiEnabled を 0 に設定すると、Hi 条件のイベントとアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmHiDisabled ドットフィールドとは逆のドットフィールドです。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

1 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 **Tag1** の Hi アラームを無効にします。

```
Tag1.AlarmHiEnabled=0;
```

参照項目

.AlarmHiDisabled、.AlarmEnabled

.AlarmHiDisabled ドットフィールド

Hi 条件のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

TagName.AlarmHiDisabled

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmHiDisabled を 1 に設定すると、Hi 条件のイベントとアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmHiEnabled ドットフィールドとは逆のドットフィールドです。

データ タイプ

論理型（読み取り／書き込み）

有効値

1 = アラームを無効化

0 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 Tag2 の Hi アラームを有効にします。

Tag2.AlarmHiDisabled=0;

参照項目

.AlarmHiEnabled、.AlarmDisabled

HiHi アラームの有効化または無効化

.AlarmHiHiEnabled および **.AlarmHiHiDisabled** ドットフィールドにより、タグ変数またはアラーム グループの HiHi アラームを設定値に応じて有効または無効にできます。この 2 つのドットフィールドには、相反する HiHi アラーム状況が関連付けられます。**.AlarmHiHiEnabled** の値を 1 に設定すると、タグ変数またはアラーム グループの HiHi アラームを有効にします。**.AlarmHiHiDisabled** の値を 1 にすると、タグ変数またはアラーム グループの HiHi アラームを無効にします。

いずれかのドットフィールドによってアラーム グループの HiHi アラームを有効化すると、そのグループに属するすべてのタグ変数の HiHi アラームが有効になります。いずれかのドットフィールドによって HiHi アラームを無効にすると、すべての HiHi アラームは無視されます。この場合、HiHi アラームはアラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。

.AlarmHiHiEnabled ドットフィールド

HiHi 条件のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

`TagName.AlarmHiHiEnabled`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmHiHiEnabled を 0 に設定すると、HiHi 条件のイベントとアラームはすべて無視されます。アラームメモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

1 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 Tag1 の HiHi アラームを無効にします。

`Tag1.AlarmHiHiEnabled=0;`

参照項目

.AlarmHiHiDisabled、.AlarmEnabled

.AlarmHiHiDisabled ドットフィールド

HiHi 条件のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

`TagName.AlarmHiHiDisabled`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmHiHiDisabled を 1 に設定すると、HiHi 条件のイベントとアラームはすべて無視されます。アラームメモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmHiHiEnabled ドットフィールドとは逆のドットフィールドです。

データ タイプ

論理型（読み取り／書き込み）

有効値

1 = アラームを無効化

0 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 **Tag2** の **HiHi** アラームを有効にします。

```
Tag2.AlarmHiHiDisabled=0;
```

参照項目

.AlarmHiHiEnabled、**.AlarmDisabled**

論理値アラームの有効化または無効化

.AlarmDscEnabled および **.AlarmDscDisabled** ドットフィールドにより、タグ変数またはアラーム グループの論理値アラームを設定値に応じて有効または無効にできます。この 2 つのドットフィールドには、相反する論理値アラーム状況が関連付けられます。**.AlarmDscEnabled** の値を 1 に設定すると、タグ変数またはアラーム グループの論理値アラームを有効にします。**.AlarmDscDisabled** の値を 1 にすると、タグ変数またはアラーム グループの論理値アラームを無効にします。

いずれかのドットフィールドによってアラーム グループの論理値アラームを有効化すると、そのグループに属するすべてのタグ変数の論理値アラームが有効になります。いずれかのドットフィールドによって論理値アラームを無効にすると、すべての論理値アラームは無視されます。この場合、論理値アラームはアラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。

.AlarmDscEnabled ドットフィールド

タグ変数が論理値アラームを生成できるかどうかを示します。

カテゴリ

アラーム

使用法

```
TagName.AlarmDscEnabled
```

パラメータ

TagName

任意の論理型、間接論理型タグ変数またはアラーム グループ

備考

.AlarmDscEnabled を 0 に設定すると、論理値条件のアラームとイベントはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmDscDisabled ドットフィールドとは逆のドットフィールドです。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

1 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 **Tag1** の論理値アラームを無効にします。

```
Tag1.AlarmDscEnabled=0;
```

参照項目

.AlarmDscDisabled

.AlarmDscDisabled ドットフィールド

タグ変数が論理値アラームを生成できるかどうかを示します。

カテゴリ

アラーム

使用法

```
TagName.AlarmDscDisabled
```

パラメータ

TagName

任意の論理型、間接論理型タグ変数またはアラーム グループ

備考

.AlarmDscDisabled を 1 に設定すると、すべての論理型条件のアラームとイベントが無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmDscEnabled ドットフィールドとは逆のドットフィールドです。

データ タイプ

論理型（読み取り／書き込み）

有効値

1 = アラームを無効化

0 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 **Tag2** の論理値アラームを有効にします。

```
Tag2.AlarmDscDisabled=0;
```

小偏差アラームの有効化または無効化

.AlarmMinDevEnabled および .AlarmMinDevDisabled ドットフィールドにより、タグ変数またはアラームグループの小偏差アラームを設定値に応じて有効または無効にできます。この 2 つのドットフィールドには、相反する小偏差アラーム状況が関連付けられます。 .AlarmMinDevEnabled の値を 1 に設定すると、タグ変数またはアラームグループの小偏差アラームを有効にします。 .AlarmMinDevDisabled の値を 1 にすると、タグ変数またはアラームグループの小偏差アラームを無効にします。

いずれかのドットフィールドによってアラーム グループの小偏差アラームを有効化すると、そのグループに属するすべてのタグ変数の小偏差アラームが有効になります。いずれかのドットフィールドによって小偏差アラームを無効にすると、すべての小偏差アラームは無視されます。この場合、小偏差アラームはアラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。

.AlarmMinDevEnabled ドットフィールド

小偏差のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

`TagName.AlarmMinDevEnabled`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmMinDevEnabled を 0 に設定すると、小偏差のイベントおよびアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

1 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 **Tag1** の小偏差アラームを無効にします。

`Tag1.AlarmMinDevEnabled=0;`

参照項目

.AlarmDisabled、**.AlarmEnabled**、**.AlarmMinDevDisabled**

.AlarmMinDevDisabled ドットフィールド

小偏差のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

`TagName.AlarmMinDevDisabled`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmMinDevDisabled を 1 に設定すると、小偏差のイベントおよびアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmMinDevEnabled ドットフィールドとは逆のドットフィールドです。

データ タイプ

論理型（読み取り／書き込み）

有効値

1 = アラームを無効化

0 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 **Tag2** の小偏差アラームを有効にします。

```
Tag2.AlarmMinDevDisabled=0;
```

参照項目

.AlarmDisabled、.AlarmEnabled、.AlarmMinDevEnabled

大偏差アラームの有効化または無効化

.AlarmMajDevEnabled ドットフィールドおよび **.AlarmMajDevDisabled** ドットフィールドにより、タグ変数またはアラーム グループの大偏差アラームを設定値に応じて有効または無効にできます。この 2 つのドットフィールドには、相反する大偏差アラーム状況が関連付けられます。**.AlarmMajDevEnabled** の値を 1 に設定すると、タグ変数またはアラーム グループの大偏差アラームを有効にします。**.AlarmMajDevDisabled** の値を 1 にすると、タグ変数またはアラーム グループの大偏差アラームを無効にします。

いずれかのドットフィールドによってアラーム グループの大偏差アラームを有効化すると、そのグループに属するすべてのタグ変数の大偏差アラームが有効になります。いずれかのドットフィールドによって大偏差アラームを無効にすると、すべての大偏差アラームは無視されます。この場合、大偏差アラームはアラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。

.AlarmMajDevEnabled ドットフィールド

大偏差のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

```
TagName.AlarmMajDevEnabled
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmMajDevEnabled を 0 に設定すると、大偏差のイベントとアラームはすべて無視されます。アラームメモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmMajDevDisabled ドットフィールドとは逆のドットフィールドです。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

1 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 Tag1 の大偏差アラームを無効にします。

```
Tag1.AlarmMajDevEnabled=0;
```

参照項目

.AlarmDisabled、.AlarmEnabled、.AlarmMajDevDisabled

.AlarmMajDevDisabled ドットフィールド

大偏差のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

```
TagName.AlarmMajDevDisabled
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmMajDevDisabled を 1 に設定すると、大偏差のイベントとアラームはすべて無視されます。アラームメモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmMajDevEnabled ドットフィールドとは逆のドットフィールドです。

データタイプ

論理型（読み取り／書き込み）

有効値

1 = アラームを無効化

0 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 **Tag2** の大偏差アラームを有効にします。

```
Tag2.AlarmMajDevDisabled=0;
```

参照項目

.AlarmDisabled、.AlarmEnabled、AlarmMajDevEnabled

変化率アラームの有効化／無効化

.AlarmROCEnabled および **.AlarmROCDisabled** ドットフィールドにより、タグ変数またはアラームグループの変化率アラームを設定値に応じて有効または無効にできます。この2つのドットフィールドには、相反する変化率アラーム状況が関連付けられます。**.AlarmROCEnabled** の値を **1** に設定すると、タグ変数またはアラームグループの変化率アラームを有効にします。**.AlarmROCDisabled** の値を **1** にすると、タグ変数またはアラームグループの変化率アラームを無効にします。

いずれかのドットフィールドによってアラームグループの変化率アラームを有効化すると、そのグループに属するすべてのタグ変数の変化率アラームが有効になります。いずれかのドットフィールドによって変化率アラームを無効にすると、すべての変化率アラームは無視されます。この場合、変化率アラームはアラームメモリに保持されないだけでなく、ディスクにも書き込まれません。

.AlarmROCEnabled ドットフィールド

変化率のイベントおよびアラームを有効または無効にします。

カテゴリ

アラーム

使用法

```
TagName.AlarmROCEnabled
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラームグループ

備考

.AlarmROCEnabled を **0** に設定すると、変化率条件のイベントおよびアラームはすべて無視されます。アラームメモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmROCDisabled ドットフィールドとは逆のドットフィールドです。

データタイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

1 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 **Tag1** の変化率アラームを無効にします。

```
Tag1.AlarmROCEnabled=0;
```

参照項目

.AlarmDisabled、.AlarmEnabled、.AlarmROCDDisabled

.AlarmROCDDisabled ドットフィールド

変化率のイベントおよびアラームを無効または有効にします。

カテゴリ

アラーム

使用法

`TagName.AlarmROCDDisabled`

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

.AlarmROCDDisabled を 1 に設定すると、変化率条件のイベントおよびアラームはすべて無視されます。アラーム メモリに保持されないだけでなく、ディスクにも書き込まれません。データの損失を避けるには、イベントやアラームをできるだけ有効にしてください。

.AlarmROCEnabled のプロパティとは逆のプロパティです。

データ タイプ

論理型（読み取り／書き込み）

有効値

0 = アラームを無効化

0 = アラームを有効化（デフォルト）

例

以下の例は、タグ変数 Tag2 の変化率アラームを有効にします。

`Tag2.AlarmROCDDisabled=0;`

参照項目

.AlarmDisabled、.AlarmEnabled、.AlarmROCEnabled

タグ変数のアラームしきい値の変更

アプリケーションの実行中にタグ変数のアラームしきい値を変更するには、以下のドットフィールドを使用します。変更できるのは、LoLo、Lo、Hi、および HiHi アラームのしきい値、大偏差および小偏差のパーセント値と基準値、および変化率の偏差です。

- [.LoLoLimit ドットフィールド](#)
- [.LoLimit ドットフィールド](#)
- [.HiLimit ドットフィールド](#)
- [.HiHiLimit ドットフィールド](#)
- [.MinorDevPct ドットフィールド](#)

- [.MajorDevPct ドットフィールド](#)
- [.DevTarget ドットフィールド](#)
- [.ROCPct ドットフィールド](#)

.LoLoLimit ドットフィールド

タグ変数の LoLo アラームしきい値を変更します。

カテゴリ

アラーム

使用法

```
TagName.LoLoLimit
```

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

WindowViewer を意図的にまたは誤ってシャットダウンした後も継続してこのドットフィールドのランタイム値を使用するには、タグ変数ディクショナリで **[パラメータ値保持]** オプションを選択してください。

データタイプ

アナログ型（読み取り／書き込み）

有効値

指定タグ変数に対する設定範囲内の値

例

このステートメントにより、タグ変数 **MyTag** に対する LoLo アラームしきい値が **10** だけ減少します。

```
MyTag1.LoLoLimit=MyTag1.LoLoLimit - 10;
```

参照項目

.Alarm、.AlarmValue、.Ack、.LoLoStatus、.LoLoSet、.AlarmDisabled、.AlarmEnabled、.AlarmLoLoDisabled、.AlarmLoLoEnabled、.AlarmLoLoInhibitor

.LoLimit ドットフィールド

タグ変数の Lo アラームしきい値を変更します。

カテゴリ

アラーム

使用法

```
Tagname.LoLimit
```

パラメータ

Tagname

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

WindowViewer を意図的にまたは誤ってシャットダウンした後も継続してこのドットフィールドのランタイム値を使用するには、タグ変数ディクショナリで **[パラメータ値保持]** オプションを選択してください。

データタイプ

アナログ型（読み取り／書き込み）

有効値

指定タグ変数に対する設定範囲内の値

例

このステートメントは、タグ変数 **MyTag** に対する Lo アラームしきい値を 10 だけ減少させます。

```
MyTag.LoLimit=MyTag.LoLimit - 10;
```

参照項目

.Alarm、.AlarmValue、.Ack、.LoStatus、.LoSet、.AlarmDisabled、.AlarmEnabled、.AlarmLoDisabled、.AlarmLoEnabled、.AlarmLoInhibitor

.HiLimit ドットフィールド

タグ変数の Hi アラームしきい値を変更します。

カテゴリ

アラーム

使用法

```
TagName.HiLimit
```

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

WindowViewer を意図的にまたは誤ってシャットダウンした後も継続してこのドットフィールドのランタイム値を使用するには、タグ変数ディクショナリで **[パラメータ値保持]** オプションを選択してください。

データタイプ

アナログ型（読み取り／書き込み）

有効値

指定タグ変数に対する設定範囲内の値

例

このステートメントは、タグ変数 **PumpTemp** の Hi 限界アラームを 212 に設定します。

```
PumpTemp.HiLimit = 212;
```

参照項目

.Alarm、.AlarmValue、.Ack、.HiHiStatus、.HiHiSet、.AlarmDisabled、.AlarmEnabled、.AlarmHiHiDisabled、.AlarmHiHiEnabled、.AlarmHiHiInhibitor

.HiHiLimit ドットフィールド

タグ変数の HiHi アラームしきい値を変更します。

カテゴリ

アラーム

使用法

TagName.HiHiLimit

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

WindowViewer を意図的にまたは誤ってシャットダウンした後も継続してこのドットフィールドのランタイム値を使用するには、タグ変数ディクショナリで [パラメータ値保持] オプションを選択してください。

データタイプ

アナログ型（読み取り／書き込み）

有効値

指定タグ変数に対する設定範囲内の値

例

以下のステートメントにより、タグ変数 **MyTag** に対する HiHi アラームしきい値が 5 だけ増加します。
`MyTag.HiHiLimit=MyTag.HiHiLimit + 5;`

参照項目

.Alarm、.AlarmValue、.Ack、.HiHiStatus、.HiHiSet、.AlarmDisabled、.AlarmEnabled、.AlarmHiHiDisabled、.AlarmHiHiEnabled、.AlarmHiHiInhibitor

.MinorDevPct ドットフィールド

タグ変数の小偏差アラームのしきい値を変更します。

カテゴリ

アラーム

使用法

TagName.MinorDevPct

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

WindowViewer を意図的にまたは誤ってシャットダウンした後も継続してこのドットフィールドのランタイム値を使用するには、タグ変数ディクショナリで **[パラメータ値保持]** オプションを選択してください。

データタイプ

実数型（読み取り／書き込み）

有効値

0 ～ 100

例

以下のステートメントは、タグ変数 **MyTag** の小偏差しきい値のプロパティを 25 パーセントに設定します。

```
MyTag.MinorDevPct=25;
```

参照項目

.AckDev、.AlarmDev、.AlarmMinDevDisabled、.AlarmMinDevEnabled、.AlarmMinDevInhibitor、.MinorDevSet、.MinorDevStatus

.MajorDevPct ドットフィールド

タグ変数の大偏差アラームのしきい値を変更します。

カテゴリ

アラーム

使用法

```
TagName.MajorDevPct
```

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

WindowViewer を意図的にまたは誤ってシャットダウンした後も継続してこのドットフィールドのランタイム値を使用するには、タグ変数ディクショナリで **[パラメータ値保持]** オプションを選択してください。

データタイプ

実数型（読み取り／書き込み）

有効値

0 ～ 100

例

以下のステートメントは、タグ変数 **MyTag** の大偏差しきい値のプロパティを 25 パーセントに設定します。

```
MyTag.MajorDevPct=25;
```

参照項目

.AckDev、.AlarmDev、.AlarmMajDevDisabled、.AlarmMajDevEnabled、.AlarmMajDevInhibitor、.MajorDevSet、.MajorDevStatus

.DevTarget ドットフィールド

タグの小偏差および大偏差アラームの基準値を変更します。

カテゴリ

アラーム

使用法

`TagName.DevTarget`

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ。

備考

WindowViewer を意図的にまたは誤ってシャットダウンした後も継続してこのドットフィールドのランタイム値を使用するには、タグ名ディクショナリで [パラメータ値保持] オプションを選択してください。

データ タイプ

実数型（読み取り／書き込み）

有効値

タグに指定された範囲内の値である必要があります。

例

以下のステートメントは、タグ **MyTag** の偏差基準値を **500** に設定します。

```
MyTag.DevTarget=500;
```

参照項目

.AckDev、.AlarmDev、.AlarmMajDevDisabled、.AlarmMajDevEnabled、.AlarmMajDevInhibitor、.MajorDevSet、.MajorDevStatus

.ROCPct ドットフィールド

タグ変数の変化率アラームしきい値を変更します。

カテゴリ

アラーム

使用法

`TagName.ROCPct`

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

タグ変数ディクショナリの [アラーム] で設定されている同じフィールドに直接対応しています。

データタイプ

整数型（読み取り／書き込み）

有効値

0 ～ 100

例

以下のステートメントは、タグ変数 **MyTag** の変化率アラームしきい値を 25 パーセントに設定します。
MyTag.ROCPct=25;

参照項目

.ROCStatus、.ROCSet

タグ変数のアラーム デッドバンドの変更

アプリケーションの実行中にタグ変数のアラーム デッドバンドの範囲を変更するには、以下のドットフィールドを使用します。

- [.AlarmValDeadband ドットフィールド](#)
- [.AlarmDevDeadband ドットフィールド](#)

.AlarmValDeadband ドットフィールド

InTouch アプリケーションの実行中に、タグのデッドバンド値を変更します。

カテゴリ

アラーム

使用法

TagName.AlarmValDeadband

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ。

備考

WindowViewer を意図的にまたは誤ってシャットダウンした後も継続してこのドットフィールドのランタイム値を使用するには、タグ名ディクショナリで [パラメータ値保持] オプションを選択してください。

データ タイプ

アナログ型（読み取り／書き込み）

有効値

タグに指定された範囲内の値である必要があります。

例

以下のステートメントは、タグ **Tag1** に対するデッドバンド値を **25** に変更します。

```
Tag1.AlarmValDeadband=25;
```

参照項目

.AlarmDevDeadband

.AlarmDevDeadband ドットフィールド

大小両方の偏差アラームに対するタグ変数の偏差率デッドバンドを変更します。

カテゴリ

アラーム

使用法

```
TagName.AlarmDevDeadband
```

パラメータ

TagName

任意の整数型、実数型、または間接アナログ型のタグ変数。

備考

WindowViewer を意図的にまたは誤ってシャットダウンした後も継続してこのドットフィールドのランタイム値を使用するには、タグ変数ディクショナリで **[パラメータ値保持]** オプションを選択してください。

データタイプ

整数型（読み取り／書き込み）

有効値

0 ～ 100

例

以下のステートメントは、偏差デッドバンドのパーセント率を **25** パーセントに変更します。

```
tag.AlarmDevDeadband=25;
```

参照項目

.AlarmValDeadband、.AlarmDev

タグ変数に関連付けられたアラーム コメントの変更

.AlarmComment ドットフィールドは、タグ変数またはアラーム グループのアラームに関連付けられたコメントの文字列を返します。

.AlarmComment ドットフィールド

タグ変数またはアラーム グループのアラームに関連付けられたコメントの文字列を返します。新規アプリケーションのデフォルト値は空白です。

ユーザー定義情報とアラーム インスタンスとの関連付け

アラーム レコードには、3つの項目（2つの数値と1つの文字列）を付加することができます。アラーム レコードに情報を追加するには、以下のドットフィールドを使用します。

- [.AlarmUserDefNumX ドットフィールド](#)
- [.AlarmUserDefStr ドットフィールド](#)

.AlarmUserDefNumX ドットフィールド

特定のタグ変数だけでなくアラーム グループにドットフィールドを設定できるので、ユーザー値の設定が簡単化できます。たとえば、InTouch で \$System アラーム グループに対して .AlarmUserDefNum1 を使ってバッチ番号を設定し、すべてのアラームにバッチ番号を付加することができます。

.AlarmUserDefNum1 と .AlarmUserDefNum2 はそれぞれ、Alarm Viewer コントロールのユーザー 1 とユーザー 2 のカラムに対応しています。

アラーム グループに対して .AlarmUserDefNum1 を設定した場合、この値はこのアラーム グループとサブグループ内のすべてのアラームに適用されます。また、特定のタグ変数に対して .AlarmUserDefNum1 の値を設定することもできます。この場合、設定した値はそのタグ変数にのみ適用され、そのタグ変数のアラーム グループ内にある .AlarmUserDefNum1 の設定は、この値で上書きされます。

カテゴリ

アラーム

使用法

`TagName.AlarmUserDefNum1`

`TagName.AlarmUserDefNum2`

パラメータ

TagName

論理型、整数型、実数型、間接論理型、間接アナログ型のタグ変数、またはアラーム グループ

備考

このユーザー定義のドットフィールドは、さまざまなタグ変数、特に論理型タグ変数、アナログ型タグ変数、およびアラーム グループに対して（定義されたアラームの有無にかかわらず）有効です。これらのアイテムを未設定のままにすること、すべてのアイテムを設定すること、あるいは個々のタグ変数、グループ、親グループに対する一部のアイテムだけを設定することが可能です。

このドットフィールドの値は、スクリプトや Poke などによって設定済みの場合のみ、アラームに付加されます。

データ タイプ

アナログ型（読み取り／書き込み）

有効値

任意の実数値、または未設定（デフォルト）

例

以下の例では、定数値が使用されています。ただし、InTouch QuickScript を使用すれば、他のタグ変数の値をこれらのユーザー定義フィールドのいずれかにコピーすることができます。また、PtAcc を使用し

て設定や確認をしたり、InTouch を I/O Server として使用して値を取得したり、設定したりすることもできます。

```
$System.AlarmUserDefNum1 = 4;  
GroupA.AlarmUserDefNum1 = 27649;
```

アラーム通知が分散アラーム システムに送信される場合は、最低レベルの設定を優先するコンセプトになっています。つまり、タグ変数の **.AlarmUserDefNum1** に値が設定されている場合、アラーム レコードはこの設定を使用して記録されます。ただし、タグ変数にこの設定がない場合は、タグ変数のアラーム グループに設定があるかがルート グループの **\$System** に達するまでさかのぼって **WindowViewer** によってチェックされます。タグ変数の設定がいずれのレベルにもない場合、アラーム レコード内のエントリは空のままとなります（数値に対してはゼロ、文字列に対しては空の文字列）。

注意：この階層検索は、アイテムごとに独立して処理されます。このため、タグ変数に **.AlarmUserDefNum2** だけが設定されていて **.AlarmUserDefNum1** は設定されておらず、親グループで **.AlarmUserDefNum1** が設定されている場合は、タグ変数は親グループから **.AlarmUserDefNum1** の設定を受け継ぎます。

参照項目

.AlarmUserDefStr

.AlarmUserDefStr ドットフィールド

.AlarmUserDefStr ドットフィールドは、Alarm DB Logger によってアラームごとにアラーム データベースに記録される情報に付加されます。**.AlarmUserDefStr** ドットフィールドは、データベースの **User3** フィールドに対応しています。**SELECT** ステートメントで「ユーザー定義」カラムを使用すると、特定のアラームの集合を選択してデータベース操作を行うことができます。たとえば、**\$System.AlarmUserDefStr** をバッチ文字列に設定し、バッチが変化するたびにこの値も変化するようにした場合、データベースの **User3** フィールドを含む選択を使用して特定のバッチのアラームを選択できます。

カテゴリ

アラーム

使用法

Tagname.AlarmUserDefStr

パラメータ

Tagname

論理型、整数型、実数型、間接論理型、間接アナログ型のタグ変数、またはアラーム グループ

備考

このユーザー定義のドットフィールドは、さまざまなタグ変数、特に論理型タグ変数、アナログ型タグ変数、およびアラーム グループに対して（定義されたアラームの有無にかかわらず）有効です。これらのアイテムを未設定のままにすること、すべてのアイテムを設定すること、あるいは個々のタグ変数、グループ、親グループに対する一部のアイテムだけを設定することが可能です。

このドットフィールドの値は、スクリプトや **Poke** などによって設定されている場合にのみアラームに付加されます。

データタイプ

メッセージ型（読み取り／書き込み）

有効値

NULL および任意の有効な文字列

例

上記の例では、定数値が使用されています。ただし、InTouch QuickScript を使用すれば、他のタグ変数の値をこれらのユーザー定義フィールドのいずれかにコピーすることができます。また、PtAcc を使用して設定や確認をしたり、InTouch を I/O Server として使用して値を取得したり、設定したりすることもできます。

```
Tag04.AlarmUserDefStr = "Joe";
```

アラーム通知が分散アラーム システムに送信される場合は、最低レベルの設定を優先するコンセプトになっています。つまり、タグ変数の .AlarmUserDefStr に値が設定されている場合、アラーム レコードはこの設定を使用して記録されます。ただし、タグ変数にこの設定がない場合は、タグ変数のアラーム グループに設定があるかがルート グループの \$System に達するまでさかのぼって WindowViewer によってチェックされます。タグ変数の設定がいずれのレベルにもない場合、アラーム レコード内のエントリは空のままとなります（数値に対してはゼロ、文字列に対しては空の文字列）。

また、この階層検索は、アイテムごとに独立して処理されます。したがって、タグ変数に .AlarmUserDefNum1 だけが設定されていて .AlarmUserDefStr が設定されておらず、その親グループには .AlarmUserDefStr の設定がある場合は、この設定がアラーム レコードで使用されます。

参照項目

.AlarmUserDefNumX

タグ変数またはアラーム グループの抑止タグ変数の確認

各タイプのアラームの抑止タグ変数を調べるには、以下のドットフィールドを使用します。

- [.AlarmDscInhibitor ドットフィールド](#)
- [.AlarmLoLoInhibitor ドットフィールド](#)
- [.AlarmLoInhibitor ドットフィールド](#)
- [.AlarmHiInhibitor ドットフィールド](#)
- [.AlarmHiHiInhibitor ドットフィールド](#)
- [.AlarmMinDevInhibitor ドットフィールド](#)
- [.AlarmMajDevInhibitor ドットフィールド](#)
- [.AlarmROCIInhibitor ドットフィールド](#)

.AlarmDscInhibitor ドットフィールド

論理値アラームに割り当てられた抑止タグ変数の名前を返します。

カテゴリ

アラーム

使用法

```
TagName.AlarmDscInhibitor
```

パラメータ

TagName

任意の論理型タグ変数またはアラーム グループ

備考

WindowMaker で設定します。ランタイム中は変更できません。

データ タイプ

メッセージ型（読み取り専用）

例

.AlarmDSCInhibitor ドットフィールドを使用するには、任意の間接型タグ変数の **.Name** にタグ変数の **.AlarmDSCInhibitor** と同じ値をセットしてから、その間接型タグ変数の値を操作します。

以下のステートメントは、論理値アラームに対するアラーム抑止タグ変数名を返します(**SomeIndirectTag** はアナログ型間接タグ変数とする)。

```
SomeIndirectTag.Name = AlarmedTag.AlarmDscInhibitor;
```

アラーム状況にあるタグ変数の抑止状態は、間接型タグ変数の値を次のように設定することによって制御できます。

```
SomeIndirectTag = 1;
```

抑止をオンにします。**AlarmedTag** の論理値アラームは無効になります。

```
SomeIndirectTag = 0;
```

抑止をオフにします。

AlarmedTag の論理値アラームが生成できるようになります。

.AlarmLoLoInhibitor ドットフィールド

LoLo アラームに割り当てられた抑止タグ変数の名前を返します。

カテゴリ

アラーム

使用法

```
TagName.AlarmLoLoInhibitor
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループタグ変数

備考

WindowMaker で設定します。ランタイム中は変更できません。

データ タイプ

メッセージ型（読み取り専用）

例

.AlarmLoLoInhibitor ドットフィールドを使用するには、任意の間接型タグ変数の **.Name** にタグ変数の **.AlarmLoLoInhibitor** と同じ値をセットしてから、その間接型タグ変数の値を操作します。

以下のステートメントは、LoLo アラームしきい値に対するアラーム抑止タグ変数名を返します(**SomeIndirectTag** はアナログ型間接タグ変数とする)。

```
SomeIndirectTag.Name = AlarmedTag.AlarmLoLoInhibitor;
```

アラーム状況にあるタグ変数の抑止状態は、間接型タグ変数の値を次のように設定することによって制御できます。

```
SomeIndirectTag = 1;
```

抑止をオンにします。

AlarmedTag の LoLo アラームが無効になります。

```
SomeIndirectTag = 0;
```

抑止をオフにします。

AlarmedTag の LoLo アラームが生成できるようになります。

参照項目

.AlarmHiInhibitor、.AlarmHiHiInhibitor、.AlarmLoInhibitor

.AlarmLoInhibitor ドットフィールド

Lo アラームに割り当てられた抑止タグ変数の名前を返します。

カテゴリ

アラーム

使用法

```
TagName.AlarmLoInhibitor
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループタグ変数

備考

WindowMaker で設定します。ランタイム中は変更できません。

データ タイプ

メッセージ型（読み取り専用）

例

.AlarmLoInhibitor ドットフィールドを使用するには、任意の間接型タグ変数の **.Name** に タグ変数の **.AlarmLoInhibitor** と同じ値をセットしてから、その間接型タグ変数の値を操作 します。

以下のステートメントは、Lo アラームしきい値に対するアラーム抑止タグ変数名を返します（SomeIndirectTag はアナログ型間接タグ変数とする）。

```
SomeIndirectTag.Name = AlarmedTag.AlarmLoInhibitor;
```

アラーム状況にあるタグ変数の抑止状態は、間接型タグ変数の値を次のように設定することによって制御できます。

```
SomeIndirectTag = 1;
```

抑止をオンにします。

AlarmedTag の Lo アラームが無効になります。

```
SomeIndirectTag = 0;
```

抑止をオフにします。

AlarmedTag の Lo アラームが生成できるようになります。

参照項目

.AlarmHiInhibitor、.AlarmHiHiInhibitor、.AlarmLoLoInhibitor

.AlarmHiInhibitor ドットフィールド

Hi アラームに割り当てられた抑止タグ変数の名前を返します。

カテゴリ

アラーム

使用法

```
TagName.AlarmHiInhibitor
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループタグ変数

備考

WindowMaker で設定します。ランタイム中は変更できません。

データ タイプ

メッセージ型（読み取り専用）

例

.AlarmHiInhibitor ドットフィールドを使用するには、任意の間接型タグ変数の **.Name** にタグ変数の **.AlarmHiInhibitor** と同じ値をセットしてから、その間接型タグ変数の値を操作します。

以下のステートメントは、Hi アラームしきい値に対するアラーム抑止タグ変数名を返します（**SomeIndirectTag** はアナログ型間接タグ変数とする）。

```
SomeIndirectTag.Name = AlarmedTag.AlarmHiInhibitor;
```

アラーム状況にあるタグ変数の抑止状態は、間接型タグ変数の値を次のように設定することによって制御できます。

```
SomeIndirectTag = 1;
```

抑止をオンにします。

AlarmedTag の Hi アラームが無効になります。

```
SomeIndirectTag = 0;
```

抑止をオフにします。

AlarmedTag の Hi アラームが生成できるようになります。

参照項目

.AlarmHiHiInhibitor、.AlarmLoInhibitor、.AlarmLoLoInhibitor

.AlarmHiHiInhibitor ドットフィールド

HiHi アラーム状況に割り当てられた抑止タグ変数の名前を返します。

カテゴリ

アラーム

使用法

```
TagName.AlarmHiHiInhibitor
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループタグ変数

備考

WindowMaker で設定します。ランタイム中は変更できません。

データ タイプ

メッセージ型（読み取り専用）

例

.AlarmHiHiInhibitor ドットフィールドを使用するには、任意の間接型タグ変数の .Name に タグ変数の .AlarmHiHiInhibitor と同じ値をセットしてから、その間接型タグ変数の値を操作 します。以下のステートメントは、HiHi アラームしきい値に対するアラーム抑止タグ変数名を返します（SomeIndirectTag はアナログ型間接タグ変数とする）。

```
SomeIndirectTag.Name = AlarmedTag.AlarmHiHiInhibitor;
```

アラーム状況にあるタグ変数の抑止状態は、間接型タグ変数の値を次のように設定することによって制御できます。

```
SomeIndirectTag = 1;
```

抑止をオンにします。

AlarmedTag の HiHi アラームが無効になります。

```
SomeIndirectTag = 0;
```

抑止をオフにします。

AlarmedTag の HiHi アラームが生成できるようになります。

参照項目

.AlarmHiInhibitor、.AlarmLoInhibitor、.AlarmLoLoInhibitor

.AlarmMinDevInhibitor ドットフィールド

小偏差アラーム状況に関連付けられているアラーム抑止タグ変数名を返します。

カテゴリ

アラーム

使用法

```
TagName.AlarmMinDevInhibitor
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループタグ変数

備考

WindowMaker で設定します。ランタイム中は変更できません。

データ タイプ

メッセージ型（読み取り専用）

例

.AlarmMinDevInhibitor ドットフィールドを使用するには、任意の間接型タグ変数の **.Name** に タグ変数の **.AlarmMinDevInhibitor** と同じ値をセットしてから、その間接型タグ変数の値を操作します。以下のステートメントは、小偏差アラームしきい値に対するアラーム抑止タグ変数名を返します(**SomeIndirectTag** はアナログ型間接タグ変数とする)。

```
SomeIndirectTag.Name = AlarmedTag.AlarmMinDevInhibitor;
```

アラーム状況にあるタグ変数の抑止状態は、間接型タグ変数の値を次のように設定することによって制御できます。

```
SomeIndirectTag = 1;
```

抑止をオンにします。

AlarmedTag の 小偏差アラームが無効になります。

```
SomeIndirectTag = 0;
```

抑止をオフにします。

AlarmedTag の 小偏差アラームが生成できるようになります。

参照項目

.AlarmMajDevInhibitor

.AlarmMajDevInhibitor ドットフィールド

大偏差アラーム状況に関連付けられているアラーム抑止タグ変数名を返します。

カテゴリ

アラーム

使用法

```
TagName.AlarmMajDevInhibitor
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループタグ変数

データ タイプ

メッセージ型（読み取り専用）

例

.AlarmMajDevInhibitor ドットフィールドを使用するには、任意の間接型タグ変数の **.Name** に タグ変数の **.AlarmMajDevInhibitor** と同じ値をセットしてから、その間接型タグ変数の値を操作します。以下のステートメントは、大偏差アラームしきい値に対するアラーム抑止タグ変数名を返します(**SomeIndirectTag** はアナログ型間接タグ変数とする)。

```
SomeIndirectTag.Name = AlarmedTag.AlarmMajDevInhibitor;
```

アラーム状況にあるタグ変数の抑止状態は、間接型タグ変数の値を次のように設定することによって制御できます。

```
SomeIndirectTag = 1;
```

抑止をオンにします。

AlarmedTag の 大偏差アラームが無効になります。

```
SomeIndirectTag = 0;
```

抑止をオフにします。

AlarmedTag の大偏差アラームが生成できるようになります。

参照項目

.AlarmMinDevInhibitor

.AlarmROCIInhibitor ドットフィールド

変化率アラーム状況に関連付けられているアラーム抑止タグ変数を返します。

カテゴリ

アラーム

使用法

TagName.AlarmROCIInhibitor

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループタグ変数

データ タイプ

メッセージ型（読み取り専用）

例

.AlarmROCIInhibitor ドットフィールドを使用するには、任意の間接型タグ変数の .Name に タグ変数の .AlarmROCIInhibitor と同じ値をセットしてから、その間接型タグ変数の値を操作します。以下のステートメントは、変化率アラームしきい値に対するアラーム抑止タグ変数名を返します（SomeIndirectTag はアナログ型間接タグ変数とする）。

```
SomeIndirectTag.Name = AlarmedTag.AlarmROCIInhibitor;
```

アラーム状況にあるタグ変数の抑止状態は、間接型タグ変数の値を次のように設定することによって制御できます。

```
SomeIndirectTag = 1;
```

抑止をオンにします。

AlarmedTag の変化率アラームが無効になります。

```
SomeIndirectTag = 0;
```

抑止をオフにします。

AlarmedTag の変化率アラームが生成できるようになります。

アクティブまたは未確認のアラーム数の確認

アクティブまたは未確認のアラーム数をアプリケーションの実行中に調べるには、以下のドットフィールドを使用します。

ドットフィールド	説明
.AlarmTotalCount ドットフィールド	タグ変数またはアラーム グループに関連付けられているアラームの数をカウントします。
.AlarmUnAckCount ドットフィールド	タグ変数またはアラーム グループに関連付けられている未確認アラームの数をカウントします。

ドットフィールド	説明
.AlarmValueCount ドットフィールド	タグ変数に関連付けられた値型アラームの数をカウントします。
.AlarmValueUnAckCount ドットフィールド	タグ変数に関連付けられた未確認の値型アラームの数をカウントします。
.AlarmDscCount ドットフィールド	論理値アラームの数をカウントします。
.AlarmDscUnAckCount ドットフィールド	未確認の論理値アラームの数をカウントします。
.AlarmDevCount ドットフィールド	偏差アラームの数をカウントします。
.AlarmDevUnAckCount ドットフィールド	未確認の偏差アラームの数をカウントします。
.AlarmROCCount ドットフィールド	変化率アラームの数をカウントします。
.AlarmROCUnAckCount ドットフィールド	未確認の変化率アラームの数をカウントします。

.AlarmTotalCount ドットフィールド

指定されたタグ変数またはアラーム グループでアクティブなアラームの合計数を追跡します。

カテゴリ

アラーム

使用法

```
TagName.AlarmTotalCount
```

パラメータ

TagName

任意のタイプのタグ変数、またはアラーム グループ

備考

カウントには、値型、偏差、変化率、および論理型の各アラームが含まれます。これには、確認済みと未確認の両方のアラームが含まれます。

データ タイプ

整数型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、Tag1 はアラームに対して設定されたアナログ型タグ変数です。ATC もまたアナログ型タグ変数で、Tag1 に存在するすべてのアクティブなアラーム（未確認と確認の両方）の合計数を取得します。

```
ATC = Tag1.AlarmTotalCount;
```

参照項目

.AlarmDevCount、.AlarmDevUnAckCount、.AlarmDSCCount、.AlarmDSCUnAckCount、.AlarmValueCount、.AlarmUnAckCount、.AlarmValueUnAckCount、.AlarmROCCount、.AlarmROCUnAckCount

.AlarmUnAckCount ドットフィールド

指定されたタグ変数またはアラーム グループの未確認アラームの合計数を追跡します。

カテゴリ

アラーム

使用法

```
TagName.AlarmUnAckCount
```

パラメータ

TagName

任意のタイプのタグ変数、またはアラーム グループ

備考

カウントには、値型、偏差、変化率、および論理型の未確認アラームが含まれます。

データ タイプ

整数型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、Tag1 はアラームに対して設定されたアナログ型または論理型のタグ変数です。AUC はアナログ型タグ変数で、Tag1 に存在する未確認アラームの合計数を取得します。

```
AUC = Tag1.AlarmUnAckCount;
```

参照項目

.AlarmDevCount、.AlarmDevUnAckCount、.AlarmDscCount、.AlarmDscUnAckCount、.AlarmValueCount、.AlarmTotalCount、.AlarmValueUnAckCount、.AlarmROCCount、.AlarmROCUnAckCount

.AlarmValueCount ドットフィールド

指定されたタグ変数またはアラーム グループでアクティブな値型アラームの合計数を追跡します。

カテゴリ

アラーム

使用法

```
TagName.AlarmValueCount
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

備考

これには、HiHi、Hi、Lo、LoLo アラームの数が含まれます。これには、確認済みと未確認の両方のアラームが含まれます。非拡張サマリ型のアラーム タグ変数の場合、このカウント数は1以下です。ただし、アラーム グループによってはカウント数が変わることがあります。

データ タイプ

整数型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、Tag1 は値型アラームに対して設定されたアナログ型タグ変数です。AVC もアナログ型タグ変数で、Tag1 に存在するすべてのアラーム値の合計数を取得します。

```
AVC = Tag1.AlarmValueCount;
```

参照項目

.AlarmDevCount、.AlarmDevUnAckCount、.AlarmDscCount、.AlarmDscUnAckCount、.AlarmROCCount、.AlarmTotalCount、.AlarmValueUnAckCount、.AlarmROCUnAckCount、.AlarmUnAckCount

.AlarmValueUnAckCount ドットフィールド

指定されたタグ変数またはアラーム グループの未確認値型アラームの合計数を追跡します。これには、HiHi、Hi、Lo、LoLo アラームの数が含まれます。

カテゴリ

アラーム

使用法

```
TagName.AlarmValueUnAckCount
```

パラメータ

TagName

整数型、実数型、間接アナログ型のタグ変数、またはアラーム グループ

データ タイプ

整数型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、Tag1 は値型アラームに対して設定されたアナログ型タグ変数です。AVUC もアナログ型タグ変数で、Tag1 に存在するすべての未確認値型アラームの合計数を取得します。

```
AVUC = Tag1.AlarmValueUnAckCount;
```

参照項目

.AlarmDevCount、.AlarmDevUnAckCount、.AlarmDscCount、.AlarmDscUnAckCount、.AlarmROCCount、.AlarmTotalCount、.AlarmValueCount、.AlarmROCUAckCount、.AlarmUnAckCount

.AlarmDscCount ドットフィールド

指定されたタグ変数またはアラーム グループでアクティブな論理値アラームの合計数を追跡します。

カテゴリ

アラーム

使用法

```
TagName.AlarmDscCount;
```

パラメータ

TagName

任意の論理型タグ変数、間接論理型タグ変数、またはアラーム グループ

備考

AlarmDscCount ドットフィールドに割り当てられたこのカウント数には、確認済みと未確認の両方のアラームが含まれています。非拡張サマリ型のアラーム タグ変数の場合、このカウント数は 常に 1 です。ただし、アラーム グループによってはカウント数が変わることがあります。

データ タイプ

整数型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、Tag1 は論理値アラームに対して設定された論理型タグ変数です。ADC はアナログ型タグ変数で、Tag1 に存在するアクティブな論理値アラーム（未確認と確認の両方）の合計数を取得します。

```
ADC = Tag1.AlarmDSCCount;
```

参照項目

.AlarmDevCount、.AlarmDevUnAckCount、.AlarmValueCount、.AlarmROCUAckCount、.AlarmTotalCount、.AlarmDscUnAckCount、.AlarmValueUnAckCount、.AlarmROCUAckCount、.AlarmUnAckCount

.AlarmDscUnAckCount ドットフィールド

指定されたタグ変数またはアラーム グループの未確認の論理値アラームの合計数を追跡します。

カテゴリ

アラーム

使用法

```
TagName.AlarmDscUnAckCount
```

パラメータ

TagName

任意の論理型タグ変数、間接論理型タグ変数、またはアラーム グループ

データ タイプ

整数型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、Tag1 は論理値アラームに対して設定された論理型タグ変数です。ADUC はアナログ型タグ変数で、Tag1 に存在する未確認の論理値アラームの合計数を取得します。

```
ADUC = Tag1.AlarmDscUnAckCount;
```

参照項目

.AlarmDevCount、.AlarmDevUnAckCount、.AlarmDscCount、.AlarmValueCount、.AlarmROCCount、.AlarmTotalCount、.AlarmValueUnAckCount、.AlarmROCUAckCount、.AlarmUnAckCount

.AlarmDevCount ドットフィールド

指定されたタグ変数またはアラーム グループでアクティブな偏差アラームの合計数を追跡します。

カテゴリ

アラーム

使用法

```
TagName.AlarmDevCount
```

パラメータ

TagName

実数型タグ変数、整数型タグ変数、間接アナログ型のタグ変数、またはアラーム グループ

備考

これには、大偏差アラームおよび小偏差アラームのカウント数が含まれます。これには、確認済みと未確認の両方のアラームが含まれます。非拡張サマリ型のアラーム タグ変数の場合、このカウント数は常に 1 です。ただし、アラーム グループによってはカウント数が変わることがあります。

データ タイプ

アナログ型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、Tag1 は偏差アラームに対して設定されたアナログ型タグ変数です。ADC もまたアナログ型タグ変数で、Tag1 に存在するアクティブな偏差アラーム（未確認と確認の両方）の合計数を取得します。

```
ADC=Tag1.AlarmDevCount;
```

参照項目

.AlarmDSCCount、.AlarmValueCount、.AlarmROCUAckCount、.AlarmTotalCount、.AlarmDSCUnAckCount、.AlarmValueUnAckCount、.AlarmDevUnAckCount、.AlarmROCUAckCount、.AlarmUnAckCount

.AlarmDevUnAckCount ドットフィールド

指定されたタグ変数またはアラーム グループの未確認の偏差アラームの合計数を追跡します。これには、大偏差アラームおよび小偏差アラームのカウント数が含まれます。

カテゴリ

アラーム

使用法

```
TagName.AlarmDevUnAckCount
```

パラメータ

TagName

実数型タグ変数、整数型タグ変数、間接アナログ型のタグ変数、またはアラーム グループ

データ タイプ

アナログ型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、Tag1 は偏差アラームに対して設定されたアナログ型タグ変数です。ADUC もまたアナログ型タグ変数で、Tag1 に存在する未確認の偏差アラームの合計数を取得します。

```
ADUC = Tag1.AlarmDevUnAckCount;
```

参照項目

.AlarmDevCount、.AlarmDSCCount、.AlarmValueCount、.AlarmROCUAckCount、.AlarmTotalCount、.AlarmDSCUnAckCount、.AlarmValueUnAckCount、.AlarmROCUAckCount、.AlarmUnAckCount

.AlarmROCCount ドットフィールド

指定されたタグ変数またはアラーム グループでアクティブな変化率アラームの合計数を追跡します。これには、確認済みと未確認の両方のアラームが含まれます。非拡張サマリ型のアラーム タグ変数の場合、このカウント数は常に 1 になります。ただし、アラーム グループによってはカウント数が変わることがあります。

カテゴリ

アラーム

使用法

```
TagName.AlarmROCCount
```

パラメータ

TagName

実数型タグ変数、整数型タグ変数、間接アナログ型のタグ変数、またはアラーム グループ

データ タイプ

整数型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、**Tag1** は変化率アラームに対して設定されたアナログ型タグ変数です。**ARC** もまたアナログ型タグ変数で、**Tag1** に存在するアクティブな変化率アラーム（未確認と確認の両方）の合計数を取得します。

```
ARC = Tag1.AlarmROCCount;
```

参照項目

.AlarmDevCount、.AlarmDevUnAckCount、.AlarmDscCount、.AlarmDscUnAckCount、.AlarmValueCount、.AlarmTotalCount、.AlarmValueUnAckCount、.AlarmROCCount、.AlarmUnAckCount

.AlarmROCCount ドットフィールド

指定されたアナログ型タグ変数またはアラーム グループの未確認の変化率アラームの合計数を追跡します。

カテゴリ

アラーム

使用法

```
TagName.AlarmROCCount
```

パラメータ

TagName

実数型タグ変数、整数型タグ変数、間接アナログ型のタグ変数、またはアラーム グループ

データ タイプ

整数型（読み取り専用）

有効値

0 または任意の正の整数

例

この例で、**Tag1** は変化率アラームに対して設定されたアナログ型タグ変数です。**ARUC** もアナログ型タグ変数で、**Tag1** に存在する未確認の変化率アラームの合計数を取得します。

```
ARUC = Tag1.AlarmROCCount;
```

参照項目

.AlarmDevCount、.AlarmDevUnAckCount、.AlarmDscCount、.AlarmDscUnAckCount、.AlarmValueCount、.AlarmTotalCount、.AlarmValueUnAckCount、.AlarmROCCount、.AlarmUnAckCount

Maintain

章 23 アプリケーションの移行およびアップグレード

このセクションでは、さまざまな InTouch アプリケーションの移行とアップグレードについて説明します。

従来のアプリケーションから新しいスタンドアロン アプリケーションへの移動

System Platform 2020 以前では、InTouch HMI ユーザーは、以下のタイプのアプリケーションを作成することができました。

- スタンドアロン
- モダン
- マネージド
- パブリッシュ済み

スタンドアロン アプリケーションは、従来のシンボルおよびコントロールを使用して構築されました。モダン アプリケーションは、従来のシンボルに加えて産業用グラフィック（以前の名称は **ArchestrA** グラフィック/シンボル）をサポートしました。マネージド アプリケーションは IDE および **Galaxy** オブジェクトを使用して構築されました。スタンドアロン アプリケーションはパッケージにパブリッシュして、他のノードに配布することができました（パブリッシュ済みアプリケーション）。

InTouch HMI 2020 では、モダン アプリケーションは、より包括的なスタンドアロン アプリケーションとして再設計されました。新しいスタンドアロン アプリケーションは、従来のスタンドアロン アプリケーションに比べて多くの機能強化が行われています。

- 容易な配布 – アプリケーション フォルダをコピーして別のノードにコピーできます。インポートやエクスポートの操作は必要ありません。
- 産業用グラフィックの使用 – 新しいスタンドアロン アプリケーションでは、従来のアプリケーションの使いやすさと最新の産業用グラフィックが組み合わされています。
- クラウド対応 – オンプレミス ノードで作成されたアプリケーションが **HTML5** 互換ブラウザで表示できるようになりました。
- 軽量 – アプリケーション ファイルは軽量なので、パフォーマンスと使用性が向上しています。

マネージド アプリケーションとパブリッシュ済みアプリケーションの動作は変更されていません。

古いアプリケーションの移行およびアップグレード

以前のバージョンの InTouch HMI で作成されたアプリケーションをサポートするために、2 つのワークフローを使用して新しいスタンドアロン アプリケーションに移行できます。

- 古いモダン アプリケーションのインプレース移行: ノードに古いモダン アプリケーションが含まれていて、ノードの製品バージョンをアップグレードする場合、アプリケーション マネージャを使用してアプリケーションを移行します。
- 以前のバージョンの InTouch HMI からエクスポートされたモダン アプリケーション .aapkg ファイルのインポート

以前の InTouch アプリケーションの現在のバージョンへの移行

古いバージョンの InTouch HMI で開発したアプリケーションを現在のバージョンに移行できます。WindowMaker または WindowViewer のどちらかで古いアプリケーションを開こうとすると、[アプリケーションの移行] ダイアログ ボックスが表示されます。このダイアログ ボックスでは、以下の操作を行うことができます。

- アプリケーションの解像度を変換する。
- 古いアプリケーションを現在のバージョンの InTouch HMI に移行する前に、バックアップ コピーを作成する。

既存のスタンドアロン、モダン、またはパブリッシュ済み InTouch アプリケーションを現在のバージョンの InTouch に移行できます。バックアップ コピーを作成するフォルダを指定したり、バックアップから除外するファイルを指定する必要があります。

1. アプリケーション リストから目的のアプリケーションをダブルクリックします。
[アプリケーションの移行] ダイアログ ボックスが表示されます。
2. アプリケーションの解像度を元の解像度から現在の解像度に変換するには、[アプリケーションの解像度を <既存の解像度> から <新しい解像度> に変換] チェック ボックスをオンにします。
3. デフォルトのバックアップ パス (<アプリケーション ディレクトリ>\Bak) を変更するには、[移行前にアプリケーションをバックアップ] チェック ボックスをオフにします。次に、[バックアップのパス] ボックスに、バックアップを保存するフォルダへのパスを入力します。フォルダが存在しない場合、フォルダを作成してからバックアップを作成する必要があります。
4. [ファイルを無視] ボックスで、バックアップから除外するファイルを指定できます。デフォルトでは、アプリケーションのディレクトリにあるすべてのファイルがバックアップされます。除外するファイルの名前を入力します。複数のファイル名を入力する場合は、各ファイル名をセミコロンで区切ります。または、名前の中の共通の文字を使って一連のファイルを除外するには、標準のワイルドカード文字（「*」および「?」）を使用します。
5. 必要なオプションを設定して [OK] をクリックします。

Application Migration

The application 'InTouch Training Application' has been developed with older version of InTouch. Select the options below to migrate the application:

Target Application Type: **Standalone**

☒ Convert the application resolution from 1280x1024 to 1600x900

☒ Backup the application before migration

Backup Path: C:\MyApps\InTouch Training Application\BAK

Ignore Files:

☒ Use default backup path

OK CANCEL

古いアラーム表示の変換

InTouch 7.11 より前のバージョンで構築されたアプリケーションを **WindowViewer** で開く場合、**WindowMaker** を実行してアプリケーションを変換することを要求するダイアログ ボックスが表示されます。変換を続行すると、デフォルト値ですべての標準のアラーム オブジェクトが分散アラーム オブジェクトに変換されます。色、フォント、式、およびアラーム クエリーの設定内容は保持されません。

アプリケーション設定の管理

アプリケーションパスなどの InTouch アプリケーション設定は、**Win.ini** ファイルに保存されます。**Win.ini** ファイルは以下のディレクトリにあります。

C:\Users\<User Name>\AppData\Local\Wonderware

WindowMaker は管理者ユーザーとして実行されます。**WindowViewer** は管理者ユーザーまたは標準ユーザーとして実行することができます。標準ユーザーは管理者ユーザー プロファイルの **Win.ini** ディレクトリにアクセスできません。したがって、アプリケーション開発者は、アプリケーションを開発するときに共通の **Win.ini** 属性を標準ユーザーの **Win.ini** プロファイルにコピーする必要があります。これにより、標準ユーザーが **WindowViewer** を起動したときに、管理者ユーザーの下で設定されているすべての属性も使用可能になります。共通の **Win.ini** 属性を変更するたびに、属性をコピーする必要があります。

InTouch アプリケーションのインポート

アプリケーション マネージャを使用して既存のモダン アプリケーションをインポートして、スタンドアロン アプリケーションに変換できます。スタンドアロン アプリケーションは別のノードにコピーできます。別のノードにコピーしたスタンドアロン アプリケーションは、[アプリケーションの検索] オプションを使用して見つけることができます。インポートやエクスポートの操作は必要ありません。

既存のモダン アプリケーションをインポートするには

1. [ファイル] メニューの [インポート] グループで [インポート] をクリックします。
[新規アプリケーションの作成:] [インポートするアプリケーションの選択] 画面が表示されます。
2. [アプリケーションの検索] セクションを使用して、インポートするアプリケーションを検索します。インポートするフォルダまたはファイルを検索します。
3. アプリケーションを選択して [次へ] をクリックします。
[アプリケーションの詳細を入力] 画面が表示されます。

The screenshot shows the 'Create New Application' dialog box with the 'Enter Application Details' tab selected. The 'Type' is set to 'Standalone'. The 'Application Name' and 'Directory Name' are both 'NewApp_001'. The 'Application Path' is 'C:\Users\Public\Windows\'. The 'Set Default Directory' checkbox is checked. The 'Resolution' is 'Screen Resolution', 'Width' is '1920', and 'Height' is '1080'. The 'Description' is 'New InTouch application with Graphic Toolbox symbols'. On the right, there is a '+ Symbol Library' button and a list showing 'STANDALONE | NewApp_001'. At the bottom are 'Back', 'Finish', and 'Cancel' buttons.

4. 必要に応じて設定を変更します。
[完了] をクリックします。
新しいアプリケーションが作成され、アプリケーション マネージャに表示されます。

注記: すべてのモダン アプリケーションはスタンドアロン アプリケーションとしてインポートされます。

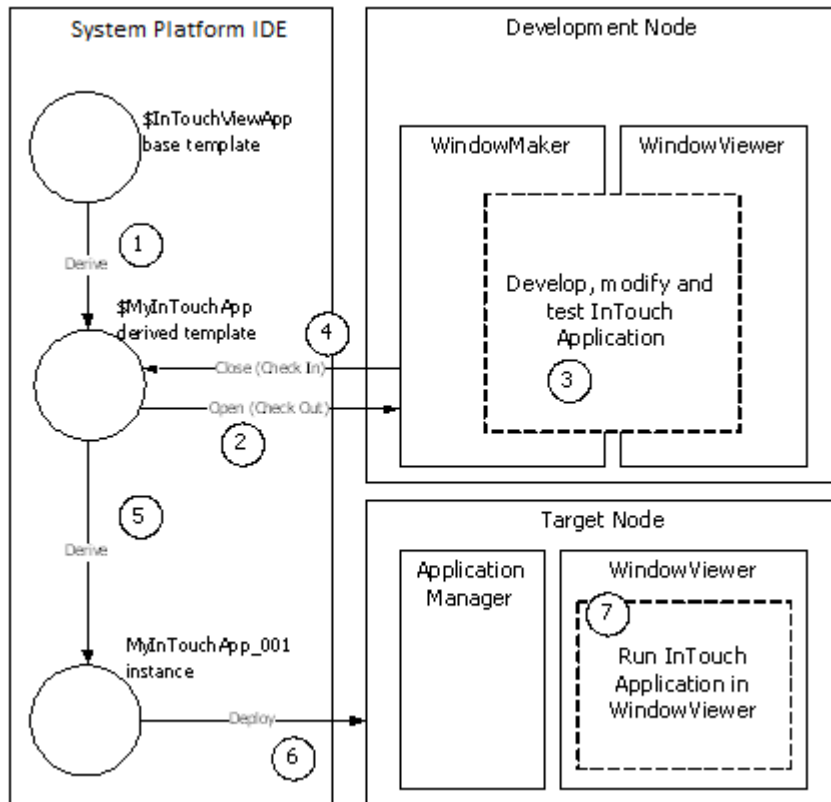
章 24 System Platform IDE を使用した InTouch アプリケーションの管理

IDE を使用して、InTouch アプリケーションを管理できます。以下の手順は、通常の方法を示しています。詳細については、「[IDE を使用した InTouch アプリケーションの管理](#)」を参照してください。

System Platform IDE の InTouch の機能は、次の 2 つのオートメーションオブジェクトによって処理されます。

- InTouchViewApp オブジェクトは、デザイン時およびランタイム時の InTouch アプリケーションを表します。
- ViewEngine オブジェクトは、Galaxy 内の対象ノードでの InTouch アプリケーションの実行方法を制御します。

以下の図は、System Platform IDE を使用して InTouch アプリケーションがどのように管理されるのかを示しています。



IDE を使用して InTouch アプリケーションを管理するには

1. マネージド InTouch アプリケーションを System Platform IDE で作成します。
2. 作成した InTouch アプリケーションを WindowMaker で開きます。
3. WindowMaker で InTouch アプリケーションを設定します。アプリケーションをテストするために、WindowViewer に切り替えることができます。

4. InTouch アプリケーションを保存し、WindowMaker と WindowViewer を終了します。
5. InTouch アプリケーションを配置するノードを決めます。
6. Galaxy 内の対象ノードに InTouch を配置します。
7. 対象ノード上の WindowViewer で InTouch アプリケーションを実行します。

InTouchViewApp オブジェクト

Application Server は、InTouchViewApp オブジェクトと呼ばれる特定のタイプの Application Server オブジェクトを使用して InTouch アプリケーションを管理します。

InTouchViewApp テンプレートは、デザイン時には 1 つの特定のマネージド InTouch アプリケーションを参照しており、ランタイム時には実行できません。

InTouchViewApp テンプレートのインスタンスを作成する必要があります。このインスタンスは、対象ノードに配置できます。ターゲット ノードとは、WindowViewer 内でマネージド InTouch アプリケーションが実行するノードのことです。

InTouch アプリケーションを分散させるには、同じテンプレートの複数のインスタンスを作成し、複数のノードに配置します。

オプションで、以下を行うことができます。

- Galaxy 間でマネージド InTouch アプリケーションを交換するために、InTouchViewApp オブジェクトをエクスポートおよびインポートします。
- タグのディクショナリ データを .csv ファイルとしてエクスポートおよびインポートします。
- 異なるタイプの InTouch アプリケーション間で、ウィンドウをエクスポートおよびインポートします。
- マネージド InTouch アプリケーションをパブリッシュします。パブリッシュ済み InTouch アプリケーションはスタンドアロン InTouch アプリケーションのように実行されますが、産業用グラフィックを含めることができます。
- Archestra 属性を持つ InTouch タグの読み取りおよび書き込みを行うには、配置された InTouchViewApp オブジェクトの属性を使用します。

InTouchViewApp オブジェクトを使用するには

1. InTouchViewApp テンプレートを \$InTouchViewApp base テンプレートから派生させます。
2. 新しい InTouch アプリケーションを作成するか、またはスタンドアロン InTouch アプリケーションをインポートすることにより、派生テンプレートを InTouch アプリケーションに関連付けます。
3. WindowMaker でアプリケーションを開きます。
4. WindowMaker でおよびアプリケーションを設定し、WindowViewer でテストします。
5. WindowMaker を保存して閉じます。InTouchViewApp テンプレートがチェック インされます。
6. InTouchViewApp テンプレートからインスタンスを派生させます。
7. これらのインスタンスを、Galaxy 内で選択した対象ノードに配置します。
8. 対象ノードでアプリケーション マネージャを実行し、WindowViewer でマネージド InTouch アプリケーションを実行します。

InTouchViewApp テンプレートと InTouch アプリケーションの関連付け

新しい InTouchViewApp テンプレートを作成した後、以下を行うことにより、InTouchViewApp テンプレートを InTouch アプリケーションに関連付けることができます。

- 新しい InTouch アプリケーションの作成。
- スタンドアロン InTouch アプリケーションのインポート

InTouchViewApp テンプレートには、タグ変数の設定および値などの InTouch アプリケーションのデータ自体は含まれていません。単にアプリケーションを参照しているだけです。

マネージド InTouch アプリケーションの編集

スタンドアロン InTouch アプリケーションで行う場合と同様に、WindowMaker を使用してマネージド InTouch アプリケーションを編集します。ただし、WindowMaker で関連付けられている InTouch アプリケーションを起動するために、InTouchViewApp テンプレートのエディタを開く場合を除きます。

InTouch アプリケーションへの変更を加えた後に WindowMaker を終了すると、InTouchViewApp オブジェクトが自動的にチェック インされます。

マネージド InTouch アプリケーションのテスト

スタンドアロン InTouch アプリケーションで行う場合と同様に、WindowViewer を使用してマネージド InTouch アプリケーションをテストできます。

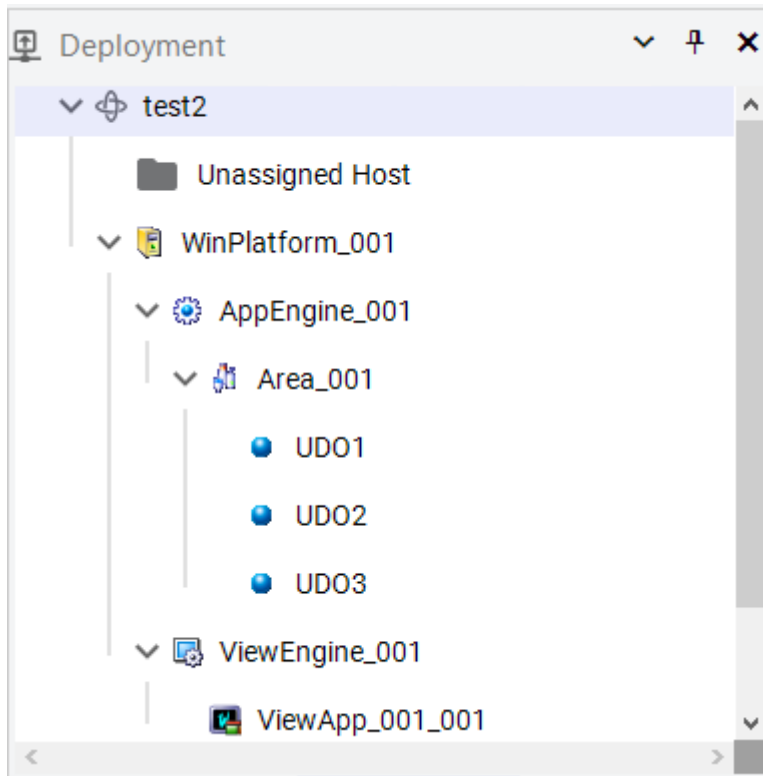
System Platform IDE から WindowMaker を開いた場合、WindowMaker と WindowViewer をすばやく切り替えてマネージドアプリケーションをテストできます。

マネージド InTouch アプリケーションに、galaxy:UDA などの Application Server データへの参照が含まれている場合、InTouch アプリケーションを編集しているノードに WinPlatform オブジェクトを配置する必要があります。そうしないと、データに値が表示されません。

InTouchViewApp オブジェクトの配置

InTouchViewApp テンプレートのインスタンスを派生させた後、ViewEngine オブジェクト下にある対象プラットフォームに割り当てることができます。

1 つの ViewEngine 下で同じ親を持つ複数の InTouchViewApp インスタンスを割り当ててすることはできません。その場合、同じ親を持つ、追加の InTouchViewApp インスタンスをホストする 2 つ目の ViewEngine インスタンスを作成します。



InTouchViewApp オブジェクトを配置した後、対象ノードの InTouch アプリケーション マネージャを開くことができます。関連付けられているマネージド InTouch アプリケーションと、[修正された日付] カラムでの最新の配置のタイム スタンプがリストに表示されます。

InTouchViewApp インスタンスを対象ノードに配置すると、InTouch アプリケーションが以下に含まれます。

- 開発ノード上のフォルダ。これには、InTouchViewApp テンプレートのソースが含まれます。
- InTouch アプリケーションが実行されている対象ノード上のフォルダ。これには、InTouch アプリケーションのインスタンスのコピーが含まれます。

InTouchViewApp オブジェクトのエクスポートとインポート

InTouchViewApp オブジェクトをエクスポートできます。これは、たとえば、マネージド InTouch アプリケーションと、それをホストする他の Galaxy 内にある InTouchViewApp オブジェクトと一緒に使用するために行います。

オブジェクトをエクスポートすると、オブジェクト、関連付けられているマネージド InTouch アプリケーション、およびアプリケーションによって使用されているすべての産業用グラフィックに関する情報が含まれているパッケージファイル (.aaPKG) が作成されます。

InTouchViewApp オブジェクトをインポートすると、System Platform IDE によって、マネージド InTouch アプリケーションもインポートされます。

InTouchViewApp オブジェクトの属性

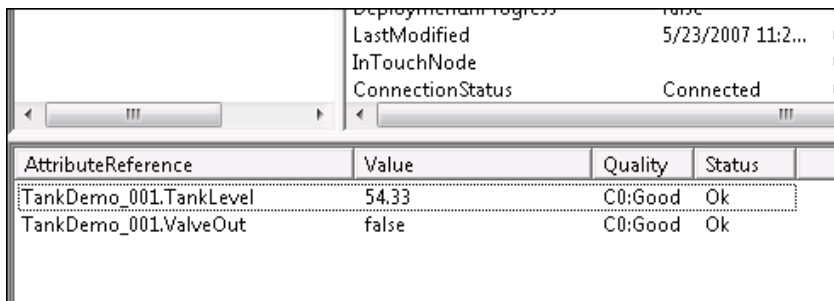
InTouchViewApp オブジェクトの Application Server 属性を使用して、関連付けられている InTouch アプリケーションのタグのランタイム データにアクセスできます。これは、Galaxy 名前空間で直接 InTouch デ

ータの読み取り、および書き込みを行ったり、InTouchProxy オブジェクトと同じ機能を提供する場合に便利です。

この例では、配置されたマネージド InTouch アプリケーションでは、タンクの充てん量レベルを報告するために実数型のタグ TankLevel が使用され、バルブの状態を制御するために論理型タグ ValveOut が使用されています。

InTouch タグを InTouchViewApp オブジェクト インスタンスから読み取りおよび書き込みを行うには

1. 配置された InTouchViewApp オブジェクトを右クリックし、次に **[モニタ]** をクリックします。
[オブジェクト ビューア] 画面が表示されます。
2. **[表示]** 領域を右クリックし、次に **[属性参照の追加]** をクリックします。
[属性参照の追加] ダイアログ ボックスが表示されます。
3. **[属性参照]** ボックスに、InTouchViewApp オブジェクトの名前、ドット、読み取りまたは書き込みを行う InTouch タグの名前の順で入力します。たとえば、TankDemo_001.TankLevel などのように入力します。
4. **[OK]** をクリックします。**[表示]** 領域に属性が追加されます。
5. 読み取りまたは書き込みを行うその他の InTouch タグに対しても、手順 2 から 4 を繰り返します。
6. InTouch タグの値が表示されます。



AttributeReference	Value	Quality	Status
TankDemo_001.TankLevel	54.33	C0:Good	Ok
TankDemo_001.ValveOut	false	C0:Good	Ok

7. InTouch タグの値を書き込むには、以下の手順を実行します。
 - a. タグをダブルクリックします。**[値の変更]** ダイアログ ボックスが表示されます。
 - b. 新しい値を入力し、**[OK]** をクリックします。実行中の InTouch アプリケーションのタグに値が書き戻されます。

InTouchViewApp オブジェクトと他のオートメーション オブジェクトとの違い

InTouchViewApp オブジェクトは、他のオートメーション オブジェクトとは異なります。他のオートメーション オブジェクトで通常行うことができるいくつかの操作を実行することができません。

- InTouchViewApp インスタンスを設定しようとする、親テンプレートを代わりに開くかどうかを確認するメッセージが表示されます。インスタンスは直接設定できません。親テンプレートのみ設定できます。
- 設定するために複数の InTouchViewApp テンプレートを 1 つのノードで一度に開こうとすると、IDE によって阻止されます。WindowMaker、WindowViewer、およびアプリケーション マネージャを終了して再試行してください。または、InTouch WindowMaker を使用して、異なるノード上にある InTouchViewApp オブジェクトを編集できます。

- WindowMaker を使用して InTouch アプリケーションを編集している間に IDE を終了した場合、変更を保存するよう要求するメッセージが WindowMaker に表示されます。終了後、InTouchViewApp テンプレートがチェック インされます。
- WindowViewer を使用して InTouch アプリケーションをテストしている間に IDE を終了した場合、WindowViewer が終了します。

以下のような場合、次の操作を実行します。

- InTouchViewApp と InTouch アプリケーション間での関連付けを変更するには、新しい派生した InTouchViewApp テンプレートを代わりに作成します。
- InTouch で ArchestrA セキュリティ (Galaxy セキュリティ) を使用するには、配置されたマネージド InTouch アプリケーションが実行されているノードに WinPlatform インスタンスを配置します。

ViewEngine オブジェクト

ViewEngine は、配置された InTouchViewApp オブジェクトをホストおよび実行する Application Server オブジェクトです。

InTouchViewApp インスタンスを対象プラットフォームに配置するには、まず ViewEngine オブジェクトに割り当てる必要があります。すると ViewEngine オブジェクトが対象 WinPlatform オブジェクトに割り当てられます。

AppEngine インスタンスがアプリケーション オブジェクトに対して行う場合と同様に、ViewEngine によって InTouchViewApp インスタンスと同じ機能が実行されます。ViewEngine によって、

- InTouchViewApp オブジェクトが最初に配置、起動されたときに、Galaxy 内の他のオブジェクトと通信できるようにするために、InTouchViewApp オブジェクトが設定および初期化されます。
- 監視、警告、履歴の記録が可能な、属性の診断が実行されます。
- Historian にデータの履歴が記録されます。

異なる ViewEngine オブジェクトを使用して、以下を行うことができます。

- 別の Historian へのデータの履歴の記録。
- 異なる走査速度での、配置された InTouch アプリケーションとの対話。これにより、InTouch タグ データが Galaxy 名前空間と対話できる頻度が設定されます。

プラットフォームは、複数の ViewEngine オブジェクトをホストできます。すべての InTouchViewApp を ViewEngine に割り当てる必要があります。

同じ ViewEngine オブジェクト下で実行する、同じ InTouchViewApp テンプレートの複数のインスタンスを作成することはできません。ただし、異なる ViewEngine オブジェクト下で、同じテンプレートの複数のインスタンスを実行することはできます。

章 25 InTouch コンポーネントのエクスポートとインポート

既存のアプリケーションのコンポーネントの一部またはすべてをインポートまたはエクスポートすることによって、より迅速に InTouch アプリケーションを構築できます。既存のアプリケーションから新しいアプリケーションにタグ変数定義、ウィンドウ、スクリプト、アプリケーションスタイルライブラリ、産業用グラフィック、クライアントコントロール、ローカリゼーション文字列、HTML5 ウィジェット、およびスクリプト関数ライブラリをインポートできます。タグ変数定義はアプリケーションマネージャからインポートおよびエクスポートできます。その他のコンポーネントは WindowMaker からインポートおよびエクスポートできます。

マネージド InTouch アプリケーションに関連付けられているタグ変数データのエクスポートとインポート

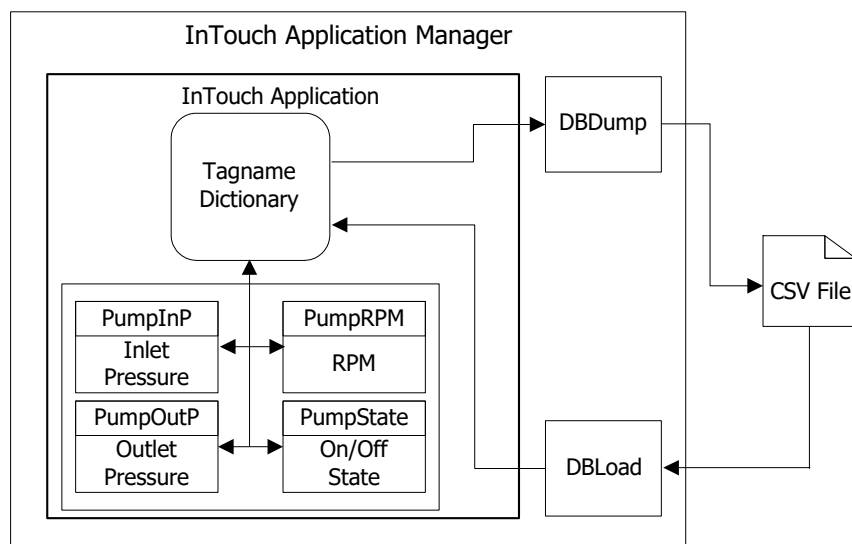
マネージド InTouch アプリケーションに関連付けられているタグ変数データを、.csv ファイルにエクスポートできます。これは、InTouch アプリケーションマネージャの DB ダンプ関数と同等です。

.csv ファイルからエクスポートされたタグ変数データは、DB ロード関数と同じ方法で、マネージド InTouch アプリケーションにインポートできます。

マネージド InTouch アプリケーションからエクスポートされた .csv ファイルと、スタンドアロン InTouch アプリケーションからエクスポートされた .csv ファイルはまったく同じです。

タグ定義のエクスポート

以下の図は、中間エクスポートファイルとアプリケーションのタグ名ディクショナリ間のタグ定義をエクスポートおよびインポートする手順を示しています。

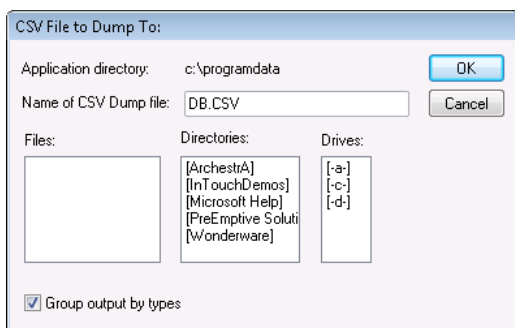


アプリケーションマネージャ内の DBDump ユーティリティを使用して、タグ名ディクショナリの内容をカンマ区切り値 (CSV) ファイルにエクスポートします。エクスポートされたファイルを Microsoft メモ帳または Microsoft Excel で表示したり、編集したりできます。編集を行った後で、アプリケーションマネージャユーティリティでもある DBLoad ユーティリティを使用して、タグ定義を InTouch アプリケーションにインポートします。

タグ定義をエクスポートする前に、InTouch HMI ソフトウェアの現在のバージョンにアプリケーションを変換する必要があります。

タグ定義をエクスポートするには

1. WindowMaker と WindowViewer を閉じます。
2. アプリケーション マネージャを起動します。[アプリケーション マネージャ] ダイアログ ボックスには、InTouch アプリケーションのリストが表示されます。
3. リストからアプリケーションを選択します。
4. [ファイル] メニューの [データ] グループで [DBDump] をクリックします。[ダンプ先 CSV ファイル名] ダイアログ ボックスが表示されます。



5. [ダンプ先 CSV ファイル名] ボックスに、.csv ファイル名の拡張子を持つファイルの名前を入力します。
6. エクスポート ファイルのデータ グループ化のタイプを選択します。
 - [タグ タイプ毎にグループ出力] チェック ボックスをオンにしてし、エクスポート ファイルのタグのタイプによってデータをグループ化します。これがデフォルトです。
 - [タグ タイプ毎にグループ出力] をクリアして、タグ名によって出力をアルファベット順にエクスポート ファイルに保存します。
7. [OK] をクリックして、タグ名ディクショナリの内容を選択したファイルに保存します。内容がファイルに正常に保存されたことを示すメッセージが表示されます。

エクスポートされたタグ定義の表示

Microsoft Excel を使用して、DBDump ユーティリティを使用して作成されたエクスポート ファイルを表示する場合、各データ レコードが個別のスプレッドシートセルに表示されます。

	A	B	C	D	E	F	G
1	modemask						
2	IOAccess	Application	Topic	AdviseActive	ODEProtocol	SecApplication	SecTopic
3	TankFarm1	VDec4TankFarm	x	Yes	No		
4	PLC1	VDec4TankFarm	x	Yes	No		
5	IOStatus	VDec4TankFarm	IOStatus	Yes	No		
6	Calvey	VDec4TankFarm	NA	Yes	MD		
7	TankFarm	VDec4TankFarm	x	Yes	No		
8	AlarmGroup	Group	Comment	EventLogged	EventLoggingPriority	LoAlarmDisable	LoAlarmDisable
9	Reactor	\$System		No	0	0	0
10	Conveyor	\$System		No	0	0	0
11	MemoryDisc	Group	Comment	Logged	EventLogged	EventLoggingPriority	RetentiveValue
12	Mixer	Reactor	Reactor mixer	No	No		
13	ConcPump	Reactor	Concentrate pump	No	No		
14	ConcValve	Reactor	Concentrate valve	No	No		
15	Auto	\$System	Automatic mode	Yes	No		
16	OutputValve	Reactor	Output valve	No	No		
17	TransferPump	Reactor	Transfer pump	No	No		
18	Factor	Connector	Barrel sector	No	No		

このファイルでは、キーワード、キーワードの属性が配置され、またタグ名ディクショナリからのデータがカラム順にキーワード属性の下に配置された状態で構成されています。

Excel スプレッドシートの例にある **:MemoryDisc** キーワードをご覧ください。このキーワードはタグ名ディクショナリからエクスポートされたメモリ論理型タグを識別します。同じスプレッドシート行で、メモリ論理型タグの属性は個別のスプレッドシート カラムに表示されています。たとえば、ログ記録された属性カラムでは、メモリ論理型タグのデータがログ記録されるかどうかが表示されています。

キーワードと属性の行のすぐ下には、エクスポートされたタグとそれに関連付けられたプロパティが配置されています。Excel スプレッドシートの例で、**OutputValve** はデータがログ記録されていないメモリ論理型タグです。

DBDump によって作成されたエクスポート ファイルは、.csv ファイル形式をサポートする任意のプログラムで表示したり、編集したりできます。一般的には、縦欄式スプレッドシート形式によって、タグデータを整理することが簡単であるため Excel が使用されます。ただし、ファイルの内容をネイティブのカンマ区切り文字列形式で表示したり、編集したりする場合は、Microsoft メモ帳を使用することもできます。

タグ定義のインポート

アプリケーション マネージャ内の DBLoad ユーティリティを使用すると、タグ定義の .csv ファイルをアプリケーションのタグ名ディクショナリにインポートできます。DBDump ユーティリティを使用して元々作成された定義ファイルをインポートできます。または、独自のインポートファイルを作成することもできます。

DBLoad ユーティリティを使用して、スーパータグ インスタンスを作成することもできます。詳細については、「[スーパータグ インスタンスの作成](#)」を参照してください。

タグ名ディクショナリ インポート ファイル形式

.csv ファイル形式をサポートする任意のプログラムを使用して、DBLoad インポート ファイルを手動で作成できます。Excel を使用してインポート ファイルを作成する場合、各エントリは個別のスプレッドシート セルに配置されます。この処理によって内容が読みやすくなり、エラーの可能性も少なくなります。

インポート ファイルの作成の詳細については、「[インポート ファイル テンプレートの作成](#)」を参照してください。

DBLoad インポート ファイルには、ファイル内のアクセス名、アラーム グループ、およびタグ変数データを整理する一組のキーワードが含まれています。

- すべてのキーワードの前には、コロン (:) が付きます。
- 行を続ける場合は、行末に円記号 (\) を入力します。
- コメントを入力するには、その前にセミコロン (;) を付けます。

以下の表には、DBLoad インポート ファイル内のキーワードが一覧表示されています。この表には、DBDump を使用してファイルを作成するときに指定した順序で、キーワードが一覧表示されています。ただし、ファイル内では任意の順序でキーワードを指定できます。

キーワード	説明
:mode	DBLoad ファイルの内容をアプリケーションのタグ名ディクショナリにインポートしているとき、重複したタグ レコードを処理する方法を指定します。

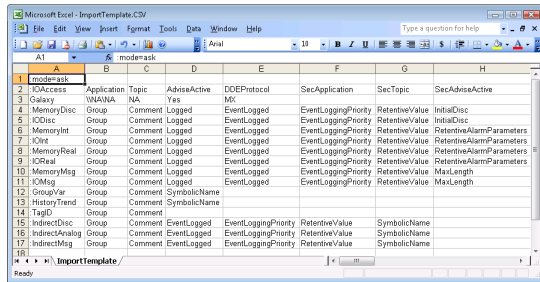
キーワード	説明
:IOAccess	InTouch アプリケーションに対して定義されたアクセス名
:AlarmGroup	InTouch アプリケーションに対して定義されたアラーム グループ
:MemoryDisc	メモリ論理型タグ
:IODisc	I/O 論理型タグ
:MemoryInt	メモリ整数型タグ
:IOInt	I/O 整数型タグ
:MemoryReal	メモリ実数型タグ
:IOReal	I/O 実数型タグ
:MemoryMsg	メモリ メッセージ型タグ
:IOMsg	I/O メッセージ型タグ
:GroupVar	グループ変数タグ変数
:HistoryTrend	履歴トレンド タグ変数
:TagID	タグ変数 ID タグ変数
:IndirectDisc	間接論理型タグ
:IndirectAnalog	間接アナログ型タグ
:IndirectMsg	間接メッセージ型タグ

各キーワードには、アクセス名、アラーム グループ、およびタグ変数のプロパティを指定する、関連付けられた一組の属性が含まれています。たとえば、**:IOAccess** キーワードには、すべての InTouch アクセス名のプロパティであるアプリケーション、トピック、および通信プロトコルを指定する属性が含まれています。

インポート ファイル テンプレートの作成

.csv ファイル形式をサポートする任意のアプリケーションを使用して、タグ名ディクショナリ インポート ファイルを手動で作成できます。ただし、インポート ファイル全体を作成することは時間のかかる作業であり、エラーが発生しやすくなります。テンプレートとして既存の .csv ファイルを使用する方が、処理が速く信頼性も高くなります。

次の図は、DBDump で作成されたテンプレート インポート ファイルを示しています。この図は、ウィンドウもタグも持たない InTouch アプリケーションから作成されたファイルを示しています。作成されたファイルには、必須キーワードとタグデータの無い属性のみが含まれます。



テンプレートを作成した後で、タグのタイプを識別するキーワードの下にタグデータを手動で追加します。タグタイプのキーワードに関連付けられた対応する属性カラムに、タグのプロパティを挿入します。

テンプレートインポートファイルを作成するには

1. アプリケーションマネージャを開きます。
2. 新しいInTouchアプリケーションを作成します。

アプリケーションを作成する手順の詳細については、「[InTouchアプリケーションの作成](#)」を参照してください。

3. アプリケーションマネージャに表示されるリストから、新しいアプリケーションを選択します。
4. DBDumpユーティリティを使用してアプリケーションのタグ名ディクショナリの内容をエクスポートします。

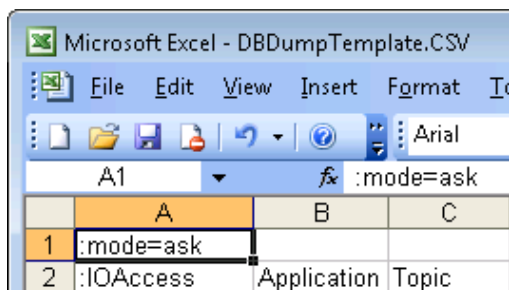
タグのエクスポートの詳細については、「[タグ定義のエクスポート](#)」を参照してください。

5. インポートするタグデータを挿入するファイルを編集します。

ディクショナリインポートファイルの操作モードの設定

インポートファイルからアプリケーションのタグ変数ディクショナリにデータをロードしている間に、重複したタグ変数レコードをDBLoadが処理する方法を指定する必要があります。

DBDumpを使用して作成されたインポートファイルテンプレートを使用する場合、ファイルの最初の実行には **:mode** キーワードが含まれます。たとえば、ExcelアプリケーションのセルA1で値askを **:mode** キーワードに割り当てることができます。



次の値を **:mode** キーワードに割り当てることができます。

```
:MODE=REPLACE
:MODE=UPDATE
:MODE=ASK
:MODE=IGNORE
:MODE=TERMINATE
```

```
:MODE=TEST
```

:MODE=REPLACE

重複したタグ変数が見つかった場合、DBLoad ユーティリティはタグ変数ディクショナリで既存のタグ変数を削除して、同じ名前を持つインポート ファイルのタグ変数で置き換えます。

:MODE=UPDATE

重複したタグ変数が見つかった場合、DBLoad ユーティリティはタグ変数ディクショナリの既存のタグ変数定義をインポート ファイルから明示的に指定されたデータのみで上書きします。タグ変数に関連付けられたその他のすべてのデータは、タグ変数ディクショナリでそのまま維持されます。

フィールドがレコード内にあり、ユーザーによって入力されたか、「:KEYWORD=value」メカニズムで設定されている場合、そのフィールドは明示的に定義されたものとみなされます。フィールドがレコードで指定されておらず、キーワードが「:KEYWORD=」コマンドを使用して再設定されている場合、現在のフィールド値は更新されません。

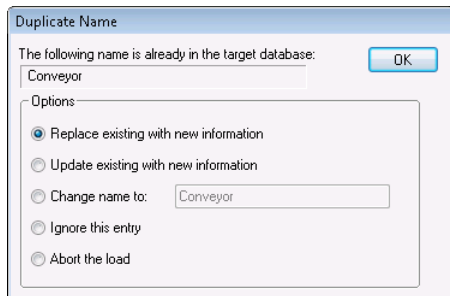
以下の例は、update モードのインポート ファイルがタグ変数ディクショナリにロード／マージされたときに何が起るかを示したものです。

```
:Mode=update
:Group=Group1
:IODisc,Group,DConversion
Tagname1,Group2,
; Tagname1 の Group のみ Group2 に更新します
Tagname2,,
; Tagname2 の Group を Group1 に更新し、DConversion はそのままにします
Tagname3,,Reverse
; Tagname3 の Group を Group1 に更新し、DConversion を "Reverse" に更新します
; 以下の行は Group フィールドをデフォルト値に再設定します
:Group=
; データ フィールド "Group" がデフォルト値に再設定されます
Tagname4,,
; Tagname4 はそのまま残ります
```

タグ変数を変更中で、そのタグ変数が使用中である場合、タグ変数タイプは互換性がある必要があります。たとえば、タグ変数がアプリケーションによって使用されている場合は、既存の履歴トレンドタグ変数を I/O 整数型に変更することはできません。また、タグ変数がアプリケーションの入力リンクに使用されている場合、このタグ変数を ReadOnly=yes に変更することはできません。これらの制限により、DBLoad ユーティリティを使用する前にインポート先アプリケーションで使用タグの検索をしてください。

:MODE=ASK

タグ変数ディクショナリをロードしている間に重複したタグ変数が見つかったとき、DBLoad が停止します。[同一名称タグ変数の処理] ダイアログボックスが表示され、重複したタグ変数进行处理するオプションのリストが表示されます。これは、デフォルトのインポート モードです。



重複を処理するオプション：

- 既存のタグ変数レコードをインポート ファイルのレコードに置き換えるには、[新しい情報で置き換える] をクリックします。
- 既存のタグ変数レコードをインポート ファイルで明示的に定義されたフィールドでのみ上書きするには、[新しい情報のみ書き込む] をクリックします。
- インポートされたタグ変数の名前を [同一名称タグ変数の処理] ダイアログ ボックスのボックスに入力した名前で置き換えるには、[名称を変更] をクリックします。
- タグ変数を見捨て、ファイルの内容のインポートを続行するには、[入力を見捨てる] をクリックします。
- インポート処理をキャンセルするには、[ロード処理を中止する] をクリックします。

:MODE=IGNORE

DBLoad インポート ユーティリティは重複したタグ変数を見捨て、インポート ファイルの残りのレコードの処理を続行します。

:MODE=TERMINATE

重複したタグ変数が見つかったとき、DBLoad インポート操作は停止します。

:MODE=TEST

DBLoad はインポート ファイルをスキャンしてエラーをチェックしますが、タグ変数定義をタグ変数ディクショナリにロードしません。DBLoad は、インポート ファイル内の行番号と場所の書式設定エラーを識別するレポートを生成します。

最初に、**:mode=test** を使用して DBLoad を実行し、インポート ファイルのエラーを識別します。エラーを修正した後、DBLoad を実行する前に、**mode** キーワード値を **:mode=replace** または **:mode=update** に変更します。

アクセス名とアラーム グループの設定

DBLoad インポート ファイルには、InTouch アプリケーションの定義済みアクセス名とアラーム グループを指定するキーワードが含まれています。

:IOAccess キーワード属性

:IOAccess キーワードは、InTouch アプリケーションで定義されたアクセス名を識別します。**:IOAccess** キーワードには、定義された InTouch アクセス名の特性を記述する一組の属性が含まれています。

以下の図は、アクセス名が **:IOAccess** キーワードを使用して Excel スプレッドシートで定義される方法を示しています。属性は個別のスプレッドシート カラムで左から右へ指定されます。

:IO Access		Topic Attribute			DDEProtocol		SecTopic Attribute		
	A	B	C	D	E	F	G	H	
1	:mode=ask								
2	:IOAccess	Application	Topic	AdviseActive	DDEProtocol	SecApplication	SecTopic	SecAdviseActive	
3	T7	View	T7	Yes	No				
4	M22	RSLINX	M22	Yes	Yes				
5	IOStatus	View	IOstatus	Yes	Yes				
6	Galaxy	WNA/NA	NA	Yes	MX				

Application Attribute Advise Active SecApplication

以下の表は、:IOAccess キーワードに関連付けられた属性のリストを示しています。この表では、DBDump ユーティリティを使用して作成されたテンプレート インポート ファイルを使用するときに指定された順序で属性を一覧表示しています。

文字列位置	属性	許可できる値	デフォルト値
1	Application	アクセス名に対して定義されたアプリケーション名	なし
2	Topic	アクセス名に対して定義されたトピック名	なし
3	AdviseActive	サーバーからポーリングする情報 No = 全てのアイテムを要求 Yes = 有効アイテムのみ要求	Yes
4	DDEProtocol	アクセス名に対して定義された通信プロトコル No = Suitelink Yes = DDE MX = Message Exchange	No
5	SecApplication	アクセス名のセカンダリ ソースに対して定義されたアプリケーション名	なし
6	SecTopic	アクセス名のセカンダリ ソースに対して定義されたトピック名	なし
7	SecAdviseActive	セカンダリ サーバーに保存された情報をポーリングするとき No = 全てのアイテムを要求 Yes = 有効アイテムのみ要求	なし

文字列位置	属性	許可できる値	デフォルト値
8	SecDDEProtocol	アクセス名のセカンダリ ソースに対して定義された通信プロトコル No = Suitelink Yes = DDE MX = Message Exchange	なし
9	FailoverExpression	TRUE の場合にアクセス名をセカンダリ ソースにを切り換えるフェイルオーバー式	なし
10	FailoverDeadband	アクセス名によって定義されたセカンダリ ソースに対してフェイルオーバーを開始する前の秒を表す整数	なし
11	DFOFlag	フェイルオーバー不可フラグ Yes = フェイルオーバー不可フラグの設定 No = フェイルオーバー不可フラグの設定なし	なし
12	FBDFlag	メインへ切り替えるフラグ Yes = フェイルオーバー条件の解除時にメインへ切り替える No = フェイルオーバー条件の解除時にメインへ切り替えない	なし
13	FailbackDeadband	フェイルオーバー条件の解除時にメインのアクセス名ソースに切り替える前の秒を表す整数	値なし

:AlarmGroup キーワード属性

DBLoad インポート ファイルには、InTouch アプリケーション用に定義されたアラーム グループを識別するキーワードが含まれています。:AlarmGroup キーワードには、InTouch アプリケーションのアラーム グループの特性を記述する一組の属性が含まれています。

以下の表は、:AccessGroup キーワードに関連付けられた属性のリストを示しています。この表では、DBDump ユーティリティを使用して作成されたテンプレート インポート ファイルを使用するときに指定された順序で属性を一覧表示しています。

文字列位置	属性	許可できる値	デフォルト値
1	Group	アラーム グループの名前	\$System

文字列位置	属性	許可できる値	デフォルト値
2	Comment	アラーム グループに割り当てられたコメント 任意のテキスト文字列	なし
3	EventLogged	イベント ログ記録の有効化または無効化 Yes または On = イベントログ記録が有効 No または Off = イベント ログ記録が無効	No
4	EventLoggingPriority	イベントに割り当てられた優先度 1 ～ 999、0 = ログ記録なし	0
5	LoLoAlarmDisable	LoLo アラームの無効化または有効化 0 = LoLo アラームが有効 1 = LoLo アラームが無効	0
6	LoAlarmDisable	Lo アラームの無効化または有効化 0 = Lo アラームが有効 1 = Lo アラームが無効	0
7	HiAlarmDisable	Hi アラームの無効化または有効化 0 = Hi アラームが有効 1 = Hi アラームが無効	0
8	HiHiAlarmDisable	HiHi アラームの無効化または有効化 0 = HiHi アラームが有効 1 = HiHi アラームが無効	0
9	MinDevAlarmDisable	小偏差アラームの無効化または有効化 0 = 小偏差アラームが有効 1 = 小偏差アラームが無効	0

文字列位置	属性	許可できる値	デフォルト値
10	MajDevAlarmDisable	大偏差アラームの無効化または有効化 0 = 大偏差アラームが有効 1 = 大偏差アラームが無効	0
11	RocAlarmDisable	変化率（ROC）アラームの無効化または有効化 0 = ROC アラームが有効 1 = ROC アラームが無効	0
12	DSCAlarmDisable	論理型アラームの無効化または有効化 0 = 論理型アラームが有効 1 = 論理型アラームが無効	0
13	LoLoAlarmInhibitor	LoLo アラームの抑止に使用されるタグ変数の名前 タグ リファレンス：任意の論理型またはアナログ型タグ変数	なし
14	LoAlarmInhibitor	Lo アラームの抑止に使用されるタグ変数の名前 タグ リファレンス：任意の論理型またはアナログ型タグ変数	なし
15	HiAlarmInhibitor	Hi アラームの抑止に使用されるタグ変数の名前 タグ リファレンス：任意の論理型またはアナログ型タグ変数	なし
16	HiHiAlarmInhibitor	HiHi アラームの抑止に使用されるタグ変数の名前 タグ リファレンス：任意の論理型またはアナログ型タグ変数	なし
17	MinDevAlarmInhibitor	小偏差アラームの抑止に使用されるタグ変数の名前 タグ リファレンス：任意の論理型またはアナログ型タグ変数	なし

文字列位置	属性	許可できる値	デフォルト値
18	MajDevAlarmInhibitor	大偏差アラームの抑止に使用されるタグ変数の名前 タグ リファレンス：任意の論理型 またはアナログ型タグ変数	なし
19	RocAlarmInhibitor	変更率アラームの抑止に使用されるタグ変数の名前 タグ リファレンス：任意の論理型 またはアナログ型タグ変数	なし
20	DSCAlarmInhibitor	論理型アラームの抑止に使用されるタグ変数の名前 タグ リファレンス：任意の論理型 またはアナログ型タグ変数	なし

タグ変数タイプのキーワードと属性の定義

タグ変数レコードは、タグ変数のタイプを識別するキーワード行で開始します。各タグ変数キーワードには、タグ変数のタイプに関連付けられたデータの特性を指定するための属性の固有セットが含まれています。

以下の例では、**:IODisc** キーワードが I/O 論理型タグ変数タイプを識別します。キーワード行の残りの値は、I/O 論理型タグ変数に関連付けられたデータの属性を識別します。この例では、メモ帳を使用してネイティブのカンマ区切り文字列形式でファイルの内容が示されています。

```
:IODisc,Group,Comment,Logged,EventLogged,EventLoggingPriority,RetentiveValue,InitialDis,OffMsg,OnMsg,AlarmState,AlarmPri,DConversion,AccessName,ItemUseTagname,ItemName,ReadOnly,AlarmComment,AlarmAckModel,DSCAlarmDisable,DSCAlarmInhibitor,SymbolicName
```

タグ変数タイプのキーワード行の下では、個別の行がそのタイプのタグ変数を一組の属性値で指定しています。以下の例では、**HDWStatus** タグ変数はインポート ファイルの I/O 論理型タグ変数タイプに属しています。

```
"HDWStatus","$System","",No,No,0,No,Off","",",,1,Direct,"HistdataViewstr",No,"Status",No,"",0,0,"",""
```

このレコードでは、引用符を使用して空白文字列を識別しています。

以下の図は、Excel スプレッドシートの同じインポート ファイルデータを示しています。**Comment** セルは、タグ変数コメントがインポート ファイルで指定されていないため空白です。

The screenshot shows a Microsoft Excel spreadsheet titled 'tagtest.csv'. The data is imported into a table with columns A through H. Row 1 contains the following values: A1: HDWStatus, B: \$System, C: (blank), D: No, E: No, F: 0, G: No, H: Off.

	A	B	C	D	E	F	G	H
1	HDWStatus	\$System		No	No	0	No	Off

タグ変数キーワード属性

以下の表は、InTouch タグ変数キーワードに関連付けられたすべての属性を一覧表示しています。この表には、各タグ変数属性とそのデフォルト値に関連付けられたデータのタイプを記述するカラムが含まれています。

これらのタグ変数属性は、付随するタグ変数データが対応する属性に一致する限り、DBLoad インポートファイルに任意の順序で指定できます。たとえば、Excel インポート ファイルに :IODisc キーワードを挿入する場合、すべての I/O 論理型タグ変数の工学単位を EngUnits 属性と同じ Excel カラムに配置する必要があります。

属性	許可できる値	デフォルト値
AccessName	タグ変数に割り当てられた InTouch アクセス名	なし
AlarmAckModel	アラーム確認モデル 整数 0 = 条件 1 = イベント指向 2 = 拡張サマリ	0
AlarmComment	タグ変数に割り当てられたアラーム コメント テキスト文字列	なし
AlarmDevDeadband	タグ変数偏差アラーム デッドバンド 実数	なし
AlarmPri	タグ変数に割り当てられたアラーム優先度 1 ~ 999	1
AlarmState	タグ変数のアラーム状況 オン (On)、オフ (Off)、なし (None)	なし
AlarmValueDeadband	タグ変数アラーム デッドバンド 実数	0
Comment	タグ変数に割り当てられたコメント テキスト文字列	なし
Conversion	タグ変数値変換 線形 (Linear) または二乗根 (Square Root)	Linear
Deadband	タグ変数に割り当てられた値のデッドバンド 実数	0

属性	許可できる値	デフォルト値
DevTarget	タグ変数の偏差目標値 実数	0
DSCAlarmDisable	論理型アラームの無効化または有効化 0 = 論理型アラームが有効 1 = 論理型アラームが無効	0
DSCAlarmInhibitor	論理型アラームの抑止に使用されるタグ変数の名前	なし
EngUnits	タグ変数に割り当てられた工学単位 テキスト文字列	なし
EventLogged	イベント ログ記録の有効化または無効化 Yes または On = イベントログ記録が有効 No または Off = イベント ログ記録が無効	No
EventLogging	タグ変数イベントログ記録の有効化または無効化 No または Off = ログ記録が無効 Yes または On = ログ記録が有効	No
EventLoggingPriority	イベントに割り当てられた優先度 1 ～ 999、0 = ログ記録なし	0
Group	タグ変数が属すアラーム グループの名前	\$System
HiAlarmDisable	Hi アラームの無効化または有効化 0 = Hi アラームが有効 1 = Hi アラームが無効	0
HiAlarmInhibitor	Hi アラームの抑止に使用されるタグ変数の名前 任意の論理型またはアナログ型タグ変数	なし
HiAlarmPri	Hi アラームに割り当てられた優先度 1 ～ 999	1
HiAlarmState	Hi アラームの有効化または無効化 No または Off = 無効 Yes または On = 有効	No

属性	許可できる値	デフォルト値
HiAlarmValue	タグ変数に割り当てられた Hi アラーム ポイント 実数	0
HiHiAlarmDisable	HiHi アラームの無効化または有効化 0 = HiHi アラームが有効 1 = HiHi アラームが無効	0
HiHiAlarmInhibitor	HiHi アラームの抑止に使用されるタグ変数の名前 任意の論理型またはアナログ型タグ変数	なし
HiHiAlarmPri	HiHi アラームに割り当てられた優先度 1 ～ 999	1
HiHiAlarmState	HiHi アラームの有効化または無効化 No または Off = 無効 Yes または On = 有効	No
HiHiAlarmValue	タグ変数に割り当てられた HiHi アラーム ポイント 実数	0
InitialDisc	論理型タグ変数に割り当てられた初期値 0、Off、False、または No = オフ 1、On、True、または Yes = オン	0
InitialMessage	初期タグ変数メッセージ テキスト文字列	なし
InitialValue	タグ変数に割り当てられた初期値 実数	0
ItemName	タグ変数に割り当てられたアイテムの名前 テキスト文字列	なし
ItemUseTagname	[タグ変数名をアイテム名として使用] ションの有効化または無効化 No または False = 無効 Yes または True = 有効	No

属性	許可できる値	デフォルト値
LoAlarmDisable	Lo アラームの無効化または有効化 0 = Lo アラームが有効 1 = Lo アラームが無効	0
LoAlarmInhibitor	Lo アラームの抑止に使用されるタグ変数の名前 任意の論理型またはアナログ型タグ変数	なし
LoAlarmPri	Lo アラームに割り当てられた優先度 1 ~ 999	1
LoAlarmState	Lo アラームの有効化または無効化 No または Off = 無効 Yes または On = 有効	No
LoAlarmValue	タグ変数に割り当てられた Lo アラーム ポイント 実数	0
LogDeadband	タグ変数に割り当てられたログ記録のデッドバンド 実数	0
Logged	タグ変数値のログ記録の有効化または無効化 No または Off = ログ記録が無効 Yes または On = ログ記録が有効	No
LoLoAlarmDisable	LoLo アラームの無効化または有効化 0 = LoLo アラームが有効 1 = LoLo アラームが無効	0
LoLoAlarmInhibitor	LoLo アラームの抑止に使用されるタグ変数の名前 任意の論理型またはアナログ型タグ変数	なし
LoLoAlarmPri	LoLo アラームに割り当てられた優先度 1 ~ 999	1
LoLoAlarmState	LoLo アラームの有効化または無効化 No または Off = 無効 Yes または On = 有効	No

属性	許可できる値	デフォルト値
LoLoAlarmValue	タグ変数に割り当てられた LoLo アラームポイント 実数	0
MajDevAlarmDisable	大偏差アラームの無効化または有効化 0 = 大偏差アラームが有効 1 = 大偏差アラームが無効	0
MajDevAlarmInhibitor	大偏差アラームの抑止に使用されるタグ変数の名前 任意の論理型またはアナログ型タグ変数	なし
MajorDevAlarmPri	大偏差アラームに割り当てられた優先度 1 ~ 999	1
MajorDevAlarmState	大偏差アラームの無効化または有効化 No または Off = 無効 Yes または On = 有効	No
MajorDevAlarmValue	タグ変数に割り当てられた大偏差アラームのパーセンテージ 実数	0
MaxEU	タグ変数に割り当てられた最大工学単位値 実数	32767
MaxLength	メッセージの最大文字数 実数	131
MaxRaw	タグ変数に割り当てられた生データ最大値 実数	32767
MaxValue	タグ変数に割り当てられた最大値 実数	32767
MinDevAlarmDisable	小偏差アラームの無効化または有効化 0 = 小偏差アラームが有効 1 = 小偏差アラームが無効	0

属性	許可できる値	デフォルト値
MinDevAlarmInhibitor	小偏差アラームの抑止に使用されるタグ変数の名前 任意の論理型またはアナログ型タグ変数	なし
MinEU	タグ変数に割り当てられた最小工学単位値 実数	-32768
MinorDevAlarmPri	小偏差アラームに割り当てられた優先度 1 ～ 999	1
MinorDevAlarmState	小偏差アラームの有効化または無効化 No または Off = 無効 Yes または On = 有効	No
MinorDevAlarmValue	タグ変数に割り当てられた小偏差アラームのパーセンテージ 実数	0
MinRaw	タグ変数に割り当てられた生データ最小値 実数	-32768
MinValue	タグ変数に割り当てられた最小値 実数	-32768
OffMsg	論理型タグ変数オフ メッセージ テキスト文字列	なし
OnMsg	論理型タグ変数オン メッセージ テキスト文字列	なし
ReadOnly	タグ変数値読み取り専用または読み取り／書き込み Yes = 読み取り専用 No = 読み取り／書き込み	No
RetentiveAlarmParameters	タグ変数のパラメータ値保持の有効化または無効化 No または Off = 無効 Yes または On = 有効	No

属性	許可できる値	デフォルト値
RetentiveValue	タグ変数の変数値保持の有効化または無効化 0、Off、False、または No = 無効 1、On、True、または Yes = 有効	No
RocAlarmDisable	変化率（ROC）アラームの無効化または有効化 0 = ROC アラームが有効 1 = ROC アラームが無効	0
RocAlarmInhibitor	変更率アラームの抑止に使用されるタグ変数の名前 任意の論理型またはアナログ型タグ変数	なし
ROCArmPri	変化率アラームに割り当てられた優先度 1 ～ 999	1
ROCArmState	変化率アラームの有効化または無効化 No または Off = 無効 Yes または On = 有効	No
ROCArmValue	パーセントによるタグ変数値の変更 実数	0
ROCTimeBase	変化率を計算する測定期間 秒（Sec）、分（Min）、時（Hr）	Min
SymbolicName	S7 Tag Creator 製品によって入力データブロックに割り当てられたシンボリック名。シンボリック名は S7 Tag Creator のシンボル表にリストされています。	なし

:MemoryDisc キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできるメモリ論理型タグ変数を定義するための :MemoryDisc キーワードが含まれています。以下の表は、メモリ論理型タグ変数のプロパティに関連付けられた :MemoryDisc キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために DBDump が使用されるときに、:MemoryDisc キーワード属性が指定される順序を示しています。属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	Logged
4	EventLogged
5	EventLoggingPriority
6	RetentiveValue
7	InitialDisc
8	OffMsg
9	OnMsg
10	AlarmState
11	AlarmPri
12	AlarmComment
13	AlarmAckModel
14	DSCAlarmDisable
15	DSCAlarmInhibitor
16	SymbolicName

:IODisc キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできる I/O 論理型タグ変数を定義するための :IODisc キーワードが含まれています。以下の表は、I/O 論理型タグ変数のプロパティに関連付けられた :IODisc キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために DBDump が使用されるときに、:IODisc キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	Logged
4	EventLogged
5	EventLoggingPriority
6	RetentiveValue

文字列位置	属性
7	InitialDisc
8	OffMsg
9	OnMsg
10	AlarmState
11	AlarmPri
12	Conversion
13	AccessName
14	ItemUseTagname
15	ItemName
16	ReadOnly
17	AlarmComment
18	AlarmAckModel
19	DSCAlarmDisable
20	DSCAlarmInhibitor
21	SymbolicName

:MemoryInt キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできるメモリ整数型タグ変数を定義するための :MemoryInt キーワードが含まれています。以下の表は、メモリ整数型タグ変数のプロパティに関連付けられた :MemoryInt キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために DBDump ユーティリティが使用されるときに、:MemoryInt キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	Logged
4	EventLogged
5	EventLoggingPriority
6	RetentiveValue
7	RetentiveAlarmParameters

文字列位置	属性
8	AlarmValueDeadband
9	AlarmDevDeadband
10	EngUnits
11	InitialValue
12	MinValue
13	MaxValue
14	Deadband
15	LogDeadband
16	LoLoAlarmState
17	LoLoAlarmValue
18	LoLoAlarmPri
19	LoAlarmState
20	LoAlarmValue
21	LoAlarmPri
22	HiAlarmState
23	HiAlarmValue
24	HiAlarmPri
25	HiHiAlarmState
26	HiHiAlarmValue
27	HiHiAlarmPri
28	MinorDevAlarmState
29	MinorDevAlarmValue
30	MinorDevAlarmPri
31	MajorDevAlarmState
32	MajorDevAlarmValue
33	MajorDevAlarmPri
34	DevTarget
35	ROCArmState
36	ROCArmValue

文字列位置	属性
37	ROCArmPri
38	ROTimeBase
39	AlarmComment
40	AlarmAckModel
41	LoLoAlarmDisable
42	LoAlarmDisable
43	HiAlarmDisable
44	HiHiAlarmDisable
45	MinDevAlarmDisable
46	MajDevAlarmDisable
47	RocAlarmDisable
48	LoLoAlarmInhibitor
49	LoAlarmInhibitor
50	HiAlarmInhibitor
51	HiHiAlarmInhibitor
52	MinDevAlarmInhibitor
53	MajDevAlarmInhibitor
54	RocAlarmInhibitor
55	SymbolicName

:IOInt キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできる I/O 整数型タグ変数を定義するための :IOInt キーワードが含まれています。以下の表は、I/O 整数型タグ変数のプロパティに関連付けられた :IOInt キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために DBDump が使用されるときに、:IOInt キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	Logged

文字列位置	属性
4	EventLogged
5	EventLoggingPriority
6	RetentiveValue
7	RetentiveAlarmParameters
8	AlarmValueDeadband
9	AlarmDevDeadband
10	EngUnits
11	InitialValue
12	MinEU
13	MaxEU
14	Deadband
15	LogDeadband
16	LoLoAlarmState
17	LoLoAlarmValue
18	LoLoAlarmPri
19	LoAlarmState
20	LoAlarmValue
21	LoAlarmPri
22	HiAlarmState
23	HiAlarmValue
24	HiAlarmPri
25	HiHiAlarmState
26	HiHiAlarmValue
27	HiHiAlarmPri
28	MinorDevAlarmState
29	MinorDevAlarmValue
30	MinorDevAlarmPri
31	MajorDevAlarmState
32	MajorDevAlarmValue

文字列位置	属性
33	MajorDevAlarmPri
34	DevTarget
35	ROCArmState
36	ROCArmValue
37	ROCArmPri
38	ROCTimeBase
39	AlarmComment
39	MinRaw
40	MaxRaw
41	Conversion
42	AccessName
43	ItemUseTagname
44	ItemName
45	ReadOnly
46	AlarmComment
47	AlarmAckModel
48	LoLoAlarmDisable
49	LoAlarmDisable
50	HiAlarmDisable
51	HiHiAlarmDisable
52	MinDevAlarmDisable
53	MajDevAlarmDisable
54	RocAlarmDisable
55	LoLoAlarmInhibitor
56	LoAlarmInhibitor
57	HiAlarmInhibitor
58	HiHiAlarmInhibitor
59	MinDevAlarmInhibitor
60	MajDevAlarmInhibitor

文字列位置	属性
61	RocAlarmInhibitor
62	SymbolicName

:MemoryReal キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできるメモリ実数型タグ変数を定義するための **:MemoryReal** キーワードが含まれています。以下の表は、メモリ実数型タグ変数のプロパティに関連付けられた **:MemoryReal** キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために **DBDump** が使用されるときに、**:MemoryReal** キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	Logged
4	EventLogged
5	EventLoggingPriority
6	RetentiveValue
7	RetentiveAlarmParameters
8	AlarmValueDeadband
9	AlarmDevDeadband
10	EngUnits
11	InitialValue
12	MinValue
13	MaxValue
14	Deadband
15	LogDeadband
16	LoLoAlarmState
17	LoLoAlarmValue
18	LoLoAlarmPri
19	LoAlarmState
20	LoAlarmValue

文字列位置	属性
21	LoAlarmPri
22	HiAlarmState
23	HiAlarmValue
24	HiAlarmPri
25	HiHiAlarmState
26	HiHiAlarmValue
27	HiHiAlarmPri
28	MinorDevAlarmState
29	MinorDevAlarmValue
30	MinorDevAlarmPri
31	MajorDevAlarmState
32	MajorDevAlarmValue
33	MajorDevAlarmPri
34	DevTarget
35	ROCArmState
36	ROCArmValue
37	ROCArmPri
38	ROCTimeBase
39	AlarmComment
40	AlarmAckModel
41	LoLoAlarmDisable
42	LoAlarmDisable
43	HiAlarmDisable
44	HiHiAlarmDisable
45	MinDevAlarmDisable
46	MajDevAlarmDisable
47	RocAlarmDisable
48	LoLoAlarmInhibitor
49	LoAlarmInhibitor

文字列位置	属性
50	HiAlarmInhibitor
51	HiHiAlarmInhibitor
52	MinDevAlarmInhibitor
53	MajDevAlarmInhibitor
54	RocAlarmInhibitor
55	SymbolicName

:IOReal キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできる I/O 実数型タグ変数を定義するための :IOReal キーワードが含まれています。以下の表は、I/O 実数型タグ変数のプロパティに関連付けられた :IOReal キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために DBDump が使用されるときに、:IOReal キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	Logged
4	EventLogged
5	EventLoggingPriority
6	RetentiveValue
7	RetentiveAlarmParameters
8	AlarmValueDeadband
9	AlarmDevDeadband
10	EngUnits
11	InitialValue
12	MinEU
13	MaxEU
14	Deadband
15	LogDeadband
16	LoLoAlarmState

文字列位置	属性
17	LoLoAlarmValue
18	LoLoAlarmPri
19	LoAlarmState
20	LoAlarmValue
21	LoAlarmPri
22	HiAlarmState
23	HiAlarmValue
24	HiAlarmPri
25	HiHiAlarmState
26	HiHiAlarmValue
27	HiHiAlarmPri
28	MinorDevAlarmState
29	MinorDevAlarmValue
30	MinorDevAlarmPri
31	MajorDevAlarmState
32	MajorDevAlarmValue
33	MajorDevAlarmPri
34	DevTarget
35	ROCArmState
36	ROCArmValue
37	ROCArmPri
38	ROCTimeBase
39	MinRaw
40	MaxRaw
41	Conversion
42	AccessName
43	ItemUseTagname
44	ItemName
45	ReadOnly

文字列位置	属性
46	AlarmComment
47	AlarmAckModel
48	LoLoAlarmDisable
49	LoAlarmDisable
50	HiAlarmDisable
51	HiHiAlarmDisable
52	MinDevAlarmDisable
53	MajDevAlarmDisable
54	RocAlarmDisable
55	LoLoAlarmInhibitor
56	LoAlarmInhibitor
57	HiAlarmInhibitor
58	HiHiAlarmInhibitor
59	MinDevAlarmInhibitor
60	MajDevAlarmInhibitor
61	RocAlarmInhibitor
62	SymbolicName

:MemoryMsg キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできるメモリ メッセージ型タグ変数を定義するための **:MemoryMsg** キーワードが含まれています。以下の表は、メモリメッセージ型タグ変数のプロパティに関連付けられた **:MemoryMsg** キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために **DBDump** が使用されるときに、**:MemoryMsg** キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	Logged
4	EventLogged
5	EventLoggingPriority

文字列位置	属性
6	RetentiveValue
7	MaxLength
8	InitialMessage
9	AlarmComment
10	SymbolicName

:IOMsg キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできる I/O メッセージ型タグ変数を定義するための :IOMsg キーワードが含まれています。以下の表は、I/O メッセージ型タグ変数のプロパティに関連付けられた :IOMsg キーワードの属性を一覧表示しています。

この表には、インポート ファイルを作成するために DBDump が使用されるときに、:IOMsg キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	Logged
4	EventLogged
5	EventLoggingPriority
6	RetentiveValue
7	MaxLength
8	InitialMessage
9	AccessName
10	ItemUseTagname
11	ItemName
12	ReadOnly
13	AlarmComment
14	SymbolicName

:GroupVar キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできるグループ変数タグ変数を定義するための :GroupVar キーワードが含まれています。以下の表は、グループ変数タグ変数のプロパティに関連付けられた :GroupVar キーワードの属性を一覧表示しています。

注意：InTouch グループ変数タグ変数は現在使用されていません。:GroupVar キーワードは、従来のアプリケーションのみをサポートするために含まれています。

この表は、インポート ファイルを作成するために DBDump が使用されるときに、:GroupVar キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	SymbolicName

:HistoryTrend キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできる履歴トレンドタグ変数を定義するための :HistoryTrend キーワードが含まれています。以下の表は、履歴トレンドタグ変数のプロパティに関連付けられた :HistoryTrend キーワードの属性を一覧表示しています。

この表には、インポート ファイルを作成するために DBDump が使用されるときに、:HistoryTrend キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	SymbolicName

:TagID キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできるタグ変数 ID タグ変数を定義するための :TagID キーワードが含まれています。以下の表は、タグ変数 ID タグ変数のプロパティに関連付けられた :TagID キーワードの属性を一覧表示しています。

この表には、インポート ファイルを作成するために DBDump が使用されるときに、:TagID キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment

:IndirectDisc キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできる間接論理型タグ変数を定義するための :IndirectDisc キーワードが含まれています。以下の表は、間接論理型タグ変数のプロパティに関連付けられた :IndirectDisc キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために DBDump が使用されるときに、:IndirectDisc キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	EventLogging
4	EventLoggingPriority
5	RetentiveValue
6	SymbolicName

:IndirectAnalog キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできる間接アナログ型タグ変数を定義するための :IndirectAnalog キーワードが含まれています。以下の表は、間接アナログ型タグ変数のプロパティに関連付けられた :IndirectAnalog キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために DBDump が使用されるときに、:IndirectAnalog キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment
3	EventLogging
4	EventLoggingPriority
5	RetentiveValue
6	SymbolicName

:IndirectMsg キーワード属性

DBLoad インポート ファイルには、タグ変数ディクショナリにインポートできる間接メッセージ型タグ変数を定義するための :IndirectMsg キーワードが含まれています。以下の表は、間接メッセージ型タグ変数のプロパティに関連付けられた :IndirectMsg キーワードの属性を一覧表示しています。

この表は、インポート ファイルを作成するために DBDump が使用されるときに、:IndirectMsg キーワード属性が指定される順序を示しています。これらの属性とそのデフォルト値に関連付けられているデータについては、「[タグ変数キーワード属性](#)」を参照してください。

文字列位置	属性
1	Group
2	Comment

文字列位置	属性
3	EventLogging
4	EventLoggingPriority
5	RetentiveValue
6	SymbolicName

インポート ファイルでの空白文字列の使用

ディクショナリ インポート ファイルの場合、空白文字列を含むフィールドとデータを含まないフィールドの間には違いがあります。空白文字列を割り当てられるキーワード属性：

Comment	Eng Units	OffMsg
InitialMessage	OnMsg	Application
ItemName	Topic	

以下の例で、空白文字列は引用符 (“ ”) によって示されています。

```
:Comment="Hi"  
:MemoryDisc,Comment,Group  
Tagname1,, $System  
Tagname2,"", $System
```

ここで、

Tagname1 の Comment フィールドの値は Hi で、Tagname2 の Comment フィールドの値は空白コメントです。

Microsoft Excel はファイルを保存するとき空白文字列を示す引用符を無視し、以下の結果となります。

```
:Comment="Hi"  
:MemoryDisc,Comment,Group  
Tagname1,, $System  
Tagname2,, $System
```

空白文字列が Excel で使用されていることを確認するには、属性値としてセルにスペースを入力します。

フィールドに対するデフォルト値の使用

キーワードを使用して、レコードの特定フィールドに対するデフォルト値を設定できます。デフォルト値は、タグ変数タイプの元々の InTouch の値です。たとえば、メモリ論理型タグ変数はデフォルト値として Group=\$System、EventLogging=Off、InitialValue=Off を使用します。

次に例を示します。

```
:KEYWORD=value
```

これによって、後続のデータ レコードすべてに対して参照されるフィールドのデフォルト値が設定されます。この機能は、多数のレコードに対して変化しないまま維持されるべきであるフィールドのデフォルト値を設定するために使用します。フィールドに定義されたデフォルト値が存在する場合、その値のレコードにデータが存在しない場合にデフォルト値が使用されます。

たとえば、**:GROUP=Reactor_Site**を設定すると、**GROUP** カラムに空白エントリを持つすべてのタグ変数が **Reactor_Site** アラーム グループに割り当てられます。たとえば、タグ変数の **GROUP** カラムに **\$System** が入力されている場合、このタグ変数はアラーム グループ **\$System** に割り当てられます。

方程式の値を省略すると、個々のキーワードを元々のデフォルト値にリセットできます。たとえば、**:GROUP=** などです。

すべてのキーワードをリセットするには、**:RESET** コマンドを使用します。このコマンドは引数を持たず、コマンド後に発生するファイルのすべてのエントリに影響を与えます。

スーパータグ インスタンスの作成

アプリケーション マネージャ内の **DBLoad** ユーティリティを使用してスーパータグを作成できます。ただし、作成するスーパータグ インスタンスはスーパータグ テンプレート定義には反映されません。

有効なスーパー タグ形式を使用する必要があります、スーパータグ インスタンスのデータ レコードはタグタイプの有効なキーワードで開始する必要があります。次に例を示します。

	A	B	C	D
1	MemoryDisc	Group	Comment	Logged
2	Turkey\EvapUnit1\FanMotor2	\$System	AccessLevel	No
3	MemoryInt	Group	Comment	Logged
4	CurrentWindow	\$System	Comment	Yes
5	MemoryReal	Group	Comment	Logged
6	Chicken01\EvapUnit1\CoilTemp	\$System	Comment	No
7	MemoryMag	Group	Comment	Logged
8	Chicken01\EvapUnit1\EvapStatus	\$System	Comment	No
9	IOInt	Group	Comment	Logged
10	d000	\$System	Comment	Yes
11	di111	\$System	Comment	No

次の例は有効な構文を示しています。

ParentInstance\ChildMember
ParentInstance\ChildMember\Submember

次の例は無効な構文を示しています。

ParentInstance
ParentInstance\ChildMember

無効な形式を使用すると、エラー メッセージが表示されます。

スーパータグ インスタンスを含む **CSV** ファイルをインポートすると、これらのインスタンスは自動的にタグ名ディクショナリに追加され、アニメーション リンクや **InTouch QuickScript** ですぐに使用できるようになります。

DBLoad を使用したタグ定義のインポート

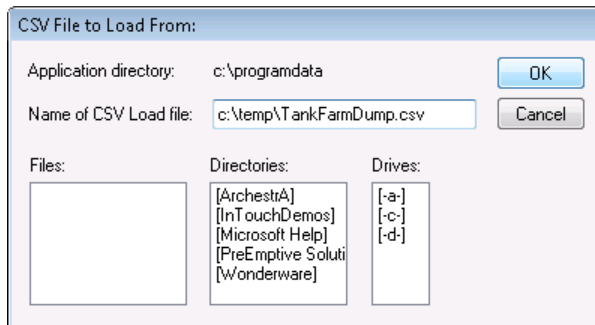
ファイルの内容を **DBLoad** を使用してインポートするとき、すべてのタグ定義は選択した **InTouch** アプリケーションのタグ名ディクショナリにインポートされます。

インポートが失敗すると、失敗の理由を説明するメッセージが表示されます。このエラー メッセージは **Logger** に記述されます。

タグ定義を InTouch アプリケーションにインポートするには

1. **WindowMaker** と **WindowViewer** を閉じます。
2. タグ名ディクショナリがインポート ファイルからのタグ定義でロードされるアプリケーションをバックアップします。

3. アプリケーション マネージャ を起動します。
4. タグ名ディクショナリがインポートされたタグ定義を受け取るリストからアプリケーションを選択します。
5. [ファイル] メニューの [データ] グループで [DBLoad] をクリックします。
InTouch アプリケーションのバックアップを確認するよう要求するメッセージが表示されます。
6. [はい] をクリックして、アプリケーションがバックアップされたことを確認します。
[ロード元 CSV ファイル名] ダイアログ ボックスが表示されます。



7. [CSV ロード ファイル名] ボックスで、インポートするファイルを探して、選択します。
8. [OK] をクリックします。

次の手順は、DBLoad が新しいタグ定義または既存のタグ定義のどちらをタグ名ディクショナリにインポートするかに基づいて異なります。

- 新しいタグ定義をインポートする場合、新しいタグデータがアプリケーションのタグ名ディクショナリにロードされます。データが正常にロードされマージされたことを確認するメッセージが表示されます。
- 既存のタグ定義をインポートする場合、:mode キーワードが :mode=ask に設定され、インポートファイルに重複したタグが含まれる場合、インポートは停止します。重複したタグを処理するオプションが表示されます。またはインポートをキャンセルすることもできます。キーワードオプションの詳細については、「[ディクショナリ インポート ファイルの操作モードの設定](#)」を参照してください。

DBLoad ユーティリティを使用したタグ定義のインポートにおける既知の制限

ファイルサイズが 2 GB を超える大きなタグ カウント newtag.tag ファイルをインポートする場合、DBLoad ユーティリティでファイルのインポートが失敗することや、パフォーマンスの問題が発生することがあります。

制限:

- newtags.tag ファイルのサイズが 2GB の制限を超える場合、DBLoad で大きなタグ カウント ファイルのインポートが失敗することがあります。
- 大きなタグ カウント CSV ファイルをインポートする場合、DBLoad でパフォーマンスの問題が発生します。

回避策:

以下のレジストリ エントリを使用して、newtags.tag ファイルへのタグのインポートを回避し、Tagname.x ファイルへのインポートを行います。

1. レジストリ エディタを開きます。
2. **HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wonderware\InTouch\Installation** に移動します。
3. ナビゲーション ペインのインストール フォルダを右クリックするか、メイン領域の任意の場所を右クリックして、**[新規]**、**[DWORD 値]** の順にクリックします。
4. 作成されたファイルの名前を **DisableWriteToNewTag** に変更します。
5. “DisableWriteToNewTag” キーを **1** に設定します。
6. DBLoad を使用して CSV ファイルからタグをインポートし、インポートが正常に完了することを確認します。

インポート機能における DisableWriteToNewTag レジストリ キーのさまざまな値の影響を以下に示します。

DisableWriteToNewTag の DWORD 値	インポートへの影響
DisableWriteToNewTag レジストリ キーが存在し、DWORD 値が 1 に設定されている	DBLoad ユーティリティは newtags.tag ファイルへのインポートをスキップします
DisableWriteToNewTag レジストリ キーが存在し、DWORD 値が 0 に設定されている	DBLoad ユーティリティはタグを newtags.tag ファイルにインポートします

注記: InTouch HMI バージョン 2023 以降、レジストリ キー DisableWriteToNewTag はデフォルトで存在しないので、DBLoad ユーティリティは newtags.tag ファイルへのインポートをスキップします。

InTouch アプリケーション間での InTouch ウィンドウのエクスポートとインポート

WindowMaker の 3 つのすべての InTouch アプリケーションタイプからウィンドウをエクスポートできますが、エクスポートされたウィンドウをインポートする場合、または InTouch アプリケーションから直接のウィンドウをインポートする場合に、いくつかの制限があります。

- スタンドアロンの場合、産業用グラフィックが含まれているパブリッシュ済みおよびマネージド InTouch アプリケーションからはウィンドウをインポートできません。警告メッセージが表示され、インポートされなかったウィンドウに関する情報がログに書き込まれます。

産業用グラフィックを含むマネージドまたはパブリッシュ済みの InTouch アプリケーションからウィンドウをインポートする場合、ウィンドウはインポートされますが、産業用グラフィックは機能せず、「見つかりません」と表示されます。

- マネージド InTouch アプリケーションの場合、パブリッシュ済み、スタンドアロン、およびその他のマネージド InTouch アプリケーションからウィンドウをインポートできます。埋め込まれた産業用グラフィックはインポートされません。
- パブリッシュ済み InTouch アプリケーションの場合、スタンドアロン InTouch アプリケーションからウィンドウをインポートできます。埋め込まれた産業用グラフィックはインポートされません。

ウィンドウのインポート

既存の InTouch アプリケーションから現在のアプリケーションにウィンドウをインポートすることによって、以前作成したウィンドウ、オブジェクト、およびウィンドウ スクリプトを再使用できるため、開発時間を削減できます。

ウィンドウをインポートする前に、InTouch HMI ソフトウェアの現在のバージョンにアプリケーションを変換する必要があります。

デフォルトでは、インポートされたウィンドウに関連付けられたタグに対してプレースホルダが作成されます。インポート後、プレースホルダはローカル タグまたはリモート タグ参照に変換できます。詳細については、「[インポートされたウィンドウとスクリプト用のタグ変数プレースホルダ](#)」を参照してください。関連付けられたタグがインポート先アプリケーションにすでに存在する場合、インポート中にこれらのタグを代わりに使用することを選択できます。

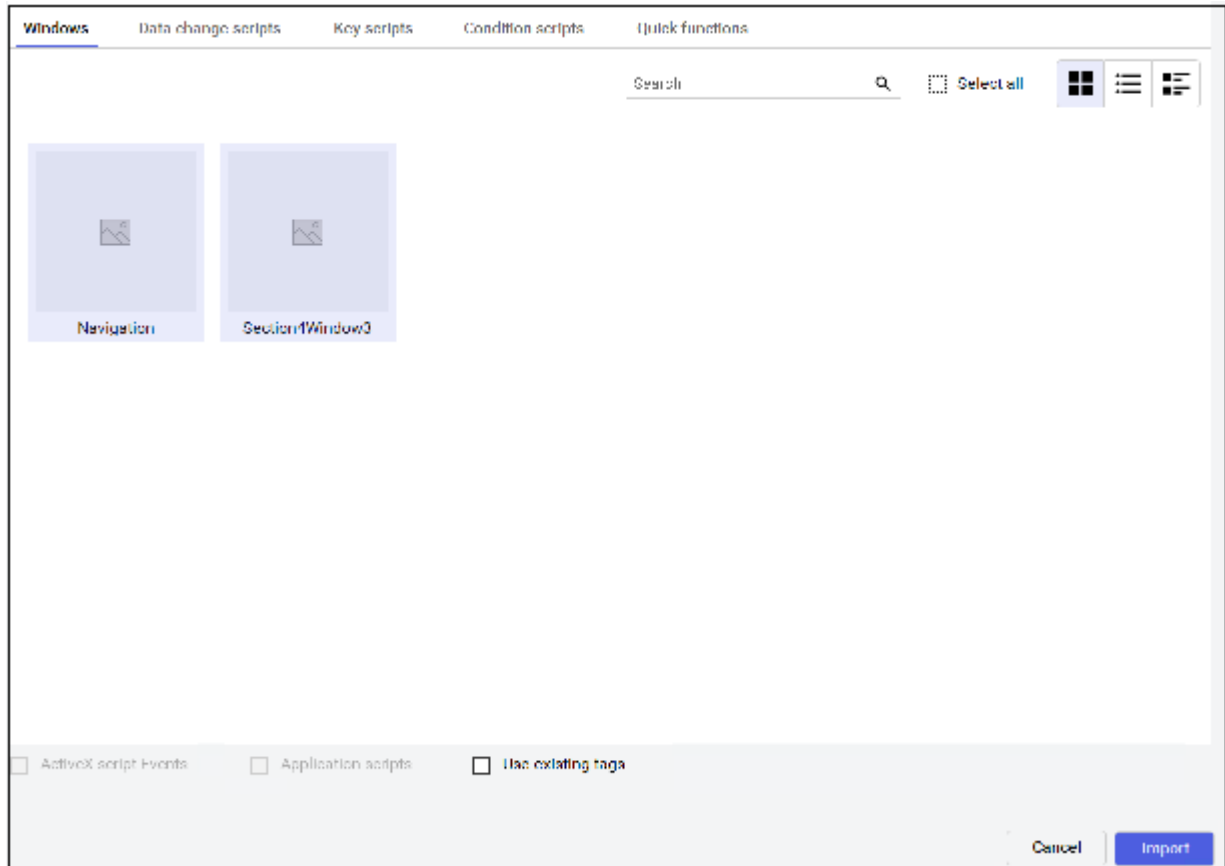
SmartSymbol を含むウィンドウをインポートする際、既存のタグを使用することを選択する場合、タグがインポート先アプリケーションで使用できる場合でも、InTouch HMI は回復されたシンボルに対してプレースホルダを維持します。

スーパータグを含むアプリケーションからウィンドウをインポートする際、ウィンドウで実際に使用されたスーパータグのインスタンスのみが新しいアプリケーションにインポートされます。スーパータグのテンプレート構造全体はインポートされません。たとえば、アプリケーションにスーパータグ メンバーのタグが何百個も定義されている場合で、インポートされたウィンドウでは 50 個しかそれらのタグが使用されていない場合、それらの 50 個のタグのみがインポートされます。

重要: インポートまたはエクスポート以外の方法を使用して InTouch ウィンドウ ファイルを移動する場合、アプリケーションのタグ名ディクショナリの内容が壊れる可能性があります。

ウィンドウをインポートするには

1. 現在使用中のアプリケーションのウィンドウをすべて閉じます。
2. [ファイル] メニューの [インポート] をクリックし、[視覚化]、[ウィンドウとスクリプト] の順にクリックします。
[フォルダを開く] ダイアログ ボックスが表示されます。
3. インポートするウィンドウを含むアプリケーションのフォルダを選択します。
4. [OK] をクリックします。
[アプリケーション データ インポート オプション] ダイアログ ボックスが表示されます。



5. [ウィンドウ] タブでインポートする個々のウィンドウを選択します。
6. インポートされたウィンドウに関連付けられたタグがアプリケーションに既に存在し、それらのウィンドウをプレースホルダの代わりに使用する場合、[既存のタグの使用] チェック ボックスをオンにします。
7. [インポート] をクリックします。
8. プレースホルダのタグをローカル タグまたはリモート タグ参照に変換します。詳細については、[「インポートされたウィンドウのプレースホルダ タグの変換」](#) を参照してください。
9. インポートされたウィンドウに 1 つまたは複数のウィザードが含まれている場合、各ウィザードをダブルクリックして、プロパティ パネルを開きます。インポートされたウィンドウに 1 つまたは複数の SmartSymbol が含まれている場合、各 SmartSymbol を編集して、新しいインスタンスを作成します。

インポートされたウィンドウのプレースホルダ タグの変換

ウィンドウまたは QuickScript を現在のアプリケーションに対してインポートまたはエクスポートする際、そのウィンドウまたは QuickScript に関連付けられているすべてのタグはウィンドウと共に転送されます。ただし、タグは新しいアプリケーションのタグ名ディクショナリには追加されません。その代わりに、[プレースホルダの節約] オプションがインポート時に選択されていない限り、タグがプレースホルダのタグとして自動的にマーク付けされます。これらのプレースホルダ タグは変換して、必要に応じて新しいアプリケーションのタグ名ディクショナリで定義する必要があります。

ウィンドウのタグを変換するには

1. WindowMaker でウィンドウを開きます。
2. F2 キーを押して、ウィンドウのすべてのオブジェクトを選択します。
3. [アニメーション] メニューの [置換] グループで [タグ] をクリックします。
[タグ名の置換] ダイアログ ボックスが表示されます。

Current Name:	Required Type	New Name:
\$Hour	Analog	\$Hour
\$Minute	Analog	\$Minute
\$System	Group	\$System
Batch%Conc	Analog	Batch%Conc
BatchNumber	Analog	BatchNumber
ConcPump	Discrete	ConcPump
Mixer	Discrete	Mixer
PassWord	String	PassWord
ReactLevel	Analog	ReactLevel
ReactTemp	Analog	ReactTemp

Buttons: OK, Cancel, Index, Convert, Replace, Prev Page, Next Page

4. [変換] をクリックします。[変換] ダイアログ ボックスが表示されます。

Convert

Local Remote

Cancel

5. タグを変換します。
 - [ローカル] をクリックして、プレースホルダ タグをローカル タグに変換します。タグ名ディクショナリで各タグを定義するよう求めるメッセージが表示されます。
 - [リモート] をクリックして、プレースホルダ タグをリモート タグ参照に変換します。[アクセス名] ダイアログ ボックスが表示されます。アクセス名を選択して、[閉じる] をクリックします。

変換した後、[タグ名の置換] ダイアログ ボックスに新しいタグが表示されます。

Current Name:	Required Type	New Name:
Auto	Discrete	PLC1:Auto
ConcPump	Discrete	PLC1:ConcPump
ConcValve	Discrete	PLC1:ConcValve
Mixer	Discrete	PLC1:Mixer
OutputValve	Discrete	PLC1:OutputValve
PassWord	String	PLC1:PassWord
ProdLevel	Analog	PLC1:ProdLevel
ReactLevel	Analog	PLC1:ReactLevel
ReactTemp	Analog	PLC1:ReactTemp
SteamValve	Discrete	PLC1:SteamValve

Buttons: OK, Cancel, Index, Convert, Replace, Prev Page, Next Page

6. [OK] をクリックします。

ウィンドウのエクスポート

アプリケーション ウィンドウをエクスポートすると、以下の操作を実行できます。

- すべてのウィンドウのライブラリ アプリケーションを作成または維持する
- 別のアプリケーションでリモート タグ参照を作成する

ウィンドウをエクスポートする前に、InTouch HMI ソフトウェアの現在のバージョンにアプリケーションを変換する必要があります。

ウィンドウをエクスポートすると、そのウィンドウに関連付けられているオブジェクトやアニメーションリンクもすべてエクスポートされます。ウィンドウのオブジェクトに関連付けられたタグはプレースホルダタグに変換され、エクスポート先アプリケーションの既存のタグが上書きされることを防ぎます。プレースホルダタグの詳細については、「[インポートされたウィンドウのプレースホルダタグの変換](#)」を参照してください。

重要: インポートまたはエクスポート以外の方法を使用して InTouch ウィンドウ ファイルを移動する場合、アプリケーションのタグ名ディクショナリが壊れる可能性があります。

ウィンドウをエクスポートするには

1. 現在使用中のアプリケーションのウィンドウをすべて閉じます。
2. [ファイル] メニューの [エクスポート] をクリックし、[視覚化]、[ウィンドウ] の順にクリックします。
3. エクスポートするウィンドウを選択して、[エクスポート] をクリックします。
[フォルダを開く] ダイアログ ボックスが表示されます。
4. ウィンドウをエクスポートするアプリケーションのフォルダを選択します。
5. [OK] をクリックします。
6. 問題が発生した場合、[エクスポート実行時の問題] ダイアログ ボックスが表示されます。実行するアクションのオプションをクリックし、[OK] をクリックします。

スクリプトのインポート

開発時間を節約するために、既存の QuickScript は InTouch アプリケーションから現在のアプリケーションにインポートできます。

スクリプトをインポートする前に、InTouch HMI ソフトウェアの現在のバージョンにアプリケーションを変換する必要があります。

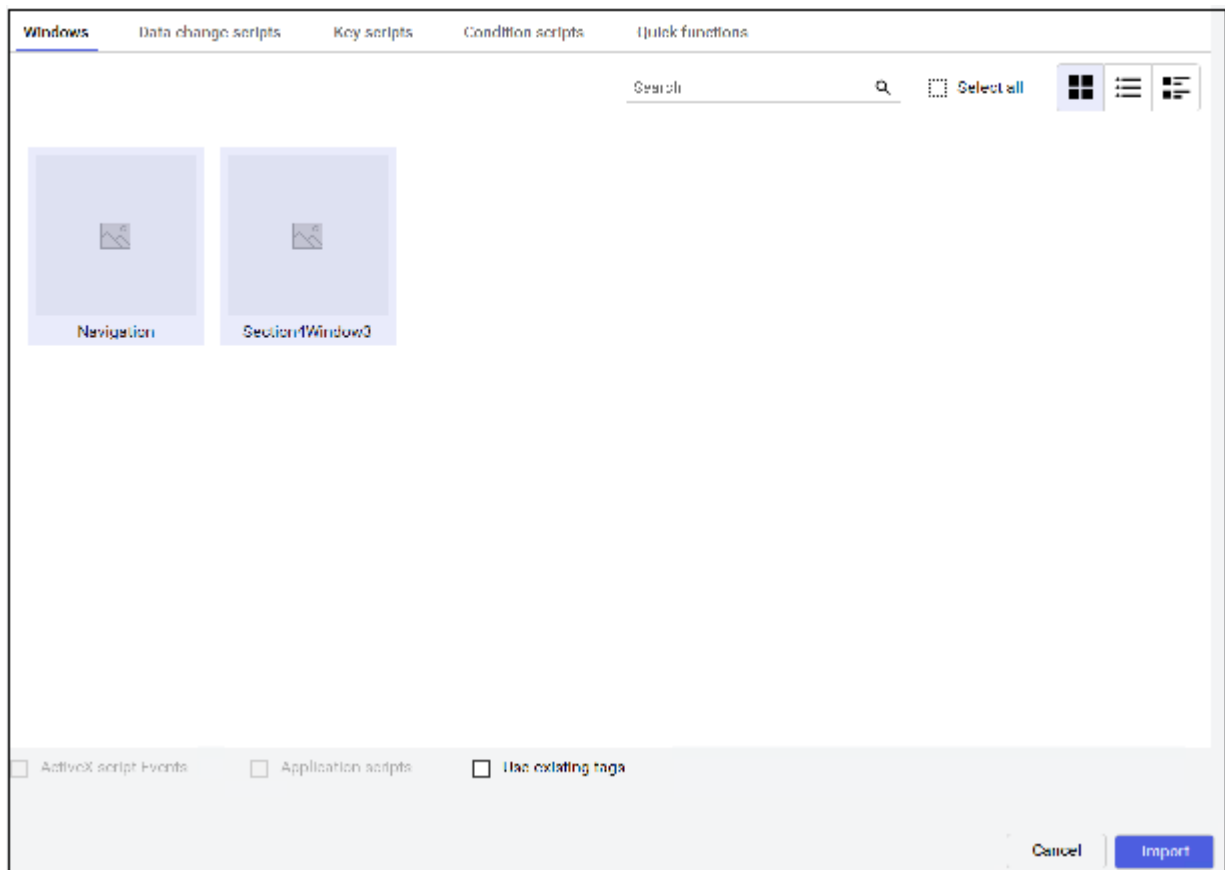
デフォルトで、インポートされた QuickScript に関連付けられたタグに対してプレースホルダが作成されます。インポート後、プレースホルダはローカル タグまたはリモート タグ参照に変換できます。詳細については、「[インポートされたウィンドウとスクリプト用のタグ変数プレースホルダ](#)」を参照してください。関連付けられたタグがインポート先アプリケーションにすでに存在する場合、インポート中にこれらのタグを代わりに使用することを選択できます。

ウィンドウ スクリプトをインポートするには、ウィンドウ全体をインポートする必要があります。

インポートされた ActiveX イベント スクリプトがインポート先のアプリケーションで正しく機能するためには、スクリプトが元々作成されたときと同じ ActiveX コントロールとイベントがインポート先のアプリケーションで使用される必要があるため、メモリにロードする必要があります。ActiveX コントロールを含むウィンドウが閉じられると、それに関連付けられたスクリプト (ActiveX イベント スクリプトまたは QuickScript) は正常に動作しません。

QuickScript をインポートするには

1. 現在使用中のアプリケーションのウィンドウをすべて閉じます。
2. [ファイル] メニューの [インポート] をクリックし、[視覚化]、[ウィンドウとスクリプト] の順にクリックします。
[フォルダを開く] ダイアログ ボックスが表示されます。
3. インポートするスクリプトを含むアプリケーションのフォルダを選択します。
4. [OK] をクリックします。[アプリケーションデータ インポート オプション] ダイアログ ボックスが表示されます。



5. インポートするクイック関数タイプのチェック ボックスをオンにし、[選択] をクリックして、インポートする個々のスクリプトを選択します。

注記: ウィンドウ スクリプトをインポートするには、ウィンドウ全体をインポートする必要があります。詳細については、「[ウィンドウのインポート](#)」を参照してください。

6. インポートされたスクリプトに関連付けられたタグがアプリケーションに既に存在し、それらのスクリプトをプレースホルダの代わりに使用する場合は、[既存のタグの使用] チェック ボックスをオンにします。
7. [インポート] をクリックします。アプリケーションに同じ名前を持つスクリプトがある場合、上書き、省略、または名前の変更などの操作を行うよう求めるメッセージが表示されます。

8. プレースホルダのタグをローカルタグまたはリモートタグ参照に変換します。詳細については、「[インポートされたスクリプトでのプレースホルダタグの変換](#)」を参照してください。

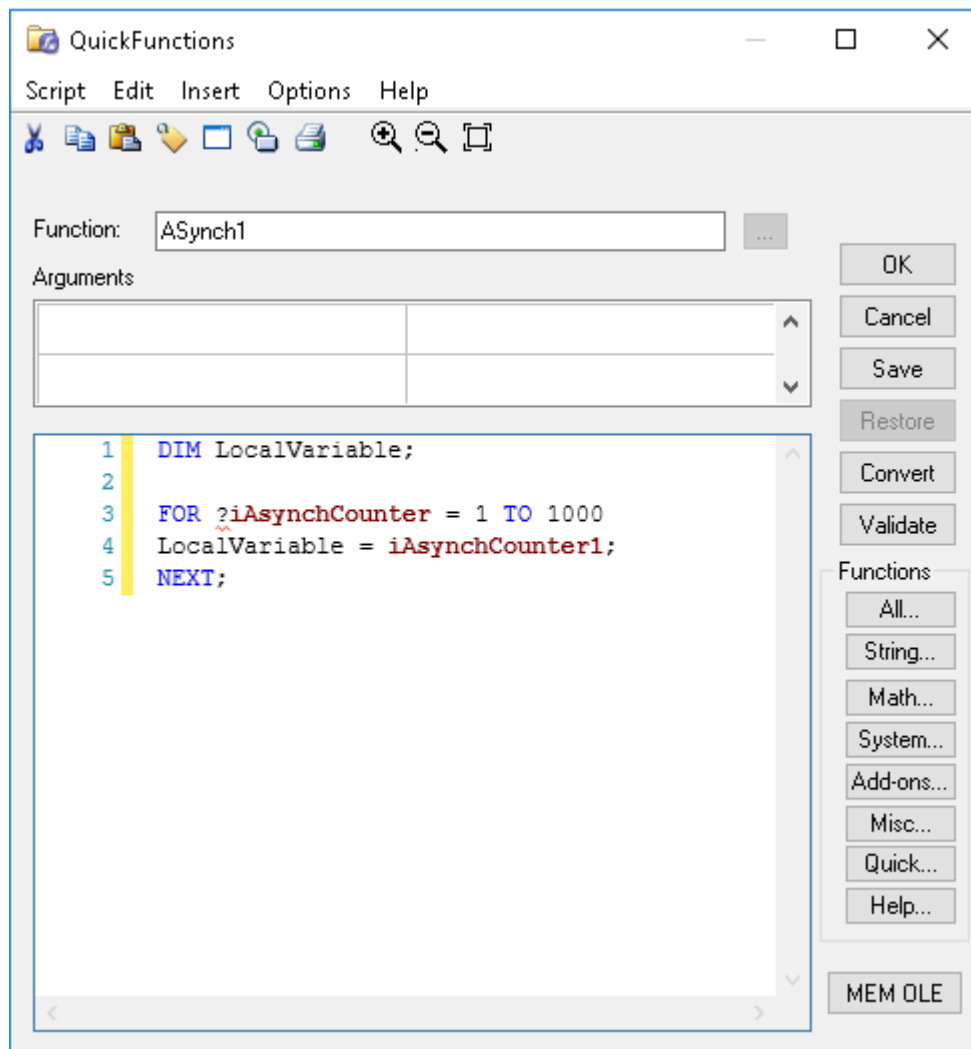
インポートされたスクリプトでのプレースホルダタグの変換

クイック関数を現在のアプリケーションに対してインポートまたはエクスポートする際、そのクイック関数に関連付けられているすべてのタグは転送されます。ただし、タグは新しいアプリケーションのタグ名ディクショナリには追加されません。その代わりに、これらのタグは自動的にプレースホルダタグとしてマーク付けされます。これらのプレースホルダタグは変換して、必要に応じて新しいアプリケーションのタグ名ディクショナリで定義する必要があります。

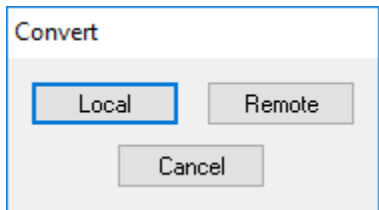
インポートされたスクリプトでプレースホルダタグを変換するには

1. [スクリプト] ペインで、インポートしたクイック関数のタイプをクリックします。

クイック関数スクリプトエディタが表示され、選択したスクリプトタイプのファイルの最初のクイック関数が表示されます。



2. [変換] をクリックします。[変換] ダイアログボックスが表示されます。



1. タグを変換します。
 - [ローカル] をクリックして、プレースホルダ タグをローカル タグに変換します。タグ名ディクショナリで各タグを定義するよう求めるメッセージが表示されます。
 - [リモート] をクリックして、プレースホルダ タグをリモート タグ参照に変換します。[アクセス名] ダイアログ ボックスが表示されます。アクセス名を選択して、[閉じる] をクリックします。
2. タグが変換された後で、QuickScript エディタの [OK] をクリックします。

インポートされたウィンドウとスクリプト用のタグ変数プレースホルダ

ウィンドウまたは QuickScript をインポートするとき、関連付けられたタグ変数を処理する方法を設定できます。

- プレースホルダ タグ変数を使用します。

デフォルトで、インポートされたタグ変数は「プレースホルダ」（または「インデックス」）タグ変数に変換されます。最大 4096 のプレースホルダが許可されています。

プレースホルダ タグ変数には、3 文字のプリフィックスが含まれます。たとえば、元のタグ変数が WaterHeater である場合、プレースホルダ タグ変数は ?d:WaterHeater です。

30、31、または 32 文字を含むタグ変数をインポートする場合、プレースホルダのプリフィックスがタグ変数の先頭に追加され、既存のタグ変数の長さは切り捨てられません。たとえば、プレースホルダ タグ変数の場合に限り、32 文字のタグ変数は 35 文字に増加します。タグ変数の長さの増加は標準タグ変数ではサポートされていません。

アプリケーションでプレースホルダ タグ変数を使用するには、以下のいずれかの手順に従います。

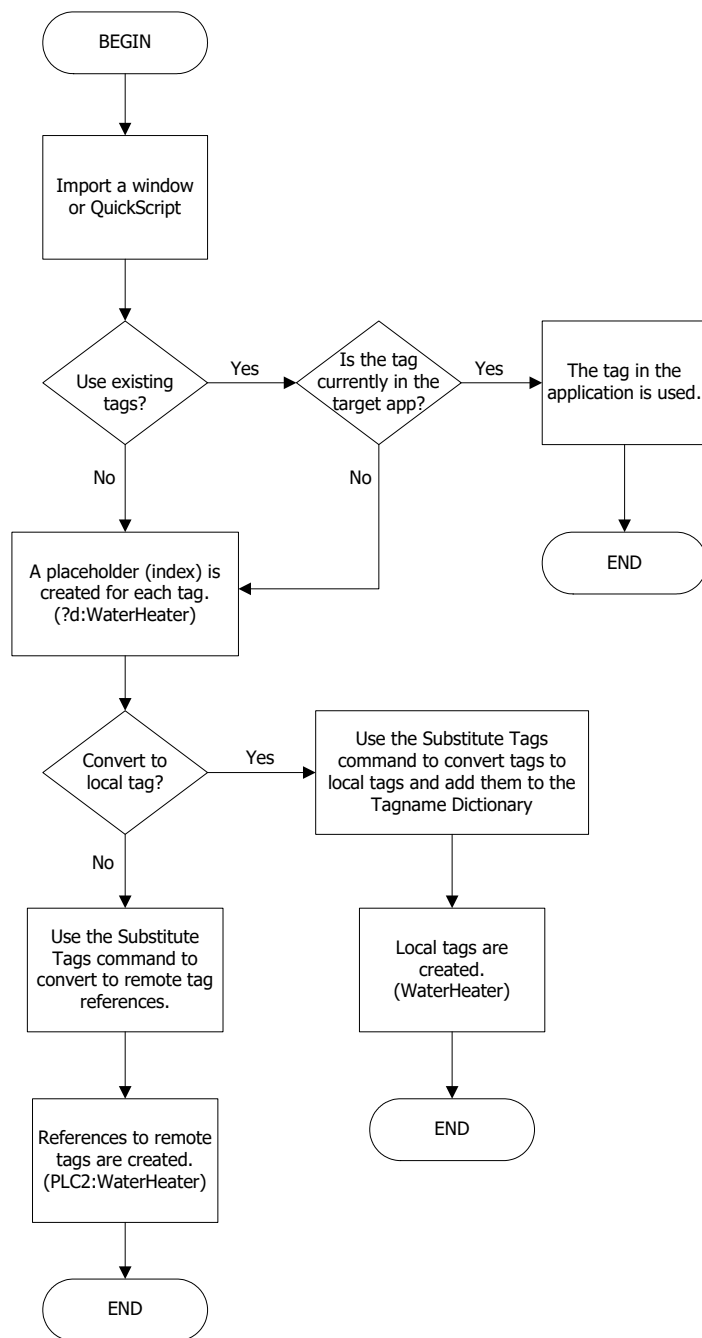
標準の（ローカル）タグ変数に変換し、タグ変数ディクショナリで定義します。

リモート タグ リファレンスに変換します。リモート タグ変数の例は、PLC2:WaterHeater です。リモート タグ リファレンスによって、アプリケーションはリモート タグ変数サーバーからデータをすぐに受け取り、ローカル タグ変数ディクショナリで単一タグ変数を定義する必要がなくなります。

- 既存のタグ変数を使用します。

インポート中、既存のタグ変数の使用を選択すると、InTouch HMI はインポートされたタグ変数がタグ変数ディクショナリにすでに存在することを確認します。タグ変数がすでに存在する場合、タグ変数は完全修飾されたタグ変数としてインポートされます。このオプションを使用することによって、プレースホルダの合計数が削減され、サイズのより大きいタグ変数データベースでアプリケーションをインポートできるようになります。

以下のフローチャートは、インポートされたウィンドウと QuickScript に対してタグ変数が処理される方法を説明しています。



アプリケーションからの産業用グラフィックのエクスポート

アプリケーションのすべての産業用グラフィックを **aaPKG** ファイルにエクスポートできます。その後、ファイルからグラフィックを同じまたは別のコンピュータ上の別のアプリケーションにインポートできます。

アプリケーションからエクスポートする産業用グラフィックを個々に選択することはできません。すべての産業用グラフィックがアプリケーションからエクスポートされます。

アプリケーションから産業用グラフィックをエクスポートするには

1. エクスポートする産業用グラフィックを含むアプリケーションを **WindowMaker** で開きます。
2. **[ファイル]** メニューの **[エクスポート]** をクリックして、**[視覚化]** グループから **[すべての産業用グラフィック]** をクリックします。
[産業用グラフィックのエクスポート] ダイアログ ボックスが表示されます。このダイアログ ボックスを使用して、エクスポート先のフォルダを指定してエクスポート ファイルに名前を付けます。
3. **aaPKG** ファイルのエクスポート先のフォルダを選択します。
4. 必要に応じて、**[ファイル名]** フィールドにエクスポート ファイルの名前を入力します。
エクスポート ファイルのデフォルト名は **IndustrialGraphics.aaPKG** です。
5. **[保存]** をクリックします。
エクスポート ファイルへの産業用グラフィックのロードの進捗状況を示す水平バーが表示されます。
6. エクスポート処理が終了した後、**Windows** エクスプローラでエクスポート先のフォルダに移動して、エクスポート ファイルが作成されていることを確認します。

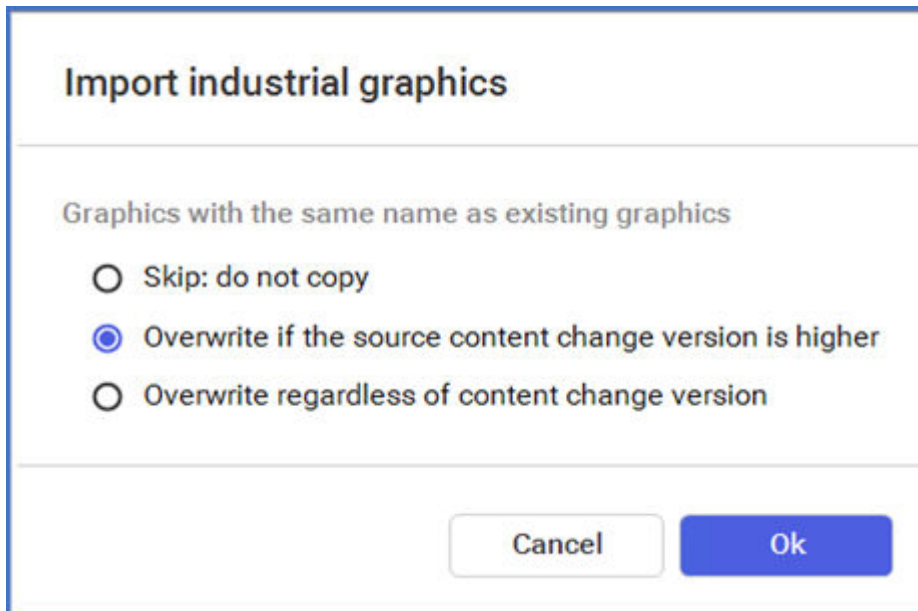
アプリケーションへの産業用グラフィックのインポート

別のモダンアプリケーションで作成された産業用グラフィックを **WindowMaker** で実行しているアクティブなアプリケーションにインポートできます。

インポートされるのは **aaPKG** ファイルに含まれる産業用グラフィックだけです。**WindowMaker** で編集用を開いているアプリケーションのグラフィックは、インポートしたグラフィックによって上書きされます。**aaPKG** ファイルにサポートされないコンポーネントが含まれる場合、インポートは失敗して、エラーを示すダイアログ ボックスが表示されます。

アプリケーションに産業用グラフィックをインポートするには

1. インポートする産業用グラフィックを含むアプリケーションを **WindowMaker** で開きます。
2. **[ファイル]** メニューの **[インポート]** をクリックして、**[視覚化]** グループから **[産業用グラフィック]** をクリックします。
産業用グラフィックのエクスポート ファイルを含むフォルダを指定する **[産業用グラフィックのインポート]** ダイアログ ボックスが表示されます。
3. **Windows** エクスプローラを使用して、エクスポートされた産業用グラフィックを含む **aaPKG** ファイルが保存されているフォルダに移動します。
4. インポートする **aaPKG** ファイルを選択します。
選択したファイルの名前が **[ファイル名]** フィールドに表示されます。
5. **[開く]** をクリックします。
[産業用グラフィックのインポート] ダイアログ ボックスが表示され、グラフィックの上書きに関する以下のオプションが表示されます。



- スキップ: コピーしない - グラフィックはインポートされません。
- ソース コンテンツの変更バージョンが新しい場合に上書きする - グラフィックは、インストールされているバージョンよりもインポートするファイルのバージョンの方が新しい場合にのみインポートされます。
- コンテンツの変更バージョンに関係なく上書きする - グラフィックがインポートされます。

6. [OK] をクリックします。

アクティブなアプリケーションへの産業用グラフィックのインポートの進行状況を示す水平バーが表示されます。処理が終了したら進捗状況を示すインジケータが消えます。

産業用グラフィック ツールボックスで選択したシンボルのエクスポート

アプリケーションの産業用グラフィック ツールボックスから選択した産業用グラフィックを aaPKG ファイルにエクスポートできます。その後、ファイルからグラフィックを同じまたは別のコンピュータ上の別のアプリケーションにインポートできます。

注: ここでは、選択した産業用グラフィックをエクスポートする手順を示します。すべての産業用グラフィックをエクスポートする手順については、「[アプリケーションからの産業用グラフィックのエクスポート](#)」を参照してください。

選択した産業用グラフィックをアプリケーションからエクスポートするには

1. エクスポートする産業用グラフィックを含むアプリケーションを WindowMaker で開きます。
2. エクスポートするシンボルを産業用グラフィック ツールボックスで選択します。
3. 選択したシンボルを右クリックしてショートカットメニューを表示します。
4. ショートカットメニューで [エクスポート] を選択して、[シンボル] を選択します。

[産業用グラフィックのエクスポート] ダイアログボックスが表示されます。このダイアログボックスを使用して、エクスポート先のフォルダを指定してエクスポート ファイルに名前を付けます。

5. aaPKG ファイルのエクスポート先のフォルダを選択します。

6. 必要に応じて、[ファイル名] フィールドにエクスポート ファイルの名前を入力します。

デフォルトのエクスポート ファイル名は、産業用グラフィック ツールボックスで最初に選択したシンボルの名前です。

7. [保存] をクリックします。

エクスポート ファイルへの産業用グラフィックのロードの進捗状況を示す水平バーが表示されます。

カスタム クライアント コントロールのインポートと埋め込み

カスタム Windows クライアント コントロールを作成して、アプリケーションの産業用グラフィックに埋め込むことができます。最初に、クライアント コントロールを WindowMaker の産業用グラフィック ツールボックスにインポートする必要があります。このセクションでは、カスタム クライアント コントロールをインポートする手順および埋め込む手順について説明します。

カスタム クライアント コントロールをインポートする方法

1. アプリケーションのカスタム クライアント コントロールを作成します。
2. InTouch WindowMaker がインストールされているコンピュータからアクセスできるフォルダにクライアント コントロールを配置します。
3. [ファイル] メニューの [インポート] をクリックして、[視覚化] グループから [クライアント コントロール] をクリックします。

重要: カスタム クライアント コントロールをインポートできるのは、スタンドアロンアプリケーションだけです。従来の InTouch アプリケーションまたはパブリッシュ済み InTouch HMI アプリケーションにカスタム クライアント コントロールをインポートすることはできません。

作成したカスタム クライアント コントロールの名前を入力するフィールドを含む [クライアント コントロールのインポート] ダイアログが表示されます。

1. Windows エクスプローラを使用して、クライアント コントロール .dll ファイルを配置したフォルダに移動します。
2. クライアント コントロール .dll ファイルを選択し、[開く] をクリックします。

WindowMaker は産業用グラフィック ツールボックスにインポートされたカスタム クライアント コントロールを更新し、表示します。

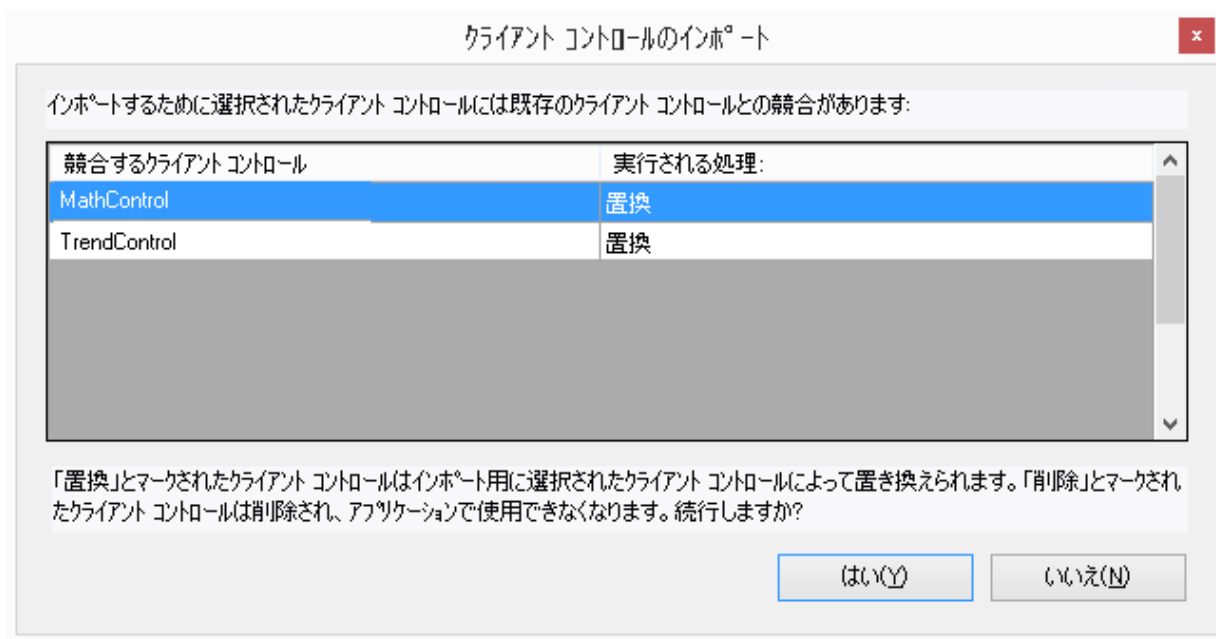
インポートされたクライアント コントロールを産業用グラフィック ツールボックスから削除することもできます。最初に、産業用グラフィック ツールボックス内でクライアント コントロールを選択します。次に、右クリックしてショートカットメニューを表示し、[削除] を選択します。

重複するクライアント コントロールをインポートする際の競合の解決

別のバージョンのクライアント コントロールをインポートして、既存のコントロールを上書きできます。既存のコントロールをホストしている .dll は、インポートするライブラリによって置換されます。競合するクライアント コントロールは、新しいクライアント コントロール .dll のインポート時に削除されます。

注記: 競合検出の条件は、コントロールの名前だけです。ライブラリのファイル名やバージョンは、競合検出に影響しません。

たとえば、**MathControl** と **TrendControl** という 2 つのコントロールを含むクライアント コントロール .dll をインポートする際に現在のライブラリに同じ名前のコントロールが含まれる場合、[クライアント コントロールのインポート] ダイアログ ボックスが表示されます。

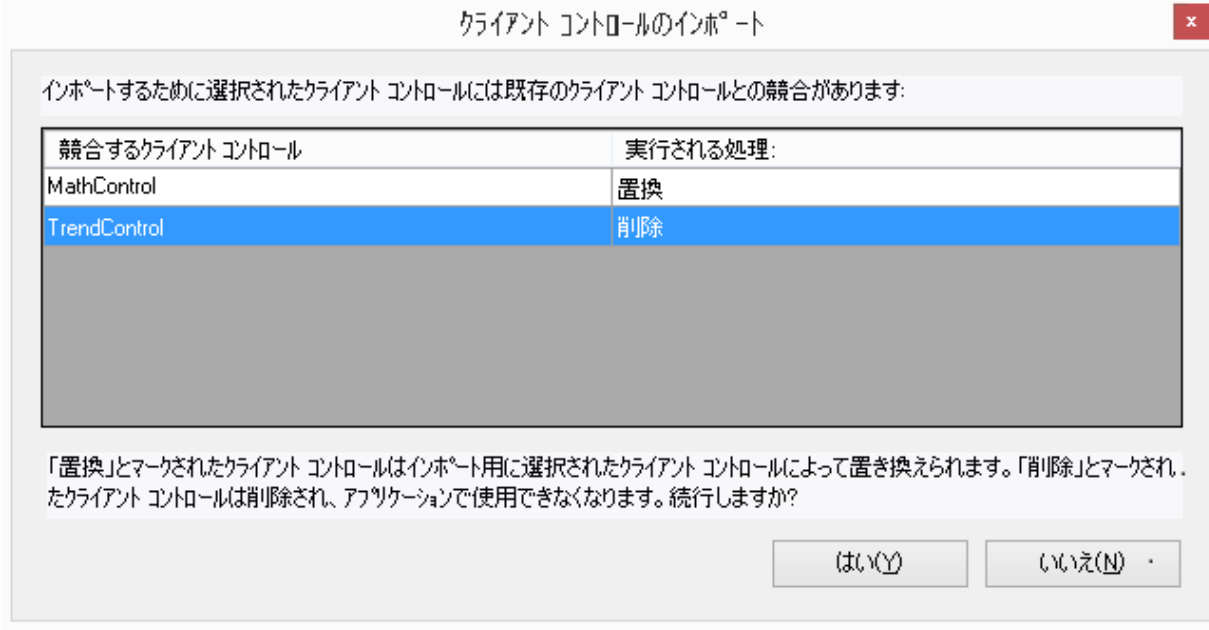


既存のクライアント コントロール .dll が置換され、新しいコントロールがライブラリで使用可能になります。

[実行される処理] カラムに「削除」と表示されている場合、インポートするライブラリに含まれないコントロールが現在のライブラリのコントロールにあることを意味します。競合するコントロールを解決するためには、ホストしている .dll を置き換える必要があるため、現在の .dll に含まれていて、インポートする .dll に含まれないライブラリはインポート処理中に削除されます。

たとえば、**MathControl** と **DatabaseControl** を含むクライアント コントロール .dll をインポートする際に、現在のライブラリに **MathControl** と **TrendControl** が含まれる場合、インポート時にライブラリから **TrendControl** が削除されます。

[クライアント コントロールのインポート] ダイアログ ボックスに削除を確認するプロンプトが表示されます。



ライブラリが置換され、インポートの完了時に TrendControl が削除されます。

WindowMaker を再起動して、グラフィック ツールボックスのコントロールを更新します。

注記: シンボルに既に埋め込まれているクライアント コントロールの新しいバージョンをインポートした場合、WindowMaker を再起動してグラフィック サムネイルを最新の情報に更新してもコントロールの内容は更新されません。新しいクライアント コントロールをサムネイルに反映させるには、新しいクライアント コントロールのシンボルを編集して保存する必要があります。

産業用グラフィックへのクライアント コントロールの埋め込み

クライアント コントロールは、産業用グラフィック ツールボックスから埋め込まれます。グラフィック ツールボックスには、既にいくつかのクライアント コントロールが含まれています。これらの既存のコントロールを産業用グラフィックに埋め込むことや、カスタム コントロールをインポートして埋め込むことができます。

産業用グラフィックにクライアント コントロールを埋め込むには

1. カスタム クライアント コントロールを埋め込むアプリケーションを WindowMaker で開きます。
2. カスタム クライアント コントロールを埋め込む産業用グラフィックを含むウィンドウを開きます。
3. 産業用グラフィックを選択します。
4. メニューバーの「産業用グラフィックの埋め込み」アイコンをクリックします。

重要: 産業用グラフィック ツールバーから産業用グラフィックにカスタム クライアント コントロールをドラッグアンドドロップすることはできません。カスタム クライアント コントロールは必ず埋め込む必要があります。

1. 必要に応じて、アプリケーション用にカスタム クライアント コントロールを設定します。

HTML5 ウィジェットのインポート

ウィジェットは、Web ページや Web サイトの機能を拡張できる小型の Web コンポーネントです。カスタム ビルドの Web サイトでも、オープン ソース コードやフレームワークを使用してウィジェットを組み込んで全体的または部分的に特定の機能を提供できます。ウィジェットは自己包含型のコードブロックで、その機能を変更することなく Web サイトに組み込むことができます。ウィジェットは、その他のプラットフォームおよびデータ ソースと統合するユーザー インターフェイス要素を提供するために最も頻繁に使用されます。ウィジェットは、一貫した配置およびユーザー インターフェイスで Web サイトの任意の Web ページで実行できます（ソーシャルメディア、天気予報、RSS、ポッドキャストウィジェットなど）。

デフォルトでは、グラフィック ツールバーのウィジェット フォルダで以下のウィジェットを使用できます。

- カラーセル
- Web ブラウザ
- QR コード スキャナ
- Map_App

スタンドアロン アプリケーションおよびマネージド アプリケーションのウィジェットをインポートできます。ファイル形式は、HTML5、CSS、および Javascript ファイルを含むカスタム ウィジェット パッケージ (.cwp) です。

HTML ウィジェットのインポート

1. WindowMaker を起動します。
2. [ファイル] メニューの [インポート] をクリックし、[視覚化]、[HTML5 ウィジェット] の順にクリックします。

[HTML5 ウィジェットのインポート] ダイアログ ボックスが表示されます。

3. インポートするウィンドウを含むアプリケーションのフォルダを選択します。
4. [OK] をクリックします。

ツールボックスにウィジェットが表示されます。

ウィジェットをインポートした後、以下の操作を行います。

1. グラフィックを作成します。
2. グラフィックを編集してウィジェットを埋め込みます。
3. [ウィジェットのプロパティ] セクションでプロパティを設定します。それぞれのウィジェットには、独自のプロパティがあります。
4. ウィジェットをウィンドウに挿入します。

ウィジェットが WindowViewer および任意の Web ブラウザで表示できるようになります。デザイン時のプロパティ設定に応じて、ウィジェットをランタイム時に操作できます。[ウィジェットのプロパティ] の下にあるカスタム プロパティを使用してウィジェットを変更するスクリプトはサポートされていません。

カラーセル ウィジェット

カルーセル ウィジェットでは、入力なしでカルーセルのようにさまざまな要素（画像やテキストのスライドなど）をサイクル表示できます。このウィジェットは、ダッシュボード、アラート、またはアラーム情報を工場作業場の大型モニタに表示するために使用できます。

注記: カルーセル ウィジェットのクライアントコントロール（アラーム クライアントコントロールおよびトレンドクライアントコントロール）画面は、**WindowViewer** および **Web Client** でサポートされていません。

プロパティ

[ウィジェットのプロパティ] では、標準のグラフィック プロパティに加えて、ウィジェット固有のプロパティも設定できます。

名前	説明	デフォルト
Autoplay	Autoplay プロパティが True に設定されている場合、起動時に カルーセル ウィジェットが自動的にロードされます。False に設定されている場合、カルーセルを開始する次の項目をユーザーが選択する必要があります。	True
BackgroundColor	ウィジェットの背景色を設定します。RGB、HTML コード (#FF0000)、または有効な HTML 色名で色の値を指定します。	白
GraphicNames	カルーセルがランタイムに実行するグラフィックのリスト（カンマ区切り）。	空
Interval	項目のサイクルを自動的に開始する際の間隔（ミリ秒）。	5000
Keyboard	Keyboard プロパティが True に設定されている場合、カルーセルはキーボード入力に反応します。	True
Loop	Loop プロパティが True に設定されている場合、グラフィックが継続的にサイクルします。それ以外の場合、1 回のサイクル後に停止します。	True
Pause	Pause プロパティが True に設定されている場合、カルーセル上のマウス ホバーまたはクリック イベントが検出されるとグラフィックのサイクルが一時停止します。グラフィックのサイクルは、カルーセルからマウスを離すと再開します。	True

カルーセル ウィジェットは **Bootstrap 4.0** のカルーセル コンポーネントをベースにしています。Bootstrap の詳細については、以下のサイトを参照してください: <https://getbootstrap.com/docs/4.0/components/carousel/>

Web ブラウザ ウィジェット

Web ブラウザ ウィジェットを使用して、Web サイトを **WindowViewer** および **Web Client** に表示できます。

Web Client が HTTPS で実行している場合、読み込むことができるのは HTTPS URL ページだけです。Web Client が HTTP で実行している場合、HTTPS と HTTP の両方のページを読み込むことができます。クロスドメインアクセスが Web サービスのポリシーによってブロックされる場合、このウィジェットは機能しません。URL を二重引用符で囲む必要はありませんが、有効な URL である必要があります。

プロパティ

URL: Web サイトのアドレス。

制限

- プロトコルを指定しない場合、デフォルトで https が使用されます。
- HTTPS を使用するように Web Client が設定されている場合、読み込むことができるのは HTTPS URL ページだけです。HTTP URL を使用する場合、Web ブラウザ ウィジェットに「Mixed Content: The page at 'https://localhost/intouchweb' was loaded over HTTPS, but requested an insecure frame 'http://*****'. this request has been blocked: the content must be served over HTTPS.」（混合コンテンツ: 'https://localhost/intouchweb' のページは HTTPS で読み込まれましたが、セキュアでないフレーム 'http://*****' に対するリクエストはブロックされました。コンテンツは HTTPS で提供される必要があります。）というメッセージが表示されます。
- Web サイト ポリシーによってクロスドメイン（クロスオリジン）アクセスがブロックされる場合、Web ブラウザ ウィジェットは機能しません。Web ページを開くリンクが別のタブに表示されます。

QR コード スキャナ

QRCode_Scanner ウィジェットはカメラに接続して QR コードをスキャンし、結果の文字列を返します。

プロパティ

プロパティ名	説明	デフォルト値
QRCode	スキャンした QR コードの結果の文字列。デフォルト値は空です。	空
AutoStart	True に設定すると、カメラが自動的に起動します。	True
AutoStop	True に設定すると、QR コードをスキャンした後にカメラが自動的に停止します。	True
Start	True に設定すると、カメラが起動します。	False
Stop	True に設定すると、カメラが停止します。	False
BackgroundColor	ウィジェットの背景色を設定します。RGB、HTML コード（#FF0000）、または有効な HTML 色名で色の値を指定します。	Black

制限事項

- デバイスにカメラが装備されている必要があります。
- 仮想マシンではなく、物理マシンの QR コードを使用することが推奨されます。

- Web Client をリモートで使用する場合は、セキュアな URL (https://) を使用して Web Client にアクセスします。

使用法

スキャンした値に基づいて、QR コードをスキャンしてグラフィックを表示するスクリプトを設定できます。

ランタイム時、[AutoStart]、[AutoStop]、および [StartStop] ボタンを含むフローティング ツールバーと共に QR コード スキャナ ウィジェットが表示されます。

AutoStart が True に設定されている場合、ウィジェットをロードするとカメラが自動的に起動します。カメラを起動したままにするには、[AutoStop] をクリックします。

手動でカメラを起動するには、[StartStop] をクリックして QR コードをスキャンします。

QR コードをスキャンした後もカメラは起動したままになるので、別の QR コードをスキャンできます。カメラを停止するには、[StartStop] をクリックします。

フローティング ツールバーには、カメラでスキャンした QR コードから派生した QRCode が表示されます。

返される QRCode に基づいてアクションのスクリプトを記述できます。

Map_App ウィジェット

Map_App ウィジェットには、実行中のアプリケーション内でシンボルを含むマップを表示できます。ランタイム中、マップは、マップのさまざまな領域のパンや、ズーム インまたはズーム アウトして詳細表示のレベル調整を行うためのコントロールとタッチサポートを提供します。一般的に、マップ内に配置されたグラフィックは、マップが示す領域内に配置されたビジネス アセットを表します。これらのグラフィックには、各ビジネスの場所のプロセスの現在の状態を示すアラームを含めることができます。

プロパティ

Map_App ウィジェットの特定のプロパティは産業用グラフィック エディタから設定できます。

プロパティ	説明
ConfigName	Map_App ウィジェット グローバル設定ファイルの名前。
InitialLatitude	初期中心点のマップ位置の緯度 (角度)。有効な緯度値は +/- 0~90 です。
InitialLongitude	初期中心点のマップ位置の経度 (角度)。有効な経度値は +/- 0~180 です。
InitialZoom	ランタイム時に最初に表示されたマップのズーム レベル パーセンテージ。
MinZoom	ランタイム時にズーム アウトできるマップの最小ズーム パーセンテージ (0~100%)。
MaxZoom	ランタイム時にズーム インできるマップの最大ズーム パーセンテージ (0~100%)。

MaxBoundsSouth	画面の視点中心点の垂直パンの動きをマップの下の境界に制約する南側のマップ境界の緯度 (+/- 0~90°)。
MaxBoundsWest	画面の視点中心点の水平パンの動きをマップの左側の境界に制約する西側のマップ境界経度 (+/- 0~180°)。
MaxBoundsNorth	画面の視点中心点の垂直パン動作をマップの上の境界に制約する北側のマップ境界の緯度 (+/- 0~90°)。
MaxBoundsEast	画面の視点中心点の水平パンの動きをマップの右側の境界に制約する東側のマップ境界経度 (+/- 0~180°)。
Asset	表示されているマップから選択されたアセットの名前。
CurrentLatitude	マップ上で表示されているアイテムの現在の緯度。
CurrentLongitude	マップ上で表示されているアイテムの現在の経度。
CurrentZoom	表示されているマップの現在のズーム レベル。
FollowCurrentAsset	<p>このプロパティを True に設定すると、MapApp ウィジェットが現在選択されているアセット（コンテキスト）に追従して自動的にマップをパンおよびズームして、アセットの位置および関連付けられたシンボルが表示されます（アセットはマップ アプリ エディタ ページの場所タブに追加し、アセット レベルを "-1" に設定する必要があります）。</p> <ul style="list-style-type: none">• マップを開くと、マップは選択されたアセットを中心に表示されます。アセットを使用して ViewApp 内で移動できます。たとえば、すべての地域を表示し、個々の状態のマーカーを表示できます。次に、マップ内で地域を選択することで、地域の詳細を示す別のペインに ViewApp のフォーカスを設定できます。• マップは、選択したアセットのズーム レイヤーに 1% を加えてズームします。• アセットが選択されていない場合、または選択したアセットがマップ上に配置されていない場合、マップは初期ズーム レイヤーとマップの中心点を表示します。 <p>Asset プロパティが設定されている場合、FollowCurrentAsset プロパティを False に設定すると、アセットの場所および関連付けられたシンボルを使用してマップをロードできます。</p>

Sources	<p>アプリのマップ設定で構成されたマップデータ ソース。(All) がデフォルト値で、Map アプリに対して指定されたすべてのマップ データ ソースが含まれます。</p> <p>注記: All は Sources プロパティ値として半角カッコ内に配置する必要があります (All)。</p> <p>一部のマップ ソースからのデータだけを表示するよう Map アプリを制限する場合、次に示すようにカンマ区切りの文字列を使用して複数のソースを指定します。</p> <p>OSM,Bing,TemperatureOverlay</p>
ZoomLayers	<p>Map アプリに対して設定されたマップ ズーム レイヤー。(All) がデフォルト値で、Map アプリに対して指定されたすべてのズーム レイヤーが含まれます。</p> <p>注記: All は ZoomLayers プロパティ値として半角カッコ内に配置する必要があります (All)。</p> <p>一部のズーム レイヤーからのデータだけを表示するよう Map アプリを制限する場合、次に示すようにカンマ区切りの文字列を使用して名前ズーム レイヤーを指定します。</p> <p>country,state,city</p>

InTouch アプリケーションへのスクリプト関数ライブラリのインポート

InTouch アプリケーションにスクリプト関数ライブラリをインポートできます。.NET (*.dll およびその他の .NET ファイル拡張子)、スクリプト ライブラリ ファイル (*.aaSLIB)、および InTouch スクリプト拡張ファイル (*.wdf) を含むさまざまなスクリプト関数ライブラリをインポートできます。

1 つのアプリケーションにインポートしたスクリプト関数ライブラリは、別のアプリケーションを作成するためにアプリケーションをエクスポートする際、自動的にエクスポートに含まれます。スクリプト関数ライブラリは、それをインポートしたアプリケーションをパブリッシュするときにも使用できます。

スクリプト関数ライブラリをアプリケーションにインポートするには

1. スクリプト関数ライブラリをインポートする InTouch アプリケーションを開きます。
2. WindowMaker のメインメニューで [ファイル] をクリックし、[インポート] をクリックして [スクリプト] をクリックします。
[スクリプト関数ライブラリのインポート] ダイアログが開きます。
3. インポートする関数ライブラリを参照します。
4. インポートするファイルを選択して [開く] をクリックすると、スクリプト関数ライブラリのインポートが開始されます。

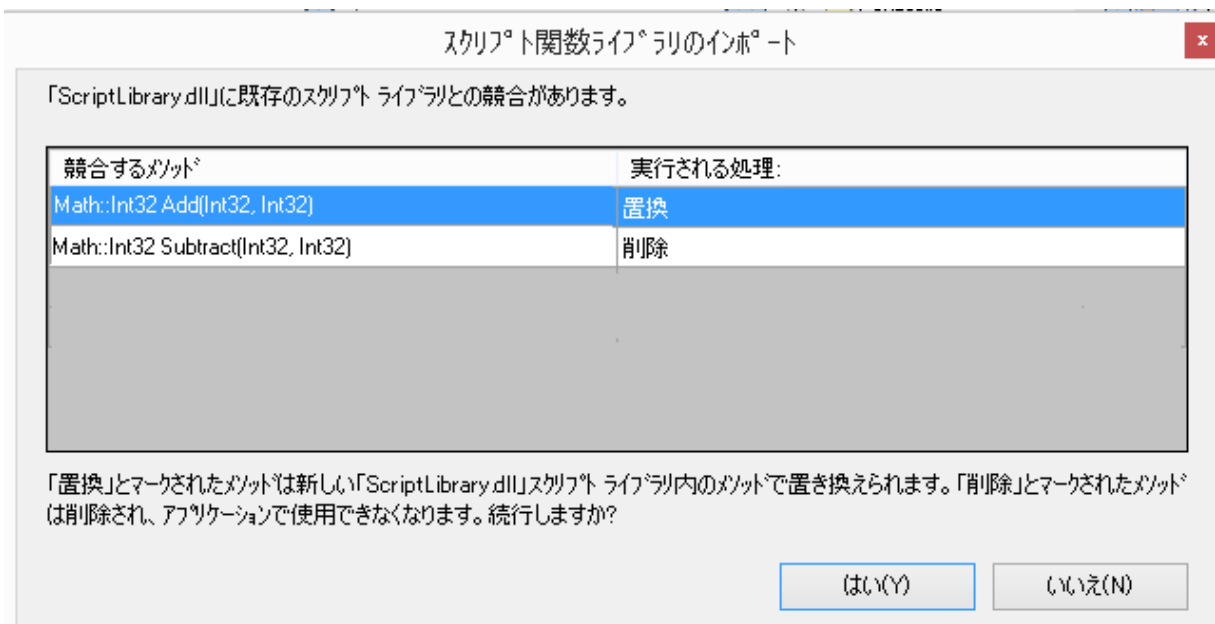
注記: インポート中は進捗状況を示すバーや進捗情報を出すウィンドウは表示されません。インポートが正常に完了すると、情報ウィンドウが開きます。

.NET スクリプト ライブラリ内の競合するメソッドのインポートの解決

.NET クラス スクリプト ライブラリをアプリケーションにインポートするとき、既存のスクリプト ライブラリはインポートするライブラリで置換されます。競合するスクリプト メソッドは、この時点で検出されます。競合検出は、名前空間、クラス名、メソッド名、およびパラメータ宣言に基づいて行われます。

注記: .dll のバージョンやファイル名はメソッド競合検出に影響しません。

インポート時に、競合するメソッドが [スクリプト関数ライブラリのインポート] ダイアログ ボックスに表示されます。



この例では、現在のライブラリに `Math::Int32 Add(Int32, Int32)` があり、インポートするライブラリのメソッドとして、同じクラス、メソッド名、およびパラメータが含まれています。この場合、「実行される処理」カラムに「置換」と表示されます。インポートを続行すると、インポートするライブラリによってアプリケーションのスクリプト ライブラリ全体が置換されます。

インポートするライブラリには `subtract` メソッドが含まれないので、`[Math::Int32 Subtract(Int32, Int32)]` は「削除」とマークされます。スクリプト メソッドの競合を解決するには、スクリプト ライブラリ全体を置換する必要があります。その結果、インポートするライブラリに含まれないメソッドが削除されることになります。

上記の例のように、既存のメソッドがライブラリから削除される結果となる場合、メソッドのインポートを個々にキャンセルできます。競合するすべてのメソッドを解決するか、インポートそのものをキャンセルする必要があります。

重要: インポート時に重複として検出できるのは、.NET クラス ライブラリ ファイルだけです。既存のライブラリ内のメソッドと競合した場合、.aaSLIB ライブラリおよび .wdf スクリプト拡張子ファイルはインポートされません。その場合、競合の通知は行われません。

アプリケーションのアプリケーション スタイル ライブラリの設定

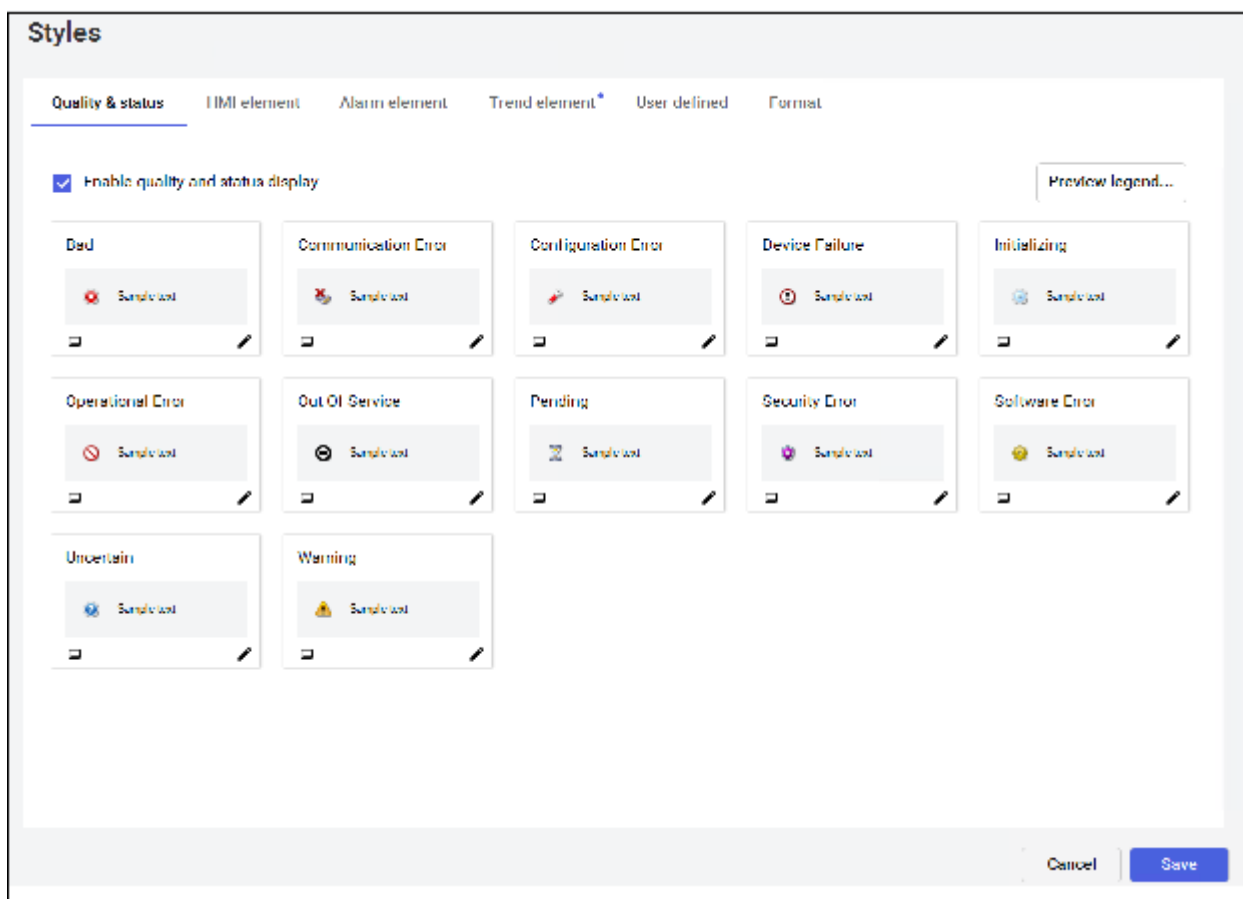
InTouch アプリケーションのグラフィックのスタイル ライブラリを設定できます。クォリティとステータス、要素スタイル、および数値フォーマット スタイルのアプリケーション スタイルを設定できます。設定の変更は、アプリケーションのリポジトリに保存されます。

- クォリティとステータス インジケータは、アプリケーション データの現在の品質、およびアプリケーション シンボルで表示される装置の状態を表すグラフィック アイコンです。
- 要素スタイルは、表示プロパティのセットを定義します。これらのプロパティは、産業グラフィックに表示されるテキスト、線、グラフィックの輪郭、および内部の塗りつぶしの外観を決定します。
- フォーマット スタイルは、産業用グラフィックで使用される一般的なタイプの数値のアプリケーション全体でのスタイルを個々に設定するオプションを提供します。

重要: このセクションでは、アプリケーションのスタイル ライブラリにアクセスするための WindowMaker 内のワークフローについて説明します。アプリケーション スタイルの編集の詳細については、WindowMaker のオンライン ヘルプまたは『産業用グラフィック エディタ ユーザー ガイド』を参照してください。

アプリケーション スタイル ライブラリを設定するには

1. WindowMaker でアプリケーションを開きます。
2. [ファイル] メニューの [設定] をクリックし、[スタイル] をクリックします。
[スタイル] 設定画面が表示されます。この画面には、クォリティとステータス インジケータ、グラフィック要素スタイル、および数値フォーマット スタイルを設定するタブがあります。



3. 編集するアプリケーションスタイルのタブを選択します。

注記: WindowViewer で一度に開くことのできるアプリケーションは1つだけです。プラットフォームがローカルノードに配置されている場合、他のスタンドアロンアプリケーションまたはマネージドアプリケーションで設定されているスタイルよりも **Galaxy** で設定したスタイルが優先されます。

アプリケーションスタイルライブラリのエクスポートとインポート

アプリケーションからアプリケーションスタイルライブラリをエクスポートして、別のアプリケーションにインポートできます。品質、要素スタイル、および数値形式の設定が XML ファイルにエクスポートされます。

アプリケーションからアプリケーションスタイルライブラリをエクスポートするには

1. WindowMaker を開きます。
2. [ファイル] メニューの [エクスポート] をクリックし、[スタイル] をクリックします。
ファイル名を指定するフィールドを含む [アプリケーションスタイルライブラリのエクスポート] ファイル参照画面が表示されます。
3. エクスポートされた XML ファイルを配置するフォルダを選択して、ファイルに名前を付けます。
4. [保存] をクリックします。

アプリケーション スタイル ライブラリが正常にエクスポートされたことを示すダイアログ ボックスが表示されます。

アプリケーションにアプリケーション スタイル ライブラリをインポートするには

1. WindowMaker を開きます。

2. [ファイル] メニューの [インポート] をクリックし、[スタイル] をクリックします。

ファイル名を指定するフィールドを含む [アプリケーション スタイル ライブラリのインポート] ファイル参照画面が表示されます。

3. インポートする XML ファイルが配置されているフォルダを選択します。そのファイル名が [ファイル名] フィールドに表示されます。

4. [開く] をクリックします。

アプリケーション スタイル ライブラリがインポートされたことを示すダイアログ ボックスが表示されます。

アプリケーションのアラーム優先度マッピングの設定

InTouch アプリケーションのアラーム優先度マッピングを設定して、各アラームの重要度の優先度範囲を設定します。

重要: ここでは、アラーム優先度範囲をアラーム重要度にマッピングするための WindowMaker 内のワークフローについて説明します。Application Server とは異なり、InTouch にはアラームの重要度の管理機能が組み込まれていませんが、ユーザーは InTouch タグを使用して、アラーム輪郭アニメーションを実装できます。その場合、ダイアログ ボックスでの優先度と重要度のマッピングは、優先度をアラーム輪郭の色とアラーム インジケータ アイコンに関連付けるための視覚的補助としてのみ使用されます。アラーム優先度のマッピングおよびアラームのシェルフの設定の詳細については、WindowMaker のオンラインヘルプまたは『産業用グラフィック エディタ ユーザー ガイド』を参照してください。

アプリケーションのアラーム優先度マッピングを設定するには

1. WindowMaker でアプリケーションを開きます。

2. [ファイル] メニューの [設定] をクリックし、[アラーム] をクリックします。

優先度の範囲を各アラームの重要度にマップするフィールドを含む [アラーム優先度] セクションが表示されます。この画面には、アラームの重要度に基づいてアラームのシェルフを有効にするためのフィールドもあります。

Alarm priority

Alarms

	Severity	Description	From Priority Range	To Priority Range	Shelve	Image	
	1	Critical	1	250	<input type="checkbox"/>		...
▶	2	High	251	500	<input type="checkbox"/>		...
	3	Medium	501	750	<input checked="" type="checkbox"/>		...
	4	Low	751	999	<input checked="" type="checkbox"/>		...

Modes

	Description	Image	
▶	Inhibited/Disabled		...
	Silenced		...
	Shelved		...

3. [優先度始点] フィールドと [優先度終点] フィールドをクリックして、1 ～ 999 の数字を入力し、各アラーム重要度のアラーム優先度範囲の下限と上限を設定します。

各優先度範囲は、優先度範囲間の重複なしで連続するものである必要があります。アラーム重要度 1 はデフォルトで優先度 1 で開始します。

4. [シェルフ] カラムで、各アラーム優先度のアラームのシェルフを有効にするチェック ボックスをオンまたはオフにします。
5. [OK] をクリックして、変更を保存します。

変更内容はアプリケーションのアプリケーション フォルダに保存されます。

アプリケーションから産業用グラフィック テキスト文字列のエクスポート

アプリケーションがランタイム時の言語切り替えをサポートすることを目的としている場合、その産業用グラフィックのテキスト文字列をディクショナリ ファイルにエクスポートできます。その後、テキストエディタ、XML エディタ、または Microsoft Excel や言語アシスタントなどのスプレッドシートプログラムを使用して、ディクショナリ ファイル内の文字列を別の言語に翻訳できます。

グラフィック テキスト文字列をエクスポートするとき、ディクショナリ ファイルの出力フォルダを指定する必要があります。文字列を別の言語に変換するディクショナリ ファイルごとに個別のフォルダを作成することをお勧めします。

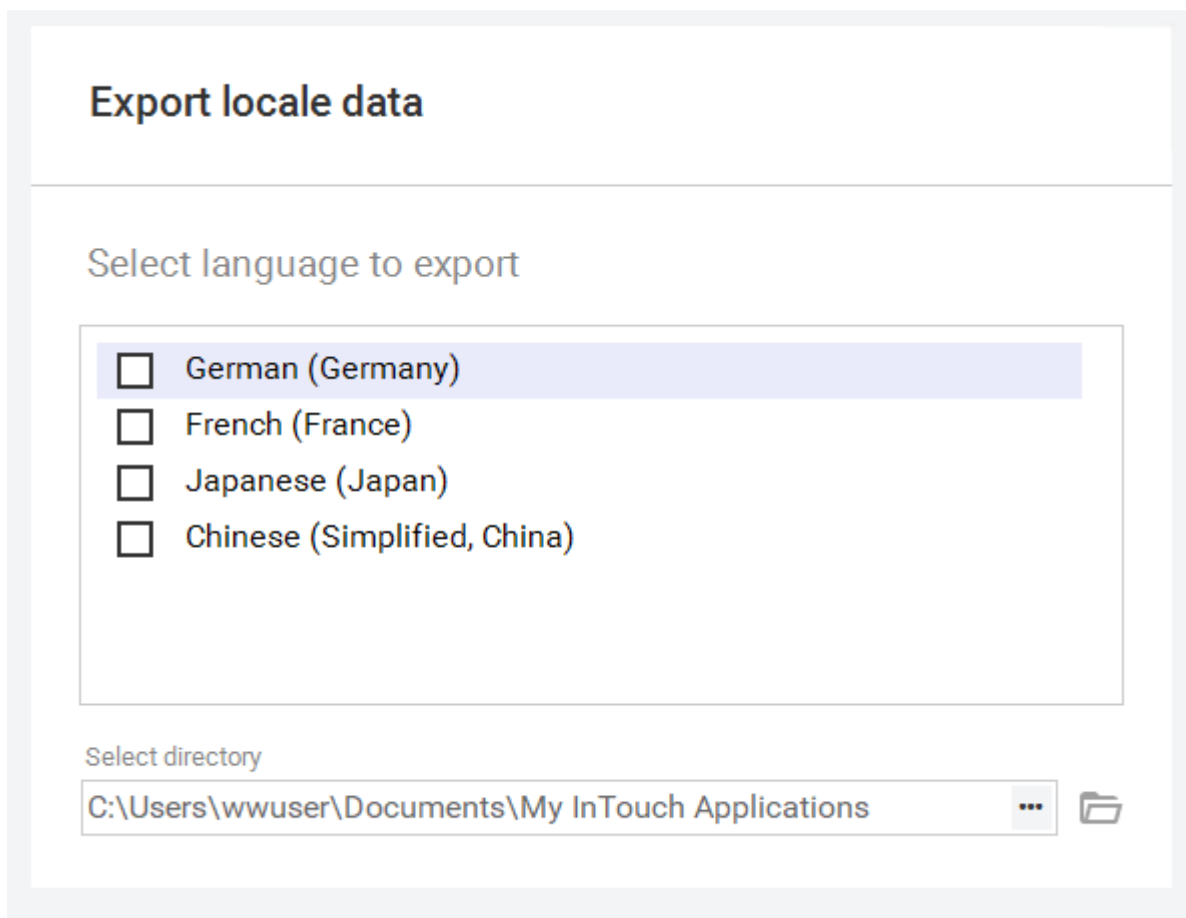
エクスポートしたすべてのディクショナリファイルには、<AppFolderName>「AA_<言語 ID>.xml」という名前が付けられます。たとえば、アプリケーションのフォルダ名が PumpStation で、エクスポートする言語がフランス語（言語 ID は 1036）の場合、ファイル名は PumpStationAA_1036.xml になります。

異なるときに異なるオブジェクトの言語文字列をエクスポートする場合、最初のエクスポートが後続のエクスポートによって上書きされるのを防止するために個別のターゲット フォルダを使用してください。

産業用グラフィック テキスト文字列をエクスポートするには

1. WindowMaker でアプリケーションを開きます。
2. [ファイル] メニューの [エクスポート] をクリックし、[ローカリゼーション]、[産業用グラフィックの翻訳] の順にクリックします。

[ロケールデータのエクスポート] 画面が表示されます。



Export locale data

Select language to export

- ☐ German (Germany)
- ☐ French (France)
- ☐ Japanese (Japan)
- ☐ Chinese (Simplified, China)

Select directory

C:\Users\wwuser\Documents\My InTouch Applications

3. エクスポートするシンボルテキスト文字列を設定します。

- **[エクスポートする言語]** リストで、エクスポートする言語ディクショナリのチェック ボックスをオンにします。デフォルトの言語はリストに表示されません。
- **[ディレクトリの選択]** フィールドにディクショナリ ファイルのエクスポート先のフォルダを入力します。

参照して既存のフォルダを選択することや新規フォルダを作成することもできます。

4. **[エクスポート]** をクリックします。

アプリケーションへの産業用グラフィックのテキスト文字列のインポート

シンボルテキストの場合、各言語の翻訳されたディクショナリ ファイルをインポートして、それらの言語のランタイム時の言語切り替えを有効にする必要があります。1つの言語のすべてのディクショナリ ファイルは同じフォルダに配置する必要があります。

一度にインポートできるファイルは1つの言語だけです。インポートするとき、目的の言語を選択して、インポートするディクショナリ ファイルを指定します。

翻訳されたディクショナリ ファイルをインポートするには

1. インポートする産業用グラフィック テキストを含むアプリケーションを開きます。

1. [ファイル] メニューの [インポート] をクリックし、[ローカリゼーション]、[産業用グラフィックの翻訳] の順にクリックします。
[ロケールデータのインポート] 画面が表示されます。

Import locale data

Select language to import

- ☒ German (Germany)
- ☐ French (France)
- ☐ Japanese (Japan)
- ☐ Chinese (Simplified, China)

Select directory

C:\Users\wwuser\Documents\My InTouch Applications

Select files to import

1. インポート設定を構成します。
 - [インポートする言語] リストで、インポートする言語ディクショナリのチェック ボックスをオンにします。
 - [ディレクトリの選択] ボックスで、インポートするディクショナリ ファイルを含むフォルダを指定します。
 - [インポートするファイルの選択] ボックスで、インポートする .xml ファイルを選択します。現在のアプリケーション フォルダ名および選択した言語のロケール ID を含むファイルだけが表示されます。
2. [インポート] をクリックします。

シンボルからのローカリゼーション文字列のエクスポート

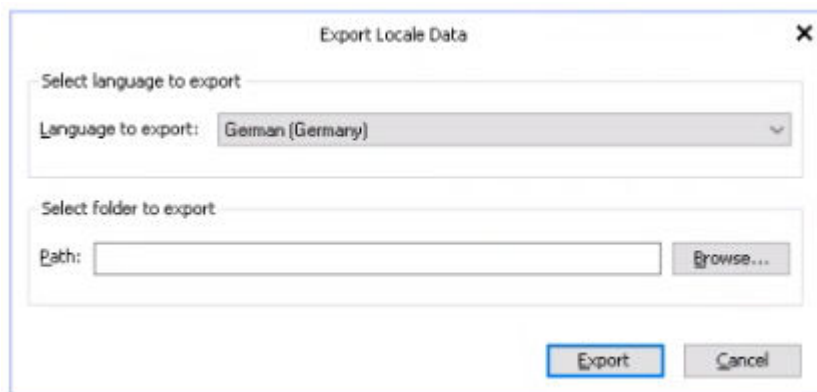
アプリケーションでランタイム時の言語切り替えをサポートする場合、産業用グラフィック ツールボックスで選択した 1 つまたは複数のシンボルのテキスト文字列をエクスポートできます。その後、テキストエディタ、XML エディタ、または Microsoft Excel などのスプレッドシート プログラムを使用して、ファイル内のエクスポート済み文字列を別の言語に翻訳できます。

シンボルからテキスト文字列をエクスポートするとき、出力フォルダを指定する必要があります。文字列をそれぞれ異なる言語に翻訳するファイルごとに個別のフォルダを作成することが推奨されます。

エクスポートしたすべてのローカリゼーション ファイルには、<AppFolderName>「AA_<言語 ID>.xml」という名前が付けられます。たとえば、アプリケーション フォルダの名前が PumpStation1 で、ローカリゼーション文字列の言語がメキシコのスペイン語（言語 ID = 2058）である場合、ファイル名は PumpStation1AA_2058.xml になります。

シンボルからローカリゼーション文字列をエクスポートするには

1. WindowMaker でアプリケーションを開きます。
2. エクスポートするローカリゼーション文字列を含むシンボルを産業用グラフィック ツールボックスで選択します。
 - 1 つのシンボルを選択するには、目的のシンボルをクリックします。
 - 複数のシンボルを選択するには、Ctrl キーを押しながら目的のシンボル名をクリックします。
 - 連続する範囲に含まれるすべてのシンボルを選択するには、範囲の最初または最後にあるファイルをクリックし、Shift キーを押しながら範囲の最後または最初にあるファイルをクリックします。
3. 選択したシンボルを右クリックしてショートカットメニューを表示します。
4. [エクスポート]、[ローカリゼーション]、[選択されたシンボル] の順に選択します。
[ロケールデータのエクスポート] ダイアログ ボックスが表示されます。



5. エクスポートするシンボル テキスト文字列を設定します。
 - [Languages to export (エクスポートする言語)] リストで、シンボルからエクスポートするローカリゼーション文字列を選択します。デフォルトの言語はリストに表示されません。
 - [パス] フィールドで、ローカリゼーション文字列をエクスポートするフォルダを入力します。
[参照] をクリックして、既存のフォルダを選択するか、新しいフォルダを作成します。

6. [エクスポート] をクリックします。エクスポート操作の進捗状況を示すバーが表示されます。
7. [詳細の表示] をクリックして、選択した各シンボル内のローカリゼーション文字列が正常にエクスポートされたことを確認します。

産業用グラフィック ライブラリのインポート

アプリケーションを開発する際、以下の条件が満たされていれば、産業用グラフィック ライブラリおよび **Situational Awareness Library** を標準のアプリケーションにインポートできます。

- アプリケーションが空白のテンプレートで作成されていて、産業用グラフィックや **Situational Awareness Library** が含まれていない。
- 古いスタンドアロンアプリケーションまたはモダン アプリケーションが移行されていて、ライブラリはインポートされていない。

アプリケーションに産業用グラフィック ライブラリをインポートするには:

- 産業用グラフィック ツールボックスで、アプリケーション名を右クリックして [産業用グラフィック ライブラリのインポート] をクリックします。

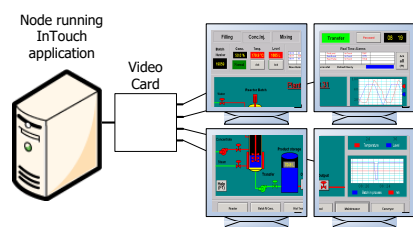
[産業用グラフィックのインポート] ダイアログが表示されます。先に産業用グラフィック ライブラリがインポートされ、その後に **Situational Awareness Library** がインポートされます。

処理が完了すると、産業用グラフィック ライブラリと **Situational Awareness Library** が産業用グラフィック ツールボックスに表示されます。

章 26 マルチ モニタ システムの設定

マルチ モニタ システムでは、複数のモニタ上に InTouch アプリケーションが同時に表示されます。また、マルチ モニタを設定すると、InTouch アプリケーションが実行されているコンピュータに接続されているすべてのモニタから構成される複合画面が作成されます。各モニタには、画面の一部、またはキーパッドのような単一のウィンドウ コンポーネントのみが表示されます。

InTouch アプリケーションの実行中、モニタ間でマウスを移動したり、あるモニタから別のモニタにウィンドウをドラッグしたりできます。また、以下の図のように、いくつかのマルチ モニタの設定では、すべてのモニタにかけて InTouch アプリケーション ウィンドウ全体を表示できます。



マルチ モニタの設定

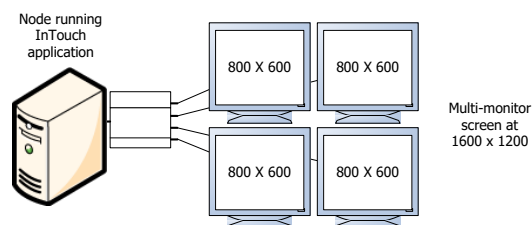
2 つの基本的なマルチ モニタの設定を使用できます。

- 単一のビデオ カード
- 複数のビデオ カード

各設定には、一意のハードウェア、ソフトウェア、および設定の必要条件があります。また、各設定では、異なるマルチ モニタ機能の設定がサポートされています。

単一のビデオ カードの設定

単一のビデオ カードの設定では、コンピュータには、モニタに接続された複数の出力ポートを持つ単一のビデオ カードがインストールされています。



複合画面の解像度は、各モニタの個々の水平および垂直の解像度の合計です。たとえば、一般的なビデオ カードでは、上下 2 台の構成のキューブとしてスタックされた 4 台の 17 インチ モニタを接続できます。上の図では、800 x 600 ピクセルの画面解像度で各モニタが実行されています。複合された仮想画面解像度は、1600 x 1200 ピクセルです。

単一のカードの設定の特性

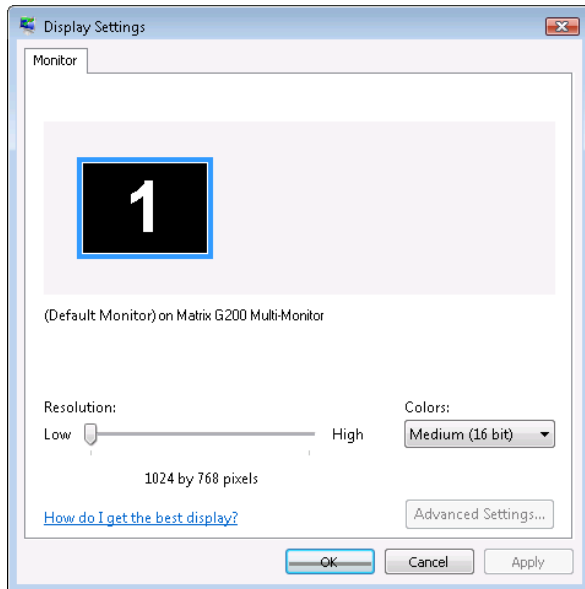
単一のビデオ カード ドライバには、次の特性があります。

- 単一のビデオ カードにより、すべてのモニタで同時に単一の大きな画面が作成されます。

- 接続されたすべてのモニタのプロパティは、単一の画面値を使用して設定できます。
- 複合画面には、すべてのモニタにわたって、設定の一番下の行に **Windows** タスクバーが表示されます。
- すべてのモニタに合わせて **Windows** アプリケーションを最大化できます。

単一のカード ドライバ特性

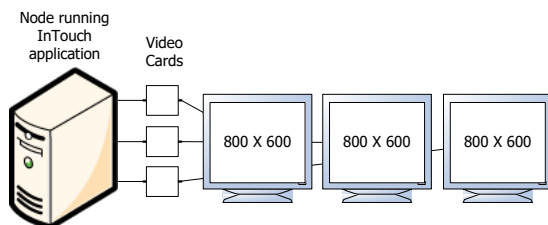
次の図は、複数の出力ポートを持つ単一のビデオ カードに接続されたすべてのモニタのドライバを設定するための、**Windows** の [画面のプロパティ] ダイアログ ボックスを示しています。



この図では、解像度の設定は、一列に並べられた 4 つのモニタ用です。各モニタの解像度は **1024 x 768** です。合計すると、複合画面の解像度は **4096 x 768** になります。設定する必要があるのは、単一のモニタの解像度、カラー深度、およびリフレッシュ レートだけです。解像度の設定は、単一のビデオ カードに接続されたすべてのモニタに適用されます。

複数のビデオ カードの設定

複数のビデオ カードの設定では、複数のビデオ カードがコンピュータにインストールされています。各ビデオ カードにより、**InTouch** アプリケーションが実行されているコンピュータが単一のモニタに接続されています。



複数のカードの設定の特性

ダイナミック レゾリューション機能（**DRC**）を他の分散機能と併用すると、画面解像度を自由に設定できるようになります。**NAD** アーキテクチャでは、開発ノードで **InTouch** アプリケーションの作成と管理

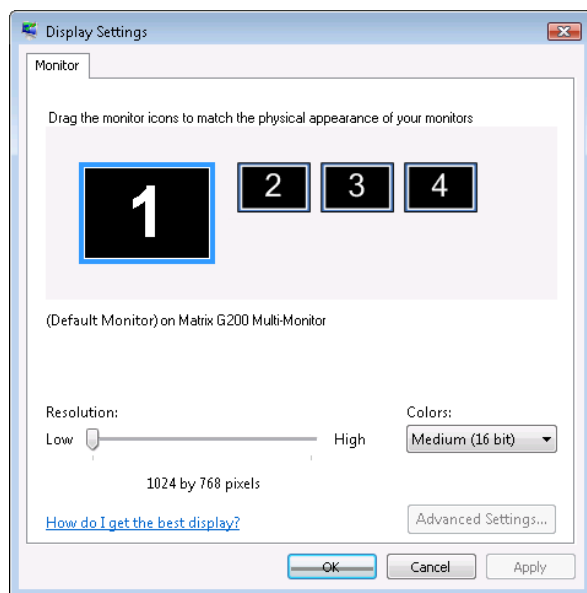
を行い、いくつかの表示ノードにコピーします。DRCを使用すると、異なる画面解像度でノードが実行されている場合でも、すべての表示ノードでアプリケーションを表示できます。

DRCを使用すれば、各表示ノードが、多くのユーザー定義オプション（解像度のカスタマイズも含む）に合わせてアプリケーションをスケーリング（拡大縮小）できます。このスケーリングは、WindowViewerがアプリケーションをコンパイルするときに行われるため、WindowMakerは必要ありません。各表示ノードで異なる DRC 設定を使用できるため、個々の表示ノードを設定する必要があります。

DRCにより、マルチ モニタ システムをサポートしやすくなります。複合画面全体またはその一部に InTouch アプリケーションを表示するよう、DRC 解像度の変換オプションから選択するだけです。

複数のカード ドライバの特性

以下の図は、InTouch アプリケーションが実行されているコンピュータにインストールされた個々のビデオカードに接続されたすべてのモニタのドライバを設定するための、Windows の「画面のプロパティ」ダイアログ ボックスを示しています。



「画面のプロパティ」ダイアログ ボックスの番号の付いた四角形をクリックして、設定するモニタを選択します。モニタの物理的な場所に一致するように、番号の付いた四角形を並べます。画面解像度、カラー深度およびリフレッシュ レートは、選択したモニタにのみ適用されます。

マルチ モニタ アプリケーションの計画

アプリケーション用に複数のモニタを設定するには、次の操作を行う必要があります。

- マルチ モニタ ビデオ カードの選択
- アプリケーションの画面解像度の決定
- アプリケーションを表示するモニタの数の決定
- アプリケーション ウィンドウの場所の決定

マルチ モニタ ビデオ カードの選択

テクニカルサポートでは、マルチ モニタ InTouch アプリケーションがサポートされている推奨ビデオ カードのリストを提供しています。

ビデオ カードを選択する前に、次の質問の回答に関する情報をテクニカル サポートから取得してください。

- どのバージョンの InTouch でビデオ カードがサポートされているか?
- カードでは、単一または複数のカードの設定がサポートされているか?
- ビデオ カードの推奨ドライバは?
- 推奨するビデオ カードの設定は?

アプリケーションの画面解像度の決定

全体の画面解像度を決定し、表示領域の正確なサイズを知っておくと、マルチ モニタ環境でアプリケーションを作成するプロセスを簡素化できます。

モニタ設定全体を表示する描画を作成します。描画には、各モニタの解像度およびすべてのモニタの解像度の合計が表示される必要があります。この描画には、各モニタの水平と垂直のピクセル範囲が表示されます。

たとえば、800 x 600 の画面解像度を持つ 2 つの水平モニタから構成される複合画面がある場合、2 番目のモニタの左上のピクセルの位置はピクセル 800 x 0 になります。最初のモニタの画面のピクセル数は 0 から 799 になり、2 番目のモニタでは 800 から 1599 になります。描画を参照しながら、複合マルチ モニタ画面のアプリケーション ウィンドウの場所を決定できます。

アプリケーションを表示するモニタの数の決定

生産環境に似た開発環境を使用すると、マルチ モニタ InTouch アプリケーションの作成を簡素化できます。すべての場合で、マルチ モニタ開発環境を使用できるとは限りません。InTouch アプリケーションの開発に使用するコンピュータに接続されているモニタが単一である場合でも、ウィンドウを開発し、表示に必要なウィンドウ サイズおよび位置を設定することにより、マルチ モニタ アプリケーションを構築できます。

WindowMaker のプロパティ ペインを使用して、ウィンドウの特性を変更します。[ウィンドウ] ペインで、変更するウィンドウを選択します。右側のナビゲーション ペインにウィンドウのプロパティ ペインが表示されます。

Name	:	Window_001
Comment	:	
Window type		Replace
▼ Location		4, 4
	X	4
	Y	4
▼ Size		1920, 1037
	Width	1920
	Height	1037
Window color	:	<input type="checkbox"/> White
Titlebar		<input checked="" type="checkbox"/>
Frame style		Single
Close button		<input checked="" type="checkbox"/>
Size controls		<input checked="" type="checkbox"/>
Template		<input type="checkbox"/>

【X座標】および【Y座標】の値により、画面上のウィンドウの左上隅の水平および垂直のピクセル位置が決まります。水平および垂直のピクセルのスケールの基点は、画面の左上隅です。

【ウィンドウ幅】および【ウィンドウ高さ】の設定により、ウィンドウ全体のサイズが決まります。たとえば、次のようにウィンドウを設定できます。

- X座標 = 1024
- Y座標 = 0
- ウィンドウ幅 = 1024
- ウィンドウ高さ = 768

マルチモニタの設定は、水平に一行に並べられた4つのモニタから構成されます。各モニタの解像度は1024 X 768です。全体的な複合画面解像度は、4096 x 768です。

ウィンドウの水平基点を1024に設定し、垂直基点を0に設定すると、ランタイム時にこのウィンドウを2番目のモニタに表示できます。ウィンドウが2番目のモニタの画面全体に表示されます。

アプリケーション ウィンドウの場所の決定

マルチモニタ環境の InTouch ウィンドウを開発する場合、いくつかの異なる設定を使用できます。

ウィンドウの表示位置の強制

ウィンドウを開発し、指定した位置に表示するように強制するのが 1 つの方法です。すべてのモニタの表示領域全体にわたって **WindowViewer** が最大化されるようにしてください。これにより、指定したモニタに **InTouch** アプリケーション ウィンドウを表示できます。

InTouch セキュリティ機能を使用すると、**Windows** デスクトップへのアクセスを拒否できます。

ウィンドウが手動で移動可能

別のオプションとしては、選択したモニタにウィンドウを手動で移動し、単一のアプリケーションを異なるモニタ設定で実行できるようにアプリケーションを開発する方法があります。この場合、次の点に留意してください。

- アプリケーション内のすべてのウィンドウは、**[ポップアップ]** タイプである必要があります。
- メインの **WindowViewer** の親ウィンドウは、すべてのモニタを表示しないように小さくすることもできます。ただし、この設定では **InTouch** が最大化されていないため、**InTouch** セキュリティを使用して **Windows** デスクトップへのアクセスを拒否することはできません。

この設定では、メインの **WindowViewer** の親ウィンドウの位置にかかわらず、どのモニタにも簡単に移動できるポップアップ ウィンドウが使用されます。ポップアップ ウィンドウは、**WindowViewer** の親ウィンドウ内に残る必要はありません。メイン ウィンドウのサイズを縮小してモニタの隅に移動すると、選択したモニタにすべてのポップアップ ウィンドウを自由に移動できます。

環境に基づいてウィンドウを自動的に配置

最後に、上記の方法に手順を追加した方法について説明します。この手順では、使用されている環境に基づいて、アプリケーションで自動的にウィンドウを配置できます。これは最も複雑な設定であり、大規模なスクリプト記述および計画が必要です。

この設定では、標軸および計算のデフォルト設定に基づいて、**ShowAt()** および **ShowAtTopLeft()** のスクリプト関数によりウィンドウが動的に配置されます。これは、アプリケーションの必要条件に応じて、さまざまな方法で設定できます。

マルチ モニタ InTouch アプリケーションの開発

マルチ モニタをサポートするには、**InTouch.ini** ファイルおよび **Win.ini** ファイル内の選択したパラメータに値を割り当てる必要があります。これらのパラメータを使用すると、**InTouch** システム ダイアログ ボックスおよびキーパッドを複合画面の適切な位置に配置できます。

マルチ モニタ パラメータの設定

マルチ モニタ サポートを有効にするには、一連の **InTouch** パラメータを **Windows** の **Win.ini** ファイルに追加します。これらのパラメータにより、**InTouch** アプリケーションが実行されているノードおよび各モニタの解像度のマルチ モニタ サポートが有効になります。

ノード上でマルチ モニタの設定を行うには

1. **InTouch HMI** ソフトウェアが実行されているコンピュータの **Windows** フォルダにある **Win.ini** ファイルを編集します。
2. **Win.ini** ファイル内の **[InTouch]** セクションを検索し、次のパラメータを追加します。

パラメータ	説明
MultiScreen=1	値 1 の場合、マルチ モニタ モードが有効になります。値 0 の場合、マルチ モニタ モードが無効になります。
MultiScreenWidth=nnnn	単一の画面の幅（ピクセル）
MultiScreenHeight=nnnn	単一の画面の高さ（ピクセル）

たとえば、2 つの水平モニタ上に 2560 x 1024 の画面解像度で InTouch アプリケーションを表示する場合、次のように入力します。

```
[InTouch]
MultiScreen=1
MultiScreenWidth=1280
MultiScreenHeight=1024
```

画面解像度の変換の設定

異なる画面解像度で実行されているノード間で移行する場合、パラメータ値を指定して InTouch アプリケーション ウィンドウの現在の解像度を保守できます。

ScaleForResolution パラメータ値により、WindowViewer が実行されているコンピュータで表示解像度を変更された後に、アプリケーション ウィンドウ (*.win) が WindowMaker によって自動的にスケールされるかどうかが決まります。ScaleForResolution パラメータは、WindowViewer ダイアログボックスの解像度には影響しません。

ノード上の画面解像度の変換を設定するには

1. InTouch が実行されているコンピュータの InTouch.ini ファイルを編集します。
2. ScaleForResolution パラメータをファイルに追加します。

```
ScaleForResolution=1
```

0 に設定すると、解像度の変換が無効になります。

1 に設定すると、解像度の変換が有効になります。

注意： ScaleForResolution パラメータが InTouch.ini ファイルに追加されない場合、デフォルト値が有効になります (ScaleForResolution=1)。パラメータを無効にすると (ScaleForResolution=0)、解像度を変換するよう要求されます。しかし、解像度の変換は発生しません。

アプリケーションの配置およびマルチ モニタの設定の確認

マルチ モニタ システムで実行されるアプリケーションを単一のモニタ システム上で開発している場合、ScaleForResolution パラメータは特に重要になります。ScaleForResolution パラメータに割り当てられた値により、ある環境から別の環境に移動したときに、アプリケーションを調整できるかどうかが決まります。

重要： 異なる環境に移動する前に、アプリケーションのコピーをバックアップしておくことをお勧めします。

たとえば、4 つモニタを一行に並べた設定を持つシステムで実行されるアプリケーション（全体の解像度は 4096 x 768）を、解像度 1024 x 768 の単一のモニタを持つコンピュータで開発した場合、アプリケーションの変換が必要です。

マルチ モニタ システムにアプリケーションを配置すると、アプリケーションの変換を要求するメッセージが表示されます。

ScaleForResolution が .ini に設定されている場合、このメッセージは表示されますが、アプリケーションは変換されずに設計したとおりに実行できます。起動を続行するには、[はい] をクリックします。

.ini が設定されていない場合、アプリケーション内のすべてのグラフィックおよびウィンドウが InTouch HMI によって新しい解像度に変換され、スケールされます。そのため、すべてのウィンドウおよびグラフィックの表示が拡大され、予期しない表示になる場合があります。

重要：アプリケーションを実行する前に、移行先のコンピュータでもマルチ モニタの Win.ini パラメータが設定されていることを確認してください。Win.ini の設定は、InTouch アプリケーションと一緒に自動的に転送されません。

ランタイム中のマルチ モニタ サポートの確認

InTouch アプリケーションが実行されるローカル ノードでマルチ モニタがサポートされているかどうかを確認するために、テクニカル サポート スクリプト ライブラリから、オプションのスクリプト関数をダウンロードできます。

WWMultiMonitorNode() 関数により、ノードでマルチ モニタがサポートされるかどうか、およびノードに接続されるモニタの数が決まります。

通常、InTouch アプリケーションが実行されるノードに割り当てられたモニタの数を決定するために、QuickScript から WWMultiMonitorNode() 関数を実行します。

次の例では、InTouch の整数型タグ変数に割り当てられた WWMultiMonitorNode() 関数の値を持つ、QuickScript ステートメントの例を示しています。WindowViewer でアプリケーションが起動したときに、QuickScript が実行されるように設定できます。

```
{整数型タグ変数として定義された MultiMonitor}  
MultiMonitors = WWMultiMonitorNode();  
{この関数の実行後 Result = 4}
```

WWMultiMonitorNode() は、ノードの Win.ini ファイルで指定された MultiScreen パラメータを読み取ります。WWMultiMonitorNode() 関数は、0 または正の整数のどちらかを返します。

- 0 の返り値

MultiScreen=0 の場合、または Win.ini ファイルの [InTouch] セクションで MultiScreenWidth パラメータまたは MultiScreenHeight パラメータが誤って 0 に設定されている場合、WWMultiMonitorNode() は 0 を返します。

- 正の整数の返り値

MultiScreen=1 の場合、および MultiScreenWidth パラメータおよび MultiScreenHeight パラメータに正しい画面解像度の値が割り当てられている場合、WWMultiMonitorNode() はマルチ モニタの設定のモニタの数を返します。

章 27 タブレット PC での InTouch の使用

Windows XP Tablet PC Edition および InTouch は、一連のポータブルタブレット PC にプレインストールされています。これらの丈夫なタブレット PC は、防水処理されており、振動耐性を持ち、多くの産業環境に適しています。InTouch アプリケーションを実行できるタブレット PC は、他のコンピュータ製造業者からも入手可能です。

オペレータはタブレット PC を持ち運んで、工場内を移動することができます。タブレット PC では、工場での実際の処理を表す InTouch アプリケーションが実行されます。画面ポインタまたは入力デバイスとして機能するペンを使用すると、画面上の InTouch オブジェクトを選択したり、キーボードの代わりに画面上に直接メモを記述したりすることができます。

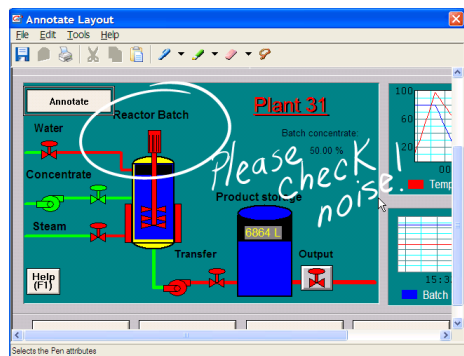


オペレータは、工場での実際の処理を直接観察し、実行中の InTouch アプリケーションにメモおよび注釈を書き込むことができます。

ビジュアル化画面に注釈を付け、電子メール メッセージで送信する

タブレット PC に表示される画面をキャプチャするには、AnnotateLayout() スクリプト関数を使用します。Windows XP Tablet PC オペレーティング システムを使用しているタブレット PC で InTouch が実行されている場合のみ、AnnotateLayout() 関数を使用できます。

AnnotateLayout() 関数は、アクティブな InTouch ウィンドウの表示されている部分の画面をキャプチャします。キャプチャされた画面は、[Annotate Layout] ダイアログ ボックスに表示されます。



[Annotate Layout] ダイアログ ボックスには、ツールバーおよびメニュー オプションが含まれています。ダイアログ ボックスでは、キャプチャされた画面がクライアント領域に表示されます。さまざまな描画ツールを使用して画像に注釈を付けたり、キャプチャされた画面を保存、印刷したり、または電子メール メッセージとして送信することができます。

ウィンドウの注釈の作成

ウィンドウに注釈を付けるには、次のツールを使用します。

- ペン：描画したりコメントを記述する。



- 蛍光ペン：半透明色を使用して、ウィンドウの領域をハイライトする。



- 消しゴム：注釈の一部を削除する。



これらの各ツールには、サイズ、色または透明度などのオプションがあります。

- これらのオプションを設定するには、各ツールのアイコンの横にある下矢印をクリックし、次にオプションのコマンドをクリックします。
- これらのオプションをデフォルト設定に復元するには、[ツール] メニューで、[デフォルトにリセット] をクリックします。

ウィンドウの注釈の選択、コピー、および削除

ウィンドウで作成した注釈を選択、コピー、および削除できます。

注釈を選択するには



1. ツールバーで [選択ツール] アイコンをクリックします。
2. スタイラス ボタンを押したまま、選択する注釈の周りの領域を描画します。

選択した注釈の切り取り、コピー、または削除を行うことができます。

注釈の切り取り、コピー、および貼り付けを行うには

- Windows 標準の [切り取り]、[コピー]、および [貼り付け] のコマンドを使用します。

注釈を削除するには

- 以下のいずれかを実行します。
 - ウィンドウのすべての注釈を削除するには、[編集] メニューで、[クリア] をポイントして、次に [すべて] をクリックします。
 - 選択ツールを使用して選択した注釈を削除するには、[編集] メニューで、[クリア] をポイントして、次に [選択] をクリックします。

注釈の付いたウィンドウの保存、印刷、電子メールによる送信

ウィンドウに注釈を付けた後、画像ファイルとして保存したり、印刷したり、または電子メールの添付ファイルとして送信することができます。

電子メール サーバーを一度設定するだけで、これを行うことができます。

注釈の付いたウィンドウを保存するには

1. [ファイル] メニューで、[保存] をクリックします。Windows 標準の [名前を付けて保存] ダイアログ ボックスが表示されます。
2. 名前およびファイル形式を入力して、[OK] をクリックします。

注釈の付いたウィンドウを印刷するには

1. [ファイル] メニューで、[印刷] をクリックします。Windows 標準の [印刷] ダイアログ ボックスが開きます。
2. 印刷オプションを指定して、[OK] をクリックします。

注釈の付いたウィンドウを電子メールの添付ファイルとして送信するには

1. [編集] メニューで、[電子メール設定] をクリックします。[電子メール設定] ダイアログ ボックスが表示されます。
2. 電子メールを送信するために使用する SMTP 電子メール サーバーのホスト名を入力します。不明な場合は、管理者に連絡してください。[OK] をクリックします。
3. [ファイル] メニューで、[電子メール] をクリックします。[E-mail] ダイアログ ボックスが表示されます。
4. 送信者および受信者のアドレスを入力して、メッセージを記述します。注釈の付いたウィンドウの画像ファイルが添付ファイルとして自動的に追加されます。
5. [送信] をクリックして、電子メール メッセージを送信します。

AnnotateLayout() 関数

[Annotate Layout] ダイアログ ボックスが表示され、このスクリプト関数が呼び出された現在の表示画面に注釈を付けることができます。この関数は、Windows XP Tablet PC Edition オペレーティング システムでのみサポートされています。

カテゴリ

システム

構文

```
AnnotateLayout()
```

備考

[Annotate Layout] ダイアログ ボックスが表示されると、WindowViewer の画面の画像がキャプチャされます。次の場合にこのダイアログ ボックスを使用します。

- ツール バーおよびメニュー項目の設定とペンを組み合わせて使用して、キャプチャされた画面に注釈を付ける場合
- .gif ファイルまたは .jpeg ファイルとして画像および注釈を保存する場合
- 画像および注釈を印刷する場合（プリンタが設定されている場合）
- 電子メール メッセージの添付ファイルとして画像および注釈を送信する場合（SMTP が設定されている場合）

画面の向きの変更

タブレット PC がタブレット設定で実行されていて、画面の解像度に対してアプリケーションの解像度が動的に変更するように **WindowViewer** を設定している場合、横方向モードで開発された InTouch アプリケーションは縦方向モードに合うように調整されます。

アプリケーションの解像度が動的に変更するように **WindowViewer** を設定していない場合、横方向のアプリケーションは調整されません。この場合、InTouch ウィンドウの一部がタブレット PC 上に表示されません。

ある設定から別の設定に切り替えると、デフォルトの値を使用して画面の解像度が切り替わります。たとえば、ラップトップの設定で実行されているタブレット PC がタブレット設定に切り替わる場合、横方向モード（1024 x 768）から縦方向モード（768 x 1024）に画面の向きが切り替わります。

章 28 InTouch サービスの管理

サービスとは、ユーザー インターフェイスを使わずに、またはユーザーがログオンしなくても、特定のシステム機能をバックグラウンドで自動的に実行する **Windows** のプロセスのことです。

Windows サービスでは、以下のスタートアップ オプションを使用できます。

- **自動。** Windows が再起動すると、サービスが自動的に開始されます。ユーザーが操作する必要はありません。
- **手動。** ユーザーまたはアプリケーション プロセスによって、明確にサービスの開始が指定されなければなりません。
- **無効。** サービスは開始されません。これはトラブルシューティングの際に便利です。

注記: InTouch WindowViewer サービスのパラメータ オプションはサポートされません。

Windows セキュリティ システムを犠牲にすることなく、サービスが開始されます。

InTouch HMI には、以下の Windows サービスが含まれています。

- Alarm DB Logger
- Alarm DB Purge/Archive
- NetDDE Helper
- SuiteLink
- WindowViewer

InTouch サービスの管理の概要

サービスとは、ユーザー インターフェイスを使わずに、またはユーザーがログオンしなくても、特定のシステム機能をバックグラウンドで自動的に実行する **Windows** のプロセスのことです。

Windows サービスでは、以下のスタートアップ オプションを使用できます。

- **自動。** Windows が再起動すると、サービスが自動的に開始されます。ユーザーが操作する必要はありません。
- **手動。** ユーザーまたはアプリケーション プロセスによって、明確にサービスの開始が指定されなければなりません。
- **無効。** サービスは開始されません。これはトラブルシューティングの際に便利です。

注意： InTouch WindowViewer サービスのパラメータ オプションはサポートされません。

Windows セキュリティ システムを犠牲にすることなく、サービスが開始されます。

InTouch HMI には、以下の Windows サービスが含まれています。

- Alarm DB Logger
- Alarm DB Purge/Archive
- NetDDE Helper
- SuiteLink

- WindowViewer

サービスとしての WindowViewer の実行

サービスとして実行されるように WindowViewer を設定した場合、アプリケーションがインストールされているコンピュータが起動したときに、WindowViewer が自動的に起動します。WindowViewer サービスはバックグラウンドで動作します。WindowViewer サービスが実行している場合、WindowViewer の別のインスタンスを開始することはできません。

WindowViewer をサービスとして実行すると、以下のようなメリットがあります。

- ほとんどの障害回復プランでは、停電が回復した後ですぐに主要のコンピュータ システムが起動することが要求されます。Microsoft Windows Server は、停電が回復した後、自動的に再起動できます。WindowViewer がサービスとして実行されていると、工場の自動化システムはすぐに実行を開始できます。WindowViewer で最後に開かれた InTouch アプリケーションは、コンピュータが再起動したときに自動的に起動します。
- WindowViewer は履歴データのログを続行し、アラーム情報を収集し、スクリプトを処理し、I/O サーバーとして機能し、別のオペレータがログオンおよびログオフしたときでも I/O クライアントとして値を書き込みます。

注記: ネットワーク アプリケーションをサービスとして実行する場合、またはネットワーク パスを履歴ログ フォルダとして使用する場合、ログオンユーザーは、ネットワークの場所への適切なアクセスが必要です。

WindowViewer がすでにサービスとして実行されている場合に、ショートカットアイコンから WindowViewer を起動しようとしたり、Windows の [スタート] メニューから [WindowViewer] をクリックして起動しようとしたりすると、Operations Control Log Viewer にメッセージが記録されます。メッセージには、サービスとして実行するよう設定された場合に WindowViewer を起動する際の制限に関する説明が表示されます。

WindowViewer がサービスとして既に実行しているときにアプリケーション マネージャまたは WindowMaker を起動しようとする、警告メッセージが Operations Control Log Viewer に記録されます。メッセージは、WindowViewer をサービスとして実行するときはアプリケーション マネージャおよび WindowMaker を開くことができない旨が示されます。

重要: WindowViewer をサービスとして実行する場合、管理者特権でサービスを実行する際に生じる潜在的なセキュリティの問題を削減するために、ユーザー アカウントの権限は非対話型に設定されます。

サービスとして起動するための WindowViewer の設定

Windows サービスとして WindowViewer を実行すると、オペレータのログオフ後の継続稼働や、システム ブート時のオペレータを介さない自動起動などの機能を使用できます。これによって、オペレーティング システムのセキュリティを犠牲にすることなく、無人で WindowViewer を起動できます。

サービスとして起動するように WindowViewer が設定されている場合、InTouch アプリケーションもサービスとしての WindowViewer 内で実行するよう指定する必要があります。アプリケーション ディレクトリは、[ノードのプロパティ] ダイアログ ボックスで指定するか、WIN.INI ファイルに手動で入力することができます。

サービスとして起動するように WindowViewer を設定するには

1. InTouch アプリケーション マネージャを起動します。
2. [ツール] メニューで、[ノードのプロパティ] をクリックします。

[ノードのプロパティ] ダイアログ ボックスが表示されます。

Node properties

App development Resolution Memory settings Performance Security

☐ None

☐ Start following application in WindowViewer as a service

Application path: C:\ProgramData\InTouchDemos\demoapp1_1280

☒ Enable network application development

Network application development

Local working directory: C:\Users\wwuser\AppData\Local\NAD

Polling period: 10 sec

Change mode

☐ Ignore changes

☐ Restart WindowViewer

☒ Prompt user to restart WindowViewer

☐ Load changes into WindowViewer

☐ Prompt user to load changes into WindowViewer

Cancel Ok

3. [以下のアプリケーションをサービスとして **WindowViewer** で起動] を選択して、サービスとして自動的に実行するよう **WindowViewer** を設定します。

[アプリケーションパス] フィールドが有効になります。

4. 省略記号ボタンをクリックしてファイル エクスプローラを表示し、目的の **InTouch** アプリケーションを選択します。

アプリケーション ディレクトリがグループ ボックスに入力されます。

5. [OK] をクリックします。

6. アプリケーション マネージャのツールボックスで [WindowViewer] アイコンをクリックします。

WindowViewer が、指定した **InTouch** アプリケーションのサービスとして実行するようになります。

注記: 上記の手順に従って[ノードのプロパティ]を設定している場合、WindowMaker から WindowViewer にすばやく切り替えて、InTouch アプリケーション用に WindowViewer サービスを起動することもできます。この操作は、アプリケーション マネージャから WindowViewer を起動するときに行うことができます。

WindowViewer でアプリケーションをサービスとして実行するための WIN.INI の編集

【ノードのプロパティ】で【以下のアプリケーションをサービスとして **WindowsViewer** で起動】オプションが有効な場合、アプリケーションディレクトリを WIN.INI ファイルに手動で入力できます。アプリケーション マネージャでアプリケーションを選択する前に WIN.INI ファイルを更新する場合、WindowViewer は、選択した後にアプリケーションのサービスとして実行します。

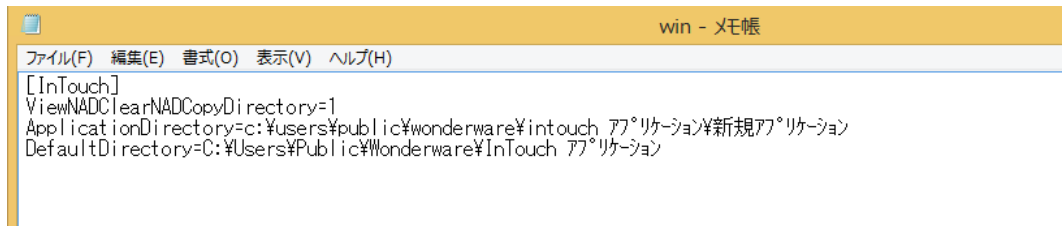
WindowMaker で開いたアプリケーションで WIN.INI を更新することもできます。その後にランタイムをすばやく切り替えると、WindowViewer はアプリケーションのサービスとして実行します。

注記: 上記の機能はマネージド InTouch アプリケーションではサポートされていません。WindowMaker からマネージドアプリケーションを切り替えて WindowViewer のサービスとして実行しようとすると、警告メッセージが **Logger** に記録されます。

WIN.INI は次の場所にあります。

C:\ProgramData\Wonderware\InTouch\Service\win.ini

サービスとして実行するアプリケーションのディレクトリを入力します。次の例を参照してください。



手動によるサービスの起動

Windows のコントロールパネルを使用して、InTouch WindowViewer サービスを手動で起動できます。

サービスとして起動するよう設定しない限り、サービス コントロール パネルには WindowViewer が表示されません。詳細については、「[サービスとして起動するための WindowViewer の設定](#)」を参照してください。

コントロールパネルを使用して **WindowViewer** サービスを起動するには

1. コントロール パネルを起動します。
2. [管理ツール] をダブルクリックし、次に [サービス] をダブルクリックします。[サービス] ダイアログ ボックスが表示されます。
3. 詳細ペインで、[Wonderware WindowViewer] サービスを右クリックし、次に [開始] をクリックします。

重要: コマンドプロンプトを使用して WindowViewer をサービスとして起動することはできません。

サービスの停止

コントロールパネルを使用して、WindowViewer サービスを手動で停止できます。

コントロールパネルを使用して **WindowViewer** サービスを停止するには

1. コントロール パネルを起動します。

2. [管理ツール] をダブルクリックし、次に [サービス] をダブルクリックします。[サービス] ダイアログ ボックスが表示されます。
3. 詳細ペインで [WindowViewer] を右クリックし、次に [サービスの停止] をクリックします。

InTouch サービスのユーザー アカウントの設定

デフォルトでは、Windows サービスはローカル システム アカウントを使用して実行されます。InTouch サービスには管理特権を持つユーザー アカウントが必要ですが、このようなユーザー アカウントはローカル システム アカウントでは提供されていない場合があります。

InTouch HMI をインストールするとき、すべての AVEVA サービスを実行する管理者のアカウントを指定します (アカウントがまだ作成されていない場合)。このアカウントは、マスタ アカウントと見なされます。InTouch サービスでは、自動的に起動するためにマスタ アカウントが使用されます。

注記: マスタ アカウントは偽装アカウントとも呼ばれます。偽装アカウントとは、サイトやサーバーの制限付きリソース「領域」へのアクセス権を持ったユーザーまたはグループアカウントのことです。

マスタ アカウントを変更するには、Change Network Account ユーティリティを使用します。

注意: マスタ アカウントの変更は、InTouch サービスだけではなく、すべての AVEVA サービスに影響します。

マスタ アカウントを変更するには

1. [スタート] メニューの [プログラム]、[AVEVA] の順にポイントし、[ネットワーク アカウントの変更] をクリックします。[ネットワーク アカウントの変更] ダイアログ ボックスが表示されます。
2. ユーザー アカウントを変更します。詳細については、ネットワーク アカウントの変更のマニュアルを参照してください。
3. [OK] をクリックします。

InTouch サービスのトラブルシューティング

あるサービスの起動が他のサービスの起動に依存している場合、そのサービスを起動する前に必要なサービスが実行されているかどうか Windows によって確認されます。

WindowViewer を実行するための必要条件に基づいて、以下の依存関係を把握する必要があります。

- Distributed Alarming または Distributed History を使用する場合、およびネットワーク DDE データにアクセスする場合、NetDDE Helper サービスが実行している必要があります。

また、NetDDE Helper サービスを使用するには、Network DDE および Network DDE DSDM の両方のサービスが手動または自動スタートアップでインストールおよび構成されている必要があります。インストール中、NetDDE Helper サービスが [手動] スタートアップに設定されます。つまり、コンピューターが起動するときに、WindowViewer でこのサービスが自動的に起動されます。

- WindowViewer を SuiteLink サーバーまたはクライアントとして使用する場合、SuiteLink サービスが実行している必要があります。

SuiteLink サービスを使用するには、Microsoft TCP/IP がインストールされている必要があります。

- WindowViewer が実行している間にメッセージまたはエラーを保存するには、Operations Control Log Viewer サービスがインストールされていることを確認する必要があります。

SuiteLink と Operations Control Log Viewer の両方のサービスは、自動スタートアップでインストールおよび設定してください。

サービスのエラー メッセージの表示

サービスに関連するエラー メッセージをトラブルシューティングするには、Windows イベント ビューアを使用します。たとえば、「1 個または複数のサービスを開始できませんでした...」というエラーが発生する場合があります。Windows イベント ビューアは Windows サービスの起動中に発生した情報提供のメッセージ、警告、またはエラーを一覧表示します。イベント ビューアの詳細については、Microsoft のマニュアルを参照してください。

InTouch サービスが起動に失敗したことによるすべての警告またはエラー メッセージを表示できます。イベント ビューアに WindowViewer サービスが起動に失敗したことが示される場合、依存関係にある必要なサービスが実行されていないことが原因であることがほとんどです。

サービス ユーザー アカウントの問題のトラブルシューティング

InTouch サービスのインストールが失敗した場合、または InTouch HMI のインストール後に起動に失敗した場合、実行中のユーザー アカウントに問題が発生する場合があります。

サービスのユーザー アカウントの問題をトラブルシューティングするには

1. Windows ユーザー マネージャ ウィンドウを開き、新しいマスタ ユーザー アカウントを作成します。

InTouch コンポーネントをサービスとして起動するには、このユーザー アカウントがローカル コンピュータの管理者権限を持っている必要があります。ドメイン リストにコンピュータのノード名が表示されない場合は、ノード名を手動で入力します。

詳細については、「[InTouch サービスのユーザー アカウントの設定](#)」を参照してください。

2. コンピュータのノード名の長さが 14 文字を超えないことを確認します。ノード名にアンダースコア (_) またはダッシュ (-) が含まれている場合は、削除します。
3. インストール中、ドメイン名を入力するよう求めるメッセージが表示されたら、ドメイン名ではなくコンピュータのノード名を入力します。その後、手順 1 で作成したユーザー名とパスワードを入力します。
4. InTouch HMI がすでにインストールされている場合でも、Archestra Change Network Account ユーティリティを実行することによって、ドメイン名、ユーザー名、およびパスワードを指定できます。
5. コンピュータを再起動します。
6. 有効なユーザー アカウントを使用して、ネットワーク ドメインにログオンします。ドメインがダウンした場合でも、ローカル コンピュータで実行している InTouch アプリケーションに影響は与えられません。

有効 I/O アイテムの使用解除

Windows オペレーティング システムを起動すると、自動的に開始するよう設定されているサービスは自動的に「バックグラウンド」で開始し、デスクトップにユーザー インターフェイスは表示されません。この状態のサービスはシステム コンテキストで実行されています。ユーザーがシステムにログオンすると、システム コンテキストで実行されていて、ユーザー インターフェイスが関連付けられているサービスは、自動的にデスクトップに表示されます。この状態では、サービスは今度はデスクトップ コンテキストで実行されています。

自動的に開始するように **WindowViewer** サービスを設定すると、オペレーティングシステムの起動時にサービスはシステム コンテキストで実行されます。その後、ユーザーがログオンすると、**WindowViewer** サービスは実行され続けますが、デスクトップ コンテキストで実行され、**WindowViewer** のユーザー インターフェイスが自動的に表示されます。

[有効アイテムのみ要求] オプションをオンにして定義された InTouch アクセス名および特定の InTouch アプリケーション ウィンドウでのみアクティブである I/O タグ変数（アプリケーションの他の場所では使用されないタグ変数）がある場合は、それらのタグ変数を「非アクティブ化」できます。たとえば、**WindowViewer** がサービスとして実行されていて、スクリプトを使用してアプリケーション ウィンドウを閉じると、ウィンドウはメモリから自動的にアンロードされ、これらのタグ変数へのリンクは解除されます。

InTouch サービスのレジストリ キー

InTouch サービスは Windows レジストリのキーとして一覧表示されます。

SuiteLink:

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SLS
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\slssvc
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SuiteLink

NetDDE Helper:

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\WWNetDDE

WindowViewer:

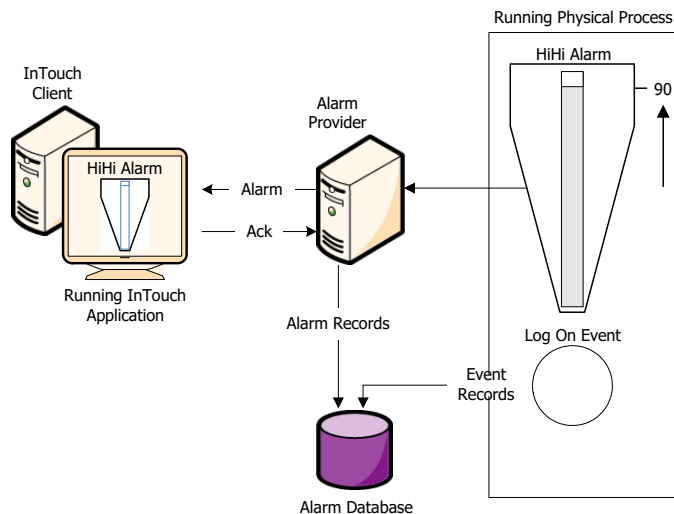
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\VIEW

章 29 アラーム

アラームとイベントを生成してオペレータにプロセスの動作状況を通知する InTouch アプリケーションを作成できます。

- 問題を引き起こす可能性のあるプロセス状態に関する警告が、アラームによってランタイム オペレータに警告されます。通常は、プロセス値が定義済みのしきい値を超えた場合にトリガするようにアラームを設定します。オペレータは、アラームを確認する必要があります。
- イベントは、通常のシステム ステータス メッセージを表します。オペレータが InTouch アプリケーションにログオンするなど、システム条件が発生するときに典型的なイベントとなります。オペレータは、イベントを確認する必要はありません。

以下の図は、アプリケーションが実行されている間、InTouch HMI でアラームとイベントが処理される方法を示しています。アラームとイベントのデータは、アラーム データベースに保存されます。



イベント監視用のタグを設定できます。イベントメッセージは、タグの値が変更するたびにアラーム システムにログ記録されます。イベントメッセージには、値がどのように変更されたかに加えて、オペレータ、I/O、スクリプト、またはシステムのいずれによってその変更が開始されたのかという情報が含まれます。

従来のアラーム システムからの移行

標準アラーム システムまたは AlarmSuite を使用して構築されたアプリケーションを移行できます。

従来のアラーム システムからの移行の概要

標準アラーム システムまたは AlarmSuite を使用して構築されたアプリケーションを移行できます。

標準アラーム システムから分散アラーム システムへの移行

標準アラーム システムを分散アラーム システムに移行するとき、マスタ/スレーブ アプリケーションのすべての標準アラーム表示は分散アラーム表示オブジェクトに移行されます。

色、フォント、式、およびアラーム クエリーの設定は移行されません。新しい分散アラーム表示オブジェクトには、以下のデフォルト クエリーがあります。ここで、ノード名はマスタ ノードの名前です。

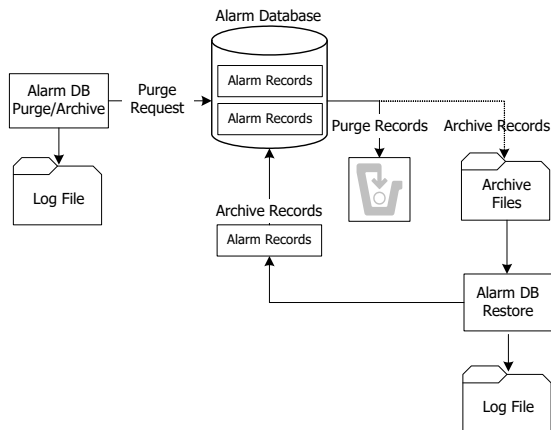
\\nodename\InTouch!\$system

確認とアラーム ステータスのドットフィールドは以前と同じように機能し続けます。I/O タグ変数が **NetDDE** または **SuiteLink** のどちらに対して設定されているかによって、**NetDDE** を有効にする必要がある可能性があります。ただし、分散アラーム表示オブジェクトを使用してアラームが確認できるようになったため、確認を発行するために個別のコントロールは必要ないと判断する場合もあります。

アラーム データベースの維持

アラーム データベースは、2 つの InTouch ユーティリティを使用して管理します。レコードをデータベースから永久に削除するか、ファイルにアーカイブするには **Alarm DB Purge-Archive** ユーティリティを使用します。データベースが破壊された場合、**Alarm DB Restore** ユーティリティを使用して、アーカイブされたレコードを復元します。

以下の図は、両方のユーティリティがレコードをパージ／アーカイブして、データベースに復元する方法を示しています。

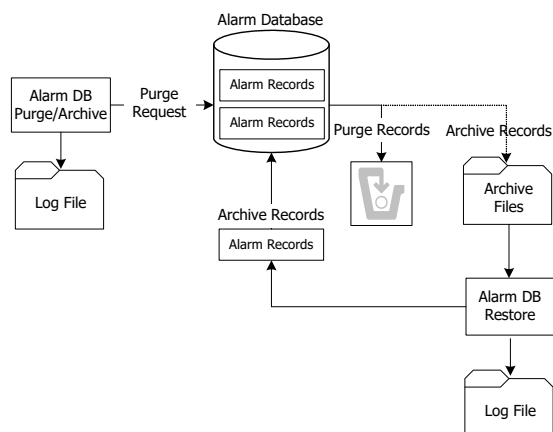


Alarm DB Purge-Archive ユーティリティを使用するには、管理者としてコンピュータにログインしている必要があります。

アラーム データベースの維持の概要

アラーム データベースは、2 つの InTouch ユーティリティを使用して管理します。レコードをデータベースから永久に削除するか、ファイルにアーカイブするには **Alarm DB Purge-Archive** ユーティリティを使用します。データベースが破壊された場合、**Alarm DB Restore** ユーティリティを使用して、アーカイブされたレコードをリストアします。

以下の図は、両方のユーティリティがレコードをパージ／アーカイブして、データベースにリストアする方法を示しています。



Alarm DB Purge-Archive ユーティリティを使用するには、管理者としてコンピュータにログインしている必要があります。

ページ設定またはアーカイブ設定

Alarm Purge-Archive ユーティリティは、以下の操作を実行するために使用します。

- アラーム データベースからパージするレコード タイプを選択する
- 毎日、毎週、または毎月のスケジュールでレコードを自動的にパージする
- パージされたデータベース レコードをオプションでファイルにアーカイブする
- 問題をトラブルシューティングするためにパージ操作またはアーカイブ操作のステータスを保存する
- パージ操作またはアーカイブ操作のステータスを表示する

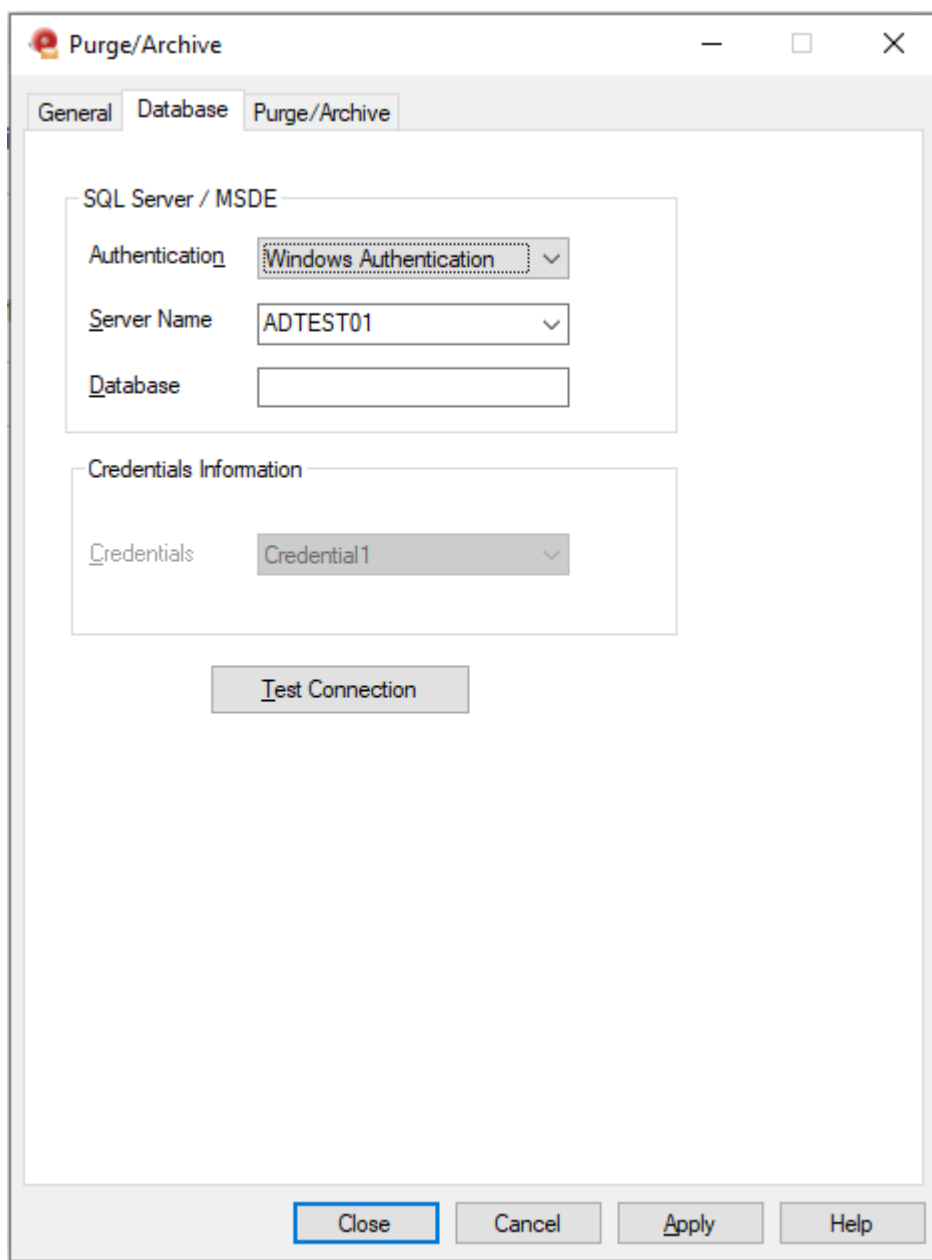
重要: Alarm DB Purge-Archive ユーティリティは Windows サービスのセットとして実行します。管理者権限で Alarm DB Purge-Archive ユーティリティを実行するときにセキュリティの脅威にさらされる可能性を削減するために、ユーザー アカウントの許可は非対話型に変更されます。

アラーム データベースへの接続

Alarm DB Purge-Archive ユーティリティを使用する前に、アラーム データベースに接続する必要があります。

データベース接続を設定するには

1. **Alarm DB Purge-Archive** ユーティリティを開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Purge-Archive] をダブルクリックします。
2. [データベース] タブをクリックします。



3. データベース接続を設定します。以下の手順を実行します。

- a. [認証] リストで認証方法（**SQL Server 認証**または **Windows 認証**を選択します（デフォルトは Windows 認証）。

注記: Windows 認証は、SQL 認証よりも優れたアプリケーションセキュリティを提供できます。この理由から、Windows 認証から SQL 認証に切り替えると、Windows 認証を使用することを推奨するポップアップダイアログが表示されます。この警告を無視して SQL 認証を使用する場合は、[OK] をクリックします。同様のメッセージが Log Viewer に記録されます。

- a. [サーバー名] リストで、サーバーのノード名をクリックします。
- b. [データベース] ボックスにアラーム データベースの名前を入力します。
- c. [資格情報] 領域の [資格情報] ドロップダウンから、認証に使用する資格情報を選択します。

注記: [資格情報] フィールドは、SQL Server 認証タイプを選択した場合にのみ有効になります。Windows 認証方法では、現在ログインしているユーザーの資格情報が使用され、[資格情報] フィールドは無効になります。スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報マネージャから取得されます。をクリックします。詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[資格情報マネージャの操作](#)」を参照してください。

4. [接続テスト] をクリックして、データベースへの接続をテストします。アラーム データベースへの接続が正常に終了したかどうかを示すメッセージが表示されます。
5. [OK] をクリックします。
6. [適用] をクリックします。

サーバーからパージするデータ量の設定

以下の操作を実行できます。

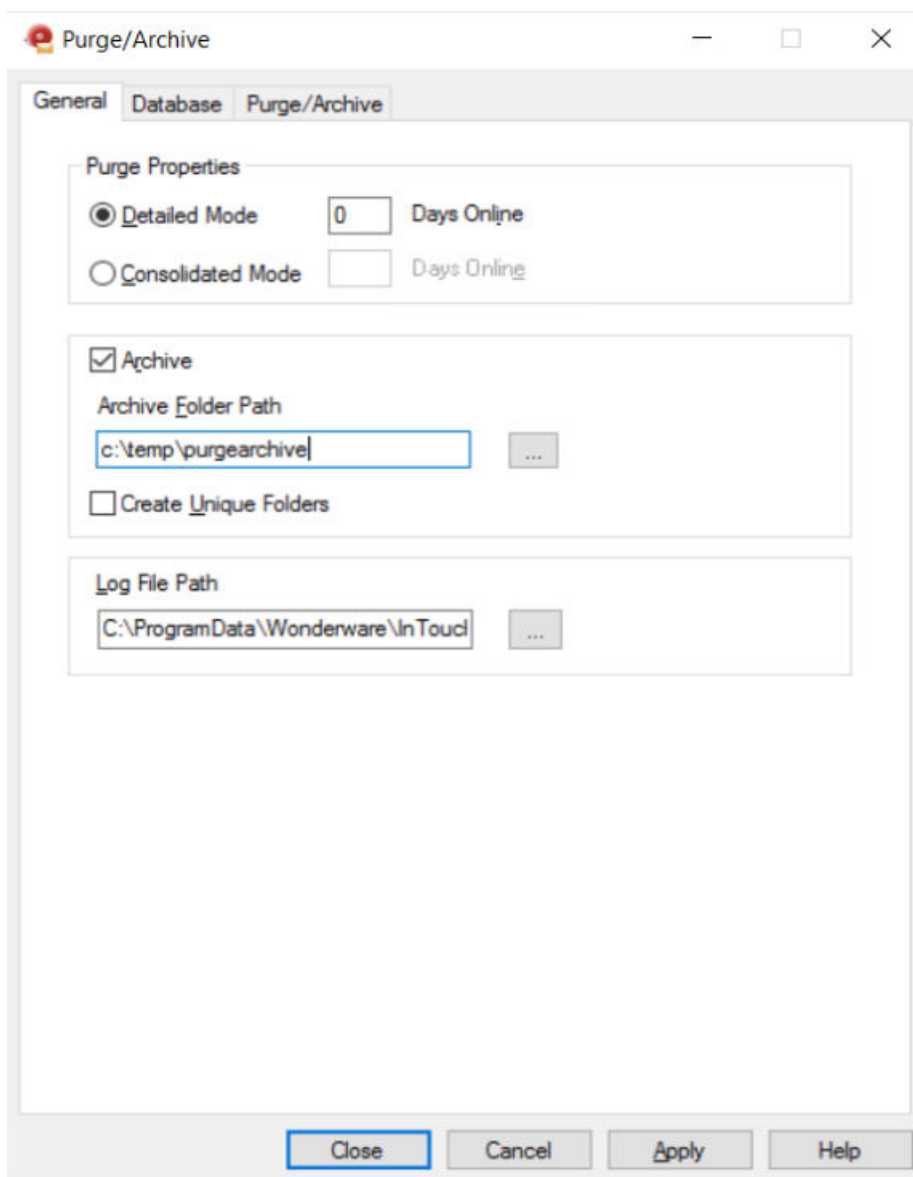
- アラーム データベースからパージするアラーム レコードを選択する
- オプションでアラーム データベースからファイルへパージされたレコードをアーカイブする
- パージ ログ ファイルを保存するフォルダ場所を選択する

パージする必要があるテーブルのタイプを AlarmDetail または AlarmConsolidated のいずれかのテーブルから選択します。

指定した日数の 1 日前のすべてのデータがパージされます。有効なエントリは 0 ～ 9999 です。0 を選択すると、現在の日付のレコードを除くすべてのレコードがアラーム データベースからパージされます。

パージするレコードを選択するには

1. **Alarm DB Purge-Archive** ユーティリティを開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Purge-Archive] をダブルクリックします。
2. [一般] タブをクリックします。



3. [パージのプロパティ] 領域で、パージするレコードのタイプを設定します。以下のいずれかを実行します。
 - 詳細モードでデータベースに保存されているアラーム レコードをパージするには、[詳細モード] をクリックします。
 - 統合モードでデータベースに保存されているアラーム レコードをパージするには、[統合モード] をクリックします。
4. [オンライン日数] ボックスに、アラーム データベースに保持するレコードの日数を入力します。
5. [適用] をクリックします。

パージされたデータのアーカイブの設定

アラーム データベースからパージされたレコードをアーカイブして、Alarm DB Restore ユーティリティを使用して、それらのレコードを復元します。

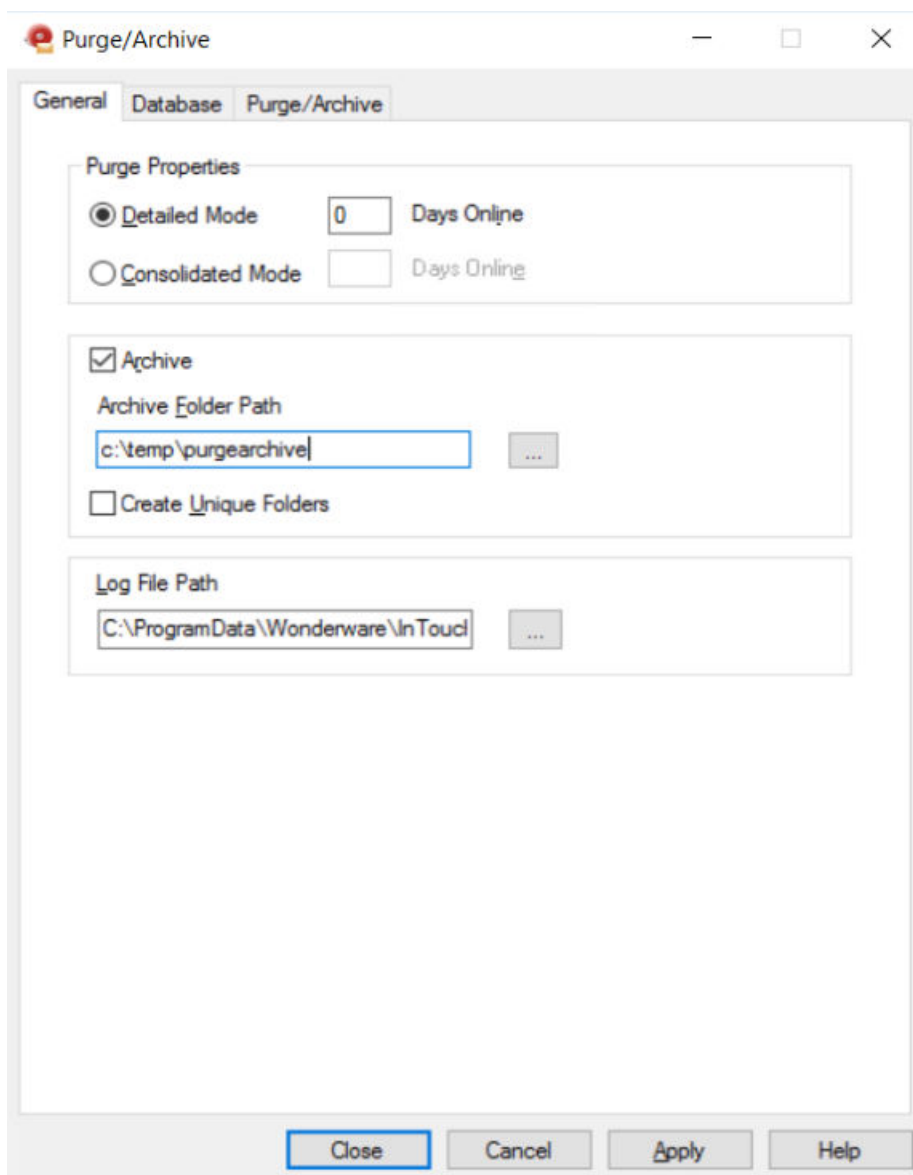
アラーム データベースをパージする場合、**Alarm DB Purge-Archive** ユーティリティではパージされたアラーム データベース テーブルに対応する 9 つのアーカイブ ファイルが自動的に作成されます。各ファイルには 1 つのテーブルのパージされたレコードが含まれます。

Alarm DB Purge-Archive ユーティリティでは、パージ操作が発生した際、テーブル名、日付、および時間に基づいてアーカイブされたファイルに名前が割り当てられます。たとえば、2007 年 6 月 22 日の午後 5 時 30 分にパージされた **AlarmMaster** テーブルに対するアーカイブ ファイルの名前の形式は、以下のようになります。

AlarmMaster_06222007_1730.txt

アーカイブを設定するには

1. **Alarm DB Purge-Archive** ユーティリティを開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Purge-Archive] をダブルクリックします。
2. [一般] タブをクリックします。



3. [アーカイブ] チェック ボックスをオンにします。
4. [アーカイブ フォルダ パス] ボックスに、アーカイブ ファイルが保存されるフォルダ場所を入力するか、参照ボタンをクリックして場所を参照します。
5. アーカイブ ファイルをアーカイブ ファイル フォルダの下の子のサブフォルダに配置するには、[一意のフォルダを作成] チェック ボックスをオンにします。
6. [適用] をクリックします。

ログ ファイルの設定

Alarm DB Purge-Archive ユーティリティでは、ページ操作中にステータス メッセージが生成されます。これらのメッセージは、ユーティリティの [ステータス] ウィンドウからオンラインで表示されます。Alarm DB Purge-Archive ユーティリティではまた、ページメッセージが **WWAlmPurge.log** という名前のページ ログ ファイルに書き込まれます。

以下の例は、ページ操作が正常に終了した後でログに保存されるメッセージを示しています。

ページ開始時刻: 12:16:48 PM 6/22/2007

トランザクションの開始...

ProviderSession テーブルをアーカイブ中...

Query テーブルをアーカイブ中...

Casue テーブルをアーカイブ中...

AlarmMaster テーブルをアーカイブ中...

OperatorDetails テーブルをアーカイブ中...

AlarmDetail テーブルをアーカイブ中...

Comment テーブルをアーカイブ中...

Events テーブルをアーカイブ中...

TagStatus テーブルをアーカイブ中...

データベースのすべてのレコードをページ中...

実行中...

ページ完了時刻: 12:16:52 PM 6/22/2007

144 AlarmMaster から取得したレコードは、別のテーブルから取得した関連するレコードと共にページされました。

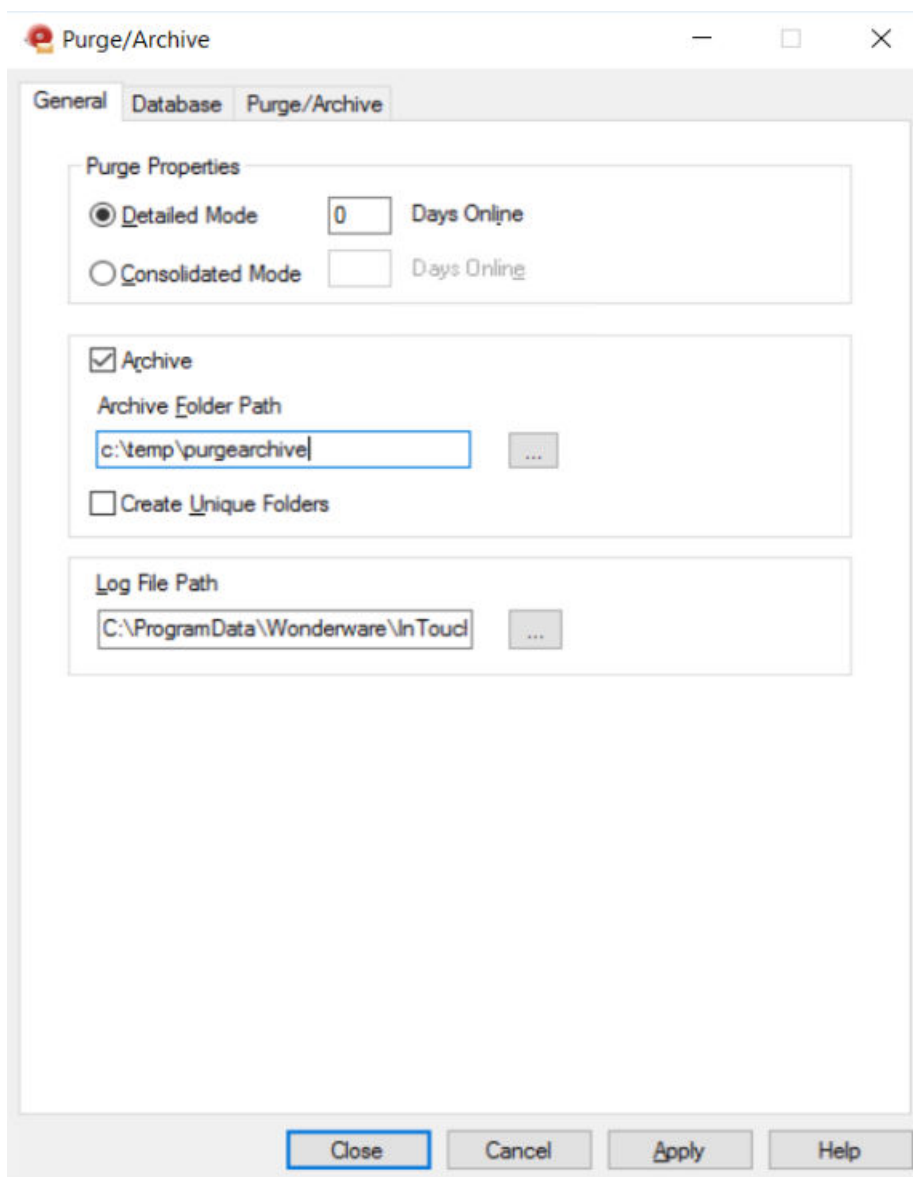
デフォルトで、ページログファイルは C:\Documents and Settings\All Users\Application Data\Wonderware\InTouch フォルダに保存されます。Microsoft Windows Vista オペレーティングシステムが実行されているコンピュータでは、デフォルトのアプリケーションフォルダは、C:\Users\UserName\Documents\My InTouch Applications です。

ページログファイルの保存場所は変更できます。

Alarm DB Purge-Archive ユーティリティでは、ページが発生するたびに新しいメッセージがログファイルに追加されます。

アーカイブ ログを設定するには

1. **Alarm DB Purge-Archive** ユーティリティを開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Purge-Archive] をダブルクリックします。
2. [一般] タブをクリックします。



3. [ログフォルダパス] ボックスに、パージログファイルが保存されるフォルダ場所を入力するか、参照ボタンをクリックして場所を参照します。
4. [適用] をクリックします。

データベースの手動によるパージとアーカイブ

アラーム データベースは手動でパージおよびアーカイブできます。これは実行日時よりも優先され、パージおよびアーカイブがすぐに開始されます。

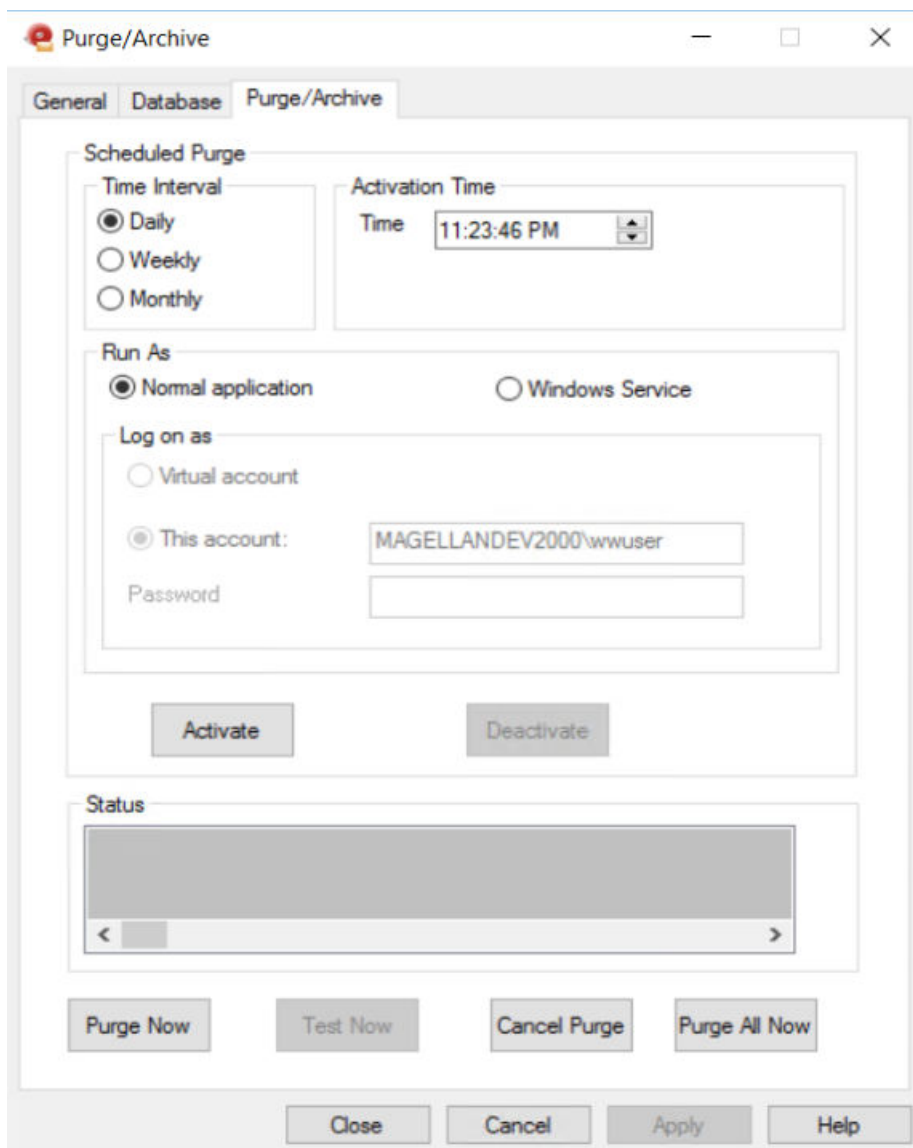
パージ操作ではアーカイブ ファイルの存在が確認され、データが追加されます。アーカイブ ファイルが存在しない場合、ファイルは命名規則に従って作成され、アーカイブに使用されます。

パージ操作では、外部キー制約を通じて **AlarmMaster** などのメイン テーブルにリンクされている **ProviderSession**、**Query**、および **Cause** などのテーブルのエントリは削除されません。これらのテーブルの関連レコードは、データの一貫性を維持し、データベースで保持するために記述されています。

注意: すべてのレコードの手動パージ（[全てを実行] オプション）は、Alarm DB Logger サービスが停止している場合にのみ実行します。Alarm DB Logger サービスが実行している間にパージ操作が正常に実行された場合、Alarm DB Logger サービスはロギングを停止して、レコードのキャッシュを開始します。

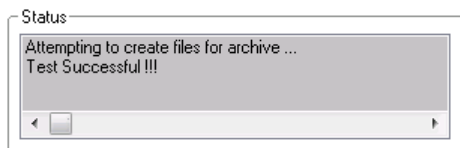
アラーム データベースから手動でレコードをパージおよびアーカイブするには

1. **Alarm DB Purge-Archive** ユーティリティを開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Purge-Archive] をダブルクリックします。
2. [パージ/アーカイブ] タブをクリックします。



3. [テスト] をクリックして、データベースとアーカイブ場所への接続を確認するテスト パージを実行します。

テスト パージによって、指定したアーカイブ フォルダに空のアーカイブ ファイルが作成されます。[ステータス] 領域に、テストが正常に終了したことを示すメッセージが表示されます。



[テスト] ボタンは、パージされたレコードをアーカイブすることを選択した場合のみ使用できます。[アーカイブ] オプションは [全般] タブに配置されています。

4. データベースからレコードをパージします。以下のいずれかを実行します。
 - 選択したレコードをパージするには、[今すぐ実行] をクリックします。
 - すべてのレコードをパージするには、[全てを実行] をクリックします。
5. パージを停止するには、[中止] をクリックします。パージをキャンセルした場合、アラーム データベースは元の状態にロールバックされます。

自動パージのスケジュール設定

Alarm DB Purge-Archive ユーティリティは、スケジュールされた間隔でレコードをアラーム データベースから自動的にパージまたはアーカイブできます。テスト パージを実行して、データベースやターゲット場所への接続の検証、およびパージの開始と停止を行うことができます。

自動パージのスケジュールを設定するには

1. **Alarm DB Purge-Archive** ユーティリティを開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Purge-Archive] をダブルクリックします。
2. [パージ/アーカイブ] タブをクリックします。

The screenshot shows the 'Purge/Archive' configuration window. The 'Purge/Archive' tab is selected. Under 'Scheduled Purge', 'Daily' is selected under 'Time Interval', and '11:23:46 PM' is set for 'Activation Time'. Under 'Run As', 'Windows Service' is selected, and 'Virtual account' is chosen for 'Log on as'. The 'This account' field shows 'MAGELLANDEV2000\wwuser'. At the bottom, there are buttons for 'Purge Now', 'Test Now', 'Cancel Purge', 'Purge All Now', 'Close', 'Cancel', 'Apply', and 'Help'.

3. [時間間隔] 領域で、毎日、毎週、毎月のいずれかからページ間隔を選択します。

[毎週] または [毎月] をクリックすると、[実行日時] 領域に [日] ボックスが表示され、曜日または日付を指定できます。

[毎日] をクリックした場合、[時間] ボックスで、ページ/アーカイブ操作を開始する時間を設定します。

4. [実行] 領域で、[標準アプリケーション] をクリックすると、Purge-Archive ユーティリティがアプリケーションとして実行されます。[Windows サービス] をクリックすると、ユーティリティはサービスとして実行されます。

[Windows サービス] を選択した場合、[仮想アカウント] を選択するか、[指定アカウント:] 領域で別のアカウントのユーザー名とパスワードを指定します。

注記: 仮想アカウントの詳細については、「[仮想アカウントの使用](#)」を参照してください。

5. ページとアーカイブの設定を保存するには、[適用] をクリックします。

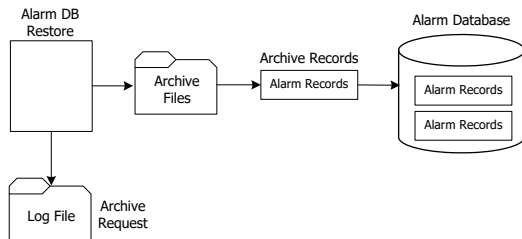
6. Alarm DB Purge-Archive ユーティリティを自動パージスケジュールに設定するには、[アクティブ化]をクリックします。

7. [閉じる] をクリックします。

自動パージスケジュールを停止し、Windows サービスを削除するには、[非アクティブ化] をクリックします。少し待つとサービスが Windows サービスから削除されます。

アラーム データベースのリストア

Alarm DB Restore ユーティリティでは、アーカイブ ファイルのアーカイブされたアラーム レコードがアラーム データベースにリストアされます。以下の図は、アラーム レコードをデータベースにリストアする手順の概要を示しています。



データベースをリストアするには、以下の操作を行う必要があります。

- アラーム データベースへの接続を設定します。
- アラーム データベースにリストアするアラームを選択します。
- アーカイブ レコードをアラーム データベースにリストアします。

Alarm DB Restore ユーティリティは最小化すると、システム トレイにアイコンで表示されます。アイコンを右クリックすると、メニューで以下のコマンドが表示されます。

コマンド	説明
リストア	リストア処理を開始します。
リストアのキャンセル	リストア処理をキャンセルします。
ステータスを消去	ステータス ウィンドウの内容をクリアします。
ウィンドウ消去	Alarm DB Restore ユーティリティを最小化し、システム トレイにアイコンで表示します。
ウィンドウ表示	Alarm DB Restore ユーティリティを開いて、最大化します。
終了	Alarm DB Restore ユーティリティを閉じます。

Alarm DB Restore ユーティリティ内を右クリックすると、同じメニューが表示されます。

データベース接続の設定

アーカイブ済みデータを復元するデータベースを選択する必要があります。指定したデータベースがサーバーに存在しない場合は、デフォルトのサーバー パラメータを使用して、新しいデータベースを作成するよう求めるメッセージが表示されます。

復元するデータベースを設定するには

1. **Alarm DB Restore** ユーティリティを開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Restore] をダブルクリックします。
2. [Configuration] (設定) タブをクリックします。

The screenshot shows the 'Restore' dialog box with the 'Configuration' tab selected. The 'SQL Server / MSDE' section has 'Authentication' set to 'SQL Server Authentication', 'SQL Server Name' set to 'BETAVM01', and 'Database' set to 'Script Test'. The 'Credentials Information' section has 'Credentials' set to 'Credential1'. A 'Test Connection' button is located below these sections. The dialog also has 'Close' and 'Help' buttons at the bottom.

3. アラーム データベースへの接続を設定します。以下の手順を実行します。
 - a. [認証] リストで認証方法 (**SQL Server 認証**または **Windows 認証**を選択します (デフォルトは Windows 認証)。

注記: Windows 認証は、SQL 認証よりも優れたアプリケーションセキュリティを提供できます。この理由から、Windows 認証から SQL 認証に切り替えると、Windows 認証を使用することを推奨するポップアップ ダイアログが表示されます。この警告を無視して SQL 認証を使用する場合は、[OK] をクリックします。同様のメッセージが Log Viewer に記録されます。

 - b. [SQL Server 名] リストで、アラーム データベースをホストするサーバーのノード名をクリックします。
 - c. [データベース] ボックスにアラーム データベースの名前を入力します。
 - d. [資格情報] 領域の [資格情報] ドロップダウンから、認証に使用するアラーム データベースの資格情報を選択します。

[資格情報] フィールドは、SQL Server 認証タイプを選択した場合にのみ有効になります。Windows 認証方法では、現在ログインしているユーザーの資格情報が使用され、[資格情報] フィールドは無効になります。

注記: スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報マネージャから取得されます。詳細については、『AVEVA™ InTouch HMI アプリケーション開発ガイド』の「[資格情報マネージャの操作](#)」を参照してください。

a. [接続テスト] をクリックして、データベースへの接続をテストします。アラーム データベースへの接続が成功したかどうかを通知するメッセージが表示されます。

b. [OK] をクリックします。

4. [閉じる] をクリックします。

復元するファイルの設定

レコードを復元する時間範囲、およびデータベース テーブルを再作成するかどうかを選択できます。

復元をキャンセルすると、データベースは元の状態にロールバックされます。

注意: データベースに既に存在するアーカイブされたアラームを復元しようとしても、アーカイブ レコードは復元されません。この処理によって、データベースに重複したアラーム/イベント エントリが生じることが防止されます。レコードに関連付けられているアラーム GUID またはイベント GUID によって、アラームまたはイベントがデータベースに既に存在しているかどうか判断されます。

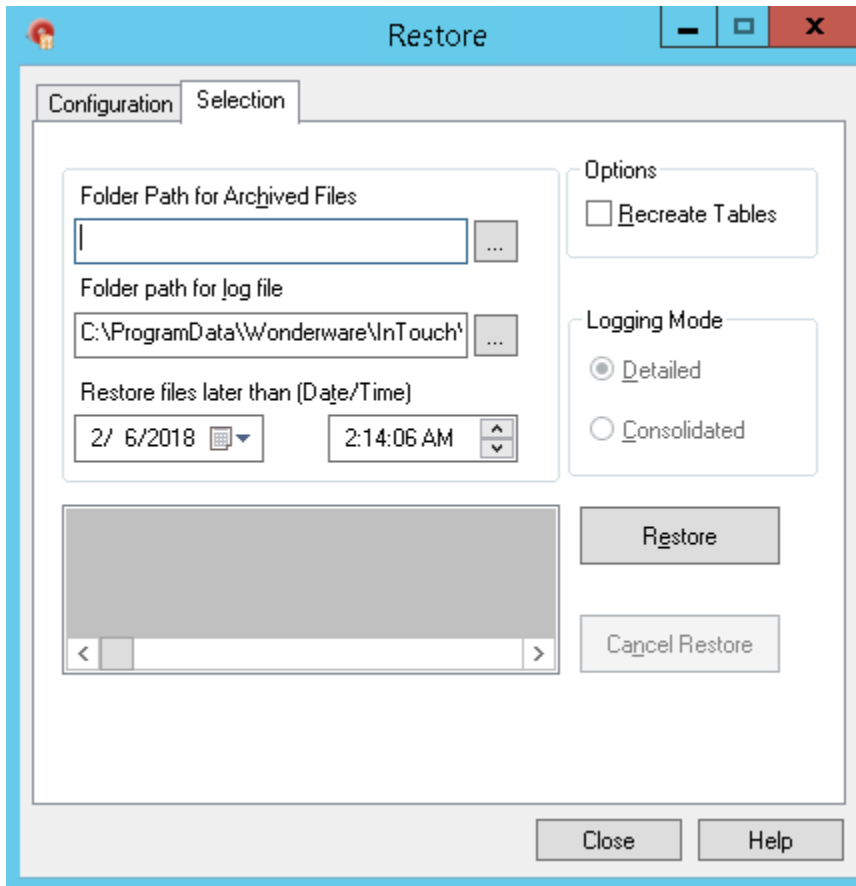
復元するデータベース レコードを選択するには

1. **Alarm DB Restore** ユーティリティを開きます。以下の手順を実行します。

a. [ツール] ビューで [アプリケーション] を展開します。

b. [Alarm DB Restore] をダブルクリックします。

2. [選択] タブをクリックします。



3. [アーカイブしたファイルのフォルダパス] ボックスに、アーカイブされたファイルの場所へのフルパスを入力します（最大 255 文字）。または、参照ボタンをクリックしてアーカイブファイルが保存されているフォルダを検索して、選択します。
4. [次の日時以降のファイルを復元] 領域で、データベースへのレコードの復元を開始する日時を選択します。
開始日時は、デフォルトで現在の日時に設定されています。
5. [ログファイルのフォルダパス] ボックスに、ログファイルを作成して保存する場所のフルパスを指定します（最大 255 文字）。または、参照ボタンをクリックして、フォルダを参照して選択します。
6. [テーブルの再生] チェック ボックスをオンにすると、指定したアラーム データベースのテーブルが再作成されます。
7. アーカイブファイルに含まれるアラーム レコードに対して選択したログモードのタイプに基づいて、以下を選択します。
 - 詳細 - 詳細モードでアラーム データベース テーブルを再作成します。
 - 統合 - 統合モードでアラーム データベース テーブルを再作成します。

重要: テーブルを再作成すると、アラーム データベースに現在保存されているすべてのレコードが上書きされます。

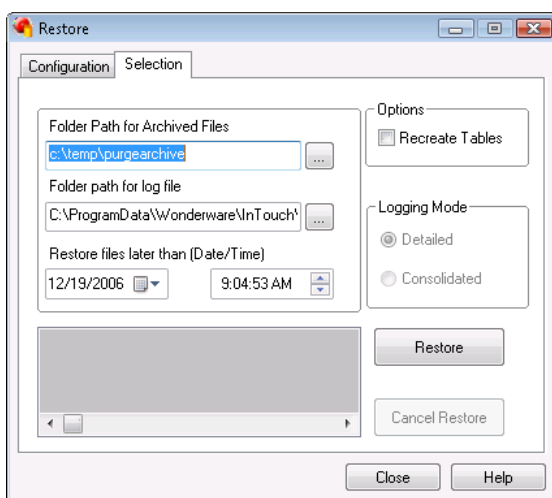
1. [復元] をクリックします。

データベース 復元操作の開始

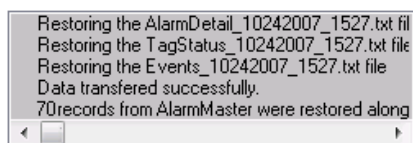
データベース接続を確立して、アーカイブされたファイル フォルダとタイム バッファを指定した後で、アーカイブされたデータベース レコードを復元します。

アーカイブからデータベース レコードを復元するには

1. **Alarm DB Restore** ユーティリティを開きます。以下の手順を実行します。
 - a. [ツール] ビューで [アプリケーション] を展開します。
 - b. [Alarm DB Restore] をダブルクリックします。
2. [選択] タブをクリックします。



3. [復元] をクリックします。復元が正常に行われたかどうか、およびデータベースに復元されたレコードの数を示すメッセージが表示されます。



付録 AD INTOUCH.ini ファイルからのアプリケーション設定のカスタマイズ

InTouch アプリケーションを初めて実行するとき、アプリケーション フォルダに INTOUCH.ini ファイルが作成されます。INTOUCH.ini ファイルが作成されるときに、個々の InTouch アプリケーションの動作特性を決定する一連のパラメータに値が割り当てられます。

WindowMaker または WindowViewer からアプリケーションの設定を続行すると、新しい INTOUCH.ini パラメータが作成されるか、または既存のパラメータが変更されます。たとえば、WindowMaker の **履歴ログの設定** ダイアログ ボックスからログを設定すると、INTOUCH.ini ファイルにログのパラメータが追加されます。

その他の設定パラメータは、INTOUCH.ini ファイルに手動で追加する必要があります。

アプリケーションのカスタマイズ後、INTOUCH.ini ファイルを別のアプリケーションのフォルダにコピーできます。このようにすると、カスタマイズに必要なすべての手順を繰り返さなくても、一貫したアプリケーションの動作特性を保つことができます。

カスタム INTOUCH.ini パラメータ

以下の表には、カスタム プロパティを InTouch アプリケーションに追加するために、INTOUCH.ini ファイルに手動で入力できる一連のパラメータが一覧表示されています。

INTOUCH.ini パラメータ	説明
16PenTrendDrawMode	16 ペン トレンドで、データ値が平均化モードまたは最小 - 最大モードのどちらで表示されるのかを決定します。
ApplicationThumbnail	アプリケーション サムネイル ファイルの名前を設定します。
AllowPubAppEdit	アプリケーション フラグを設定し、パブリッシュ済みアプリケーションを編集できるようにします。値が 1 の場合、パブリッシュ済みの InTouch ファイルを編集できます。
CommentRetentive	[アラーム コメント] フィールドへのランタイムの変更が保存されるかどうかを決定します。
ForceLogCurrentValue	ForceLogging パラメータによって設定された間隔で、ログ済みタグの現在の値が履歴ログ ファイルに書き込まれるかどうかを決定します。
ForceLogging	現在の値にかかわらず、タグの値が履歴ログ ファイルに定期的書き込まれるときの間隔の長さを設定します。
LoopTimeOut	InTouch スクリプトでの FOR-NEXT ループ処理のタイムアウト期間を設定します。

INTOUCH.ini パラメータ	説明
MarkAppReadOnlyNonRDS	このパラメータが非 RDS ノードで 1 に設定されている場合、そのノードは読み取り専用ノードとみなされ、InTouchView アプリケーションの読み取り専用ライセンスが使用されます。
MouseMustBeOnObjectForOnKeyUp	[離れた時]アクションをトリガするためにマウスがオブジェクト上にある必要があるかどうかを決定します。デフォルトでは、この値は '1' です。
NoKeyboardResize	数値キーボードのサイズが WindowViewer の画面解像度用に変更されるかどうかを決定します。
OldRightMouseBehavior	WindowMaker で右マウス ボタンを有効にするかどうかを決定します。
PrintScreenWait	WindowViewer から画面を印刷する前の待機時間を設定します。
PrintWindowWait	WindowViewer から InTouch ウィンドウを印刷する前の待機時間を設定します。
RemoteTagsLogEvents	InTouch アプリケーションが、リモート参照されたタグアラームおよびイベントをログ記録するかどうかを決定します。
RemoteTagsNoIOEvents	InTouch アプリケーションが、リモート参照されたタグアラームをログ記録するかどうかを決定します。
ScaleForResolution	異なる画面解像度を持つノードに変更する場合には、InTouch アプリケーションのウィンドウが自動的にサイズ変更されるかどうかを決定します。
ViewLicenseRetryCount	セットアップ時および使用可能なライセンスがないときに WindowViewer がバックグラウンドでライセンス取得を試行する回数を決定します。
WindowNameWithSpecialCharacters	このパラメータを 1 に設定すると、新しいウィンドウを作成する際にウィンドウ名に特殊文字を使用できます。

カスタム ログ プロパティの設定

タグ変数の値がどのように InTouch の履歴ログ ファイルに保存されるのかを指定する INTOUCH.ini ファイルに、一連のパラメータを追加できます。これらのパラメータに割り当てられられた値により、ログの頻度、およびリモート参照されたタグ変数の値がログ済みかどうかが決まります。

ログの頻度の設定

次の 2 つの条件に基づいて、InTouch HMI によって履歴ログ ファイルにエントリが書き込まれます。

- タグ変数値がログ デッドバンド値より大きい工学単位値によって変更されると、InTouch HMI によってすぐにログ エントリが書き込まれます。
- InTouch HMI によって、すべてのログ済みのタグ変数の現在の値が固定間隔で書き込まれます。デフォルトの固定間隔は 60 分です。

周期を変更するには、次の 2 つのパラメータを INTOUCH.ini ファイルに追加します。

- **ForceLogging**

ForceLogging は、ログ記録間隔の長さを分単位で指定します。**ForceLogging** の値は、5 ～ 120 分の間で設定できます。デフォルトは、**ForceLogging=60** です。

- **ForceLogCurrentValue**

ForceLogCurrentValue を使用すると、現在の値がログのデッドバンドの範囲以下であっても、InTouch HMI によってすべてのログ済みのタグ変数のログ エントリが書き込まれます。デフォルトは、**ForceLogCurrentValue=0** です。

次の例では、15 分の周期で、またはタグ変数の値が変更されたときに、現在のタグ変数の値が履歴ログ ファイルに書き込まれます。

```
ForceLogging=15
```

```
ForceLogCurrentValue=1
```

リモート参照されたタグ変数のログ記録

デフォルトでは、リモート参照されたタグ変数はイベント ログ ファイルにログ記録されません。リモート参照されたタグ変数をログ記録するには、イベント ログを有効にし、**RemoteTagsLogEvents** パラメータを INTOUCH.ini ファイルに追加する必要があります。

```
RemoteTagsLogEvents=1
```

I/O タグ変数がログ記録されないようにするには、**RemoteTagsNoIOEvents** パラメータを INTOUCH.ini ファイルに追加します。**RemoteTagsLogEvents** パラメータが 1 に設定されている場合のみ、**RemoteTagsNoIOEvents** パラメータが適用されます。

```
RemoteTagsNoIOEvents=1
```

WindowMaker ショートカット メニューの無効化

デフォルトでは、選択したオブジェクト上でマウスを右クリックしたときに、WindowMaker にショートカット メニューが表示されます。以前のバージョンの InTouch HMI と同じマウスの動作を使用してアプリケーションを開発する場合、INTOUCH.ini ファイルの **oldrightmousebehavior** パラメータを 1 に設定することにより、WindowMaker の右クリックの動作をオフにできます。

```
oldrightmousebehavior=1
```

カスタム WindowViewer プロパティの設定

次を行うために、WindowViewer の動作を設定する一連の INTOUCH.ini ファイル パラメータを追加できます。

- スクリプト ループの処理
- 異なる画面解像度用に InTouch ウィンドウを調整
- ウィンドウまたは画面を印刷するための待ち時間の設定
- アラーム コメントへのランタイムでの変更のログ記録
- 16 ペン トレン드의描画モードの設定

- 数値キーパッドのサイズ変更
- アナログおよび文字列のデータ入力リンクの入力フィールドのサイズ変更

スクリプトループタイマーの追加

デフォルトでは、InTouch スクリプト内の FOR-NEXT ループは 5 秒以内に完了する必要があります。FOR-NEXT ループ処理がタイムアウト制限までに終了していない場合、WindowViewer でスクリプトが自動的に停止します。このタイムアウト制限により、スクリプトエラーによる無限ループを防ぐことができます。

FOR-NEXT ループコード処理が 5 秒のタイムアウト制限を超えるスクリプトを記述する必要がある場合もあります。LoopTimeout パラメータを INTOUCH.ini ファイルに追加することにより、タイムアウト制限の長さを変更できます。

この例では、ループ処理が最大 20 秒間続行します。

```
LoopTimeout=20
```

異なる画面解像度への InTouch ウィンドウの調整

異なる画面解像度で実行されている他のノードにアプリケーションを移行する場合、INTOUCH.ini ファイルにパラメータを追加して、InTouch ウィンドウの現在の解像度を保持できます。

ScaleForResolution パラメータ値により、WindowViewer が実行されているコンピュータで表示解像度を変更された後に、アプリケーション ウィンドウが WindowMaker によって自動的にスケールされるかどうかが決まります。ScaleForResolution パラメータは、WindowViewer ダイアログ ボックスの解像度には影響しません。ScaleForResolution パラメータが 1 に設定されている場合、解像度の変換が有効になります。

```
ScaleForResolution=1
```

印刷の待ち時間の長さの設定

印刷するウィンドウまたは画面を選択する場合、選択したウィンドウまたは画面が WindowViewer によってメモリにロードされます。次に WindowViewer は、ウィンドウまたは画面に表示されたすべての DDE 変数が更新されるまで 10 秒間待機します。待ち時間が終了した後、WindowViewer によってウィンドウまたは画面がプリンタに送信されます。

WindowViewer の印刷の待ち時間は、PrintWindowWait パラメータまたは PrintScreenWait パラメータを INTOUCH.ini ファイルに追加することにより変更できます。両者のパラメータの待機時間は、ミリ秒で表されます。

```
PrintWindowWait=15000
```

```
PrintScreenWait=20000
```

アラームコメントのログ記録

アラームを確認するとき、オペレータはコメントを追加できます。タグ変数データベースの [アラームコメント] フィールドにランタイムでの変更を書き込むには、現在のアプリケーションの INTOUCH.ini ファイルに次の行を追加します。

```
CommentRetentive=1
```

16 ペントレンドの描画モードの設定

16PenTrendDrawMode パラメータの値に基づいて、16 ペントレンドの線描画モードを選択できます。

- 平均化モード：16PenTrendDrawMode=0

16 ペントレンドの時間枠およびバッファ サイズにより、トレンド上の各ピクセルは数秒相当のデータを表しています。各周期には、異なる値を持つ複数のサンプルが含まれている場合もあります。

結果として、トレンドのデータ ポイントは、その周期内に監視された最大値と最小値の間の垂直線として表示されます。

最小から最大までの垂直線が描画された後、計算された周期の平均値にトレンド ペンが移動します。平均値からトレンドの次の周期に線を描画すると、次の周期が開始します。垂直の最小から最大までの線が描画され、その周期に対して計算された平均値でペンが止まります。このプロセスは各サンプリング間隔に対して繰り返されます。

INTOUCH.ini ファイルで **16PenTrendDrawMode** が指定されていない場合、平均化は **16 ペン** トレンドのデフォルトの描画モードです。

- 最小 - 最大モード : **16PenTrendDrawMode=1**

最小 - 最大の描画モードでは、各データの収集期間のエンドポイントを直接繋ぐことによってトレンドの線が描画されます。

数値キーパッドのサイズ変更

InTouch アプリケーションの数値キーパッドのサイズを変更できるかどうかを決定する INTOUCH.ini ファイルにパラメータを追加できます。キーパッドのサイズをより高い画面解像度 (1280 x 1024) にすると、キーパッドに表示されるテキストが読みやすくなります。しかし、アプリケーションで画面スペースが制限されていると、キーパッドのサイズに現実的な制限が設定されている場合があります。

NoKeyboardResize パラメータを INTOUCH.ini ファイルに追加できます。デフォルトでは、パラメータは含まれていません。デフォルト値は次のとおりです。

```
NoKeyboardResize=0
```

デフォルト値を使用すると、画面解像度に応じて数値キーパッドのサイズを変更できます。

パラメータに割り当てることができる別の値は次のとおりです。

```
NoKeyboardResize=1
```

この場合、画面解像度に基づいてキーパッドのサイズは変更されず、数値キーパッドのサイズは固定されたままです。

アナログおよび文字列のデータ入力リンクの入力フィールドのサイズ変更

Resizable InputLink パラメータを INTOUCH.ini ファイルに追加すると、アナログまたは文字列のデータ入力リンクの入力ボックスをマウスでサイズ変更できます。**Resizable InputLink** パラメータは、ゼロ以外の値に設定する必要があります。

入力フィールドを初めてサイズ変更した後、**WindowViewer** によって INTOUCH.ini ファイルに **Resizable InputLink Width** パラメータおよび **Resizable InputLink Height** パラメータが追加されます。これらのパラメータにより、入力ボックスの幅および高さがピクセルで指定されます。

例 :

```
Resizable InputLink = 1  
Resizable InputLink Width=300  
Resizable InputLink Height=50
```

また、これらのパラメータに割り当てられた値を手動で変更して INTOUCH.ini ファイルを編集できます。

反応しなくなったアプリケーション ボタンまたは表示値の問題の解決

パラメータを InTouch.ini ファイルに追加して、InTouch アプリケーション ボタンが押されたままになる問題や、表示値が変化しなくなる問題を解決できます。この問題が発生すると、マウスを繰り返しクリックしてもボタンと値が反応しなくなります。

この問題は、**OnKeyDown** スクリプトを含むグラフィック要素によってウィンドウが非表示になるため、**OnKeyUp** スクリプトが実行しなくなることが原因で発生することがあります。また、反応しなくなったボタンの問題は、ビットを設定する **OnKeyDown** とビットをクリアする **OnKeyUp** の 2 つのスクリプトがある場合に発生することがあります。この場合、オペレータがボタンをクリックしたとき、マウス ボタンを離す前にボタンを含むウィンドウが閉じます。

このような問題を解決するには、以下の手順を実行します。

- **UseLegacyOnKeyUp=1** パラメータを **Intouch.ini** ファイルに挿入します。
- **[WindowViewer のプロパティ]** ダイアログ ボックスの **[メモリ内ウィンドウのキャッシュを使用する]** チェック ボックスをオンにします。

Diagnose

章 31 QuickScript のトラブルシューティング

Log Viewer を使用してタグ変数のランタイム値を表示することによって、QuickScript をトラブルシューティングできます。

Log Viewer へのメッセージのログ記録

Log Viewer は、QuickScript のデバッグに役立ちます。Operations Control Log Viewer は InTouch HMI をインストールする際にインストールされ、System Management Console にあります。

QuickScript をデバッグする 1 つの方法を以下に示します。

1. QuickScript でチェック ポイントを設定して、Log Viewer に値をログ記録します。
2. Operations Control Log Viewer を開いて、値を表示します。

別の方法として、タグ値を Log Viewer に記録するキー スクリプトを作成する方法があります。

QuickScript でチェック ポイントを設定するには

1. エラーの原因と考えられる QuickScript を開きます。
2. チェック ポイントを設定する行を検索します。
3. その行の後に、以下のいずれかのコードの一部を挿入します。
 - `LogMessage(messagetag);`
このスクリプトでは、messagetag は、値をログ記録するメッセージ型タグの名前です。
 - `LogMessage(StringFromIntg(inttag,10));`
このスクリプトでは、inttag は、値をログ記録する整数型タグの名前です。
 - `LogMessage(Text(realtag,"#.#####"));`
このスクリプトでは、realtag は、値をログ記録する実数型タグの名前です。
 - `LogMessage(DText(disctag,"TRUE","FALSE"));`
このスクリプトでは、disctag は、値をログ記録する論理型タグの名前です。
 - 識別子やタグ名など、チェックポイントで LogViewer に詳細情報をログ記録します。例:

```
LogMessage("DEBUG tag:"+ind.name+" value:"+Text(ind,"#.#####"));
```


このスクリプトでは、ind は、アナログ型間接タグです。

LogMessage() 関数

ユーザー定義のメッセージを Log Viewer に書き込みます。

カテゴリ

その他

構文

```
LogMessage("Message_Tag");
```

パラメータ

Message_Tag

Log Viewer にログ記録する文字列です。実際の文字列またはメッセージ型タグです。

備考

これは、InTouch スクリプトのトラブルシューティングに非常に役立つ関数です。LogMessage() 関数をスクリプトに戦略的に配置することで、QuickScript の実行順序やスクリプトのパフォーマンスを判断し、タグが変更される前と QuickScript によって影響を与えられた後のタグの値を識別できます。Log Viewer に記録される各メッセージには、正確な日付および時刻が示されます。

重要: パーセント (%) 文字は、スクリプトのデバッグ時に Log Viewer に表示される診断メッセージをフォーマットします。パーセント文字がログ文字列または関数パラメータ内にある場合、WindowViewer が応答しなくなることがあります。パーセント文字によって生じるエラーを排除するには、2 つのパーセント文字 (%%) を使用します。

例

```
LogMessage("Report Script is Running");
```

上記のステートメントは、以下の情報を Log Viewer に出力します。

```
94/01/14 15:21:14 WWSCRIPT Message:Report Script is Running.
```

```
LogMessage("The Value of MyTag is " + Text(MyTag, "#"));
```

```
MyTag = MyTag + 10;
```

```
LogMessage("The Value of MyTag is " + Text(MyTag, "#"));
```

Log Viewer メッセージの表示

Log Viewer は InTouch HMI をインストールする際にインストールされ、Operations Control Management Console にあります。

Log Viewer にログ記録された値を表示するには

1. [スタート] をクリックし、[プログラム] をポイントしてから [AVEVA] をポイントし、[System Platform Management Console] をクリックします。System Management Console が表示されます。
2. 左ペインで、[Log Viewer]、[デフォルトグループ] の順に展開し、[ローカル] をクリックします。Log Viewer メッセージが詳細ペインに表示されます。
3. LogMessage() 関数からログ記録された値を検索します。

注記: リモート InTouch HMI ノードでスクリプトをデバッグしている場合、ノード名を Log Viewer のノードグループに追加して、そのノードの Log Viewer メッセージを表示する必要があります。

章 32 エラー メッセージの理解

Tag Viewer では、要求の実行に失敗するとエラー メッセージが表示されます。このエラー メッセージは、エラーの説明を示すダイアログ ボックスに表示されます。

メイン ウィンドウ

機能	説明	原因
クイック 検索	検索されたタグ名／アラーム グループが見つかりませんでした。	システムは一致するタグ名/アラーム グループを見つけることができませんでした。
監視ウィンドウ	監視ウィンドウに追加できるリファレンスの最大数は 2000 です。	監視ウィンドウに 2000 個以上のアイテムを追加しようとしてしました。
	監視ウィンドウの最大数は 50 です。	50 個以上の監視ウィンドウを追加しようとしてしました。
	<File1> は有効な監視リスト ファイルではありません。	監視ウィンドウを開く際に、無効なファイル名を指定しました。
	<File1> がすでに存在します。置き換えますか?	監視ウィンドウを保存する際に、既存のファイル名を指定しました。
	現在の InTouch アプリケーションには以下のリファレンスは存在しません。またこれらのリファレンスは監視ウィンドウには追加されていませんでした。 <リファレンス リスト>	監視ウィンドウに読み込まれたリファレンスには、現在の InTouch アプリケーションで無効なものや存在しないものがあります。

タグ リファレンスを追加

機能	説明	原因
タグ リファレンスを追加	<abcd> に <f> が見つかりません	[変更] ボックスに文字列 <f> を入力しましたが、この文字列はタグ リファレンスの文字列 <abcd> にはありません。
	<始点> の値は <終点> の値以下である必要があります	[始点] ボックスの値よりも小さい値を、[終点] ボックスに入力しました。
	<始点> に数値を入力してください または <終点> に数値を入力してください	[変更] ボックスに文字列を入力しましたが、[始点] ボックスまたは [終点] ボックスのどちらかが空白です。

機能	説明	原因
	現在の InTouch アプリケーションには以下のリファレンスは存在しません。またこれらのリファレンスは監視ウィンドウには追加されていませんでした。 <リファレンス リスト>	無効なリファレンスを 1 つ入力したか、または入力した複数のリファレンスのうちのいくつかが無効です。
	監視ウィンドウに追加できるリファレンスの最大数は 2000 です。	監視ウィンドウに 2000 個以上のアイテムを追加しようとした。

タグ値の変更

機能	説明	原因
整数値を変更	<1a> は有効な数値ではありません。	[値] ボックスに無効なエントリである <1a> を入力して、[適用] をクリックしました。
	入力値は、整数型のデータ タイプの有効な範囲外です。	許容範囲外の数値を入力しました（有効な数値は -2147483648 ～ 2147483647）。
実数値を変更	入力値は、実数型データ タイプの有効な範囲外です。	許容範囲外の数値を入力しました（有効な数値は -3.402823466e38 ～ 3.402823466e38）。
[タグ リファレンス] ダイアログ ボックス	このアプリケーションにタグ リファレンス <InvalidReference> は存在しません。	[タグ リファレンス] ダイアログ ボックスに <InvalidReference> を入力して、[OK] をクリックしました。
	設定するデータは、誤ったデータ タイプのものです。	リファレンスは読み取り専用のタグですが、値を変更しようとした。 または タグ リファレンスの選択したタイプに対して無効な値を入力しました。 たとえば、整数型タグに対して文字列値を入力した場合などです。

監視ウィンドウの名前の変更

機能	説明	原因
監視ウィンドウ	<Watch List 1> という名前の監視ウィンドウは、すでに存在します。	監視ウィンドウの名前を変更する際に、既存の名前を入力しました。
	「ReadonlyReference」は読み取り専用であり、変更できません。	監視ウィンドウに読み取り専用タグを追加して、そのタグをダブルクリックして値を変更しようとした。

章 33 Managing the InTouch Web Client

このセクションには、以下の内容が含まれます。

[Web Client でのエラー処理](#)

[ブラウザとモバイルのサポート](#)

Web ブラウザでのグラフィックの表示に関するトラブルシューティング

[無効な証明書エラー](#)

Web Client でのエラー処理

Web Client は、予期されるエラーを表示する一貫したエラー ページが提供します。エラー ページは、以下のシナリオで表示されます。

- **アプリケーションがない** - WindowViewer で最後に実行したアプリケーションの必要なサポート ファイルが Web Client で検出されませんでした。必要なアプリケーションを WindowMaker で開きます。
- **権限が拒否された** - 現在ログインしているユーザーに Web ページを表示するための適切な権限がありません。ユーザーを aalInTouchUsers または aalInTouchRWUsers ユーザー グループに追加します。
- **使用可能なライセンスがない** - 現在の Web セッションの有効なライセンス ステータスが取得されませんでした。有効な Web Client ライセンスが使用可能であることを確認します。
- **高速切り替えモード** - Web Client アプリケーションは高速切り替えモードですが、ユーザーが <http://localhost/InTouch> 以外の URL を使用して Web Client にアクセスしようとしています。

ブラウザとモバイルのサポート

ブラウザのサポート:

InTouch Web クライアントは、以下の Web ブラウザでテストされています。

- Google Chrome
 - デスクトップ用 Chrome バージョン 64.0.3282.186 以降
 - Android スマートフォンおよびタブレット用 Chrome バージョン 64.0.3282.137 以降
- Microsoft Internet Explorer 11.0.9600.18861 以降
- Microsoft Edge
 - Microsoft Edge (Windows 10) バージョン 41.16299.15.0 以降
 - Microsoft Edge (Surface) バージョン 41.16299.248.0 以降
- Apple Safari for iOS バージョン 11.2.6

サポートされていないブラウザの場合はエラー メッセージが表示されます。

モバイルデバイスのサポート

InTouch Web Client は、以下のモバイルデバイスおよびそのデフォルト ブラウザをサポートします。

- Apple iPhone

- Apple iPad
- Android スマートフォンおよびタブレット
- Microsoft Surface

Web ブラウザでのグラフィックの表示に関するトラブルシューティング

Web Client が表示されない場合、以下の設定を確認してください。

- **InTouch Web** サービスがノード上で実行している。
- **InTouch Web** ページへのアクセスに使用するユーザー アカウントがいずれかのユーザー グループに含まれている。
- 同じマシン上で IIS が実行している場合、**InTouch Web Server** と競合する可能性があります。
 - IIS が実行していて使用されていない場合、IIS を停止してサービスを無効にします。IIS を実行する必要がある場合は、競合を防ぐためにポート 80 以外のポートを使用するよう設定します。

HTTP エラー 404.0 - 見つかりませんというエラーが表示された場合:

- 選択したノードで **Web Client** が有効化されていることを確認してください。管理者に連絡してください。

グラフィック プリミティブまたはアニメーションが表示されない場合、または予期した通りに動作しない場合:

- サポートされていない機能のリストをチェックして、**Web Client** でサポートされているかどうかを確認してください。
- <http://localhost/InTouch> から **Web Client** をローカルで実行し、シンボルを参照します。次に、互換性アイコンを探して、グラフィック内のサポートされていないアイテムのリストを表示します。
- サポートされていないグラフィックで使用されているプリミティブまたはアニメーションは削除され、メッセージがログに記録されます。記録されたメッセージには、生成された CSV レポート ファイルのパスが含まれます。詳細については、ログメッセージにアクセスしてレポートを参照してください。

選択したグラフィックが表示されていてもデータが表示されない場合やアニメーションが更新されない場合:

- グラフィックが **InTouch** タグをポイントする場合は、**InTouch WindowViewer** が実行していることを確認してください。
- グラフィックがアプリケーション オブジェクト参照をポイントする場合は、**Application Server** が実行していることを確認してください。
- **InTouch iData** サービスが実行していることを確認してください。

無効な証明書エラー

エラー: 証明書エラー: このサイトへの接続は安全ではありません。

Web Client にアクセスするときに有効な証明書がクライアントで使用できない場合、このエラーがブラウザに表示されます。

解決方法:

1. サーバー マシンに接続し、System Platform コンフィグレータを起動します。
2. System Management Server ページで [詳細] をクリックします。
[証明書] ダイアログ ボックスが表示されます。
3. [詳細] タブで [ファイルにコピー] をクリックします。
4. [証明書のエクスポート ウィザード] の指示に従ってファイルを保存します。
5. 証明書ファイルをクライアント マシンにコピーします。
6. クライアント マシンで、**Microsoft 管理コンソール (mmc)** を起動します。
7. 証明書をインポートします。
8. 証明書をインポートした後にブラウザを起動します。

Manage

章 34 AVEVA Operations Control 接続エクスペリエンス

Operations Control 接続エクスペリエンスについて

Operations Control 接続エクスペリエンスオプションでは、CONNECT クラウド機能を介して、特定のノードにおけるすべての Operations Control 製品間でのシングルサインオン (SSO) エクスペリエンスが有効になります。Operations Control 接続エクスペリエンスには、有効な Operations Control サブスクリプションおよびユーザー管理機能のある CONNECT アカウントが必要です。

- 統合ユーザー管理：ノード上で Operations Control 接続エクスペリエンスを有効にすると、そのノードのすべての対象 Operations Control 製品の動作が変更され、ノード上で最初の製品を起動するときに CONNECT でのログイン認証が必要になります。以降にノード上で起動する製品では CONNECT を使用した認証が行われ、認証済みユーザーはシングルサインオン (SSO) が可能になります。CONNECT ベースの承認は、Operations Control 接続エクスペリエンスで使用可能な唯一のセキュリティモードです。
- ノード間の互換性：Operations Control 接続エクスペリエンスは、システムのすべてのノードで有効にする必要があります。Operations Control 接続エクスペリエンスが有効化されていないノードで以前に構築されたアプリケーションは、Operations Control 接続エクスペリエンス環境で機能するように再設定する必要があります。

Operations Control 接続エクスペリエンスはいつでも無効にできますが、Operations Control 接続エクスペリエンスはシステムのすべてのノードで無効にする必要があります。Operations Control 接続エクスペリエンスの下で構築されたアプリケーションは、認証方法と製品ライセンスなど、Operations Control 接続エクスペリエンスが有効でない環境で機能するように再設定する必要があります。

Operations Control 接続エクスペリエンスの設定

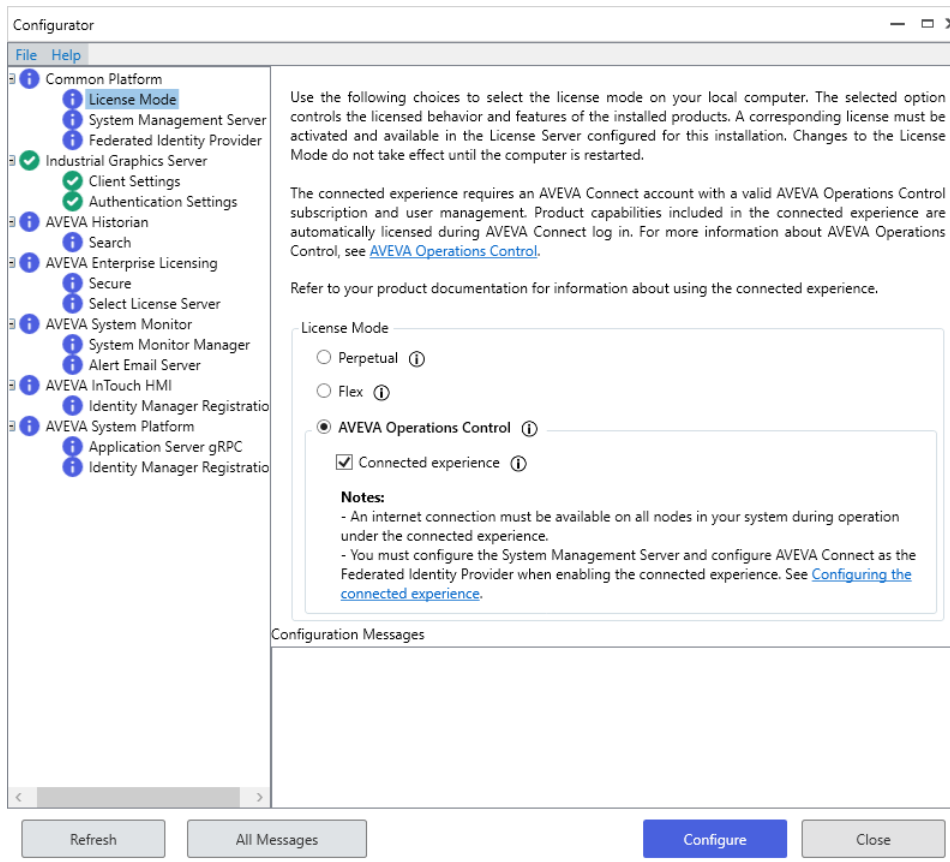
AVEVA Operations Control の下で InTouch およびその他の System Platform 製品を（製品サブスクリプションのみとして、または Operations Control 接続エクスペリエンスを使用して）運用するには、インストールされたライセンス済み製品を設定する必要があります。この場合でも、ヘッドレス サーバーが適切に機能するにはライセンス サーバーを設定する必要があります。

詳細については、以下を参照してください。

- [コンフィグレータ ヘルプ](#)
- [AVEVA Identity Manager ヘルプ](#)

Operations Control 接続エクスペリエンスを設定するには

1. [スタート] > [AVEVA] > [Configurator] からコンフィグレータを開きます。
2. 左側のペインで [ライセンス モード] を選択してライセンス モードを選択します。
 - a. 2 つの AVEVA Operations Control パッケージ (Edge および Supervisory) の少なくとも 1 つをサブスクリライブするには、[AVEVA Operations Control] ラジオ ボタンを選択します。これには、定義済みのユーザーのセットに対する製品パッケージのすべての製品の無制限の使用が含まれます。
 - b. CONNECT クラウド機能 (製品で既定で使用可能) を備えたノード上ですべての Operations Control 製品でシングルサインオン (SSO) エクスペリエンスを有効にするには、[接続エクスペリエンス] チェックボックスをオンにします。



3. System Management Server を設定するには、左側のペインの [共通プラットフォーム] で [System Management Server] を選択します。

注： System Management Server のユーザー資格情報の入力が必要の場合、DomainName\UserName の形式でユーザー名を入力します。ドメイン管理特権があってもローカル マシンの管理者でない場合は、ユーザー資格情報が要求されることがあります。System Management Server を設定するには、Administrators または aaAdministrators OS グループのメンバーである必要があります。

- a. 既存の System Management Server に接続するには、[既存の System Management Server に接続] を選択します。使用可能な System Management Server のリストから目的の System Management Server を選択します。リストには、ネットワーク上で System Management Server として機能するように設定されたすべてのマシンが表示されます。ほとんどの場合、1 つの System Management Server だけが表示されます。
- b. マシンを冗長 SSO (RSSO) サーバーとして設定するには、[既存の System Management Server に接続] で [このマシンを Redundant SSO サーバーとして設定] を選択します。

または

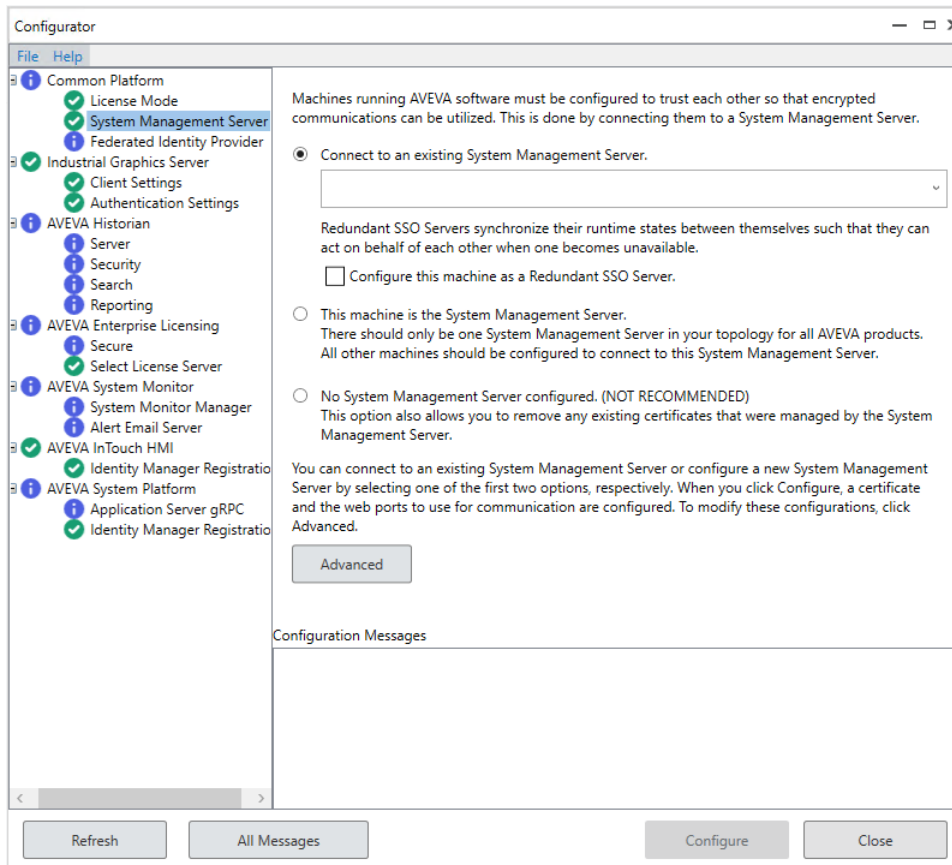
- c. 現在のマシンを System Management Server として設定するには、[このマシンを System Management Server にする] オプションを選択します。
- d. [設定] をクリックします。

注：

- Operations Control 接続エクスペリエンスでは「**System Management Server** は設定されていない」オプションはサポートされません。Operations Control 接続エクスペリエンスを使用するには **System Management Server** を設定する必要があります。

- ネットワークで **System Management Server** として識別および設定できるのは 1 つのマシンだけです。AVEVA 製品を実行するマシンは、その **System Management Server** に接続して信頼関係を確立し、暗号化された通信を構成できます。

- 冗長 SSO を設定しても別の **System Management Server** は設定されません。



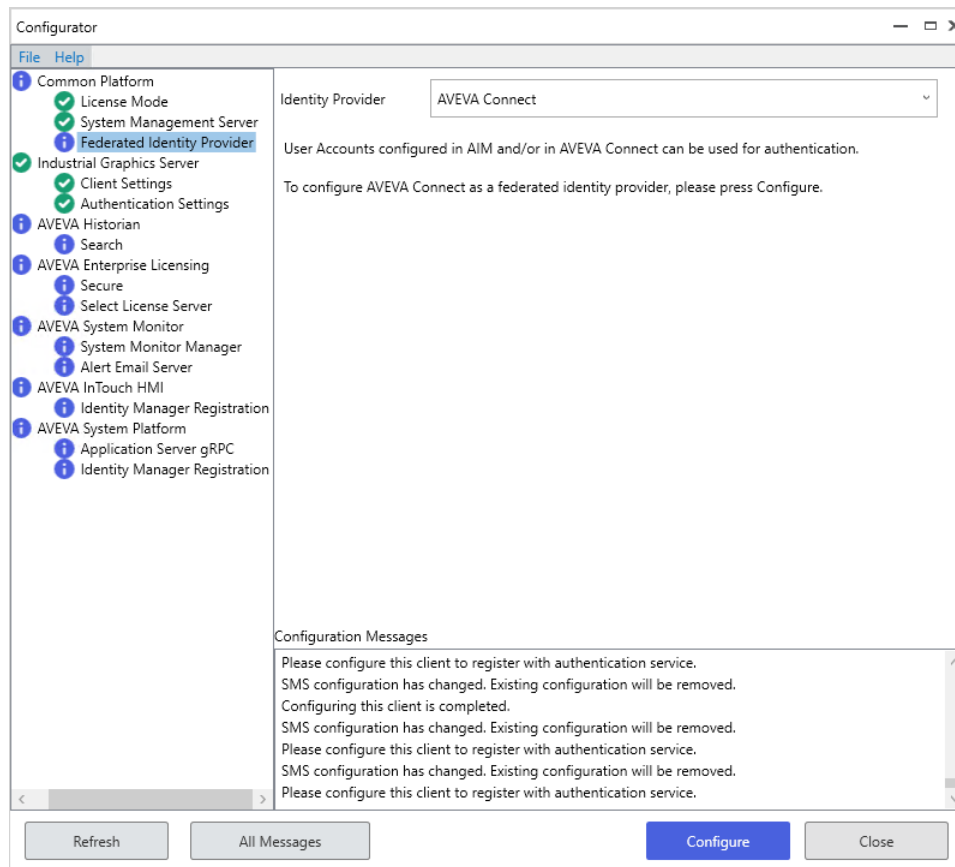
4. フェデレーション ID プロバイダを設定するには、左側のペインの「共通プラットフォーム」で「フェデレーション ID プロバイダ」を選択します。

System Management Server (SMS) および冗長 SSO (RSSO) マシンで使用可能な Platform Common Services (PCS) Framework の Identity Manager コンポーネントは、CONNECT との「フェデレーション」ログイン用に設定できます。したがって、ユーザーが CONNECT アカウントに登録されている E メールアドレスを Identity Manager のログインフォームに入力すると、ログインするために CONNECT にリダイレクトされます。現在、Operations Control 接続エクスペリエンスでは、CONNECT とのフェデレーションだけがサポートされています。Azure AD はサポートされていません。

注：

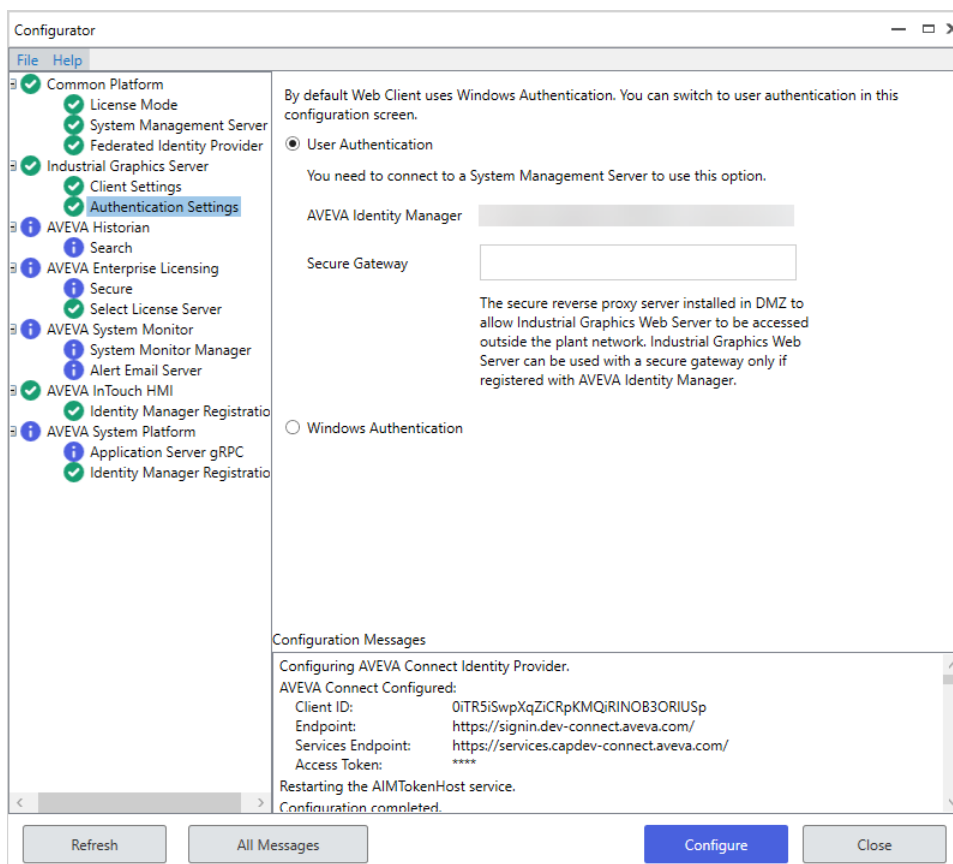
- フェデレーション ID 接続の場合、有効な CONNECT アカウントが必要です。また、ユーザーはそのアカウントの管理者である必要があります。

- この手順は、現在のマシンが **System Management Server** として設定されているノードにのみ適用されます。ノードが既存の **System Management Server** に接続する場合、フェデレーション ID プロバイダを設定する必要はありません。



複数の **CONNECT** アカウントがある場合、認証後、ユーザーが属する複数の **CONNECT** アカウントの名前の一覧を含むアカウント選択ダイアログが表示されます。フェデレーションを設定するアカウントを選択すると、それ以降は、マシンとすべての **Operations Control** 製品は、その選択したアカウントのサブスクリプションに対して認証または検証されます。**CONNECT** アカウントが 1 つだけの場合、アカウント選択ダイアログは表示されません。

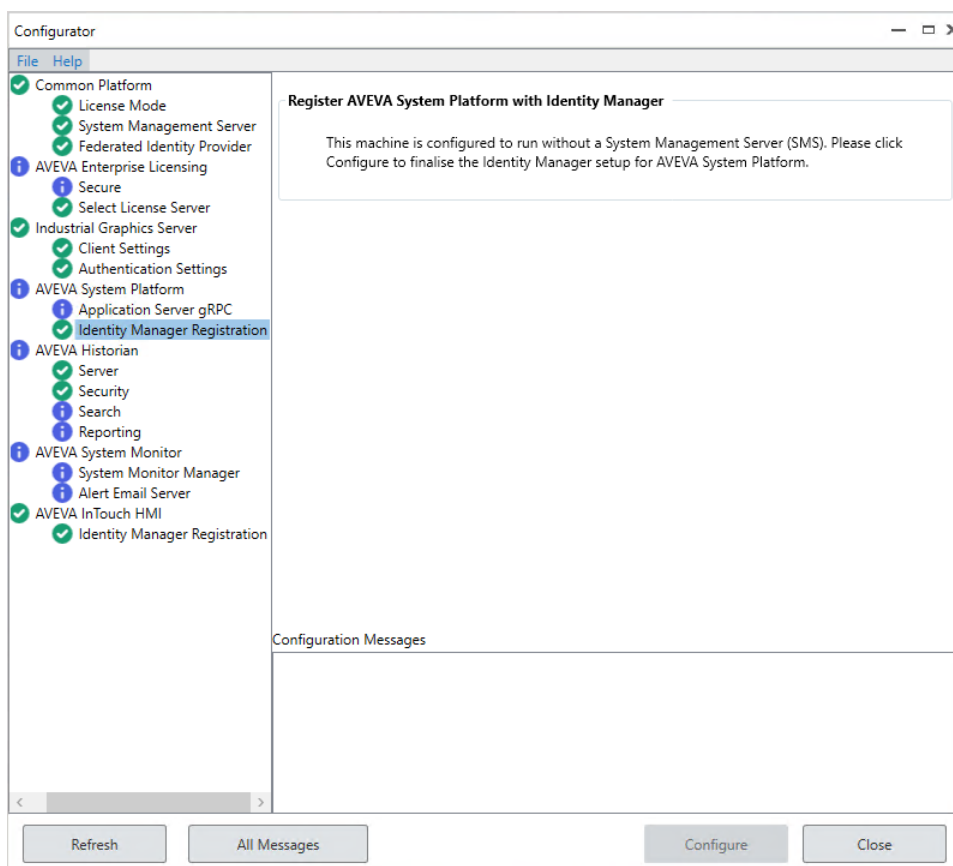
5. ユーザー認証を使用するように InTouch Web Client を設定するには、左側のペインの [産業用グラフィック サーバー] > [認証設定] から [ユーザー認証] を選択し、[設定] をクリックします。



6. System Platform を AVEVA Identity Manager に登録するには、左側のペインの [AVEVA System Platform] で [Identity Manager 登録] を選択します。

Identity Manager への AVEVA System Platform の登録を行う前に、必ず System Management Server の設定を完了してください。

[設定] をクリックして Identity Manager に登録します。



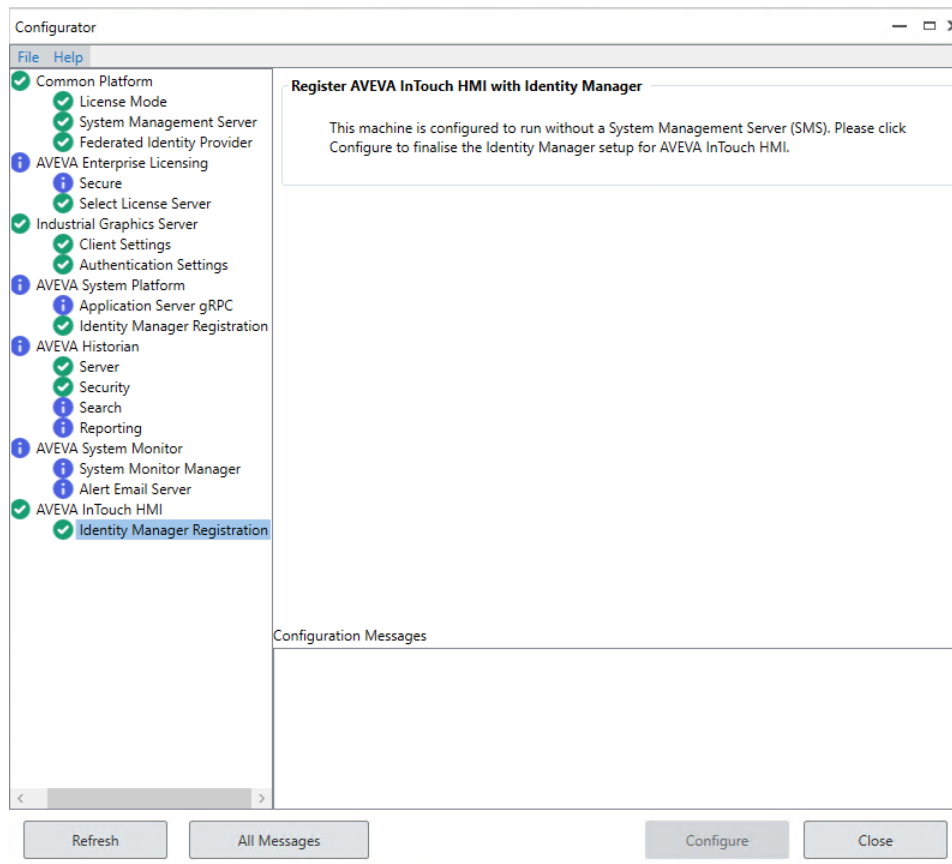
7. InTouch with AVEVA Identity Manager を登録するには、左側のペインの **[AVEVA InTouch HMI]** で **[Identity Manager 登録]** を選択します。

Identity Manager への InTouch HMI の登録を行う前に、必ず前の手順（下記参照）を完了してください。

- ライセンス モードとして **AVEVA Operations Control** 接続エクスペリエンスを有効化します。
- **System Management Server** を設定します。
- フェデレーション ID プロバイダで **CONNECT** を選択します。

[設定] をクリックして Identity Manager に登録します。

注：System Management Server を設定していない（非推奨）場合でも InTouch は設定できますが、Operations Control 接続エクスペリエンスは使用できません。



認証とエンタイトルメント

認証、エンタイトルメント、承認は、組織で使用する AVEVA 製品へのユーザーアクセスを制御するためのセキュリティエレメントです。

- **認証**とは、一意の資格情報のセット（ユーザー名とパスワード）を使用してユーザーの本人確認を行うことです。
- **エンタイトルメント**とは、組織がサブスクリプションを持っており、認証されたユーザーがアクセスを許可されている製品です。ユーザーが有効な **Operations Control** サブスクリプションを持っていない場合、ユーザーは認証されていても特定製品の使用を許可されない場合があります。
- **承認**とは、認証されたユーザーに関連付けられている権限のレベルのことです。通常、これはユーザーをユーザーグループに割り当てるか、フェデレーション ID 管理を介して行われます。

フェデレーション ID はフェデレーション ID プロバイダ (FIDP) を介して管理され、AVEVA 製品のポートフォリオ全体のユーザーに対する一元的なユーザー管理とシングルサインオンを可能にします。これはコンフィグレータを通じて有効になります。通常はインストール時に完了しますが、いつでも再設定できます。

選択したライセンスモードは、**System Platform** 環境における認証、エンタイトルメント、承認の処理において重要な役割を果たします。詳細については、「ライセンスモード」を参照してください。

Operations Control 接続エクスペリエンスが有効になっている場合、ユーザーが IDE、Historian、OMI ViewApp などの **System Platform** コンポーネントにログインするたびに、ログインが **CONNECT** 監査ログにイベントとして保存されます。

データ ログの表示

監査ログにより、組織と AVEVA 両方から承認されたユーザーが、アカウントで発生したイベントまたは操作のログを **CONNECT** ポータルに表示できます。ログには、選択した時間範囲に組織のアカウントで行われたすべての操作が表示されます。監査ログを表示するには、アカウント管理者、または **Report Viewer** の役割を持つユーザーである必要があります。

注：監査ログには、アカウントで監査機能が使用可能になった時点以降のデータが記録されます。

監査ログを表示するには、[CONNECT](#) の **Web** サイトに移動します。サイトナビゲーションメニューで [監査] を選択します。[監査ログ] ページが表示されます。

監査ログには以下の情報が表示されます。

- 操作が実行されたときのタイムスタンプ（日付と時刻）。
- ユーザー名、マシン名、アプリケーション名（例：AVEVA OMI や System Platform IDE）など、操作の詳細。ログに記録される情報には、どの認証されたユーザーがどの製品にアクセスしたかに加え、そのユーザーの使用可能なエンタイトルメントアクセスが含まれます。
- 監査ログデータは **CONNECT** のエンタイトルメントチェック機能を使用してフィルタリングできます。この機能は、接続エクスペリエンスを介して製品の使用状況に関連するすべてのデータが表示します。

注：データをユーザーインターフェイスに表示する際はローカル時間を使用されます。ただし、操作の詳細およびデータがエクスポートされる際は UTC 時間が表示されます。

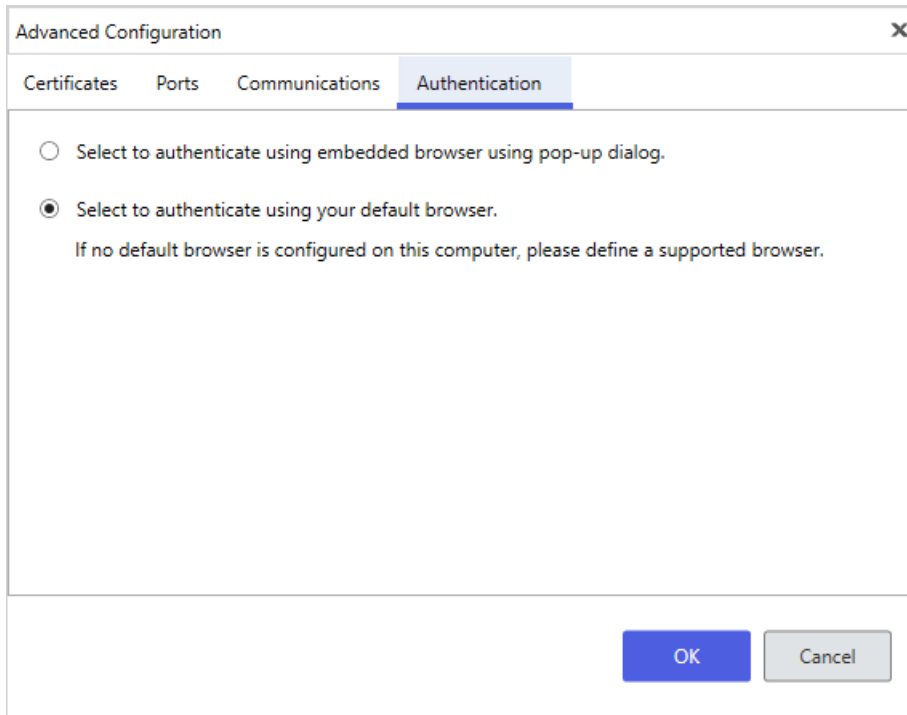
[監査ログ](#)の詳細については、**CONNECT** のヘルプを参照してください。

AVEVA Identity Manager を使用した認証

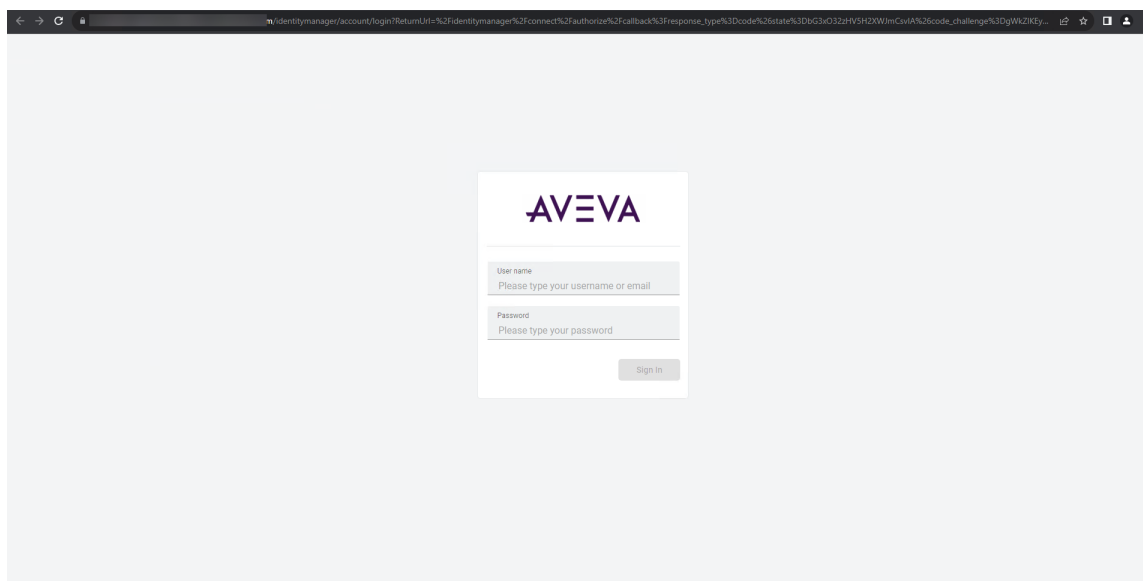
コンフィグレータで **Operations Control** 接続エクスペリエンスが有効化されている場合、アプリケーションマネージャ、WindowMaker、または WindowViewer の初回起動時に **AVEVA Identity Manager** での認証を受けるよう求めるメッセージが表示されます。その後の起動では、**SSO** での認証が行われます。

Web ブラウザで AVEVA Identity Manager を使用して認証するには

1. Web ブラウザで AVEVA Identity Manager サインイン画面を開くには、コンフィグレータで [共通プラットフォーム] > [System Management Server] > [詳細] > [認証] の [デフォルトブラウザを使用して認証する] を選択する必要があります。



2. いずれかの AVEVA Operation Control アプリケーションを起動すると、システムのデフォルトブラウザで AVEVA Identity Manager ログイン ページが開きます。
3. AVEVA Identity Manager へのログイン用に CONNECT アカウントに登録されている E メールアドレスを AVEVA Identity Manager ログインページの[ユーザー名]フィールドに入力します。
4. [パスワード] フィールドにパスワードを入力します。
5. [サインイン] をクリックします。



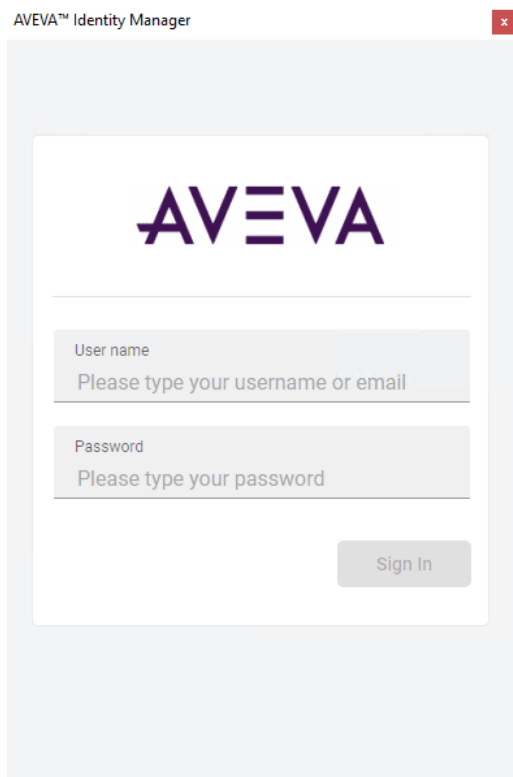
埋め込まれたブラウザで **AVEVA Identity Manager** を使用して認証するには

1. 埋め込まれたブラウザで AVEVA Identity Manager サインイン画面を開くには、コンフィグレータで [共通プラットフォーム] > [System Management Server] > [詳細] > [認証] の [ポップアップを使用して、埋め込まれたブラウザを使用して認証する] オプションを選択する必要があります。既定ではこのオプションが選択されます。

InTouch Access Anywhere の場合、セキュリティの理由から埋め込まれたブラウザを使用することをお勧めします。

The screenshot shows a configuration window titled "Advanced Configuration" with a close button (X) in the top right corner. Below the title bar are four tabs: "Certificates", "Ports", "Communications", and "Authentication". The "Authentication" tab is selected and highlighted. The main content area of the "Authentication" tab contains two radio button options: the first is "Select to authenticate using embedded browser using pop-up dialog." and is selected with a filled radio button; the second is "Select to authenticate using your default browser." and is unselected with an empty radio button. Below these options is a text instruction: "If no default browser is configured on this computer, please define a supported browser." At the bottom of the window are two buttons: "OK" (blue) and "Cancel" (gray).

2. いずれかの AVEVA Operation Control アプリケーションを起動すると、埋め込まれたブラウザで AVEVA Identity Manager ログイン ページが開きます。
3. AVEVA Identity Manager へのログイン用に CONNECT アカウントに登録されている E メールアドレスを AVEVA Identity Manager ログインページの[ユーザー名]フィールドに入力します。
4. [パスワード] フィールドにパスワードを入力します。
5. [サインイン] をクリックします。



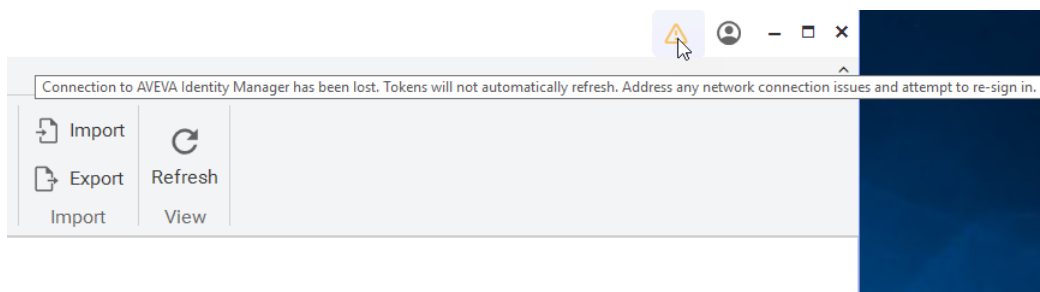
接続損失時の動作

接続が失われた場合:

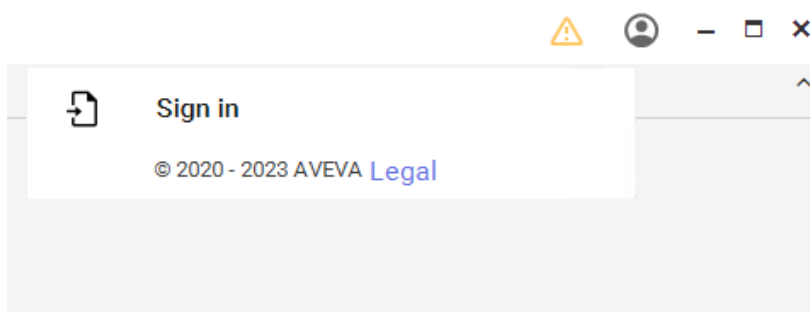
- 認証が正常に行われ、アプリケーションが正常に実行した後に AVEVA Identity Manager と CONNECT の間の接続が失われた場合でも、オンプレミス機能は引き続き機能します。
- 認証が正常に行われ、アプリケーションが正常に実行しているときに InTouch アプリケーションと AVEVA Identity Manager の間の接続が失われた場合、Operations Control 接続エクスペリエンス機能が停止します。[サインイン] オプションを使用して再度サインインする必要があります。
- 認証が正常に行われた後に接続が失われた場合、WindowMaker でアプリケーションを開こうとすると WindowMaker はデモ モードで実行します。
- 認証が正常に行われた後に接続が失われた場合、WindowViewer でアプリケーションを開こうとすると WindowViewer は読み取り専用モードで実行します。
- AVEVA Identity Manager にサインインしたことがない状態で接続が失われると、WindowMaker または WindowViewer を起動しようとする、WindowMaker は実行せず、WindowViewer はデモ モードで実行します。

接続損失の問題に対処するには

1. CONNECT または AVEVA Identity Manager への接続の問題がある場合、アプリケーションマネージャと WindowMaker に警告アイコンが表示され、接続状態が通知されます。警告アイコンの上にカーソルを置くと、接続損失に関するメッセージを含むツールヒントが表示されます。



2. ネットワーク接続の問題に対処してプロファイルアイコンをクリックし、[サインイン] をクリックして再度サインインします。



接続が回復してアプリケーションが再起動すると、ツールチップは表示されなくなります。

ライセンスとエンタイトルメント

Operations Control 接続エクスペリエンスでの InTouch のライセンス取得は、マネージドまたはスタンドアロンアプリケーションの場合と同じです。2 種類のサブスクリプションが使用可能です。

AVEVA Operations Control Edge サブスクリプション

InTouch アプリケーションが ViewApp として設定されておらず、コンフィグレータでライセンスモードが Operations Control 接続エクスペリエンスに設定されている場合、WindowViewer の起動にはこのサブスクリプションが必要です。

AVEVA Operations Control Supervisory サブスクリプション

InTouch アプリケーションが ViewApp として設定されており、コンフィグレータでライセンスモードが Operations Control 接続エクスペリエンスに設定されている場合、WindowViewer の起動にはこのサブスクリプションが必要です。

どちらのサブスクリプションでも、無制限の数のセッションが許可され、WindowMaker の完全な機能が有効になります。

データ ログの表示

監査ログには、アカウントのイベントまたは操作のログが含まれています。ログには、選択した時間帯にアカウントで行われたすべての操作が表示されます。監査ログを表示できるのは、アカウント管理者、または Report Viewer の役割に割り当てられたユーザーだけです。

注：監査ログには、アカウントで監査機能が使用可能になった時点以降のデータが記録されます。

監査ログを表示するには:

- サイト ナビゲーション メニューで [監査] を選択します。[監査ログ] ページが表示されます。

監査ログの以下の情報が表示されます。

- 日付 – 操作が実行された日
- メッセージ – 操作の詳細 (どのユーザーが何の操作を実行したかなど)
- 機能 – 操作によって影響を受ける機能

注: データをユーザーインターフェイスに表示する際はブラウザのローカル時間が使用されます。しかし、.csv ファイルにデータをエクスポートするときには UTC タイム ゾーンが使用されます。

詳細については、のヘルプを参照してください。

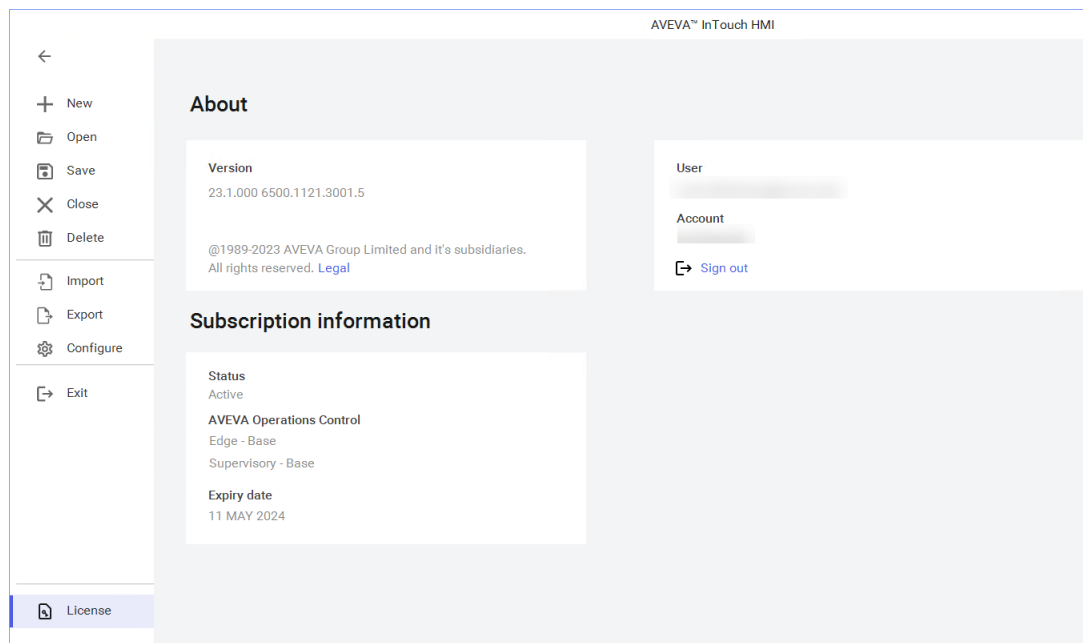
サブスクリプション情報の表示

WindowMaker または WindowViewer によって使用されている現在のサブスクリプションの具体的な情報を表示できます。

WindowMaker のサブスクリプション情報を表示するには

1. WindowMaker を開きます。
2. 画面左下隅にある [ファイル] メニューの [ライセンス] をクリックします。

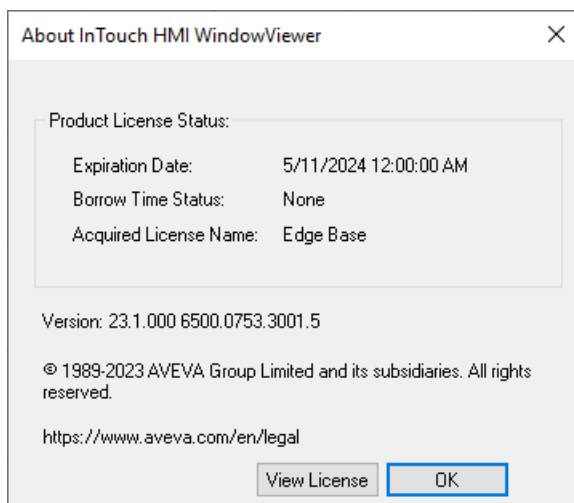
[バージョン情報] 画面が表示されます。[バージョン情報] 画面には、WindowMaker のバージョンおよびサブスクリプション情報が表示されます。



WindowViewer のサブスクリプション情報を表示するには:

1. WindowViewer を開きます。
2. [ファイル]、[WindowViewer のバージョン情報] の順にクリックします。

[WindowViewer のバージョン情報] ダイアログ ボックスが表示されます。



3. [ライセンスを表示] をクリックして、サブスクリプションの詳細を表示します。

WindowMaker の起動と実行

このセクションでは、Operations Control 接続エクスペリエンスでの WindowMaker の動作について説明します。

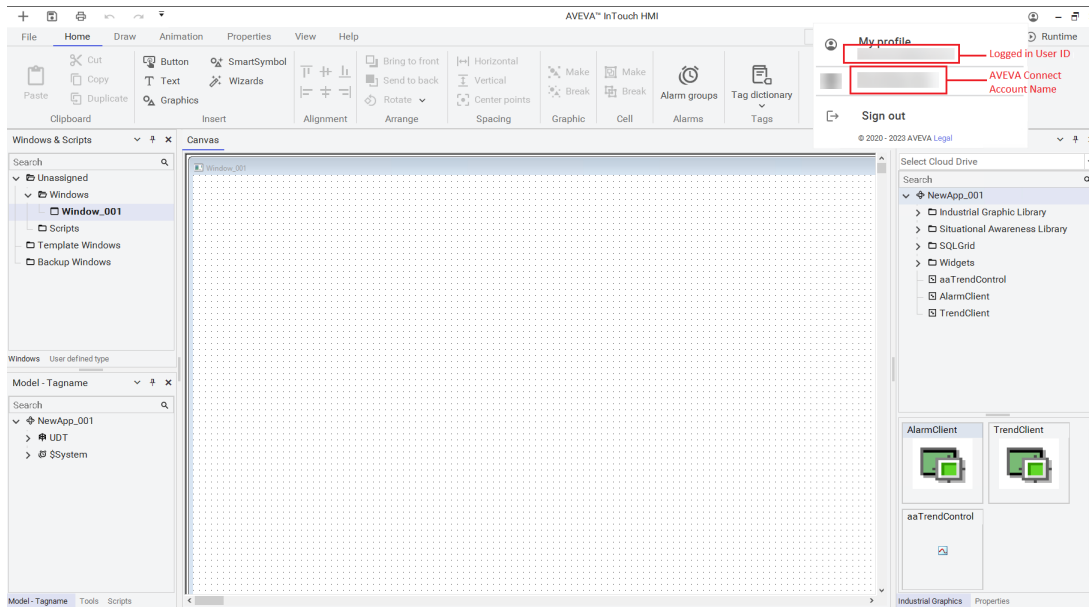
Operations Control 接続エクスペリエンスで WindowMaker を操作するには

1. WindowMaker を起動します。

以前に他の Operations Control 製品が起動していない状態で、特定のノードで認証を受けていない場合に同じノードで WindowMaker を初めて起動すると、(コンフィグレータで System Management Server の [認証] タグを設定する際に行った選択に基づいて) Web ブラウザまたは埋め込まれたブラウザを使用して AVEVA Identity Manager で認証するよう求めるメッセージが表示されます。このノードで認証済みの WindowMaker を既に起動している場合、または以前に同じノードで他の Operations Control 製品を起動して認証したことがある場合、シングルサインオンを使用した認証が行われます。詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

2. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。

正常にサインインした後、Edge または Supervisory サブスクリプションのエンタイトルメントがある場合、WindowMaker のタイトルバーにユーザー プロファイルのサインイン情報が表示されます。マイプロファイルには、ログインユーザーのユーザー ID と CONNECT アカウントの名前が表示されます。



3. 有効な資格情報が入力されない場合、または認証がキャンセルされた場合、エラーメッセージが表示されます。CONNECT 管理者に資格情報を確認してサインインをもう一度試してください。[OK]をクリックすると WindowMaker が閉じます。



- AVEVA Identity Manager での認証で起動する埋め込まれたブラウザのログイン ページを閉じると、認証失敗のダイアログが直ちに表示されます。
 - 資格情報が入力されない場合、または AVEVA Identity Manager での認証で起動した Web ブラウザを閉じた場合、3 分後に認証失敗のダイアログが表示されます。
4. Edge または Supervisory サブスクリプションがない場合、またはサブスクリプションの有効期限が切れている場合:
 - アプリケーションに設定されているタグが 64 個以下（システムタグを除く）で、設定されているウィンドウが 32 個以下の場合、WindowMaker はデモモードで実行されます。
 - アプリケーションに 64 個を超えるタグと 32 個以下のウィンドウが設定されている場合、WindowMaker は終了します。

CONNECT 管理者に問い合わせ有効なサブスクリプションを取得してください。サブスクリプションの詳細については、「」を参照してください。

5. WindowMaker が正常に実行している場合、AIM および CONNECT の更新トークンの有効性がチェックされます。

有効な AIM トークンが確認された場合、AVEVA Identity Manager と CONNECT の接続が失われるとオンライン機能が機能します。CONNECT 産業用グラフィックドライブにはアクセスできません。CONNECT の産業用ドライブにアクセスするには、AVEVA Identity Manager にサインインする必要があります。

WindowViewer の起動と実行

このセクションでは、Operations Control 接続エクスペリエンスでの WindowViewer の動作について説明します。

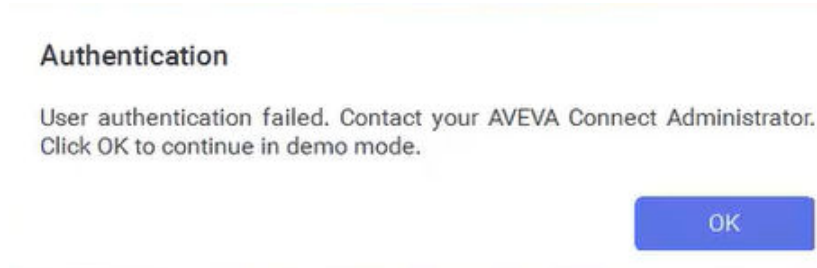
Operations Control 接続エクスペリエンスで WindowViewer を操作するには:

1. WindowViewer を起動します。

以前に他の Operations Control 製品が起動しておらず、特定のノードで認証を受けていない場合に同じノードで WindowViewer を初めて起動すると、(コンフィグレータで System Management Server の [認証] タグを設定する際に行った選択に基づいて) Web ブラウザまたは埋め込まれたブラウザを使用して AVEVA Identity Manager で認証するよう求めるメッセージが表示されます。このノードで認証済みの WindowViewer を既に起動している場合、または以前に同じノードで他の Operations Control 製品を起動して認証したことがある場合、シングルサインオンを使用した認証が行われます。詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

2. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。

3. 認証に失敗した場合、または以前にサインインしたことがない場合、WindowViewer はデモ モードで起動します。



4. 認証に失敗した場合、または認証をキャンセルした場合でも、以前にそのノードで少なくとも 1 回は正常に認証およびログインしたことがあれば WindowViewer は読み取り専用* モードで起動します。

- AVEVA Identity Manager での認証で起動する埋め込まれたブラウザのログイン ページを閉じると、認証失敗のダイアログが直ちに表示されます。
- 資格情報が入力されない場合、または AVEVA Identity Manager での認証で起動した Web ブラウザを閉じた場合、3 分後に認証失敗のダイアログが表示されます。

Authentication

User authentication failed. Contact your AVEVA Connect Administrator.
Click OK to continue in read-only mode.

OK

5. AVEVA Identity Manager との接続が失われた場合、WindowViewer は読み取り専用* モードで実行します。
6. 正常にサインインした後、Edge または Supervisory サブスクリプションがある場合、WindowViewer のタイトルバーにサインイン情報が表示されます。
7. 正常にサインインしても Edge または Supervisory サブスクリプションがない場合、以下の処理が行われます。
 - アプリケーションに設定されているタグが 64 以下の場合、WindowViewer はデモ モードで起動できます。アプリケーションで 64 以上のタグが設定されている場合、WindowViewer はデモ モードで開かずに終了します。
8. WindowViewer が正常に実行している場合、AVEVA Identity Manager および CONNECT の更新トークンの有効性がチェックされます。有効な AVEVA Identity Manager トークンだけを受け取り、AVEVA Identity Manager が CONNECT から切断された場合は次のようになります。
 - WindowViewer は読み取り専用モードで実行します。警告アイコンは表示されません。接続のステータスを確認するには、`GetTokenConnectionStatus()` 関数スクリプトを使用する必要があります。スクリプト メソッドを使用して、ユーザー インターフェイスおよびその適切なステータスを表すグラフィックを設定できます。接続が失われると、設定されているスクリプトが実行し、WindowViewer ユーザーには、現在の接続状態に関連付けられている適切なユーザー インターフェイスが表示されます。`GetTokenConnectionStatus()` スクリプトの詳細については、AVEVA™ InTouch HMI アプリケーション開発ガイド (ITBuild.pdf) の「[GetTokenConnectionStatus\(\) function](#)」セクションを参照してください。

注：

- Operations Control 接続エクスペリエンスでは、WindowMaker で [ファイル] > [設定] > [WindowViewer] > [ウィンドウ] の [ユーザーの設定] および [パスワードの変更] オプションは無効化されます。これらのオプションは Operations Control 接続エクスペリエンスで適用されないため、これらのオプションを WindowMaker で設定することはできません。これらのオプションはデフォルトでオフになり無効化され、WindowViewer で使用することはできません。

- サービスとしての WindowViewer は CONNECT のサブスクリプションを使用せず、従来の永続ライセンスが使用されます。

*読み取り専用モード:

読み取り専用とは I/O タグへの書き込みができない状態を指しますが、メモリタグへの書き込みは引き続き可能です。読み取り専用モードでは画面を表示できますが、変更を行うことはできません。このモードでは、認証が失敗した場合でもアプリケーションを WindowViewer で実行できます。この読み取り専用モードは WindowViewer 上の実装で、市場でサポートされている通常の読み取り専用モードとは異なります。この読み取り専用モードは、InTouch 製品の設定を有効にする読み取り専用は提供せず、読み取り専用ライセンスにも関連しません。

以前に少なくとも 1 回はサインインしたことがあり、サブスクリプションの有効期限が切れていないとき、以下のいずれかの条件に一致する場合、WindowViewer は読み取り専用モードで実行します。

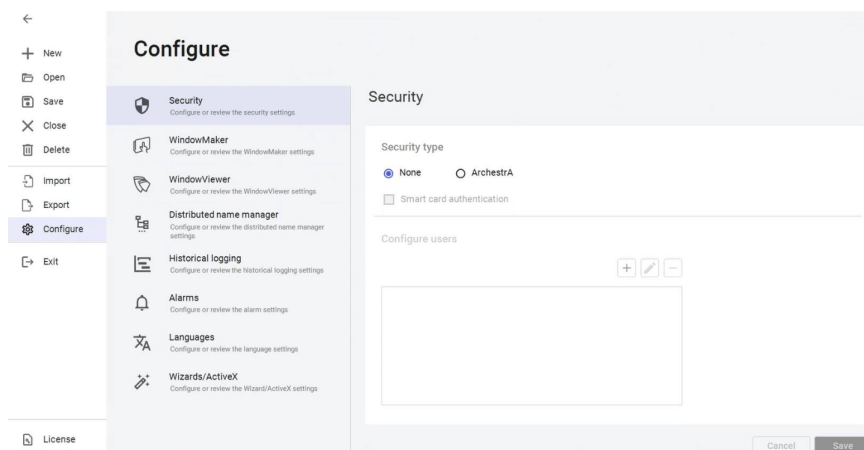
- 認証に失敗した
- 認証を望まない
- ネットワーク障害またはタイムアウトが発生した

サブスクリプションの有効期限が切れると、WindowViewer はデモ モードで実行します。

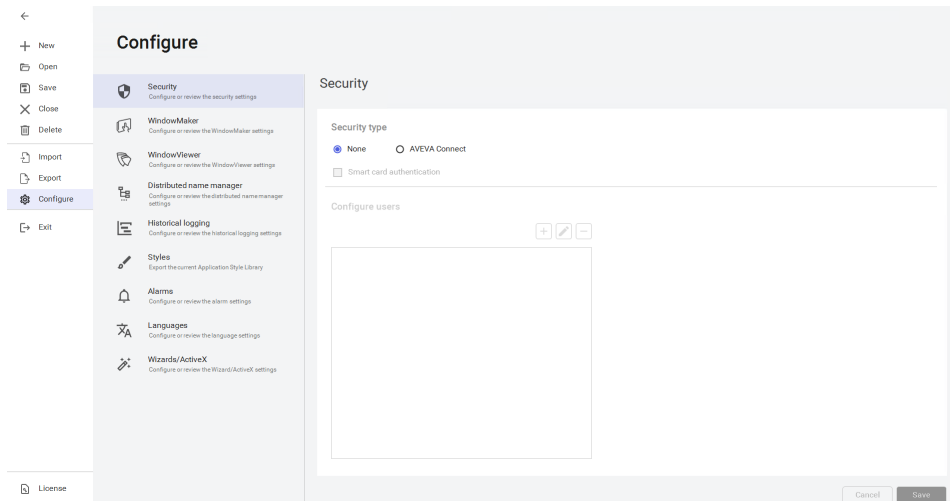
Operations Control 接続エクスペリエンスでのアプリケーション セキュリティの設定

WindowMaker が Operations Control 接続エクスペリエンス モードで機能するには、以下のいずれかのセキュリティ タイプを設定する必要があります。

- [マネージドアプリケーションでのセキュリティ設定](#)
 - なし
 - ArchestrA ([認証モード] タブの CONNECT ベースのセキュリティグループ設定を System Platform IDE で設定する必要があります。詳細については、「[マネージドアプリケーションでのセキュリティ設定](#)」を参照してください。



- [スタンドアロン アプリケーションでのセキュリティ設定](#)
 - なし
 - AVEVA Connect

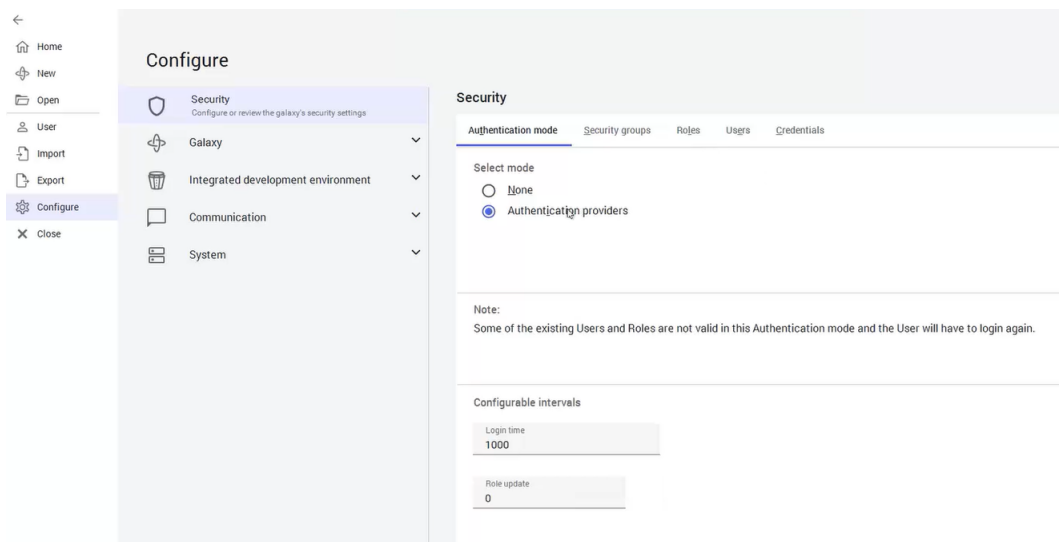


Operations Control 接続エクスペリエンス モードが有効で、互換性のないセキュリティ タイプが設定されている場合、WindowViewer または WindowMaker を開くと、上記のサポートされているセキュリティ タイプのいずれかで WindowMaker のセキュリティを再設定するよう求めるメッセージが表示されます。[OK] をクリックすると WindowViewer が閉じ、セキュリティ タイプを編集するオプションが WindowMaker に表示されます。

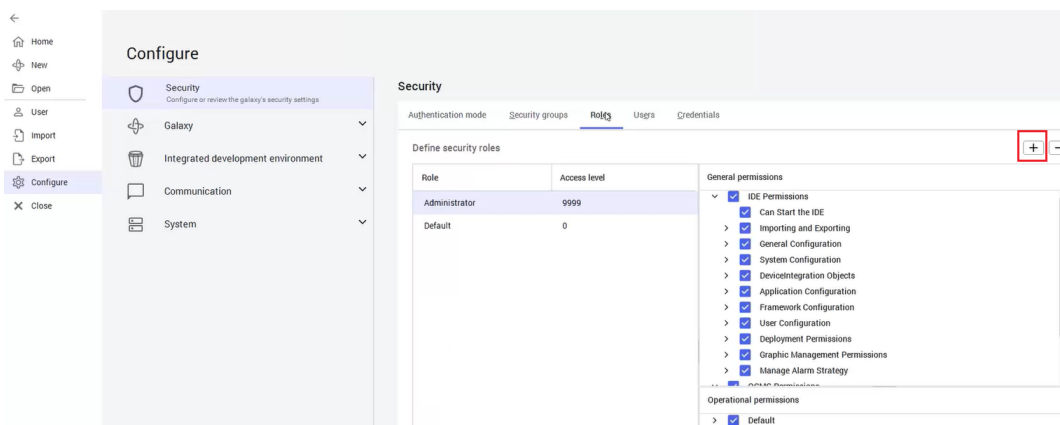
マネージド アプリケーションでのセキュリティ設定

マネージド アプリケーションでセキュリティを **ArchestrA** として設定するには

1. System Platform IDE を起動します。
2. [Galaxy] > [設定] > [セキュリティ] に移動して [セキュリティ] ページを開きます。[認証モード] タブで [認証プロバイダ] オプションを選択します。



3. アクセス レベルを CONNECT グループにマップするには、[設定]>[セキュリティ]>[役割]に移動してプラスアイコンをクリックします。



4. [リストから選択]ドロップダウンで[AVEVA Connect]が選択されていることを確認します。CONNECTのアカウントで利用できるグループのリストが表示されます。

Select groups

Enter the Authentication provider group name
<Authentication provider name>\<groupname> +

Select from the list
AVEVAConnect ▼

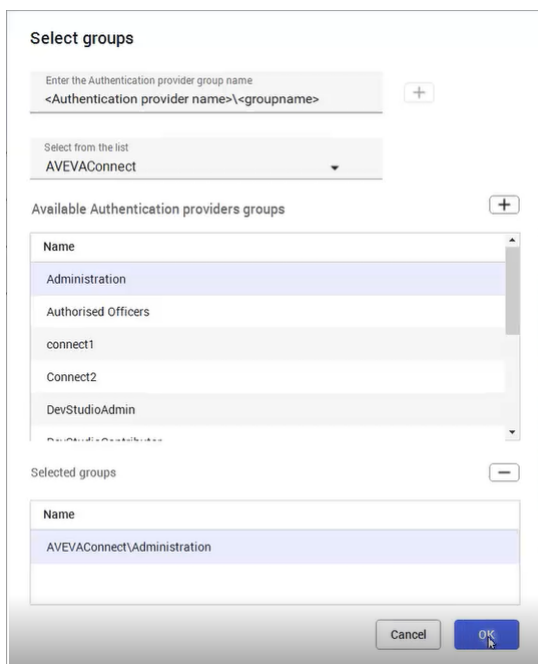
Available Authentication providers groups +

Name
Administration
Authorised Officers
connect1
Connect2
DevStudioAdmin
DevStudioContributor

Selected groups +

Name

5. 使用可能なリストから目的のグループを追加すると、[選択されたグループ] リストに表示されます。終了したら [OK] をクリックします。



6. グループに対して適切なアクセス レベル、一般権限、および操作権限を設定し、[保存] をクリックします。
7. ViewApp を選択して WindowMaker を起動します。
8. WindowMaker の [ファイル] > [設定] > [セキュリティ] でセキュリティ オプションとして [Archestra] を選択し、[保存] をクリックします。

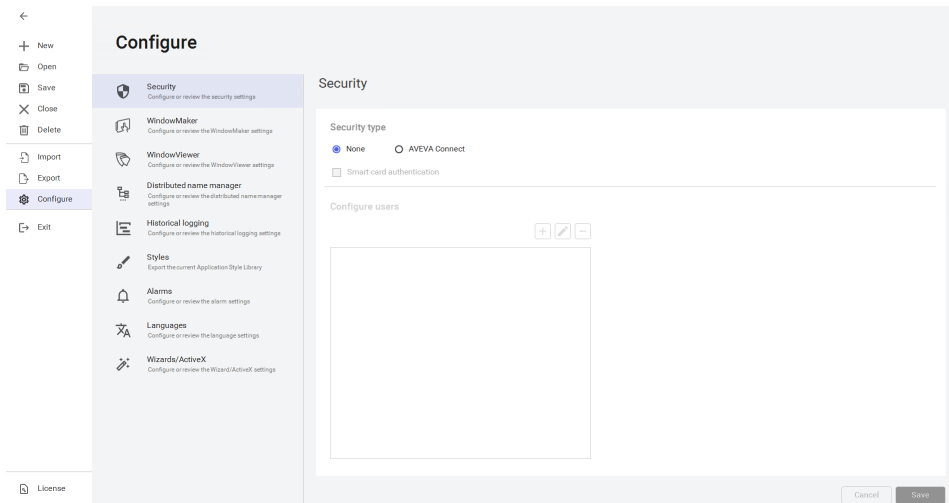
マネージドアプリケーションでセキュリティを「なし」として設定するには

1. System Platform IDE を起動します。
2. ViewApp を選択して WindowMaker を起動します。
3. [ファイル] > [設定] > [セキュリティ] でセキュリティ オプションとして [なし] を選択し、[保存] をクリックします。

スタンドアロンアプリケーションでのセキュリティ設定

スタンドアロンアプリケーションでセキュリティを設定するには

1. WindowMaker の[ファイル]>[設定]>[セキュリティ]でセキュリティオプションとして[なし]または [AVEVA Connect]を選択し、[保存]をクリックします。



Operations Control 接続エクスペリエンスが有効で、互換性のないセキュリティタイプが設定されている場合、WindowViewer または WindowMaker を開くと、上記のサポートされているセキュリティタイプのいずれかで WindowMaker のセキュリティを再設定するよう求めるメッセージが表示されます。**[OK]**をクリックすると WindowViewer が閉じ、セキュリティタイプを編集するオプションが WindowMaker に表示されます。**[InTouch]**と**[オペレーティングシステム]**の各セキュリティタイプは Operation Control 接続エクスペリエンスではサポートされていません。また、非接続エクスペリエンスでは CONNECT セキュリティタイプを使用できません。

2. スタンドアロン InTouch でグループにアクセス レベルにマップするには、スクリプト関数 **AddPermission()** を使用します。

AddPermission() メソッドが Operations Control 接続エクスペリエンスで受け入れるのは2つのパラメータだけです。

- AVEVA Connect グループ
- アクセス レベル

Operations Control 接続エクスペリエンスの AddPermission() のスクリプト関数を以下に示します。

```
DiscreteTag=AddPermission("", "AVEVA Connect group", AccessLevel);
```

ランタイムユーザーが CONNECT の複数のグループのメンバーである場合、アクセスレベルは最上位のアクセスレベルを持つグループによって決定されます。

混合モードでの操作

このセクションでは、セキュリティタイプに Operations Control 接続エクスペリエンスとの互換性がない場合の WindowMaker および WindowViewer の動作について説明します。

互換性のないセキュリティタイプでの WindowMaker の起動:

1. WindowMaker を起動します。

以前に他の Operations Control 製品が起動していない状態で、特定のノードで認証を受けていない場合に同じノードで WindowMaker を初めて起動すると、(コンフィグレータで System Management Server の**【認証】**タグを設定する際に行った選択に基づいて) Web ブラウザまたは埋め込まれたブラウザを使用して AVEVA Identity Manager で認証するよう求めるメッセージが表示されます。このノードで認証済みの WindowMaker を既に起動している場合、または以前に同じノードで他の

Operations Control 製品を起動して認証したことがある場合、シングルサインオンを使用した認証が行われます。詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

2. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。
3. 正常にサインインして有効なエンタイトルメントがある場合、セキュリティ タイプに互換性がなければ、セキュリティを設定するよう求めるメッセージが表示されます。

注：[InTouch]および[OS]の各セキュリティタイプは Operation Control 接続エクスペリエンスではサポートされていません。

4. [はい] をクリックして続行し、セキュリティ タイプを変更します。[いいえ] をクリックすると終了します。

セキュリティ タイプを変更するには、WindowMaker でセキュリティを設定します。セキュリティの設定方法の詳細については、[Operations Control 接続エクスペリエンスでのアプリケーションセキュリティの設定](#)」を参照してください。

注：スクリプト関数に AVEVA Operations Control 接続エクスペリエンスとの互換性があることを確認する必要があります。詳細については、「[サポートされている InTouch HMI 機能](#)」セクションを参照してください。

互換性のないセキュリティ タイプでの WindowViewer の起動:

1. WindowViewer を起動します。
2. Web ブラウザを使用して AVEVA Identity Manager で認証を受けるよう求めるメッセージが表示されます。
3. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。
4. 正常にサインインして有効なエンタイトルメントがある場合、セキュリティ タイプに互換性がなければ、WindowMaker でセキュリティを設定するよう求めるメッセージが表示されます。[OK] をクリックして WindowViewer を終了します。

Incompatible security configuration

This application's security type is not compatible with connected experience. Please open WindowMaker and change the security setting to a supported type. Click OK to close the application.

OK

注：[InTouch]および[OS]の各セキュリティタイプは Operation Control 接続エクスペリエンスではサポートされていません。

5. WindowMaker でセキュリティ タイプを設定します。セキュリティの設定方法の詳細については、[Operations Control 接続エクスペリエンスでのアプリケーションセキュリティの設定](#)」を参照してください。

6. WindowViewer を起動します。

Application Manager の起動と実行

このセクションでは、Operations Control 接続エクスペリエンスでのアプリケーション マネージャの動作について説明します。

Operations Control 接続エクスペリエンスでアプリケーション マネージャを操作するには

1. アプリケーション マネージャを起動します。

以前に他の Operations Control 製品が起動しておらず、特定のノードで認証を受けていない場合に同じノードでアプリケーション マネージャを初めて起動すると、(コンフィグレータで System Management Server の [認証] タグを設定する際に行った選択に基づいて) Web ブラウザまたは埋め込まれたブラウザを使用して AVEVA Identity Manager で認証するよう求めるメッセージが表示されます。このノードで認証済みのアプリケーション マネージャを既に起動している場合、または以前に同じノードで他の Operations Control 製品を起動して認証したことがある場合、シングル サインオンを使用した認証が行われます。詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

2. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。

- 認証が正常に行われた場合、以下の処理が行われます。
 - 以降のログインはシングル サインオンで認証されます。
 - アプリケーション マネージャのリボンにログイン ユーザーの情報が表示されます。
 - タグのパブリッシュ、AVEVA 産業用グラフィック ドライブと資格情報マネージャへのアクセス、読み取り専用または読み取り/書き込みモードでの View アプリケーションの実行、および DBLoad の実行が可能です。
- 認証に失敗した場合、または認証がキャンセルされた場合、以下の処理が行われます。

Authentication

User authentication failed. Contact your AVEVA Connect Administrator.

OK

- アプリケーション マネージャはログイン ユーザーなしの状態で引き続き表示されます。
- WindowMaker または WindowViewer を起動するには、AVEVA Identity Manager で認証を受ける必要があります。
- WindowViewer は読み取り専用モードでのみ実行できます。読み取り専用モードではアプリケーションを表示できますが、変更を行うことはできません。WindowMaker は終了します。
- AVEVA Identity Manager での認証で起動する埋め込まれたブラウザのログイン ページを閉じると、認証失敗のダイアログが直ちに表示されます。
- 資格情報が入力されない場合、または AVEVA Identity Manager での認証で起動した Web ブラウザを閉じた場合、3 分後に認証失敗のダイアログが表示されます。

アプリケーション マネージャからのアプリケーションの選択と実行

このセクションでは、Operations Control 接続エクスペリエンスでアプリケーション マネージャを使用してアプリケーションを開く動作について説明します。

Operations Control 接続エクスペリエンスでアプリケーション マネージャを使用してアプリケーションを開くには:

1. アプリケーション マネージャを起動します。アプリケーション マネージャを起動するには、[スタート] をクリックし、[プログラム] をポイントして、[AVEVA InTouch HMI] をポイントし、次に [InTouch HMI アプリケーション マネージャ] をクリックします。

以前に他の Operations Control 製品が起動しておらず、特定のノードで認証を受けていない場合に同じノードでアプリケーション マネージャを初めて起動すると、(コンフィグレータで System Management Server の [認証] タグを設定する際に行った選択に基づいて) Web ブラウザまたは埋め込まれたブラウザを使用して AVEVA Identity Manager で認証するよう求めるメッセージが表示されます。このノードで認証済みのアプリケーション マネージャを既に起動している場合、または以前に同じノードで他の Operations Control 製品を起動して認証したことがある場合、シングルサインオンを使用した認証が行われます。詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

AVEVA アプリケーション マネージャが開きます。

2. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。認証が正常に完了するか認証に失敗するとアプリケーション マネージャが起動します。
3. 実行するアプリケーションを選択します。

前の手順で認証に失敗した場合、AVEVA Identity Manager で認証を受けるよう求めるメッセージが表示されます。

認証が成功し、アプリケーションのセキュリティ タイプに Operations Control 接続エクスペリエンスとの互換性がない場合、セキュリティ タイプに互換性がないことを示すメッセージが表示されます。このメッセージでセキュリティ タイプを変更するか、終了するかを選択する必要があります。

4. [はい] をクリックすると、アプリケーションが WindowMaker で開きます。
5. [セキュリティ タイプ] を設定します。

セキュリティ タイプの詳細については、「[Operations Control 接続エクスペリエンスでのアプリケーションセキュリティの設定](#)」を参照してください。

InTouch Access Anywhere の起動と実行

このセクションでは、Operations Control 接続エクスペリエンスでの InTouch Access Anywhere の動作について説明します。

Operations Control 接続エクスペリエンスで InTouch Access Anywhere を操作するには

1. Web ブラウザを使用して InTouch Access Anywhere を起動します。以下のアドレスにアクセスします。

`http://ITAA_Server_Node_Name:8080/`

または

http://ITAA_Server_IP_Address:8080/

1. ログオンページでユーザー名とパスワードを入力し、表示する InTouch アプリケーションを [アプリケーション名] フィールドのドロップダウンリストから選択します。
2. [接続] をクリックして InTouch Access Anywhere ブラウザセッションを開始します。

以前に他の Operations Control 製品が起動しておらず、特定のノードで認証を受けていない場合に同じノードで InTouch Access Anywhere を初めて起動すると、(コンフィグレータで System Management Server の [認証] タグを設定する際に行った選択に基づいて) Web ブラウザまたは埋め込まれたブラウザを使用して AVEVA Identity Manager で認証するよう求めるメッセージが表示されます。セキュリティの理由から埋め込まれたブラウザを使用することをお勧めします。このノードで認証済みの InTouch Access Anywhere を既に起動している場合、または以前に同じノードで他の Operations Control 製品を起動して認証したことがある場合、シングルサインオンを使用した認証が行われます。詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

3. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。認証が正常に完了すると、WindowViewer が起動します。
4. 認証が正常に行われた場合:
 - 以降のサインインはシングルサインオンで認証されます。
 - WindowViewer では、Edge エンタイトルメントアクセスが使用可能であることが検証されます。
 - エンタイトルメントが確認された場合、WindowViewer では Edge の使用が CONNECT の Web サイトの[監査ログ]ページに記録されます。詳細については、のヘルプを参照してください。

ネットワーク アプリケーション開発 (NAD) アプリケーションの起動

ネットワーク アプリケーション開発 (NAD) は Operations Control 接続エクスペリエンスをサポートします。NAD が Operations Control 接続エクスペリエンスで機能するには、システムのすべてのノードで Operations Control 接続エクスペリエンスを有効にする必要があります。

Operations Control 接続エクスペリエンスで NAD アプリケーションを操作するには:

1. NAD クライアントを起動します。

以前に他の Operations Control 製品が起動しておらず、特定のノードで認証を受けていない場合に同じノードで NAD を初めて起動すると、(コンフィグレータで System Management Server の [認証] タグを設定する際に行った選択に基づいて) Web ブラウザまたは埋め込まれたブラウザを使用して AVEVA Identity Manager で認証するよう求めるメッセージが表示されます。このノードで認証済みの NAD を既に起動している場合、または以前に同じノードで他の Operations Control 製品を起動して認証したことがある場合、シングルサインオンを使用した認証が行われます。詳細については、「[AVEVA Identity Manager を使用した認証](#)」セクションを参照してください。

2. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。

正常にサインインし、Edge または Supervisory サブスクリプションがある場合、シングルサインオン (SSO) によるサインインが行われ、アプリケーションが実行します。以降のログインでは資格情報を再度入力する必要はありません。

注:

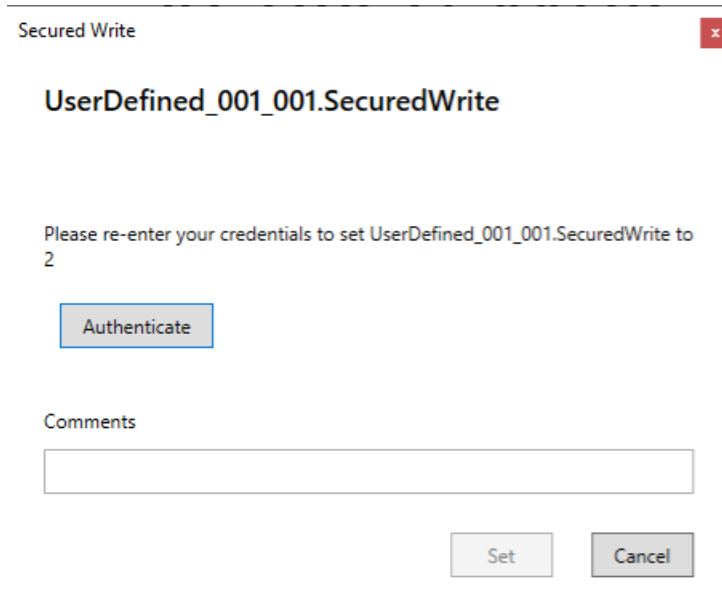
-サインインが正常に行われ、有効なエンタイトルメントがない場合、管理者に有効なエンタイトルメントを要求してください。サインインに失敗した場合、エラーメッセージが表示されます。管理者に有効な資格情報を問い合わせてください。

-マシンの接続が失われた場合、再認証を行うことができなくなるので、アプリケーションを実行できなくなります。

セキュリティ書き込みおよび確認書き込みの実行

実行セキュリティ書き込みの実行:

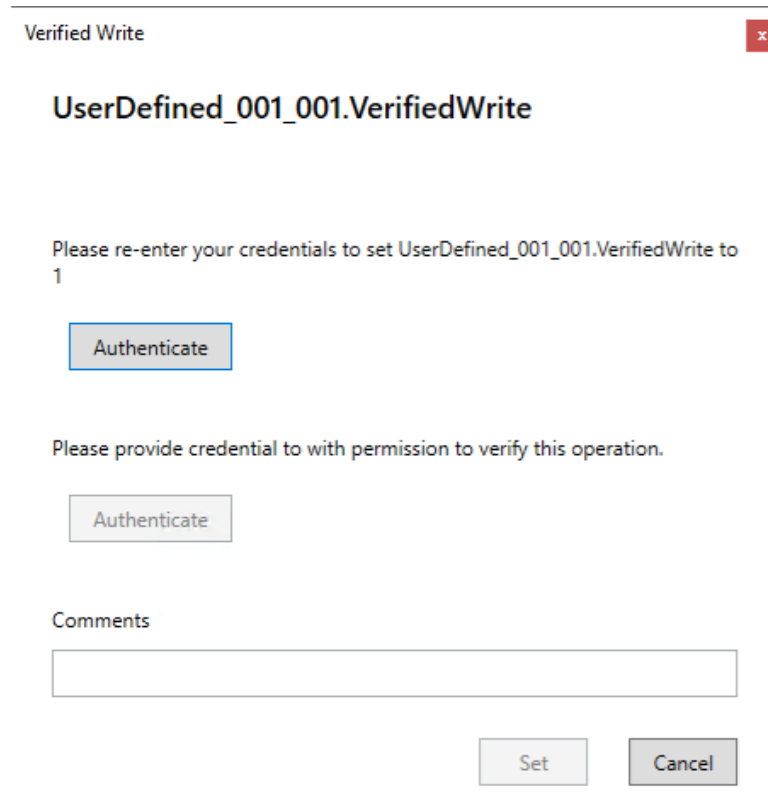
1. 値を変更します。
2. [セキュリティ書き込み] ダイアログが表示されます。
3. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。



注：セキュリティ書き込み操作では、シングルサインオンはサポートされていません。したがって、オペレータはセキュリティ書き込み操作で視覚情報を明示的に入力する必要があります。

確認書き込みの実行:

1. 値を変更します。
2. [確認書き込み] ダイアログが表示されます。
3. オペレータは[認証]をクリックし、AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。
4. 確認者は[認証]をクリックし、AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。



Verified Write

UserDefined_001_001.VerifiedWrite

Please re-enter your credentials to set UserDefined_001_001.VerifiedWrite to 1

Authenticate

Please provide credential to with permission to verify this operation.

Authenticate

Comments

Set Cancel

注：確認書き込み操作では、シングルサインオンはサポートされていません。したがって、オペレータと確認者の両方は確認書き込み操作で視覚情報を明示的に入力する確認必要があり両方です。

Web Client（オンプレミス）の起動と実行

コンフィグレータで Operations Control 接続エクスペリエンスが有効化されている場合、Web Client の初回起動時に AVEVA Identity Manager での認証を受けるよう求めるメッセージが表示されます。サインインには AVEVA ID とパスワードを使用する必要があります。Web Client のその後の起動では、シングルサインオン（SSO）で認証されます。その他の AVEVA アプリケーション（アプリケーションマネージャや WindowMaker など）を起動したときに AVEVA Identity Manager にサインインしていて、有効なエンタイトルメントと権限がある場合、SSO を使用したサインインが行われます。デスクトップアプリケーションと Web Client の間の SSO は、デスクトップアプリケーションで認証にシステムブラウザが使用されている場合にのみ機能します。SSO はブラウザ固有です。したがって、たとえば、1 つのブラウザ（Google Chrome）に資格情報を入力した後に別のブラウザ（Microsoft Edge）で Web Client を開いた場合、または Web Client をシークレットモードで開いた場合、資格情報をもう一度入力する必要があります。

注：System Platform 2023 から System Platform 2023 R2 にアップグレードする際、System Platform 2023 で AVEVA Identity Manager を既に設定している場合、Web Client が Operations Control 接続エクスペリエンスで機能するようにするには、アップグレードした後に Windows 認証に切り替え、次にユーザー認証に切り替えて、ユーザー認証を再設定する必要があります。最初に [産業用グラフィック サーバー] > [認証設定] で [Windows 認証] を選択し、[設定] をクリックします。次に、[産業用グラフィック サーバー] > [認証設定] で [ユーザー認証] を選択し、[設定] をクリックします。

Operations Control 接続エクスペリエンスで Web Client（オンプレミス）を操作するには:

1. Operations Control 接続エクスペリエンスで Web Client を有効にするには、コンフィグレータの左側のペインで [産業用グラフィック サーバー] > [認証設定] に移動し、[ユーザー認証] を選択して [設定] をクリックします。
2. ブラウザで Web Client を起動します。

以前に他の Operations Control 製品が起動しておらず、特定のノードで認証を受けていない場合に同じノードで Web Client を初めて起動すると、（コンフィグレータで System Management Server の [認証] タグを設定する際に行った選択に基づいて）Web ブラウザまたは埋め込まれたブラウザを使用して AVEVA Identity Manager で認証するよう求めるメッセージが表示されます。このノードで認証済みの Web Client を既に起動している場合、または以前に同じノードで他の Operations Control 製品を起動して認証したことがある場合、シングルサインオンを使用した認証が行われます。詳細については、[「AVEVA Identity Manager を使用した認証」](#) セクションを参照してください。

3. AVEVA Identity Manager へのサインイン用に CONNECT アカウントに登録されている E メールアドレスとパスワードを入力します。

認証が正常に行われ、有効なエンタイトルメントがある場合、Web Client が実行します。以降のログインではシングルサインオン（SSO）が使用されるので、資格情報を入力する必要はありません。

注：資格情報が正しくない場合、エラーメッセージが表示され、入力を求められます。管理者に資格情報を問い合わせ、正しい資格情報を入力してください。

無効なエンタイトルメントのシナリオ

このセクションではエンタイトルメントが有効でない場合に発生する可能性のあるさまざまなシナリオについて説明します。

- エンタイトルメントへのアクセス権がない場合、WindowViewer はデモモードで起動します。

注：アプリケーションに設定されているタグが 64 個以下の場合はデモモードで起動できます。アプリケーションに設定されているタグが 64 個を超える場合はデモモードで起動できません。

- 現在の認証が失敗しても、ノードで以前に少なくとも 1 回は正常な認証とログインが行われたことがある場合、WindowViewer は読み取り専用モードで起動します。読み取り専用モードは、ローカルに保存された有効期限まで機能します。有効期限を過ぎると、アプリケーションはデモモードで実行します。
- WindowViewer を起動して認証をキャンセルした場合、およびログインタイムアウトがある場合、WindowViewer はデモモードまたは読み取り専用モードで起動します。
- 認証を試みたときに有効な AVEVA Identity Manager トークンだけを受け取り、CONNECT から切断された場合、オンプレミス機能は機能しますが、クラウドアクセスは失敗します。クラウドアクセスを取得するにはログインする必要があります。
- 認証を試みたときに有効な AVEVA Identity Manager トークンを受け取らず、AVEVA Identity Manager から切断された場合、WindowViewer はデモモードまたは読み取り専用モードで起動します。
- WindowViewer が正常に実行されている場合、AVEVA Identity Manager および CONNECT の更新トークンの有効性がチェックされます。有効な AVEVA Identity Manager トークンだけを受け取り、AVEVA Identity Manager と CONNECT から切断された場合、オンプレミスの機能は機能しますが、クラウドアクセスは失敗します。
- AVEVA Identity Manager から切断された場合、WindowViewer はデモモードまたは読み取り専用モードで起動します。

サポートされていない機能

以下の機能は、Web Client ではサポートされていません。

- Windows 認証は Operations Control 接続エクスペリエンスでサポートされていません。
- 共有機能は Operations Control 接続エクスペリエンスでサポートされていません。
- Web Client では、冗長シングルサインオン機能はいずれのライセンスモードでもサポートされていません。

Operations Control 接続エクスペリエンスでの Web Client のライセンスとエンタイトルメント

他のライセンス モードから Operations Control 接続エクスペリエンスに切り替えた場合、[認証] 設定を再構成する必要があります。

Operations Control 接続エクスペリエンスで [InTouchView アプリケーション] チェック ボックスがオンの場合、Web Client を起動するには、アカウントに Supervisory サブスクリプションのエンタイトルメントが必要です。Web Server を起動した後に [InTouchView アプリケーション] チェックボックスの設定を変更した場合、CONNECT 産業用グラフィック Web Server を再起動する必要があります。

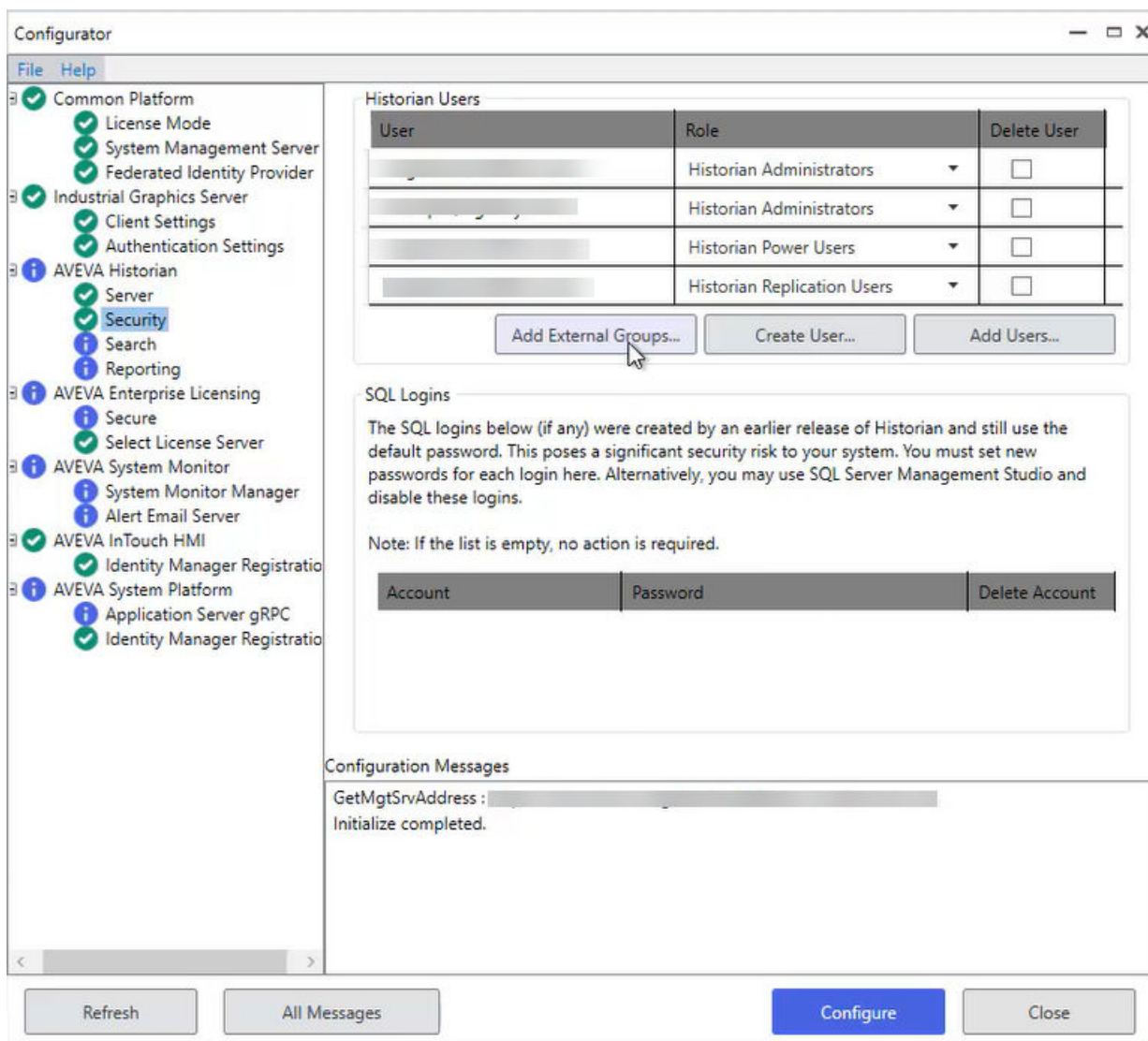
トレンド コントロールの起動と実行

トレンド コントロールはシングル サインオン機能を活用するので、WindowViewer からのシームレスなシングル サインオンが実現します。このセクションでは、Operations Control 接続エクスペリエンスでのトレンド コントロールの動作について説明します。

Operations Control 接続エクスペリエンスでトレンド コントロールを操作するには
コンフィグレータの設定:

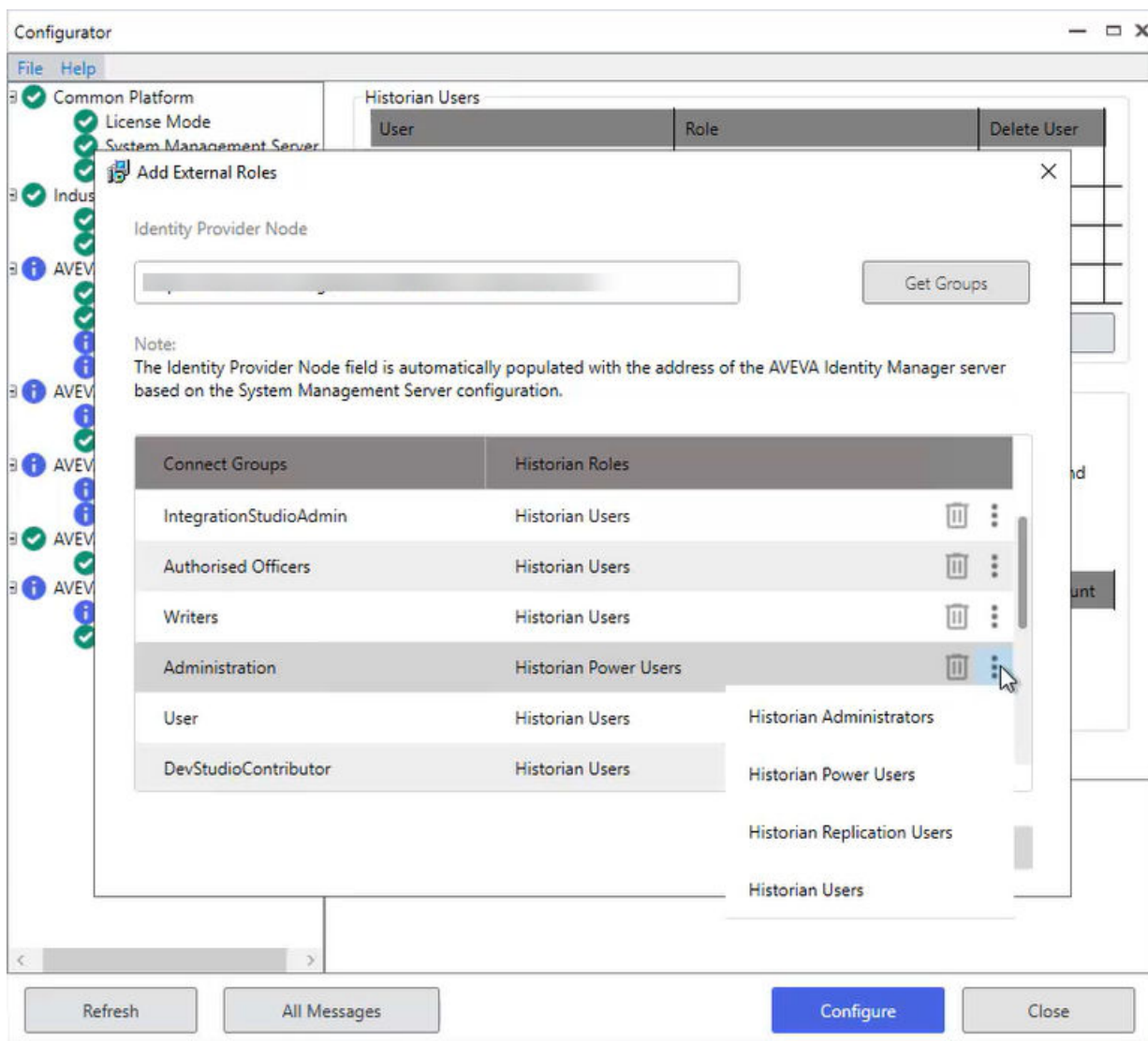
「[Operations Control 接続エクスペリエンスの設定](#)」セクションに示されているその他のコンフィグレータ設定とは別に、以下を設定する必要があります。

1. AVEVA Historian Server を設定します。詳細については、[コンフィグレータ](#) のヘルプを参照してください。
2. コンフィグレータで [AVEVA Historian] > [セキュリティ] に移動します。
3. [外部グループを追加] を選択します。



4. [外部役割を追加]ウィンドウに CONNECT に含まれるグループが一覧表示されます。[Connect グループ]列でトレンドコントロールユーザーを追加する CONNECT グループを選択し、垂直の省略記号（3つのドット）をクリックして[Historian 管理者]または[Historian パワーユーザー]のいずれかを選択します。

注：トレンドコントロールが Operations Control 接続エクスペリエンスで機能するには、トレンドコントロールのユーザーは CONNECT グループに属していること、そのグループが **Historian 管理者**または **Historian パワーユーザー**としての **Historian** の役割を持っている必要があります。



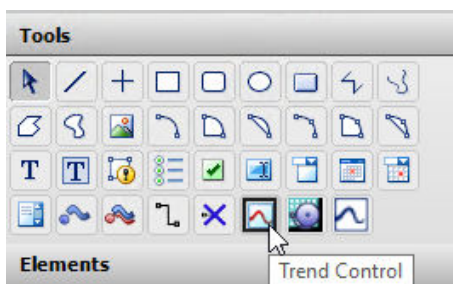
5. [保存] を選択し、[設定] を選択します。

履歴ログの設定

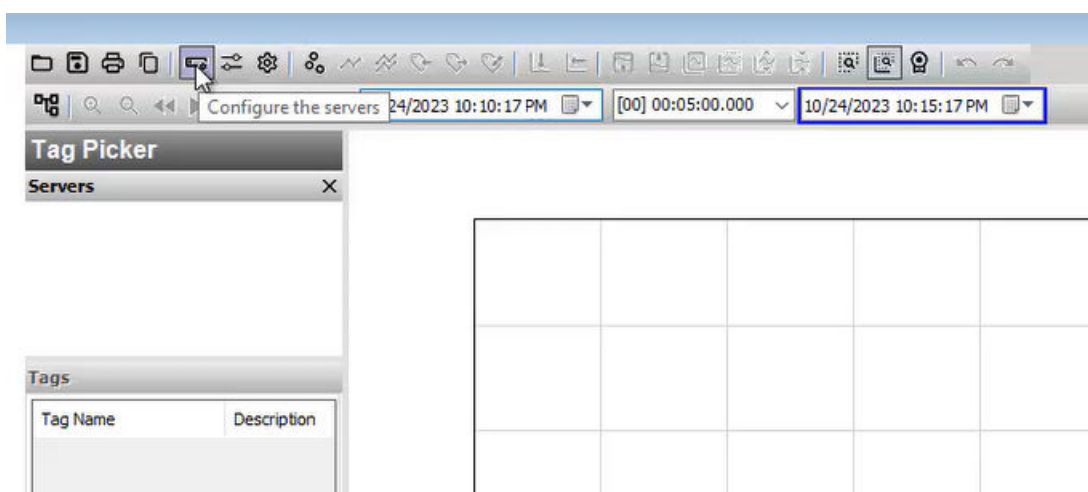
コンフィグレータを設定した後、WindowMaker で履歴ログを設定する必要があります。詳細については、AVEVA™ InTouch HMI アプリケーション開発ガイド (ITBuild.pdf) の「[一般的なログプロパティの設定 - Historian への保存](#)」セクションを参照してください。

サーバー リスト設定でログインをシングル サインオンとして設定します。

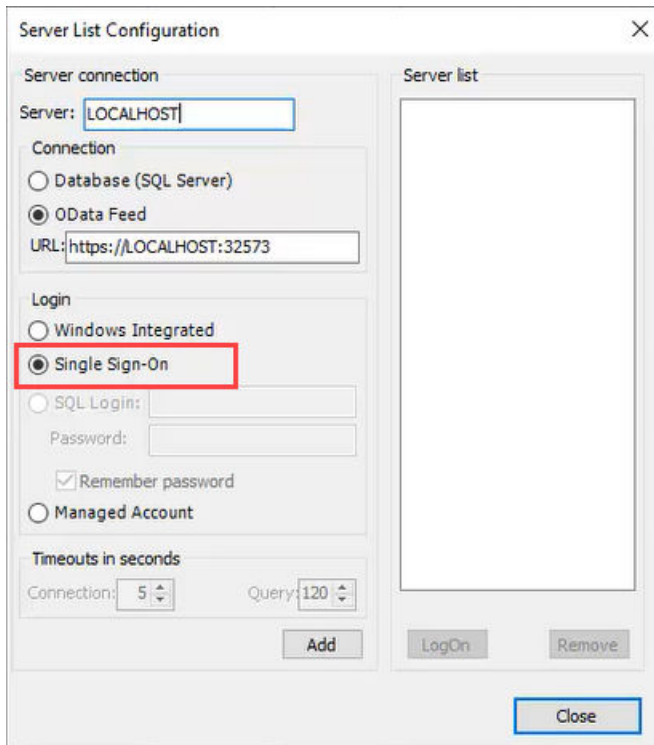
1. WindowMaker の [産業用グラフィック] ペインで右クリックし、[新規グラフィック] を選択してグラフィックを作成します。
2. 新しく作成したグラフィックを右クリックして [開く] を選択するか、グラフィックをダブルクリックして産業用グラフィック エディタで開きます。
3. [ツール] ペインから [トレンドコントロール] を選択してキャンバスに追加します。グラフィックを保存して閉じます。



4. WindowMaker で、新しく作成したグラフィックをキャンバスに追加して保存します。
5. WindowMaker の右上隅にある **ランタイム** をクリックして、WindowViewer でウィンドウを開きます。
6. **サーバーを設定** をクリックします。



7. **サーバー リストの設定** ダイアログ ボックスの **サーバー** フィールドに入力すると、入力内容にしたがって **OData フィールド** の **URL** が更新されます。
8. **ログイン** セクションで **シングルサインオン** を選択し、**追加** をクリックします。



サーバーが [サーバー リスト] に追加されます。

9. [ログオン] をクリックします。

AVEVA アプリケーションからのサインアウト

[プロファイル] アイコンの [サインアウト] オプションをクリックして、InTouch、ViewApp、またはその他の Operations Control 接続エクスペリエンス アプリケーションからサインアウトできます。アプリケーションからサインアウトすると、以下の処理が行われます。

- **CONNECT** セッションが閉じるとサインアウトします。
- デスクトップ アプリケーションからサインアウトすると、その特定のデスクトップ アプリケーションからのみサインアウトします。その他のデスクトップ アプリケーションは、現在のサインインユーザーで引き続き実行します。
 - 後でデスクトップ アプリケーションを起動すると、再度認証を受けるよう求めるメッセージが表示されます。
- 埋め込まれたブラウザを使用してデスクトップ アプリケーションからサインアウトすると、Web アプリケーションは引き続き実行します。
- サインアウトせずにアプリケーションを閉じると、セッションには影響しません。閉じるのは、そのアプリケーションインスタンスだけです。セッションは **CONNECT** サーバー上で存続します。そのアプリケーションを再度開くと、セッションは利用可能で、ユーザーはシングル サインオンで認証されます。
 - セッションの有効期限が切れている場合、アプリケーションを起動すると、再度認証を受けるよう求められます。
- WindowMaker からサインアウトすると、セッションが閉じて WindowMaker が終了します。

- WindowViewer からサインアウトすると、WindowViewer は読み取り専用モードで実行します。

サポートされている InTouch HMI 機能

サポートされているクライアント コントロール

以下の InTouch HMI クライアント コントロールが AVEVA Identity Manager 認証 を使用した Operations Control 接続エクスペリエンスでサポートされています。

- **アラームクライアントコントロール**：アラームクライアントコントロールは SSO 機能を活用するので、履歴ブロックのビューアプリケーションからシングルサインオンをシームレスに利用できます。エンタイトルメントで認証されたユーザーは、WindowViewer で Historian アラームを表示できます。

注：アラームクライアントコントロールが Operations Control 接続エクスペリエンスで機能するには、CONNECT から外部役割を設定する必要があります。

- **トレンドコントロール**：トレンドコントロールは SSO 機能を活用するので、ビューアプリケーションからシングルサインオンをシームレスに利用できます。
- **トレンドペン**：トレンドペンは SSO 機能を活用するので、ビューアプリケーションからシングルサインオンをシームレスに利用できます。エンタイトルメントで認証されたユーザーは、WindowViewer で履歴トレンドを表示できます。

サポートされているシステム タグ

以下の InTouch HMI システム タグが AVEVA ID を使用した Operations Control 接続エクスペリエンスでサポートされています。

- **\$AccessLevel**：\$AccessLevel システムタグが更新され、AddPermission()メソッドとともに使用されるようになりました。ランタイムユーザーが CONNECT の複数のグループのメンバーである場合、アクセスレベルは最上位のアクセスレベルを持つグループによって決定されます。
- **\$AccessTokenChanged**：\$AccessTokenChanged システム論理型タグは読み取り専用タグです。
- **\$Operator**：\$Operator システムタグは CONNECT のサインインユーザーを表示します。
- **\$OperatorName**：\$OperatorName システムタグは CONNECT のサインインユーザーを表示します。

スクリプト メソッド

InTouch HMI は、Operations Control 接続エクスペリエンス モードでスクリプト メソッドもサポートします。Operations Control 接続エクスペリエンスをサポートするために以下のスクリプト メソッドが追加または変更されています。

- GetAccessToken()
- GetAccessSecureToken()
- AddPermission()

AddPermission() メソッドが Operations Control 接続エクスペリエンスで受け入れるのは 2 つのパラメータだけです。

- AVEVA Connect グループ
- アクセス レベル

Operations Control 接続エクスペリエンスの AddPermission() のスクリプト関数を以下に示します。

```
DiscreteTag=AddPermission("", "AVEVA Connect group", AccessLevel);
```

ランタイムユーザーが **CONNECT** の複数のグループのメンバーである場合、アクセスレベルは最上位のアクセスレベルを持つグループによって決定されます。

- `GetTokenConnectionstatus()`
- [PostLogonDialog\(\) 関数](#)
- `SignedWrite()`
- `IsAssignedRole()`
- `Logoff()`

注： `ChangePassword()` スクリプト関数は **Operations Control** 接続エクスペリエンスでサポートされていないため、無効になります。

CONNECT 産業用グラフィックドライブ

Operations Control 接続エクスペリエンスは、**WindowMaker** で **CONNECT** 産業用グラフィックドライブのシングルサインオン機能を提供します。接続エクスペリエンスの **CONNECT** で認証されている場合は、**WindowMaker** の起動時に産業用グラフィックドライブを表示できます。その場合、再認証を行う必要なく **CONNECT** 産業用グラフィックドライブを **WindowMaker** で使用できます。したがって、**Operations Control** 接続エクスペリエンスでは、**WindowMaker** の[ファイル]メニューに[AVEVA Connect]タブは表示されません。

- **Operations Control** 接続エクスペリエンスで **WindowMaker** を起動すると、資格情報を使用した **CONNECT** での認証またはシングルサインオンでの認証が正常に行われ、かつ有効なサブスクリプションがある場合、**WindowMaker** が開きます。
 - グラフィックツールボックスで使える **CONNECT** 産業用グラフィックドライブを表示できます。ドロップダウンメニューを使用して、アカウント内の他のテナントに移動することもできます。
 - **CONNECT** 産業用グラフィックドライブに対して設定されている権限に基づいて、産業用グラフィッククラウドドライブを操作できます。

注： **CONNECT** 産業用グラフィックドライブにアクセスするには、コンテンツ投稿者の役割が必要です。これは、**Operations Control** 接続エクスペリエンスと非接続エクスペリエンスの両方で使用する場合があります。

セキュリティ書き込み/確認書き込み

- **WindowViewer** は、AVEVA ID を介した認証でセキュリティ書き込み/確認書き込みをサポートします。
- セキュリティ書き込みアクションまたは確認書き込みアクションを実行するには、**CONNECT** で認証を受ける必要があります。
- セキュリティ書き込み/確認書き込み操作では、SSO はサポートされていません。

ネットワーク アプリケーション開発 (NAD)

NAD は、した **Operations Control** 接続エクスペリエンスおよび **AVEVA Identity Manager** での認証でサポートされています。

InTouch Access Anywhere

InTouch Access Anywhere は、**Operations Control** 接続エクスペリエンスおよび **AVEVA Identity Manager** での認証でサポートされています。

Web Client

- Web Client の URL にアクセスすると、CONNECT での認証を受けるよう求めるメッセージが表示されます。
- 正常にログインした後、有効なエンタイトルメントと権限がある場合、Web Client が実行します。

サポートされていない InTouch HMI 機能

このリリースの時点では、以下の InTouch 製品と機能は Operations Control 接続エクスペリエンスでサポートされていません。

- Insight Publisher
- ウィジェット

以下の InTouch 製品と機能は Operations Control 接続エクスペリエンスでサポートされていません。

- Alarm DB Logger
- Alarm DB Purge-Archive
- Alarm DB Restore
- Alarm Hot Backup Manager
- スマートカード
- アプリケーション マネージャ 資格情報 マネージャ

章 35 Operation Control モード以外での InTouch の使用

このセクションでは、AVEVA Operations Control 接続エクスペリエンスの認証およびエンタイトルメントによって制御されない機能について説明します。

InTouch HMI でのライセンス

InTouch HMI では、InTouch でライセンスを使用するために AVEVA Enterprise License Server を使用します。AVEVA Enterprise License Manager は、1 つまたは複数の License Server を管理します。

InTouch HMI でライセンスを使用できるようにするには、以下の手順を実行します。

1. ライセンスの購入時に提供されたエンタイトルメント XML ファイルをインポートします。
2. **License Manager** のインターフェイスを使用して、License Server 上でアクティブ化するエンタイトルメントのライセンスを選択します。
3. アクティブ化されたライセンスは、**WindowMaker** または **WindowViewer** の起動時に使用可能になります。

アクティブ化されたライセンスは、License Manager の License Grid に表示されます。

InTouch は、以下の場合に使用中のライセンスを解放して License Server に返します。

- InTouch を実行するマシンがシャットダウンした場合
- InTouch アプリケーションがシャットダウンした場合

注記: InTouch HMI が異常シャットダウンした場合、ライセンスは返却されません。ライセンスを返すには、InTouch HMI を再起動して、手動でシャットダウンする必要があります。

License Manager と License Server は、InTouch HMI と共にインストールされます。デフォルトでは、InTouch HMI はローカルの License Server をポイントします。この設定は、インストール後のコンフィグレータで変更できます。詳細な手順については、『*AVEVA Enterprise Licensing Guide*』を参照してください。

InTouch ライセンスは、アプリケーションを実行するときに使用できるさまざまな数のタグ変数に基づいています。InTouch のライセンス体系でタグ変数がどのように数えられるのかを理解する必要があります。一連の関数を使用して、アプリケーション内のリモート参照タグの数を予測して計算できます。

重要: ライセンス数は随時変わる場合があります。

ライセンス タグの数について

InTouch アプリケーションの実行中は、タグのハンドルがメモリ データベースに保存されます。各タグには、ハンドルを割り当てる必要があります。タグのハンドルが初期化され、**WindowViewer** によって使用されますが、アプリケーションの停止後はディスクに永続的に保存されません。

InTouch HMI は、無制限の数のタグをサポートするようになりました。したがって、ランタイム データベースには理論的に無制限の数のライセンス タグを保存できます。これには、ローカル タグとリモート タグ ソースを参照するタグの両方が含まれます。ただし、テストされているのは最大 **300000** のタグハンドル (**300K**) です。

InTouch アプリケーションは、このランタイム データベース内で利用可能なメモリ ハンドルを超える数のアクティブなタグを同時に持つことができません。InTouch ライセンスによって、ランタイム データベース内でハンドルを割り当てることができる、ローカル タグおよびリモート タグの数が決まります。

また、アプリケーション内のアクティブなタグの最大数は、ランタイム データベースの機能的な制限によって制限されます。実際のタグの最大数は、タグ ハンドルの理論上の最大数より少なくなります。考えられるタグの最大数から、一連の定数が減算されます。

- **WindowViewer** ランタイム データベース内で無効なハンドル値が発生したかどうかを示すために、無効なタグのハンドル ビットが予約されます。
- **InTouch バージョン 10** には、ユーザー定義のタグによって置き換えることができない、**34** 個のシステム タグが含まれています。バージョン **7.11** 以前のアプリケーションを現在のバージョンの **InTouch** に移行する場合、システム タグの数は **37** 個です。
- 設定時に、プレースホルダ タグに保存するために **4096** 個のデータベース ハンドルが予約されます。設定時にウィンドウ、スクリプト、またはシンボルをインポートすると、このメモリ セグメントにプレースホルダ タグが割り当てられます。ランタイム中に、リモート参照タグに割り当てられるために、**4096** 個のすべてのプレースホルダ ハンドルを利用できます。

InTouch ライセンス オプションは、アプリケーションで使用できるローカルおよびリモート参照タグの最大数に基づきます。

InTouch タグ ライセンスが **60K** タグ未満用である場合、スティッキー タグ ライセンス スキーマが適用されます。スティッキー タグは、**WindowViewer** がリモート参照のデータ変更の通知を受信したときに、ランタイム時にバインドされるリモート タグ リファレンスです。**WindowViewer** によって、ランタイム時にリモート タグ リファレンスが **InTouch** ライセンスの上限まで更新されます。**WindowViewer** では、ライセンス制限を超えた追加のリモート タグ リファレンスは更新されません。**WindowViewer** では、ウィンドウが閉じているときは、リモート参照タグの数は削減されません。各リモート参照タグの数は、アプリケーションの実行中に固定されます。

InTouch ライセンスのリモート参照タグの最大数を超えると、単一のメッセージが表示されます。ライセンスの最大数に達した後は、無効なリモート参照タグに関連付けられている値はアプリケーションで更新されません。ライセンス制限に対してカウントされるリモート タグ リファレンスにまだ関連付けられていない 1 つまたは複数のリモート タグ リファレンスが含まれている他のウィンドウを開く前に、アプリケーションを停止して再起動する必要があります。

60K 以上のタグを許可するライセンスでは、スティッキー タグは適用されず、タグの数や、**InTouch** アプリケーション内で使用できるローカル タグとリモート タグの組み合わせの制限はありません。

たとえば、**60K** ライセンスを使用した場合のローカル タグおよびリモート参照タグの使用制限を以下に示します。

- 可能なローカル タグの合計

$$61404 = 65535 - (4096 + 37 + 1)$$

- 可能なリモート参照タグの合計

$$\text{最大} = 65535 - (37 + 1 + \text{ローカル タグの数})$$

したがって、**60K** ライセンスでアプリケーションを実行すると、ローカル タグ データベース内のユーザー定義のすべてのタグの潜在的なリモート参照と効率的にやり取りできます。ランタイム設定では、常に **4,096** 個以上の可能なリモート参照タグを利用できます。**100000** 個のタグを使用できるライセンスの場合、最大数 = $100000 - (37 + 1 + \text{ローカル タグの数})$ です。

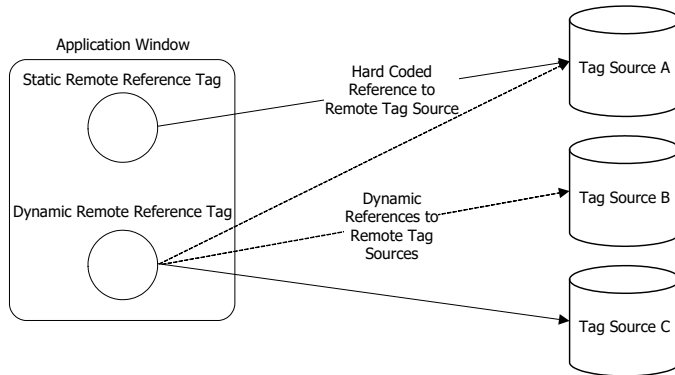
InTouch リモート参照の制限の理解

InTouch リモート参照タグには、2 つのタイプがあります。静的リモート参照は、タグ名ディクショナリからタグを定義するときに、固定リモート アドレスにハードコード化されます。静的リモート参照は、アプリケーションが実行を開始するときに、タグ データベース内のタグのハンドルに割り当てられます。静的リモート参照タグの数は、アプリケーションの実行中に固定されます。

動的リモート参照によって、アプリケーションの実行中に対象アドレスが解決されます。動的リモート参照タグがデータベース ハンドルに割り当てられた場合、**.Reference** ドットフィールドまたは

IOSetRemoteReference() 関数をスクリプト内で使用することにより、ランタイム中に対象アドレスを変更できます。

以下の図は、300K ライセンス下で実行されている、スティッキー リモート参照タグを持たない InTouch アプリケーションの例を示しています。静的リモート参照タグの数は、アプリケーションの実行中に固定されます。しかし、動的リモート参照タグの数は、アクティブなタグ ソース用のみとなります。リモート タグ ソースへの以前の接続は固定されず、リモート参照またはタグの合計数には数えられません。



InTouch 60K ライセンスでは、動的リモート タグ参照の数に制限を課すスティッキー タグの数は使用されません。これにより、アプリケーションはアプリケーションの実行中に 60K を超えるタグに動的にアクセスできるようになります。動的リモート参照のタグの使用数は、リモート参照が使用されているウィンドウを開いたり閉じたりするたびに変動します。しかし、アプリケーションは、ランタイム タグ データベースの使用制限を超える数のアクティブなタグを同時に持つことができません。

動的参照タグの数は、WindowViewer がディスク保存領域を使用して実行中のアプリケーションの内容を保存する際にのみ増減します。WindowViewer が InTouch ウィンドウまたは産業用グラフィックをキャッシュするよう設定されている場合、ウィンドウがキャッシュから削除された場合以外、リモート参照タグの数は減少しません。

ウィンドウが表示されていない場合でも、リモート タグ参照がそのソースにバインドされていないわけではありません。しかし、ウィンドウ キャッシュが完全に無効化され、高優先度ウィンドウが指定されていない場合は、WindowViewer は従来の「常にディスクからロード」シナリオと同様に機能します。その場合、ウィンドウが閉じられ、動的リモート タグ参照が 60K ライセンス環境で再要求されたとき、すべてのウィンドウがメモリから削除されます。

I/O タグのリモート参照は、InTouch ライセンスのスティッキー リモート参照数には含まれません。I/O タグのリモート参照は、スティッキー リモート参照の制限数を超えずに変更できます。回数の制限はありません。

セカンダリ タグ ソースへのフェイルオーバーが発生した場合、アプリケーションは、ライセンスされたタグの数を増やさなくても、セカンダリ ソースから同じアイテムにアクセスできます。フェイルオーバー後、セカンダリ タグ ソースから新しいアイテムにアクセスすると、タグの数が増えます。これらのアイテムは、メイン タグ ソースにフェイルバック後にアクセス可能になります。

タグの数がライセンスされた最大数に達すると、メインまたはセカンダリ タグ ソースからアイテムにアクセスできるとしても、それ以上アイテムをアクティブにすることはできません。

InTouch HMI で使用できるライセンス

InTouch HMI は、さまざまなシナリオを管理するための各種ライセンスを提供します。ライセンスは、以下のようなさまざまなパラメータに基づいて決定されます。

1. **コンソールタイプ:** コンソールタイプ（RDS/TSE または非 RDS ノード）を指定します。RDS/TSE は、ターミナルサーバーで設定されるマシン上で実行するコンソールです。非 RDS は、ターミナルサーバーで設定されないマシン上で実行するコンソールです。詳細については、「ターミナルサービスとリモートデスクトップサービスの配置と操作」を参照してください。
2. **アクセスタイプ:** ノードが設定されているアクセスタイプ（読み取り専用または読み取り/書き込み）を指定します。詳細については、「[リモートセッションで実行するアプリケーションへのユーザーアクセスの設定](#)」を参照してください。
3. **データソース:** アプリケーションが使用するデータソース（Galaxy または InTouch Tag Server）を指定します。詳細については、「[InTouchView アプリケーション](#)」を参照してください。

InTouch HMI の無制限 RDS ライセンス

InTouch HMI の無制限 RDS ライセンスは、WindowViewer における無制限のクライアントセッションで使用されます（RDS 対応のマシンのみ）。WindowViewer がライセンスを取得すると、アプリケーションがライセンスされます。読み取り/書き込みアクセスは、リモートアクセス設定に応じて異なります。同一の無制限ライセンスでは、読み取り専用と読み取り/書き込みアクセスの両方が提供されます。ライセンスが取得されていない場合は、ライセンスは既存の RDS 処理およびリモートアクセス設定に依存します。

RDS と非 RDS 環境の InTouch ライセンス

InTouch アプリケーションがリモートデスクトップサービス（RDS）で有効化されたサーバーノード上で実行している場合、コンソールは RDS クライアントセッションと同様に動作します。各セッションでは 1 つのライセンスが使用されます。セッションごとに 1 つの InTouch 開発ライセンスも別途使用されます。

この場合、InTouch アプリケーションの ReadWrite（読み取り/書き込み）機能はリモートアクセス設定によって拒否され、使用されたライセンスによって確認されます。たとえば、RDS クライアントセッションで ReadOnly（読み取り専用）リモートアクセス設定が WindowViewer で起動された場合、ReadOnly InTouch ライセンスが検索されます。License Server で RDS ReadOnly ライセンスが使用できない場合、起動時のライセンス検証は失敗します。

RDS が有効でないノードでは、オペレーティングシステムで許可されている RDS クライアントセッションでログインすることもできます。この非 RDS 環境で InTouch アプリケーションが実行している場合、クライアントセッションはコンソールと同様に動作します。この場合、アプリケーションのリモートアクセス設定では、ReadWrite アクセスは決定されません。ReadWrite アクセスは、非 RDS 環境のライセンスによってのみ決定されます。

RDS 以外の環境でタグカウントが 64 以下の場合：

- アプリケーションは常に読み取り・書き込みモードで実行されます。
- ReadOnly（読み取り専用）InTouch ライセンスの場合、移行後、InTouch アプリケーションはライセンスを使用せず、常に読み取り・書き込みモードで実行されます。

注：アプリケーションのタグカウントが 64 以下の場合、アプリケーションを読み取り専用モードでは実行することはできません。

InTouchView アプリケーション ライセンスについて

InTouchView アプリケーションには、Application Server 環境で使用するために特に設計された視覚的なインターフェイスが表示されます。このアプリケーションタイプの詳細については、「[InTouchView アプリケーション](#)」を参照してください。

InTouchView アプリケーションで使用されるライセンスのタイプは、その InTouchView アプリケーションが RDS 環境で実行しているかどうかによって依存します。

InTouchView アプリケーションが RDS クライアントセッションで実行している場合、アプリケーションのリモートアクセス設定に応じて、ReadOnly または ReadWrite クライアント接続ライセンスが検索されます。

1 つの RDS セッションで使用されるのは、1 つの接続ライセンスだけです。

InTouchView アプリケーションが RDS 以外のクライアントセッションで実行されており、タグカウントが 64 以下の場合、アプリケーションは常に読み取り・書き込みモードになります。

注： Galaxy データソースで設定されている場合、InTouchView アプリケーションは Graphic Run Time Module の ViewApp アプリケーションと同じライセンスを使用します。

InTouchView アプリケーション ライセンス

InTouch HMI アプリケーションは InTouchView アプリケーションとして設定し、InTouch タグ変数サーバーまたは Galaxy に対するクライアントとして使用できます。

InTouch タグ変数サーバーのデータにアクセスするように InTouchView アプリケーションを設定した場合に使用可能なライセンスを以下に示します。

	InTouch HMI クライアント読み取り/書き込みライセンス	InTouch HMI クライアント読み取り専用ライセンス	InTouch HMI の無制限クライアントライセンス*
リモート アクセス	読み取り/書き込み	読み取り専用	読み取り/書き込み & 読み取り専用
RDS/TSE	可	Yes	可
非 RDS	可	可**	不可
MarkAppReadOnlyNonRDS のサポート	可	可	該当なし

* このライセンスは、無制限の数の RDS クライアントで使用できます。

** タグカウントが 64 を超える場合。

Galaxy のデータにアクセスするように InTouchView アプリケーションを設定した場合に使用可能なライセンスを以下に示します。

	監視クライアント読み取り/書き込みライセンス	監視クライアント読み取り専用ライセンス	監視クライアントサーバーライセンス
リモート アクセス	読み取り/書き込み	読み取り専用	読み取り/書き込み & 読み取り専用

RDS/TSE	可	Yes	可
非 RDS	可	使用可能*	不可
OMI との共有	可	Yes	可
MarkAppReadOnlyNonRDS のサポート	可	可	該当なし

*タグカウントが 64 を超える場合。

サーバー上の InTouch ライセンスの管理に関するベスト プラクティス

InTouch ライセンスを管理する際、InTouch ライセンスの使用を予測できるようにするためのいくつかのベスト プラクティスがあります。ライセンスの使用を予測することにより、特定のシステムで適切なライセンスをオンデマンドで使用できます。このようなライセンス使用を行うと、サーバー ベースの AVEVA Enterprise Licensing システムを使用した InTouch ライセンス管理が容易になります。

予測的なライセンス使用に関する 2 つのベスト プラクティスは、ライセンス予約と浮動ライセンスです。詳細については、以下のセクションを参照してください。

ライセンスの予約

License Manager で特定のデバイスにライセンスを予約できます。特定のデバイスにライセンスを予約すると、別の InTouch アプリケーションではライセンスを取得できなくなり、アプリケーションが中断されることや実行できなくなることを防止できます。

ユーザー ベースのライセンス予約

AVEVA Enterprise License Manager のライセンス予約ページで、特定のユーザー向けにライセンスを予約できます。予約ページでは、この特定の構成が可能です。InTouch OMI と InTouch HMI ViewApp のいずれでもユーザー ベースのライセンス予約がサポートされないことを理解しておくことが重要です。最終的な結果として、ソフトウェアは予約されたライセンスを取得できません。したがって、Supervisory クライアントライセンスにはデバイス ベースの予約を使用してください。

デバイス ベースのライセンス予約

特定のデバイス用に Supervisory クライアントライセンスを予約する場合、デバイス名は InTouch HMI/OMI ViewApp を実行するコンピュータの名前にする必要があります。ViewApp が RDS またはターミナル サーバー内で実行している場合、デバイス名は以下の命名規則に従う必要があります。

<RDS ホスト名>-<RDP クライアント名>-<インデックス>

ここで、<RDS ホスト名> は RDS またはターミナル サーバーの名前を示し、<RDP クライアント名> は RDP クライアント ソフトウェアを実行する PC を指します。<インデックス> は、単一のクライアント マシンから複数の RDP セッションを実行する場合以外は 1 です。単一のクライアント マシンから複数の RDP セッションを実行する場合、<インデックス> は、それぞれの RDP クライアントの予約ごとに 1 から特定の RDP クライアントから RDP セッションの総数まで増分します。

例 1: "ControlRoomA" というホスト名のコンピュータで InTouch OMI が実行している場合

デバイス名: "ControlRoomA"

例 2: "ControlRoomB" という名前のコンピュータが単一のリモート デスクトップ クライアント (RDP) を実行し、"PrimaryRDS" というホスト名のリモート デスクトップ サーバー (ターミナル サーバー) に接続する場合

デバイス名: "PrimaryRDS-ControlRoomB-1"

例 3: 2 台のコンピュータ ("SupervisorPC1" および "LineMgrA") がそれぞれ単一のリモート デスクトップ クライアント (RDP) を実行し、"PrimaryRDS" というホスト名のリモート デスクトップ サーバー (ターミナル サーバー) に接続する場合

デバイス名:

ライセンス予約 1: "PrimaryRDS-SupervisorPC1-1"

ライセンス予約 2: "PrimaryRDS-LineMgrA-1"

例 4: "ExecutiveDesktop" というホスト名のコンピュータが 4 つのリモート デスクトップ クライアント (RDP) を実行し、"PrimaryRDS" というホスト名のリモート デスクトップ サーバー (ターミナル サーバー) に接続する場合

デバイス名:

ライセンス予約 1: "PrimaryRDS-ExecutiveDesktop-1"

ライセンス予約 2: "PrimaryRDS-ExecutiveDesktop-2"

ライセンス予約 3: "PrimaryRDS-ExecutiveDesktop-3"

ライセンス予約 4: "PrimaryRDS-ExecutiveDesktop-4"

RDS 負荷分散をサポートする場合、複数の RDS クライアントセッションがポイントできる単一の License Server 上ですべての RDS ライセンスをアクティブ化することができます。サーバー上のライセンスは、各 RDS クライアントセッション間で共有できるように同じ機能である必要があります。内部パラメータの値が同じである場合、ライセンスは同じ機能であるとみなされます。このシナリオでは予約は必要ありません。RDS クライアントセッションごとに異なる種類のライセンスが必要な場合、各 RDS サーバーに License Server をインストールする必要があります。

ライセンス予約の詳細な手順については、『AVEVA Enterprise Licensing Guide』を参照してください。

浮動ライセンス

浮動ライセンスは、いずれのマシンに対しても予約されていないライセンスです。製品名と機能が同じ浮動ライセンスを単一の License Server 上に配置することが推奨されます。たとえば、同じ容量のいくつかの InTouch 2017 ランタイム 60K タグライセンスが有効化された License Server を設定できます。同様に、同じ容量の複数の InTouch 2023 ランタイム無制限 (Unlimited) タグライセンスが有効化された License Server を設定できます。これは、計画的なライセンス使用を行うために推奨される方法です。

しかし、製品名が同じで機能が異なるライセンスを同一の License Server で有効にすることは推奨されません。たとえば、InTouch 2023 ランタイム無制限 (Unlimited) タグが有効なライセンスと InTouch 2017 ランタイム 500 タグが有効なライセンスを同一のライセンス サーバーに混在させることができます。このシナリオでは、どのインスタンス WindowViewer でタグの数が多い方のライセンスが使用されるかを把握することはできません。

ライセンス情報の表示

WindowMaker または WindowViewer によって使用されている現在のライセンスの特定の情報を表示できます。

WindowMaker ライセンス情報を表示するには

1. WindowMaker を開きます。
2. 画面左下隅にある [ファイル] メニューの [ライセンス] をクリックします。

[バージョン情報] 画面が表示されます。[バージョン情報] 画面には、WindowMaker のバージョンおよびライセンス情報が表示されます。

About

Version

23.0.000 6000.1121.0000.0000

@2002-2020 AVEVA group plc and it's subsidaires.
All rights reserved. [Legal](#)

License information



InTouch HMI WindowMaker
Development Studio License

Product text

InTouch HMI 2023 Development Unlim Tags, Flex

Expire date

12/30/2024 11:59:59 PM

Tag count

Unlimited

Runtime Timeout

None

Window count

65535

Language Lock

No

Read only

No

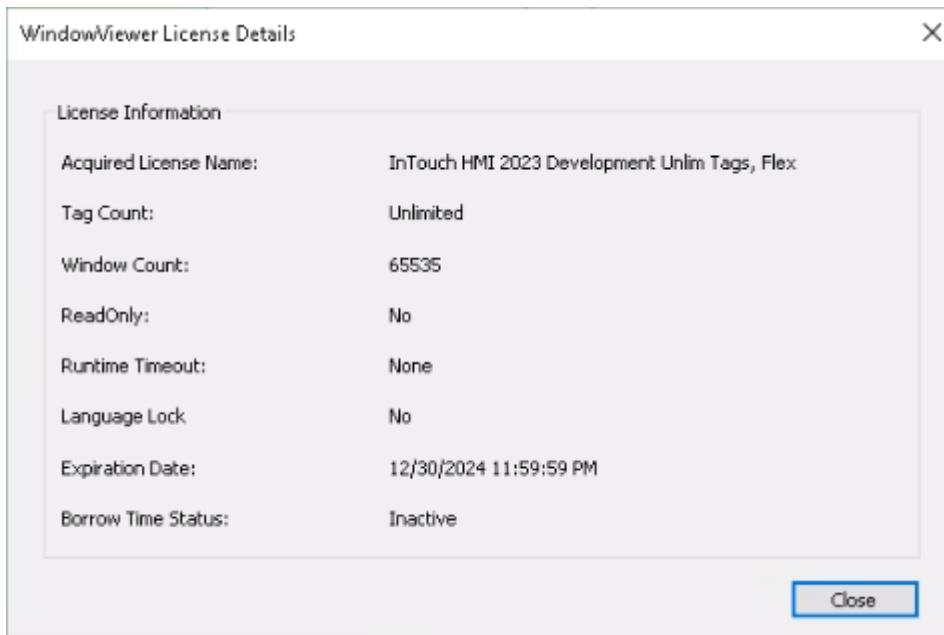
Borrow Time Status

Inactive

このダイアログには、会社名とライセンスのシリアル番号は表示されません。これらの情報は、AVEVA Enterprise License Manager のインターフェイスに表示されます。

WindowViewer ライセンス情報を表示するには:

1. WindowViewer を開きます。
2. [ファイル]、[WindowViewer のバージョン情報] の順にクリックします。
[WindowViewer のバージョン情報] ダイアログ ボックスが表示されます。
3. [ライセンス契約書を表示] をクリックしてエンドユーザー ライセンス契約書を表示します。
4. [ライセンスを表示] をクリックして、ライセンスの詳細を表示します。[ライセンス情報] ダイアログ ボックスが表示されます。



ライセンス情報に表示されるパラメータを以下に示します。

- **取得済みライセンスの名前:** 製品が License Server から使用しているライセンスの完全な名前です。
- **タグ カウント:** 使用中のライセンスで許可されているタグの数。
- **ウィンドウ カウント:** 使用中のライセンスで許可されているウィンドウの数。
- **読み取り専用:** ライセンスで許可されている I/O 読み取り/書き込み権限が表示されます。アプリケーションが I/O タグに書き込めることを示すものではありません。
- **ランタイム タイムアウト:** ライセンスによって割り当てられているアプリケーション ランタイム。タイムアウト時間に達すると InTouch セッションが終了します。
- **言語ロック:** 中国語のオペレーティング システム上の InTouch のライセンスにのみ適用されます。中国語のオペレーティングシステム上で実行される InTouch には中国語か英語のライセンスが必要です。

言語ロックの制限は、接続ライセンスには適用されません。

- **有効期限:** 使用中のライセンスの有効期限日。

- **猶予時間のステータス:** 割り当てられた猶予期間に関係なく、ライセンスが 50% の状態のときユーザーに通知することを意図します。ステータスは、50%に達すると**アクティブ**に変わります。猶予期間が完全に過ぎてもライセンスが更新されない場合、InTouch のライセンスが切れます。

注: InTouch アプリケーションマネージャでは**[View License]**オプションは無効になっています。アプリケーションマネージャはライセンスを使用しないので、表示できるのは EULA (エンドユーザー ライセンス契約書) だけです。**[アプリケーションマネージャ]** ダイアログボックスには、ライセンス関連の情報は表示されません。

起動後における別のライセンスの使用の管理

標準アップグレードとメンテナンスアクティビティの一環として、**License Manager** で個々のライセンスをアクティブ化/非アクティブ化することができます。起動後に有効なライセンスを使用した **WindowMaker** または **WindowViewer** でライセンスを更新できない場合、別のライセンス (当初割り当てられたライセンスや消費されたライセンス) を使用できます。その場合、InTouch は、新しいライセンスの機能が適切かどうかを検証する必要があります。最後の有効なライセンスと現在使用されているライセンスのライセンス機能の比較によって、猶予期間に入ることなく **WindowMaker** または **WindowViewer** の実行を続けることができるかどうかが決定されます。猶予期間を終了する方法については、「[猶予期間での操作](#)」を参照してください。

以下のセクションでは、起動後に別のライセンスを使用する 2 つのシナリオを示します。

シナリオ 1: 起動後に機能の低いライセンスが使用された

このシナリオでは、起動後に使用されたライセンスの機能が最後の有効なライセンスよりも低くなります。これは、ライセンスのダウングレードとみなされ、**WindowMaker** または **WindowViewer** が猶予期間に入ります。

ライセンスは、新しいライセンスのパラメータが以下の条件に当てはまる場合にダウングレードしたとみなされます。

猶予期間をトリガするパラメータの変更を以下に示します。

- タグ カウント: タグ カウントが減少した場合、猶予期間がトリガされます。
- ウィンドウ カウント: ウィンドウ カウントが減少した場合、猶予期間がトリガされます。
- ランタイム タイムアウト: タイムアウトの値が **[なし]** からその他の値に変わった場合、猶予期間がトリガされます。
- 言語ロック: 言語ロックの値が **[いいえ]** から **[はい]** に変わった場合 (またはその逆の場合)、猶予期間がトリガされます。
- 読み取り専用: 読み取り専用パラメータが **[いいえ]** から **[はい]** に変わった場合 (またはその逆の場合)、猶予期間がトリガされます。

ライセンスのダウングレードが発生すると、ダウングレード (「違反」) したパラメータが **[ライセンスを表示]** ダイアログボックスに表示されます。たとえば、最初に **WindowMaker** でタグ カウントが **6000** のライセンスを使用し、**3000** タグ カウントのライセンスにダウングレードした場合、タグ カウント パラメータは次のように表示されます。

重要: アプリケーションは、最後の有効なライセンスの機能で実行し続けます。この例では、猶予期間に入ったとき、元の **6000** タグ カウントが維持されます。

シナリオ 2: 起動後に機能の高いライセンスが使用された

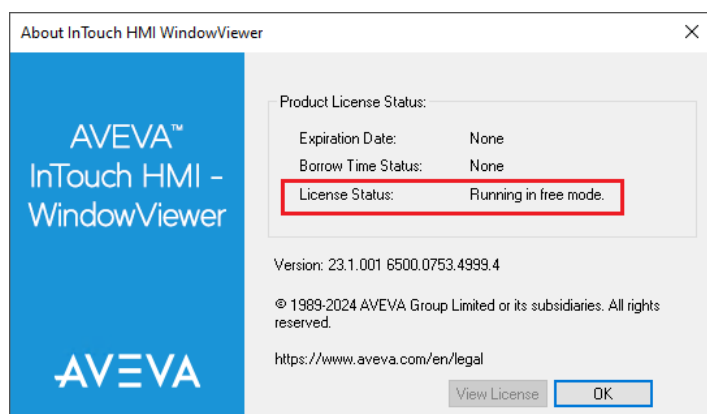
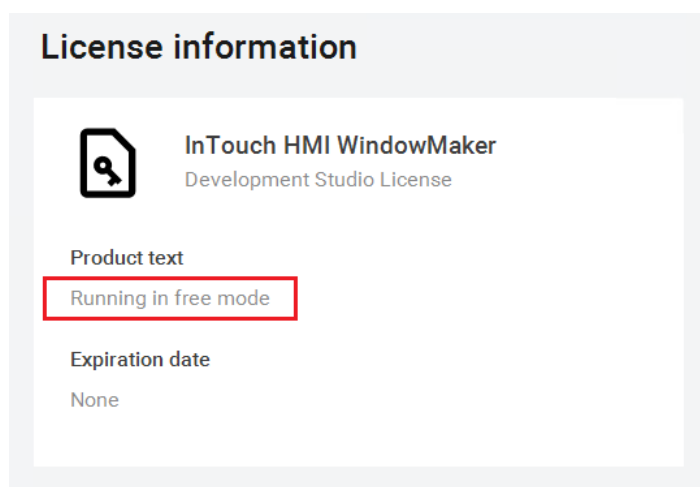
このシナリオでは、起動後に使用されたライセンスの機能が最後の有効なライセンスと同じか、それよりも高くなります。これはライセンスのアップグレードで、WindowMaker または WindowViewer が猶予期間に入りません。

ライセンスのアップグレードが発生した場合、[ライセンスを表示] ダイアログボックスに表示されるすべてのパラメータが更新され、高い機能の値が表示されます。

フリーモードとデモモードでの操作

アプリケーションのタグカウントが 64 以下の場合、ライセンスチェックなしで WindowMaker と WindowViewer の全機能を利用できます。これはフリーモードと呼ばれています。アプリケーションのタグカウントが 64 を超える場合でも、WindowMaker はフリーモードで実行されます。

注意：フリーモードは Operations Control 接続エクスペリエンスでサポートされていません。



アプリケーションのタグカウントが 64 を超えており、起動時に有効なライセンスを使用できない場合、WindowViewer はデモモードで実行されます。WindowViewer の起動中に有効なライセンスが存在しない場合、ライセンスを取得できなかったことを示すポップアップメッセージが表示されます。以下のいずれかを実行します。

- 有効なライセンスを更新して[再試行]をクリックします。
- [Demo Mode]をクリックして 120 分間のデモモードで続行します。

- **[終了]**をクリックして **WindowViewer** を終了します。

デモモードの実行中はタグカウントとウィンドウカウントの上限はありません。ただし、タイムアウトまでに **WindowViewer** をデモモードで実行できるのは 2 時間のみです。デモモードのタイムアウトに達すると、終了するよう求めるメッセージが表示されます。

注：デモモードでは、有効なライセンスをアクティブ化した場合でも、有効なライセンスを使用するには **WindowViewer** を終了して **WindowViewer** を再起動する必要があります。

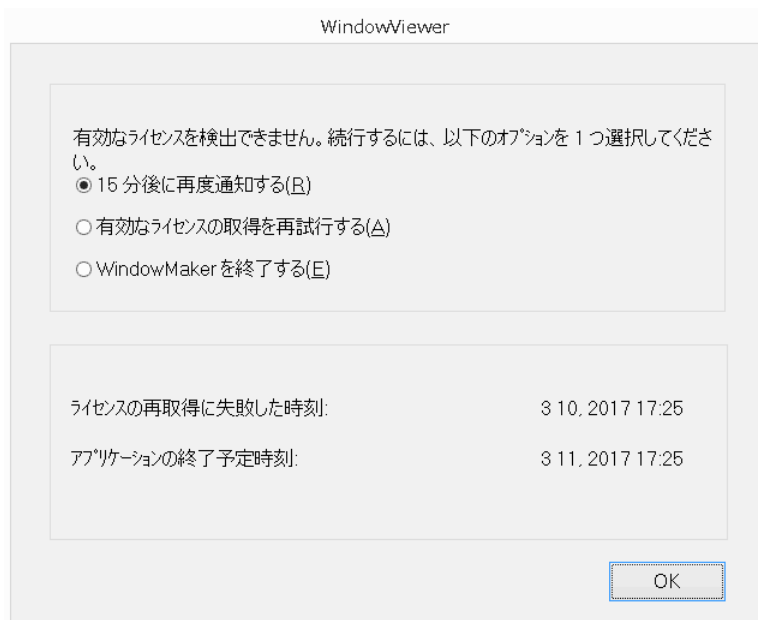
InTouch.ini ファイルで **ViewLicenseRetryCount** キーを設定すると、**WindowViewer** は、パラメータで指定された回数だけバックグラウンドでライセンス取得の試行を続けます。ライセンスが取得された場合、ダイアログが閉じて **WindowViewer** が起動します。

猶予期間での操作

猶予期間は、特定の条件が発生した後に最後の有効なライセンスの機能で InTouch の実行を継続できる期間（24 時間）です。猶予期間は、**WindowMaker** と **WindowViewer** の両方で使用できます。猶予期間が終了すると、割り当てられた時間内に有効なライセンスが取得されない場合、InTouch が終了します。

猶予期間は、以下のシナリオによってトリガされます。24 時間以内に有効なライセンスが取得されない場合、**WindowMaker** または **WindowViewer** は終了します。

WindowMaker または **WindowViewer** が猶予期間に入ると、次のダイアログが表示されます。



通知を後で再度表示するかライセンスの取得を試みることができます。アプリケーションを終了することもできます。

シナリオ 1: 使用していたライセンスが失われた

License Server から取得した有効なライセンスが製品の実行中に無効になった場合、**WindowMaker** または **WindowViewer** は猶予期間に入ります。

猶予期間を終了して通常の操作を再開するには、License Server で有効なライセンスを有効にします。ライセンスを使用すると、**WindowMaker** または **WindowViewer** の猶予期間が終了し、通常の操作が再開されます。

シナリオ 2: ライセンスの有効期限が切れた

License Server から取得した有効なライセンスの有効期間が切れた場合、WindowMaker または WindowViewer は猶予期間に入ります。猶予期間を終了して通常の操作を再開するには、License Server で有効なライセンスを有効にします。

WindowMaker または WindowViewer でライセンスを取得できなかった場合、最初のダイアログが再度表示されます。

シナリオ 3: ライセンスがダウングレードされた

AVEVA Enterprise License Manager はサーバー ベースのライセンス システムなので、ライセンスは定期的に更新する必要があります。低い機能のライセンスへの更新中に WindowMaker または WindowViewer ライセンスがダウングレードされた場合、WindowMaker または WindowViewer は猶予期間に入ります。ダウングレード シナリオの詳細については、「[起動後における別のライセンスの使用の管理](#)」を参照してください。

猶予期間を狩猟して通常の操作を再開するには、最後の有効なライセンスまたはそれ以上のライセンスを取得してください。

重要: 最後の有効なライセンスによって有効化された機能は、猶予期間でも維持されます。

上記のすべてのシナリオにおいて、適切なライセンスが正常に取得された場合、ライセンスが正常に取得されたことを示すメッセージが表示されます。

InTouch でライセンスを取得できなかった場合、最初のダイアログが再度表示されます。

注: InTouch.ini ファイルで ViewLicenseRetryCount キーを設定すると、WindowViewer は、パラメータで指定された回数だけバックグラウンドでライセンス取得の試行を続けます。ライセンスが取得された場合、ダイアログが閉じて WindowViewer が起動します。

リモート タグを数える関数

InTouch HMI には、アプリケーションがライセンスのリモート タグの要件に準拠しているかどうかを確認する一連の関数が含まれています。生産環境に配置する前に、アプリケーションで考えられるライセンスの問題をテストおよび識別するための関数が含まれている一時スクリプトを記述できます。アプリケーションにライセンス上の問題がないことを確認したら、スクリプトを削除します。

IORRGetSystemInfo() 関数

IORRGetSystemInfo() 関数は、実行中の InTouch アプリケーションのタグの数を返します。引数値に基づいて、IORRGetSystemInfo() 関数は以下のような数値を返します。

- InTouch ライセンスによって指定された、リモート タグアドレスの最大数。
- InTouch アプリケーションの実行中、ライセンスに対してカウントされたリモート タグアドレスの数。
- InTouch アプリケーションで現在アクティブ化されているリモート タグの数。
- 実行中の InTouch アプリケーションで利用可能なリモート タグの数。
- 現在無効になっているリモート参照タグの数。
- 実行中の InTouch アプリケーション内のローカル タグの数。
- 関数呼び出し中にエラーが発生した場合、または *Option* 引数に無効な値が割り当てられている場合は -1。

カテゴリ

その他

構文

IORRGetSystemInfo(Option);

引数

オプション

返すリモート参照タグの数のタイプを指定する、整数型タグまたは整数の定数。可能な値は以下のとおりです。

1	<p>InTouch ライセンスに基づいて、許可された最大数のリモート タグ アドレスを返します。ローカル I/O タグは、リモート タグの数には数えられません。</p> <p>この数は、InTouch アプリケーションの実行中は一定しています。</p>
2	<p>InTouch アプリケーションの実行中にアクティブ化されている、ライセンス制限を超えた数の独自のリモート タグ アドレスの数を返します。ローカル I/O タグは、リモート タグの数には数えられません。</p> <p>60K を超えるリモート参照タグがライセンスによって許可された場合、アクティブ化されているリモート タグ アドレスの数にかかわらず、この数は 0 になる場合があります。無制限のライセンス下で実行されている場合、WindowViewer はアクティブ化されているリモート タグ アドレスを数えません。</p> <p>リモート タグの制限を持つライセンス下でアプリケーションを実行している場合、この数はリモート タグのライセンス数の上限まで増加します。リモート タグの制限に達した後は、追加的なリモート タグ アドレスをさらにアクティブ化することはできません。現在アクティブ化されているアドレスのみ再度アクティブ化できます。ライセンス制限を超えた数のリモート参照タグのリストを取得するには、Option 引数を 3 に設定して IORRWriteState を使用します。</p>
3	<p>InTouch アプリケーション内で現在アクティブ化されているリモート参照タグの数を返します。</p>
4	<p>リモート タグのハンドルを使い切ることなく InTouch アプリケーションでアクティブ化できるリモート参照タグの数を返します。この数は通常アプリケーションの実行中に変わります。リモート参照タグが含まれているスクリプトを停止および開始したり、ウィンドウを開いたり閉じたりすると、リモート参照タグの数に影響します。</p> <p>この数は、特にライセンスが無制限である場合、ライセンスに残っている数より少なくなる場合があります。これは、同時にアクティブ化できるリモート参照タグの数に内部制限があるため発生します。</p>
5	<p>InTouch アプリケーション内で現在無効な状態にあるリモート タグの数を返します。</p>
6	<p>現在 InTouch アプリケーション内にあるローカル タグの数を返します。</p>

例

以下の例は、InTouch アプリケーションの実行中にライセンス制限を超えた数のリモート参照タグの数を返します。返されたリモート参照タグの数は、RRTagCount 整数型タグの値として割り当てられます。


```
RRTagCount = IORRGetSystemInfo(2);
```

IORRWriteState() 関数

IORRWriteState() 関数により、アプリケーションのリモート タグの現在の状態に関する情報がテキスト ファイルに保存されます。ファイルが存在しない場合、関数によって作成されます。関数が含まれているスクリプトが実行されるたびに、新しい情報がファイルに連結されます。

どのようなリモート タグ情報がファイルに保存されるのかを指定できます。また、関数の戻り値は、情報がファイルに正常に追加されたかどうかを示します。

カテゴリ

その他

構文

```
IORRWriteState(FilePath, Option, " ");
```

引数

FilePath

アプリケーションのリモート タグに関する情報が含まれているテキスト ファイルへの完全フォルダ パス。FilePath 引数は、文字列定数またはメッセージ型タグ変数になります。

Option

リモート タグの数の情報を指定する整数型タグ変数または整数の定数が、ファイルに書き込まれます。可能な値は以下のとおりです。

1	現在のリモート タグ アドレスのリスト。情報には、各リモート タグの状態、アクティブ化された時間、非アクティブ化された時間も含まれます。
2	現在アクティブなすべてのアドレスのリストと、それらがアクティブ化された時間。
3	アクティブ化され、ライセンス制限を超えた数の、すべてのリモート リファレンス タグのアドレスのリスト。 新しいリモート タグ アドレスがアクティブ化されると、新しいアイテムがリストに追加されます。ライセンス制限に達すると、アイテムをリストに追加することができなくなります。 ただし、リモート タグのライセンス制限が無制限である場合、このリストにはアドレスは追加されません。
4	リモート リファレンス タグの数がライセンス制限を超えるため、または内部のタグ変数のハンドル制限に達したため、現在のアドレスのリストがアクティブ化されません。 リモート タグのライセンス制限が無制限である場合、ライセンスに関連するアドレスを返しません。 ライセンスが無制限である場合、使用制限のため、リストには現在アクティブでないすべてのアイテムが含まれます。このリスト内のアイテムがアクティブ化されると、リストから削除されます。ライセンス制限のためにアイテムが非アクティブ化されると、リストから削除されます。このリストは、InTouch アプリケーションの実行中に更新されます。

空の文字列""

この引数は今後使用するために予約されていますが、スクリプトでは `IORRWriteState()` 関数に含める必要があります。

結果

`IORRWriteState()` 関数呼び出しによってファイルに保存された情報を解釈するためのいくつかの例を以下に示します。

現在のアドレスのリスト

出力ファイルの以下の行では、完全にアクティブ化された、更新可能なリモートリファレンスタグの例を示しています。以下の行は、`TestProt:di000` リモートリファレンスタグにアドレス `65535` が割り当てられていることを示しています。

```
65535 <TestProt:di000> (RAA) {C:5/23/2007 9:58:35 AM} {A:5/23/2007 9:58:35 AM}
```

以下のリモートリファレンスタグ名は、括弧で囲まれた3つのフラグです。

- 最初のフラグ **R** は、タグ変数のリモートリファレンスが正常に解決されたことを示します。タグ変数のリモートリファレンスがまだ保留である場合に、最初のフラグに値 **X** が割り当てられます。
- 2番目のフラグは、リモートタグが現在アクティブである場合は **A**、アクティブでない場合は **D** を示します。
- 3番目のフラグは、アドレスが許可されている場合は **A**、ライセンス制限のために許可されていない場合は **D** を示します。

C の後ろの日付は、リモートタグが作成された日付を示します。**A** の後ろの日付は、タグ変数が最近アクティブ化された日付を示します。**InTouch** アプリケーションの実行中にリモートリファレンスタグが非アクティブ化された場合、非アクティブ化された時間が行の後ろに含まれます。

出力ファイルの以下の行では、**InTouch** ライセンスのタグ変数の数の制限を超える、アクティブなリモートリファレンスタグの例を示しています。タグ変数のリモートリファレンスが正常に解決され、タグ変数が現在アクティブになっています。

```
65414 <TestProt:di121> (RAD) {C:5/23/2007 9:58:35 AM} {A:5/23/2007 9:58:35 AM}
```

しかし、リモートリファレンスタグに割り当てられたアドレスが、**InTouch** ライセンスのタグ変数の数の制限を超えるため、アプリケーション内でタグ変数の値の更新は発生しません。

アクティブなアドレスのリスト

出力ファイルの以下の行では、割り当てられた値によって **InTouch** アプリケーションが更新される、完全にアクティブ化されたリモートタグの例を示しています。

```
65429 <TestProt:di106> (A) {C:5/23/2007 9:58:35 AM} {A:5/23/2007 9:58:35 AM}
```

最初の数値はリモートタグのハンドルであり、その後ろに、アドレス、フラグ、**A**（許可されている場合）または **D**（許可されていない場合）が表示されます。出力行のフラグの後ろには、作成日時、最近アクティブ化された日時、および最近非アクティブ化された日時が表示されます。アプリケーションの実行中にリモートタグが非アクティブ化されていない場合、非アクティブ化された時間は表示されません。

出力ファイルの以下の行では、ライセンス制限を超えるアクティブなリモートタグの例を示しています。

```
65342 <TestProt:di193> (D) {C:5/23/2007 9:58:35 AM} {A:5/23/2007 9:58:35 AM}
```

ライセンスされたアドレス

リストの以下の行は、リモートリファレンスタグに割り当てられたアドレス、およびリストに追加された日時を示しています。


```
<testprot:di000> {C:5/23/2007 9:58:36 AM}
```

拒否されたアドレス

使用制限のため、またはタグ変数の数がライセンスの最大数を超えるため、拒否されたアドレスがリストに表示されます。

この例では、ライセンス制限を超えるリモート タグ アドレスを示しています。

```
testprot:di125 [1] (L) {F:5/23/2007 9:58:39 AM} {R:5/23/2007 9:58:39 AM}
```

アドレス、およびアイテムを参照するために試行された回数を示す数がリストに表示されます。フラグはアドレスがリスト内にあるかどうかを示します。ライセンスを超えることが原因である場合はL、または内部の使用制限が原因である場合はIが表示されます。2つの時間は、リストに最初に追加された時間、および最新のアクセス時間を表します。

例

この例では、現在のアクティブ化されているリモート タグ アドレスが、`c:\intouch\data` フォルダにあるファイルに書き込まれます。Return Value タグ変数に整数が割り当てられます。これは、関数呼び出しによってリモート タグ情報がファイルに正常に書き込まれたことを示します。

```
ReturnValue = IORRWriteState("c:\intouch\data", 2, "");
```

IORRGetItemActiveState() 関数

IORRGetItemActiveState() 関数は、指定したリモート タグ アドレスのステータスを返します。

カテゴリ

その他

構文

```
IORRGetItemActiveState(ItemPath, Option);
```

引数

ItemPath

ItemPath は対象となるアドレスを表す文字列です。ItemPath には、文字列定数またはメッセージ型タグ変数を指定できます。

Option

返すリモート リファレンス タグの数のタイプを指定する、整数型タグ変数または整数の定数。可能な値は以下のとおりです。

1	現在のリモート タグ アドレスが現在アクティブであるかどうか判断されます。 アドレスが現在のものであり、アクティブな場合、戻り値は1です。アドレスが現在のものでない場合、戻り値は-1です。 アドレスが現在のものであるけれどもアクティブでない場合、戻り値は0です。
2	アプリケーションの実行中に、現在のリモート タグ アドレスがアクティブ化されたことがあるかどうか判断されます。 アドレスが現在のものであり、少なくとも一度アクティブ化されたことがある場合、戻り値は1です。アドレスが現在のものでない場合、戻り値は-1です。アドレスが現在のものであるけれども、アクティブ化されたことがない場合、戻り値は0です。

3	<p>現在のリモート タグ アドレスが非アクティブ化されているかどうか判断されます。</p> <p>アドレスが現在のものでない場合、戻り値は -1 です。</p> <p>アドレスが現在のものであるけれども、非アクティブ化されたことがない場合、戻り値は 0 です。</p>
4	<p>現在のリモート タグ アドレスが無効化されているかどうか判断されます。</p> <p>アドレスが現在のものであり、少なくとも一度非アクティブ化されたことがある場合、戻り値は 1 です。アドレスが現在のものでない場合、戻り値は -1 です。アドレスが無効化されていない場合、戻り値は 0 です。アドレスが現在のものであり、無効化されている場合、戻り値は 1 です。</p>
5	<p>アドレスが許可リストにあるかどうか判断されます。</p> <p>アドレスがリストにない場合、戻り値は 0 です。</p> <p>アドレスがリストにある場合、戻り値は 1 です。</p>
6	<p>アドレスが否認リストにあるかどうか判断されます。</p> <p>アドレスがリストにない場合、戻り値は 0 です。</p> <p>アドレスがリストにある場合、戻り値は 1 です。</p>

例

この例では、TestProt:di000 リモート タグ アドレスが現在アクティブであるかどうか判断されます。
ReturnValue = IORRGetItemActiveState("TestProt:di000", 1);

この例では、TestProt:di121 リモート タグ アドレスが現在無効化されているかどうか判断されます。
ReturnValue = IORRGetItemActiveState("TestProt:di121", 4);

この例では、TestProt:di001 リモート タグ アドレスが現在ライセンス制限を超えた数であるかどうか判断されます。
ReturnValue = IORRGetItemActiveState("TestProt:di001", 5);

InTouch Web Client のライセンス付与

ログインしてアプリケーション グラフィックを Web ブラウザで表示するには、有効な Web Server ライセンスが必要です。ライセンスは、外部 Web サイトでのアプリケーション グラフィックのホスティングにも適用されます。ライセンスの詳細については、*AVEVA Enterprise Licensing* のヘルプを参照してください。Web Server は、無制限の数のセッションに接続して Web ブラウザでアプリケーション グラフィックの表示および操作を行うことができる次のようなさまざまなタイプのライセンスを提供します。

- InTouch Web Server *Flex* ライセンス
- InTouch Web Server 接続の読み取り/書き込みライセンス
- InTouch Web Server 無制限の読み取り/書き込みライセンス
- InTouch Web Server 無制限の読み取り専用ライセンス

ライセンス ステータスの変更

有効なライセンスを取得した **Web Server** がライセンスの更新に失敗した場合、**Web Client** は猶予期間モードになります。通知メッセージが通知ページに表示されます。メッセージは、**Web Client** によるライセンス更新と失敗が発生するたびにログに記録されます。

InTouch HMI アプリケーションの場合、**Web Client WindowViewer** に接続して InTouch タグ データを受信するために **WindowViewer** は読み取り/書き込みライセンスで実行する必要があります。

ライセンスの取得

ライセンスの取得は 2 段階のプロセスです。**Web Server** が起動すると、最初にユーザー認証が行われ、次にユーザーの承認が決定されます。匿名アクセスが有効になっている場合、認証ステップがバイパスされ、InTouch **Web Client** は「ゲスト」ユーザー用の読み取り専用モードで起動します。

認証:

1. InTouch **Web Client** は Windows 認証をサポートしています。**Web Server** では、**Web Client** の使用の認証に使用される "aallnTouchUsers" または "aallnTouchRWUsers" のいずれかにユーザーが含まれているかどうかを検証されます。両方のユーザー グループはインストール時に作成されます。リモート認証サーバーの場合、サーバーにドメイン ユーザー グループを作成する必要があります。
2. **Web Client** のインストール時のログイン ユーザーは、両方のグループに自動的に追加されます。後でその他のユーザーを追加できます。新しいユーザーをグループに追加した後、変更を適用するには、新しいユーザーはログオフした後に再度ログインする必要があります。

取得:

起動時、およびユーザー認証後、**Web Server** は以下のワークフローでライセンスを取得します。

1. **Flex** オプションがコンフィグレータで有効化されているかどうか **Web Server** によってチェックされます。

Flex オプションが有効化されていない場合、**Web Server** は無制限の読み取り/書き込みライセンスの取得を試みます。(ステップ 4 に進む)
2. **Flex** および **Enterprise** オプションがコンフィグレータで有効化されている場合、**Web Server** は **Enterprise** ライセンスの取得を試みます。

コンフィグレータで **Flex** オプションが有効化されていて、**Enterprise** オプションが有効化されていない場合、**Web Server** は **Flex** ライセンスの取得を試みます。(ステップ 3 に進む)
3. ライセンスのない **Web Server** は **Flex** ライセンスの取得を試みます。

Flex ライセンスが使用可能な場合、**Web Server** は **Flex** ライセンスを取得します。

Flex ライセンスが使用できない場合、**Web Server** はライセンスのない状態になります。**Web Server** にライセンスがないときに **Web Client** が接続を試みると、ライセンス エラー ページが表示されます。
4. ライセンスのない **Web Server** は無制限の読み取り/書き込みライセンスの取得を試みます。

無制限の読み取り/書き込みライセンスが使用可能な場合、**Web Server** は無制限の読み取り/書き込みライセンスを取得します。

無制限の読み取り/書き込みライセンスが使用できない場合、**Web Server** は接続の読み取り/書き込みライセンスの取得を試みます。

5. 接続の読み取り/書き込みライセンスが使用可能な場合、**Web Server** は接続の読み取り/書き込みライセンスを取得します。

接続の読み取り/書き込みライセンスが使用できない場合、**Web Server** は無制限の読み取り専用ライセンスの取得を試みます。

6. 無制限の読み取り専用ライセンスが使用可能な場合、**Web Server** は無制限の読み取り専用ライセンスを取得します。

無制限の読み取り専用ライセンスが使用できない場合、**Web Server** は読み取り専用のシングルセッションモードになります。

ライセンス機能

InTouch Web Client は以下のライセンス機能をサポートしています。

- Flex ライセンス
- 読み取り/書き込みライセンス
- 無制限の読み取り専用ライセンス

各モードの特徴を以下に示します。

取得されたライセンス	サポートされている機能	条件
Flex ライセンス	<ul style="list-style-type: none">• 読み取り/書き込みライセンスでは無制限の接続が許可されます• InTouch と AppServer ネームスペースの両方の個人用作業領域	以下の「注記」を参照してください。

読み取り/書き込み ライセンス	<ul style="list-style-type: none"> • AppServer 属性や InTouch タグなどの外部参照への書き込み • アラームの確認とそのユーザー資格情報は、アラームを確認したオペレータとして記録されます。 • 接続の読み取り/書き込みライセンスでは、Web Server に対する複数の InTouch Web Client 接続（5、10、15 などのセット）が許可されています。 	<p>以下の条件が満たされている場合、書き込み機能およびアラームの確認機能にアクセスできます。</p> <ul style="list-style-type: none"> • Web Server が Flex、Brand Neutral 読み取り/書き込みライセンス、無制限の読み取り/書き込みライセンス、または接続の読み取り/書き込みライセンスを取得している。 • 許可されている読み取り/書き込み接続内で InTouch Web Client セッションが行われている。 • Web Client 読み取り/書き込みグループに属する InTouch Web Client セッションにユーザーがログインしている。
読み取り専用ライセンス	<ul style="list-style-type: none"> • クライアントセッションは外部参照への書き込みやアラームの確認を行うことができません。 	<p>ユーザーが読み取り専用モードにアクセスできるのは、次のいずれかの場合だけです。</p> <ul style="list-style-type: none"> • Web Server が Flex、Brand Neutral 読み取り/書き込み、無制限の読み取り/書き込みライセンス、または接続の読み取り/書き込みライセンスを取得していて、ログインユーザーが InTouch Web Client aaInTouchRWUsers ユーザーグループに属していない場合 • Web Server で無制限の読み取り専用ライセンスが取得されている場合 • Web Server でライセンスが取得されず、その Web Client セッションが最初のクライアントセッションである場合。

注：Flex ライセンスではフォールバックは許可されていません。Flex ライセンスを取得できない場合、Web Server はライセンスのない状態になります。Web Server にライセンスがない状態で InTouch Web Client が接続を試みると、ライセンスエラーページが表示されます。

シングルセッション モード

起動時にライセンスが取得できない場合、**Web Server** のはシングル **Web** セッション モードで動作し、順番に割り当てられます。

- 有効なライセンスがなく、最初のセッション ライセンスが既に取得されている場合、ライセンスが使用できないことを示すメッセージが **InTouch Web** クライアントに表示されます。
- **Web Server** で無制限のライセンスが取得された後にライセンスが失われた場合、セッションの猶予期間が設けられます。

Application Server データのサブスクリプションには、**Supervisory** ライセンスが必要です。**Supervisory** ライセンスがない場合、**Application Server** データのサブスクリプションは 2 時間後に終了します。

猶予期間

以下の条件が満たされる場合、**Web Server** は猶予期間に入ります。

- 1 つの有効なライセンスが取得された後にライセンスの有効期間が終了した。
- 1 つの有効なライセンスが取得された後、そのライセンスを取得できなくなった。

猶予期間中、接続済みおよび新規セッションですべての **InTouch Web Client** の機能を使用できます。**Web Server** が猶予期間モードであることを示す通知が **Web Client** ページに表示されます。猶予期間は 14 日間です。猶予期間が終了した後、**Web Server** は 1 つのセッション用の単一ライセンスに直ちに切り替わって機能し続けます。この場合、ライセンスは要求順に割り当てられます。同じライセンスが取得された場合、**Web Server** の猶予期間モードが終了します。

定期的な更新

有効なライセンスが取得された後、定期的なライセンス更新が行われます。定期的なライセンス更新操作では、同じライセンスを更新できるかどうかを確認されます。同じライセンスを更新できない場合、**Web Server** は猶予期間モードになります。

更新の頻度は、以下に示すライセンス タイプに依存します。

ライセンス タイプ	更新の頻度
Flex ライセンス 無制限の読み取り/書き込みライセンス 無制限の読み取り専用ライセンス	7 分ごと
接続の読み取り/書き込みライセンス	24 時間ごと

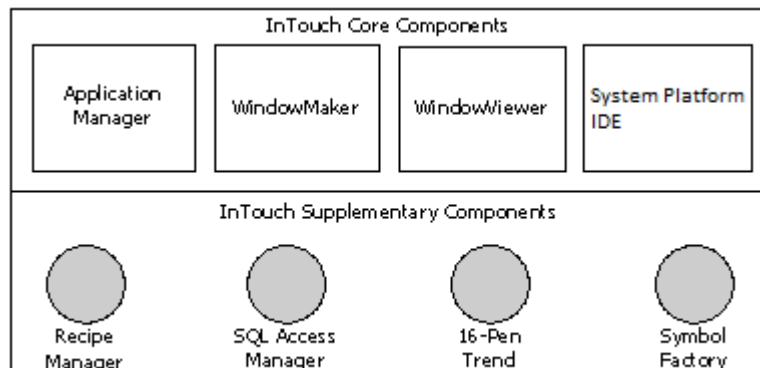
更新中、ライセンス ステータスのアップグレードやダウングレードは行われません。別のライセンスを取得するには、**Web Server** を再起動して、ライセンス取得の起動シーケンスを開始します。

ライセンスの解放

Web Server がシャットダウンされた場合、クライアントセッションの数や **Web Server** ライセンスの現在の状態に関係なく、**Web Server** で取得されたすべてのライセンスが解放されます。すべての **Web Client** も直ちに終了します。

補足コンポーネントについて

オプションで、一連の 4 つの補足コンポーネントを InTouch HMI コア コンポーネントと共にインストールすることもできます。これらの補足コンポーネントにより、InTouch HMI アプリケーションの追加的な機能が提供されます。

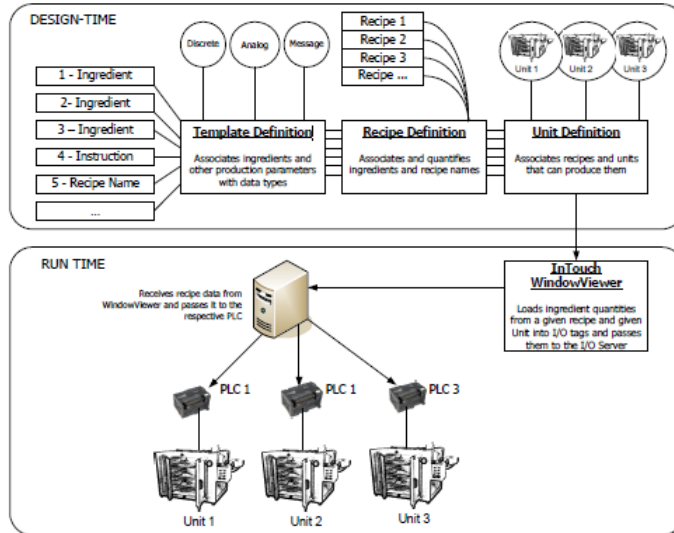


- レシピ マネージャには、製造レシピを作成するための一連のスプレッドシートおよびスクリプト関数が含まれています。
- SQL アクセス マネージャは、データベースに InTouch データを保存するためのプログラムと一連の SQL 関数で構成されます。
- 16 ペン トレンドには、リアルタイムトレンドおよび履歴トレンドを作成するための、トレンドウィザードおよびスクリプト関数が含まれています。
- Symbol Factory では、一連の産業用シンボルが提供されており、プロセス コンポーネントを表すために InTouch アプリケーションに配置することができます。

レシピ マネージャの使用

製造業者は、標準の数量の原料を使用する繰り返し可能なプロシージャに従って製品を製造します。つまり、製品はレシピに従って製造されます。レシピは原料、その数量、およびそれらをどのように組み合わせる最終的な製品を製造するかについて説明しています。最もわかりやすい例として、たとえばベーカリーではクッキーを作るためのすべての原料と手順を一覧表示した基本レシピに従って製造を行っています。

レシピ マネージャは、製造レシピを作成するプロセスを簡略化するために使用できる InTouch HMI 用の補足コンポーネントです。以下の図は、製品を作成するプロセスを管理するためにレシピ マネージャがレシピテンプレートから情報を取得する方法を示しています。



レシピ マネージャの概要

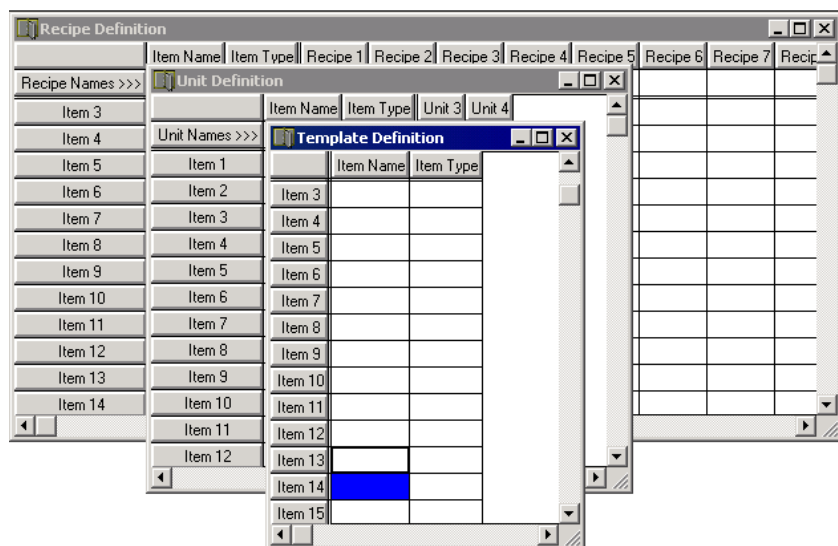
レシピ マネージャは InTouch と共にオプション コンポーネントとしてインストールできます。レシピ マネージャは、レシピ マネージャ ユーティリティと InTouch レシピ スクリプト関数のセットで構成されます。

レシピ マネージャ ユーティリティには、WindowMaker から起動するか、C:\Program Files (x86)\Wonderware\InTouch\ から Recipe.exe を起動することによってアクセスできます。レシピ マネージャ ユーティリティにはレシピ テンプレートを作成および編集するためのインターフェイスが含まれています。レシピ マネージャはテンプレートをレシピ ファイルに保存します。

一般的に、製造プロセスに関連付けられているタグ変数は QuickScript を使用してレシピ テンプレート ファイル内のデータにアクセスします。レシピ マネージャは、テンプレート ファイルに含まれている製造レシピを選択、ロード、変更、作成、および削除するための QuickScript 関数のセットを含みます。

レシピ マネージャ ユーティリティ

レシピ マネージャ ユーティリティは、レシピ テンプレート ファイルを作成および維持するためのスプレッドシートに似たユーザー インターフェイスを提供します。ファイルは 3 つのテンプレートで構成されています。これらのテンプレートは、各テンプレートのスプレッドシートのセルにあるデータを追加または変更することによって作成または編集できます。



これらのテンプレートは、カンマ区切り値（CSV）ファイルに保存されます。レシピテンプレート定義は、メモ帳または Excel など .csv ファイル形式をサポートする任意のプログラムで作成または編集できます。ただし、レシピマネージャでは、テンプレートを確実に簡単に作成および維持するための書式設定済みスプレッドシートおよび編集ツールのセットを提供しています。

Recipe Template Files

レシピマネージャテンプレートファイルには、以下の情報が含まれています。

- レシピで使用される原料の名前およびそのデータタイプ。
- InTouch タグ変数をレシピ原料値と関連付けるユニット名。
- レシピインスタンスで使用される各原料の数量または値を含むレシピ名。

Template Definition

テンプレート定義テンプレートは、すべてのレシピ原料を定義します。データタイプは各レシピ原料と関連付けられています。すべての原料データタイプは、アナログ型、論理型、またはメッセージ型のいずれかです。原料名は InTouch タグ変数である必要はありません。

Unit Definition

ユニット定義テンプレートは、InTouch タグ変数をレシピ原料に関連付けます。多くの異なるロード定義を作成できます。これらの定義はユニットと呼ばれます。RecipeLoad() 関数を使用すると、レシピの特定のインスタンスに関連付けられている InTouch タグ変数にロードできます。ユニット定義は、ファイルに定義されているすべての原料またはこれらの原料のサブセットで構成されます。

注意：ユニットタグ変数は、InTouch ウィンドウで表示、編集できるメモリ型であるか、PLC に直接ロードできる I/O タグ変数のいずれかです。

Recipe Definition

レシピ定義テンプレートは、各レシピの名前およびレシピで使用される原料数量を指定します。レシピインスタンスは、ランタイムでレシピ関数を使用して変更、作成、または削除できます。

レシピマネージャでのレシピデータの編集

製造レシピは、一連の連続タスクを完了することによって作成します。以下のリストは、レシピを作成するレシピマネージャのタスクおよびタスクが完了する順番を示しています。

- レシピ マネージャの編集グリッドの設定
- テンプレートのデータの編集
- テンプレート定義テンプレートへの原料名およびユニットタイプの割り当て
- ユニット定義テンプレートの原料への InTouch タグ変数のマッピング
- レシピ定義テンプレート内のレシピ原料への値の割り当て

レシピ マネージャの編集グリッドの設定

製造レシピを作成する前に、レシピ マネージャを設定する必要があります。レシピ マネージャ編集関数を設定するには、2 つのタスクがあります。

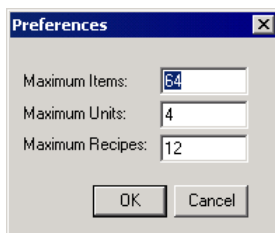
- テンプレート アイテムの最大制限を設定します。
- Enter キー スクロール機能を設定します。

レシピを作成する前に、レシピ テンプレートに入力できるアイテムの最大数を設定する必要があります。アイテム、ユニット、およびレシピ名の最大制限セットを割り当てる必要があります。

テンプレートは最大で 9999 のアイテム、ユニット、およびレシピ名を含むことができます。ただし、最大制限数が大きくなると、システムのパフォーマンスに影響が与えられる可能性があります。また、設定した最大制限がコンピュータの使用可能メモリよりも多くのメモリを必要とした場合にはエラーメッセージが表示されます。

レシピ テンプレートの最大制限を設定するには

1. 以下のいずれかの方法で、レシピ マネージャを起動します。
 - WindowMaker を起動します。[ツール] ビューで、[アプリケーション] を展開し、[レシピ マネージャ] を選択します。
- [レシピ マネージャ] ダイアログ ボックスが表示されます。
2. [オプション] メニューで、[環境設定] をクリックします。[環境設定] ダイアログ ボックスが表示されます。



3. [最大アイテム数] ボックスに、テンプレート定義テンプレートで許容されているアイテム名の最大数を入力します。
4. [最大ユニット数] ボックスに、ユニット定義テンプレートで許容されているユニットの最大数を入力します。
5. [最大レシピ数] ボックスに、レシピ定義テンプレートで許容されているレシピ名の最大数を入力します。

注意: [環境設定] ダイアログ ボックスに設定した値は、作成するすべてのレシピ テンプレート ファイルに適用されます。これらの値を変更すると、既存のレシピ テンプレート ファイルもすべて変更されます。

1. [OK] をクリックします。

レシピ マネージャには、レシピ テンプレートへのデータ入力を簡潔化するオプションが含まれています。[入力後セル移動] オプションを選択した場合、Enter キーを押すとカーソルがカラム内の次のセルに移動します。

Enter キー テンプレート スクロールを設定するには

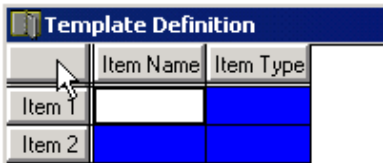
1. レシピ マネージャを開きます。

デフォルトでは、レシピ マネージャはテンプレート スプレッドシート内の次のセルには自動的にスクロールしません。

2. [オプション] メニューで、[入力後セル移動] をクリックして、セル スクロールを設定します。
3. セル スクロールをオフにする場合は、[入力後セル移動] を再びクリックします。

編集グリッドの操作

レシピ マネージャには、テンプレートからのデータを追加、変更、または削除する編集コマンドのセットが含まれます。一般に、テンプレートで編集するデータを選択して、編集作業を行います。以下の表には、テンプレート データを入力して選択するためのレシピ テンプレートの一般的な機能が説明されています。

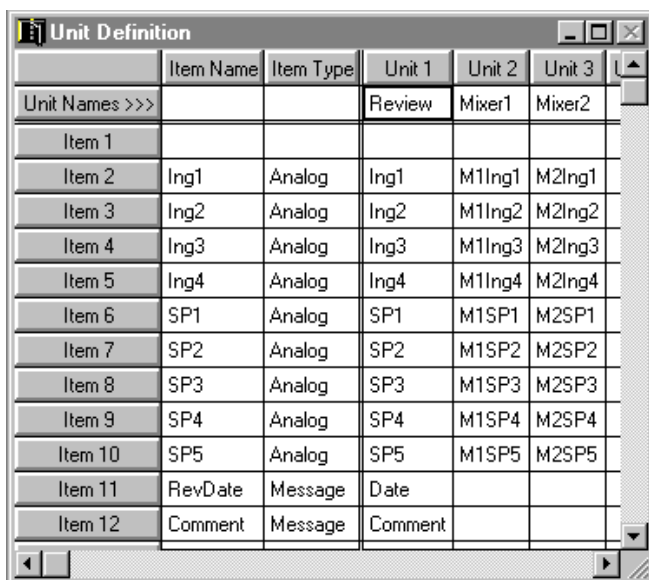
機能	説明
入力ボックス	選択したテンプレート セル用のデータを入力するために使用するテキスト入力ボックス。テンプレート セルが選択されると、その内容が [レシピ マネージャ] ダイアログ ボックスの上部付近のテキスト入力ボックスに表示されます。
すべてのセルの選択	すべてのセルを選択するには、テンプレートの左上のセルをクリックします。 
すべての行の選択	行内のすべてのセルを選択するには、テンプレートの行名をクリックします。
すべてのカラムの選択	カラム内のすべてのセルを選択するには、テンプレートのカラム見出しをクリックします。
すべてのカラムの自動サイズ調節	テンプレートのすべてのカラムを最も長いエントリの幅に自動的にサイズ変更するには、テンプレートをダブルクリックします。
カラムの自動サイズ調節	カラムを最も長いエントリの幅にサイズ変更するには、見出しをダブルクリックします。 注記: テンプレート定義テンプレート内のデータ タイプ カラムは自動サイズ変更できません。

テンプレートを編集するときには、以下の手順を実行します。

- 一定範囲のセルからデータを消去する
- 一定範囲のセルからのデータを近接する選択した一定範囲のセルへコピーする
- テンプレート定義テンプレートに行を挿入する
- テンプレート内にカラムを挿入する
- 行をテンプレート定義テンプレートから削除する
- テンプレートからカラムを削除する

一定範囲のセルからデータを消去するには

1. テンプレートから一定範囲のセルを選択します。



	Item Name	Item Type	Unit 1	Unit 2	Unit 3
Unit Names >>>			Review	Mixer1	Mixer2
Item 1					
Item 2	Ing1	Analog	Ing1	M1Ing1	M2Ing1
Item 3	Ing2	Analog	Ing2	M1Ing2	M2Ing2
Item 4	Ing3	Analog	Ing3	M1Ing3	M2Ing3
Item 5	Ing4	Analog	Ing4	M1Ing4	M2Ing4
Item 6	SP1	Analog	SP1	M1SP1	M2SP1
Item 7	SP2	Analog	SP2	M1SP2	M2SP2
Item 8	SP3	Analog	SP3	M1SP3	M2SP3
Item 9	SP4	Analog	SP4	M1SP4	M2SP4
Item 10	SP5	Analog	SP5	M1SP5	M2SP5
Item 11	RevDate	Message	Date		
Item 12	Comment	Message	Comment		

2. [編集] メニューで、[クリア] をクリックします。選択した一定範囲のセルがクリアされることを確認するよう促すメッセージが表示されます。
3. [はい] をクリックします。テンプレートによって選択した一定範囲のセルからデータがクリアされます。

一定範囲のセルを近接の選択範囲へコピーするには

1. コピーするセルまたは一定範囲のセルを選択します。
2. データをコピーする近接の一定範囲のセルを選択します。

Recipe Definition					
	Item Name	Item Type	Recipe 1	Recipe 2	Recipe 3
Recipe Names >>>			Recipe 1		
Item 1					
Item 2	Ing1	Analog	21		
Item 3	Ing2	Analog	22		
Item 4	Ing3	Analog	23		
Item 5	Ing4	Analog	24		
Item 6	SP1	Analog	21		
Item 7	SP2	Analog	22		
Item 8	SP3	Analog	23		
Item 9	SP4	Analog	24		
Item 10	SP5	Analog	25		
Item 11	RevDate	Message	7/6/97 3:36:39 PM		
Item 12	Comment	Message	Comment for Recipe 2		

選択した範囲は同じサイズであり、選択したオリジナルのセルの範囲の上、下、右、または左である必要があります。

3. **[編集]** メニューで、適切な塗りつぶしコマンドを選択します。データが選択した範囲のセルにコピーされます。

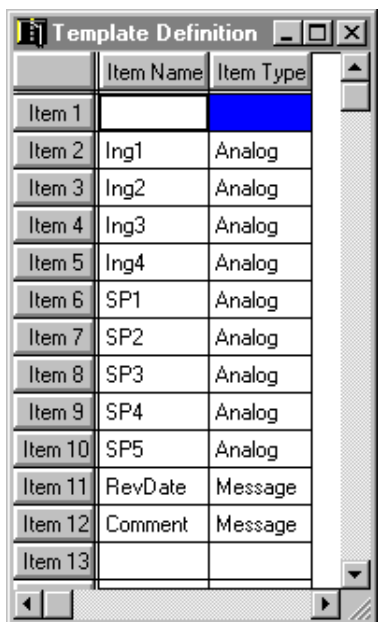
注記: データがコピーされる新しいカラムが最大エントリを含むのに十分な大きさではない場合、カラムの見出しをダブルクリックして、幅を最も長いエントリに合わせて変更します。

テンプレート定義テンプレートに行を挿入するには

1. テンプレート定義テンプレートを選択します。
2. テンプレート定義テンプレートで、**アイテム番号**をクリックして、上に新しい行を挿入する行を選択します。

レシピ定義テンプレートまたはユニット定義テンプレートでは、直接行を挿入することはできません。その代わりに、テンプレート定義へのすべての変更はレシピ定義テンプレートおよびユニット定義テンプレートによって自動的に継承されます。

3. **[編集]** メニューで、**[挿入]** をクリックします。新しい行が選択した行のすぐ上に挿入され、後続のすべての行の番号は自動的に変更されます。



	Item Name	Item Type
Item 1		
Item 2	Ing1	Analog
Item 3	Ing2	Analog
Item 4	Ing3	Analog
Item 5	Ing4	Analog
Item 6	SP1	Analog
Item 7	SP2	Analog
Item 8	SP3	Analog
Item 9	SP4	Analog
Item 10	SP5	Analog
Item 11	RevDate	Message
Item 12	Comment	Message
Item 13		

注記: レシピマネージャの「環境設定」に設定された最大値に達した場合、**「挿入」** コマンドは非アクティブになります。レシピテンプレートにアイテム/ユニット/レシピを追加するには、指定した数値を増加させる必要があります。

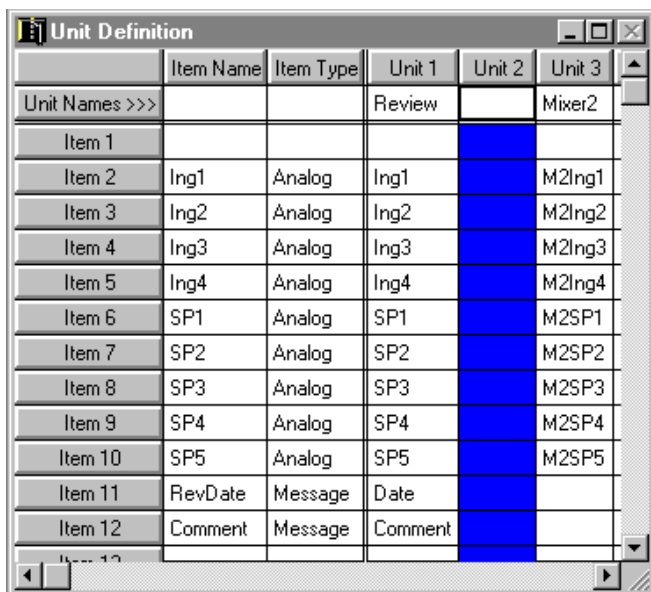
「環境設定」を変更すると、変更は既存のすべてのレシピテンプレートファイルに適用されます。

カラムを挿入するには

1. ユニット番号またはレシピ番号をクリックして、挿入したカラムの右側に配置するカラムを選択します。

カラムはレシピ定義テンプレートまたはユニット定義テンプレートに挿入できます。

2. **「編集」** メニューで、**「挿入」** をクリックします。新しいカラムが選択したカラムの左側に挿入されます。



	Item Name	Item Type	Unit 1	Unit 2	Unit 3
Unit Names >>>			Review		Mixer2
Item 1					
Item 2	Ing1	Analog	Ing1		M2Ing1
Item 3	Ing2	Analog	Ing2		M2Ing2
Item 4	Ing3	Analog	Ing3		M2Ing3
Item 5	Ing4	Analog	Ing4		M2Ing4
Item 6	SP1	Analog	SP1		M2SP1
Item 7	SP2	Analog	SP2		M2SP2
Item 8	SP3	Analog	SP3		M2SP3
Item 9	SP4	Analog	SP4		M2SP4
Item 10	SP5	Analog	SP5		M2SP5
Item 11	RevDate	Message	Date		
Item 12	Comment	Message	Comment		
Item 13					

この例では、**Mixer2** データが **[ユニット 3]** カラムに移動され、新しいカラムが **[ユニット 2]** として挿入されています。

カラムを削除するには

1. **ユニット番号**または**レシピ番号**カラム見出しをクリックして、削除するカラムを選択します。
レシピ定義テンプレートまたはユニット定義テンプレートからカラムを削除できます。
2. **[編集]** メニューで、**[削除]** をクリックします。削除を確認する確認メッセージ ダイアログ ボックスが表示されます。
3. **[はい]** をクリックします。カラムがテンプレートから削除され、残りのカラムの番号が変更されます。

行を削除するには

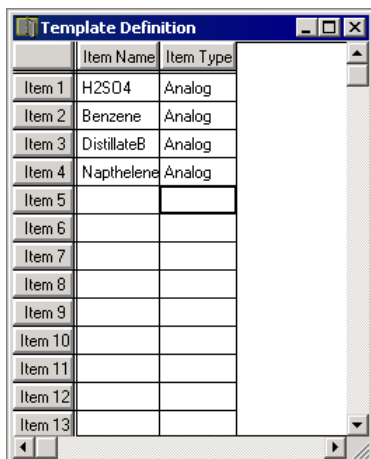
1. **テンプレート定義**テンプレートを選択します。
テンプレート定義テンプレートからは行を削除できますが、**レシピ定義**テンプレートまたは**ユニット定義**テンプレートからは行を削除することはできません。
2. **アイテム番号**行見出しをクリックして、削除する行を選択します。
3. **[編集]** メニューで、**[削除]** をクリックします。削除を確認する確認メッセージ ダイアログ ボックスが表示されます。
4. **[はい]** をクリックします。テンプレートから行が削除されます。

原料名とデータ タイプの定義

テンプレート定義テンプレートには、各原料に関連付けられているレシピ原料およびアイテム タイプが一覧表示されています。データをその他のレシピ テンプレートに追加する前に、テンプレート定義テンプレートを完了する必要があります。

テンプレート定義テンプレートを定義するには

1. レシピ マネージャを起動します。
2. **[ファイル]** メニューで、**[新規]** をクリックします。3 つのレシピ マネージャ テンプレートが表示されます。
3. **[テンプレート定義]** タイトル バーをクリックして、テンプレート ウィンドウを選択します。
4. **[アイテム名]** カラム セルに、レシピ原料に選択した名前を入力します。
各セルには 1 つの原料のみを入力できます。
5. **[アイテム タイプ]** カラム セルに、それぞれのレシピ原料に対する有効なアイテム タイプを入力します。

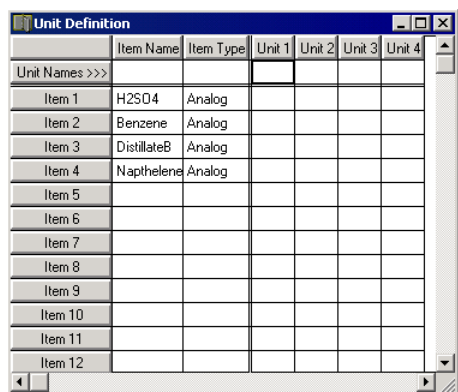


	Item Name	Item Type
Item 1	H2SO4	Analog
Item 2	Benzene	Analog
Item 3	DistillateB	Analog
Item 4	Napthelene	Analog
Item 5		
Item 6		
Item 7		
Item 8		
Item 9		
Item 10		
Item 11		
Item 12		
Item 13		

有効なアイテムタイプは、アナログ型、論理型、またはメッセージ型です。アナログ型の場合は **A**、論理型の場合は **D**、メッセージ型の場合は **M** を入力します。**Enter** キーを押すと、レシピマネージャによって残りのアイテムタイプが自動的に完了されます。

原料への InTouch タグのマッピング

ユニット定義テンプレートは、InTouch タグを特定のユニットのレシピ原料に関連付けます。次の図に示すように、ユニット定義テンプレートの最初の 2 つのカラムには、テンプレート定義テンプレートからのアイテム名およびアイテムタイプが一覧表示されています。



	Item Name	Item Type	Unit 1	Unit 2	Unit 3	Unit 4
Unit Names >>>						
Item 1	H2SO4	Analog				
Item 2	Benzene	Analog				
Item 3	DistillateB	Analog				
Item 4	Napthelene	Analog				
Item 5						
Item 6						
Item 7						
Item 8						
Item 9						
Item 10						
Item 11						
Item 12						

ユニットに定義されているタグは、PLC データを I/O サーバーから取得するメモリ型タグまたはリモートタグとなります。

RecipeLoad() 関数を InTouch QuickScript で使用する場合、ユニット名を指定する必要があります。レシピ名定義に含まれる値は、QuickScript が実行されると、ユニット名に指定されているタグにロードされます。

ユニット定義テンプレートを定義するには

1. **[ユニット定義]** テンプレートのタイトルバーをクリックして、テンプレート ウィンドウを選択します。

	Item Name	Item Type	Unit 1	Unit 2	Unit 3
Unit Names >>>			Review	Mixer 1	Mixer 2
Item 1	Ing1	Analog	Ing1	M1Ing1	M2Ing1
Item 2	Ing2	Analog	Ing2	M1Ing2	M2Ing2
Item 3	Ing3	Analog	Ing3	M1Ing3	M2Ing3
Item 4	Ing4	Analog	Ing4	M1Ing4	M2Ing4
Item 5	SP1	Analog	SP1	M1SP1	M2SP1
Item 6	SP2	Analog	SP2	M1SP2	M2SP2
Item 7	SP3	Analog	SP3	M1SP3	M2SP3
Item 8	SP4	Analog	SP4	M1SP4	M2SP4
Item 9	SP5	Analog	SP5	M1SP5	M2SP5
Item 10	RevDate	Message	Date		
Item 11	Comment	Message	Comment		

2. [ユニット名] 行に、定義する各ユニットの名前を入力します。
3. [ユニット番号] カラムセルで、以下のいずれかの方法を使用して、それぞれ対応するレシピ原料に InTouch タグの名前を入力します。
 - タグ名を入力します。
 - WindowMaker を実行している場合は、セルをダブルクリックして、[タグの選択] ダイアログボックスを表示します。次に、セルに挿入する目的のタグをダブルクリックするか、選択して、[OK] をクリックします。
4. それぞれのユニット／レシピの組み合わせに対しても同じ手順を繰り返します。

異なるレシピ内の原料の値の定義

レシピ定義テンプレートは、各レシピの名前およびレシピで使用される原料数量を指定します。レシピ定義テンプレートは、以前定義されたテンプレート定義テンプレートからのアイテム名およびデータタイプ情報を表示します。

原料値は、**RecipeLoad()** 関数が InTouch QuickScript 内で実行されると InTouch タグ変数内にロードされます。

レシピ定義テンプレートを定義するには

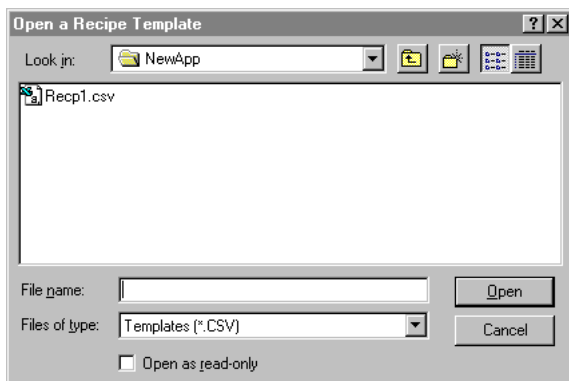
1. レシピ定義テンプレートのタイトルバーをクリックして、テンプレート ウィンドウを選択します。
2. [レシピ名] 行に、定義する各レシピの名前を入力します。

	Item Name	Item Type	Recipe 1	Recipe 2
Recipe Names >>>			Recipe 1	Recipe 2
Item 1	Ing1	Analog	11	21
Item 2	Ing2	Analog	12	22
Item 3	Ing3	Analog	13	23
Item 4	Ing4	Analog	14	24
Item 5	SP1	Analog	11	21
Item 6	SP2	Analog	12	22
Item 7	SP3	Analog	13	23
Item 8	SP4	Analog	14	24
Item 9	SP5	Analog	15	25
Item 10	RevDate	Message	7/15/97 3:19:56 PM	7/6/97 3:36:39 PM
Item 11	Comment	Message	A Comment	Comment for Recipe 2

3. [レシピ番号] カラムセルに、[アイテム名] カラム内の対応する各レシピ原料の値を入力します。
4. [ファイル] メニューで、[保存] をクリックして、レシピテンプレートファイルを保存します。

既存のレシピテンプレートファイルを開くには

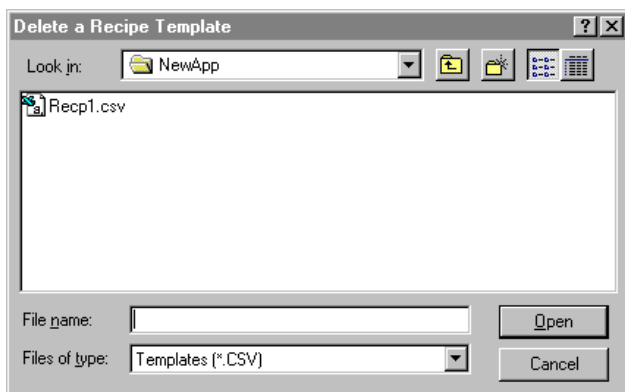
1. レシピマネージャを開きます。
2. [ファイル] メニューで、[開く] をクリックします。[レシピテンプレートを開く] ダイアログボックスが表示されます。



3. レシピファイルを検索して選択し、[開く] をクリックします。ファイルの3つのレシピテンプレートがレシピマネージャに表示され、編集できます。

レシピテンプレートファイルを削除するには

1. [ファイル] メニューで、[削除] をクリックします。[レシピテンプレートを削除] ダイアログボックスが表示されます。



2. レシピファイルを検索して選択し、[開く] をクリックするか、ファイル名をダブルクリックします。削除を確認するメッセージが表示されます。

注: 開いているレシピテンプレートファイルは削除できません。

1. [はい] をクリックして、ファイルを削除します。

別のアプリケーションでのレシピデータの編集

レシピテンプレート定義は、カンマ区切りデータをサポートする任意のプログラムで作成および編集できます。Microsoft Excel またはメモ帳を使用して、レシピマネージャテンプレートファイルを作成および編集できます。

Excel でのレシピ テンプレート ファイルの使用

レシピ マネージャ ユーティリティを使用しない場合は、Excel を使用してレシピ テンプレートを作成または編集できます。Excel で作成または編集したレシピ マネージャ テンプレートは、.csv ファイル名拡張子を付けたファイルに保存する必要があります。

Microsoft Excel で既存のレシピ テンプレート ファイルを開くには

1. Excel を起動します。
2. [ファイル] メニューで、[開く] をクリックします。[開く] ダイアログ ボックスが表示されます。
3. .csv ファイルを検索して選択し、[開く] をクリックするか、ファイル名をダブルクリックします。Excel にファイルの内容が表示されます。

	A	B	C	D	E	F	G
1	Ingredient	Ingredient	Unit	Unit	Unit	Recipe	Recipe
2	Names		Review	Mixer1	Mixer2	Recipe 1	Recipe 1
3							
4	Ing1	Analog	Ing1	M1Ing1	M2Ing1	21	21
5	Ing2	Analog	Ing2	M1Ing2	M2Ing2	22	22
6	Ing3	Analog	Ing3	M1Ing3	M2Ing3	23	23
7	Ing4	Analog	Ing4	M1Ing4	M2Ing4	24	24
8	SP1	Analog	SP1	M1SP1	M2SP1	21	
9	SP2	Analog	SP2	M1SP2	M2SP2	22	
10	SP3	Analog	SP3	M1SP3	M2SP3	23	
11	SP4	Analog	SP4	M1SP4	M2SP4	24	
12	SP5	Analog	SP5	M1SP5	M2SP5	25	
13	RevDate	Message	Date			7/6/97 15:36	
14	Comment	Message	Comment			Comment for Recipe 2	

4. レシピ ファイルの内容を編集して、変更内容を保存します。

Excel で新しいレシピ テンプレート ファイルを作成するには

1. Excel を起動します。
2. 新しいワークブックを作成します。
3. 次の図に示すように、スプレッドシートにレシピ データを入力します。

	A	B	C	D	E	F	G
1	IngredientName	IngredientType	Unit	Unit	Unit	Mixer 3	Recipe
2	Names		Review	Mixer 1	Mixer 2	Mixer 3	Recipe 1
3	Ing1	Analog	Ing1	M1Ing1	M2Ing1	M3Ing1	11
4							
5							
6							
7							

エントリは、図に示されている順番で作成する必要があります。ユニット名はレシピ名を含むカラムの左にあるカラムで定義する必要があります。

4. スプレッドシートを .csv ファイル名拡張子を付けて保存します。

メモ帳でのレシピ テンプレート ファイルの使用

レシピ マネージャ ユーティリティを使用しない場合は、メモ帳を使用してレシピ テンプレートを作成または編集できます。メモ帳で作成または編集したレシピ マネージャ テンプレートは、.csv ファイル名拡張子を付けたファイルに保存する必要があります。

メモ帳で既存のレシピ テンプレート ファイルを開くには

1. メモ帳を起動します。
2. [ファイル] メニューで、[開く] をクリックします。[開く] ダイアログ ボックスが表示されます。
3. レシピ ファイルを検索して選択し、[開く] をクリックするか、ファイル名をダブルクリックします。
4. レシピ ファイルの内容を編集して、変更内容を保存します。

メモ帳で新しいレシピ テンプレート ファイルを作成するには

1. メモ帳を起動します。
2. [ファイル] メニューで、[新規] をクリックします。
3. 次のデータをこの形式で入力します。

```
:IngredientName,IngredientType[,Unit]...[,Recipe]...
:Names,,[,UnitName]...[,RecipeName]...
IngredientName,{Analog,Discrete,Message},{[,tag]...[,value]}
```

注記: すべてのユニット名は、レシピ名が定義される前にファイル内に定義する必要があります。

4. ファイルを .csv ファイル名拡張子を付けて保存します。

複雑な構造を作成するためのレシピのネスト

複数のレシピ テンプレート ファイルを InTouch QuickScript を使用して相互にリンクして、複雑なアプリケーションを作成できます。レシピ テンプレート ファイルは、ユニット名 テンプレート内のメッセージ型タグ変数に関連付けられている原料名を定義することによってリンクします。次に、レシピの名前を持つメッセージ型タグ変数をロードします。

レシピ テンプレート ファイルをリンクすると、異なるレシピ ファイルの様々なレシピによって使用されるマシン設定パラメータを定義するマスタ レシピ テンプレート ファイルを作成できます。このようなタイプの情報を 1 つの中央ファイルに保持すると、変更するたびにデータを維持および更新するための時間を大幅に削減できます。

次の図では、アイテム名 **Setup** タグ変数がメッセージ型として定義され、ユニットにはこのアイテム用の **Setup** メッセージ型タグ変数が含まれます。各レシピには、レシピが選択されたときに **Setup** タグ変数にロードされる別のレシピ ファイルに 2 番目のレシピ名が定義されます。

RECFILEA.CSV									
	1	2	3	4	5	6	7	8	9
1	Item Name	Item Type	Unit	Unit	Unit	Unit	Recipe	Recipe	Recipe
2	:Names		Review	Mixer 1	Mixer 2	Mixer 3	Recipe 1	Recipe 2	Recipe 3
3	Ing1	Analog	Ing1	M1Ing1	M2Ing1	M3Ing1	11	21	31
4	Ing2	Analog	Ing2	M1Ing2	M2Ing2	M3Ing2	12	22	99
5	Ing3	Analog	Ing3	M1Ing3	M2Ing3	M3Ing3	13	23	66
6	Ing4	Analog	Ing4	M1Ing4	M2Ing4	M3Ing4	14	24	34
7	SP1	Analog	SP1	M1SP1	M2SP1	M3SP1	11	21	31
8	SP2	Analog	SP2	M1SP2	M2SP2	M3SP2	12	22	32
9	SP3	Analog	SP3	M1SP3	M2SP3	M3SP3	13	23	33
10	SP4	Analog	SP4	M1SP4	M2SP4	M3SP4	14	24	34
11	SP5	Analog	SP5	M1SP5	M2SP5	M3SP5	15	25	35
12	Setup	Message	Setup	LinkFile	LinkFile	LinkFile	Setup2A	Setup3A	Setup1A
13									

この処理を実行するには、次のスクリプトを入力します。

```
RecipeName="Recipe2";
RecipeLoad("c:\recipe\recfilea.csv", "Review", RecipeName);
```

このスクリプトを実行すると、**Setup** タグ変数の値が **Setup3A** となり、**Review** ユニットのロードされます。次のスクリプトを実行することによって、**Setup** タグ変数の値が、マシンセットアップパラメータを **PLC1** ユニットに対して定義されているタグ変数にロードする次のレシピロードのレシピ名として使用されます。

```
RecipeLoad("c:\recipe\machine.csv", "PLC1", Setup);
```

MACHINE.CSV							
1	Item Name	Item Type	Unit	Recipe	Recipe	Recipe	
2	Names		PLC1	Setup1A	Setup2A	Setup3A	
3	PARM1	Analog	PARM1	11	21	31	
4	PARM2	Analog	PARM2	12	22	99	
5	PARM3	Analog	PARM3	13	23	66	
6	PARM4	Analog	PARM4	14	24	34	
7	PARM5	Analog	PARM5	11	21	31	
8	PARM6	Analog	PARM6	12	22	32	
9	PARM7	Analog	PARM7	13	23	33	
10	PARM8	Analog	PARM8	14	24	34	
11	PARM9	Analog	PARM9	15	25	35	
12							

InTouch でのレシピの使用

InTouch QuickScript を使用して、レシピテンプレートファイルと対話します。レシピマネージャには、QuickScript に挿入できる一組のスクリプト関数が含まれます。これらの関数を含むスクリプトを使用すると、レシピテンプレートファイルのレコードを選択、変更、挿入、または削除できます。

レシピ関数は、InTouch スクリプトエディタを使用して、任意のタイプのスクリプトに追加できます。次の図は、レシピ関数を一覧表示する InTouch スクリプトエディタのウィンドウを示しています。すべてのレシピ関数は、関数名の一部として **Recipe** プリフィックスによって識別されます。

Choose function...		Find: RecipeDelete		
RecipeDelete	SPCMoveScooter	SQLClearTable	SQLInsertEnd	SQLSetParamDate
RecipeGetMessage	SPCSaveSample	SQLCommit	SQLInsertExecute	SQLSetParamDateTime
RecipeLoad	SPCSelectDataset	SQLConnect	SQLInsertPrepare	SQLSetParamDecimal
RecipeSave	SPCSelectProduct	SQLCreateTable	SQLLast	SQLSetParamFloat
RecipeSelectNextRecipe	SPCSetControlLimits	SQLDelete	SQLLoadStatement	SQLSetParamInt
RecipeSelectPreviousRecipe	SPCSetMeasurement	SQLDisconnect	SQLManageDSN	SQLSetParamLong
RecipeSelectRecipe	SPCSetProductCollected	SQLDropTable	SQLNext	SQLSetParamNull
RecipeSelectUnit	SPCSetProductDisplayed	SQLEnd	SQLNumRows	SQLSetParamTime
SPCConnect	SPCSetRangeLimits	SQLErrorMsg	SQLPrepareStatement	SQLSetStatement
SPCDataSetDlg	SPCSetSpecLimits	SQLExecute	SQLPrev	SQLTransact
SPCDisconnect	SQLAppendStatement	SQLFirst	SQLRollback	SQLUpdate
SPCDisplayData	SQLClearParam	SQLGetRecord	SQLSelect	SQLUpdateCurrent
SPCLocateScooter	SQLClearStatement	SQLInsert	SQLSetParamChar	

InTouch レシピ関数は、レシピテンプレートファイルに直接読み込みしたり、書き込みしたりします。したがって、レシピ関数が InTouch QuickScript で適切に実行されるようにするために、レシピマネージャプログラムを実行する必要はありません。

レシピテンプレートファイルが InTouch HMI によって使用されている場合、作成する新しいレシピまたは既存のレシピへの変更はレシピテンプレートファイルには書き込まれません。レシピマネージャは、レシピテンプレートファイルのみを作成します。レシピテンプレートファイルを作成した後は、レシピマネージャを閉じます。

レシピ関数をスクリプトに自動的に挿入するには

1. InTouch スクリプトエディタを開きます。
2. スクリプト内でレシピ関数を挿入する場所にカーソルを置きます。

3. [関数] で、[アドオン] をクリックします。[関数の選択] ダイアログ ボックスが表示されます。
4. QuickScript に挿入するレシピ関数をクリックします。ダイアログ ボックスが閉じ、関数がカーソル位置に挿入されます。

レシピ ファイルに対するレシピ データのロードと保存

レシピ マネージャは、レシピ ファイル内のレシピ データをロードおよび保存する別の InTouch QuickScript 関数を提供しています。

RecipeLoad() 関数

RecipeLoad() 関数は、レシピからのデータを InTouch アプリケーションの特定のタグ変数ユニットにロードします。

カテゴリ

レシピ

構文

```
RecipeLoad("Filename","UnitName","RecipeName");
```

引数

FileName

レシピ テンプレート ファイルの名前。FileName に関連付けられる値は、文字列定数またはレシピ テンプレート ファイルの名前を含むメッセージ型タグ変数となります。

UnitName

指定したレシピ テンプレート ファイル内の特定ユニットの名前。RecipeSelectUnit() 関数は、このパラメータに値を返します。UnitName に関連付けられる値は、文字列定数またはユニットの名前を含むメッセージ型タグ変数です。

RecipeName

指定したレシピ テンプレート ファイルの特定レシピの名前。RecipeName に関連付けられる値は、文字列定数またはレシピの名前を含むメッセージ型タグ変数です。

例

次のステートメントは、Recipe1 レシピの値を Unit に定義されているタグ変数にロードします。

```
RecipeLoad("c:\recipe\recfile.csv", "Unit1", "Recipe1");
```

RecipeSave() 関数

RecipeSave() 関数は、新しいレシピまたは既存のレシピに行った変更を指定したレシピ テンプレート ファイルに保存します。

カテゴリ

レシピ

構文

```
RecipeSave("Filename","UnitName","RecipeName");
```

引数

FileName

レシピ テンプレート ファイルの名前。FileName に関連付けられる値は、文字列定数またはレシピ テンプレート ファイルの名前を含むメッセージ型タグ変数となります。

UnitName

関数によって使用される指定したレシピテンプレートファイルの特定ユニットの名前。

RecipeSelectUnit() 関数は、このパラメータに値を返します。**UnitName** に関連付けられる値は、文字列定数またはユニットの名前を含むメッセージ型タグ変数です。

RecipeName

指定したレシピテンプレートファイルの特定レシピの名前。**RecipeName** に関連付けられる値は、文字列定数またはレシピの名前を含むメッセージ型タグ変数です。

例

次の例は、Recipe3 レシピに行われた変更を **recfile.csv** ファイルに保存します。Recipe3 が現在 **recfile.csv** ファイルに存在しない場合は、作成されます。値は、Unit2 に対して定義されたタグ変数の値を設定します。

```
RecipeSave("c:\recipe\recfile.csv", "Unit2", "Recipe3");
```

レシピファイルからのレシピの削除

RecipeDelete 関数を使用して、指定したレシピテンプレートファイルからレシピを削除します。

RecipeDelete() 関数

RecipeDelete 関数は、指定したレシピテンプレートファイルからレシピを削除します。

カテゴリ

レシピ

構文

```
RecipeDelete("Filename", "RecipeName");
```

引数

FileName

レシピテンプレートファイルの名前。**FileName** に関連付けられる値は、文字列定数またはレシピテンプレートファイルの名前を含むメッセージ型タグ変数となります。

RecipeName

指定したレシピテンプレートファイルの特定レシピの名前。**RecipeName** に関連付けられる値は、文字列定数またはレシピの名前を含むメッセージ型タグ変数です。

例

次のステートメントは、Distlt1 レシピを **recfile.csv** ファイルから削除します。

```
RecipeDelete("c:\recipe\recfile.csv", "Distlt1");
```

ユニットの選択（タグ変数原料マッピング）

RecipeSelectUnit() 関数を使用して、現在のレシピ値がロードされるタグ変数のユニットを選択します。

RecipeSelectUnit() 関数

RecipeSelectUnit() 関数は、ランタイムユーザーがユニットを選択できるように、[ユニットの選択] ダイアログボックスを開きます。選択したユニット名はメッセージ型タグに返されます。

RecipeSelectRecipe() 関数および **RecipeSelectUnit()** 関数はどちらも **RecipeLoad()** 関数と共に使用されます。

カテゴリ

レシピ

構文

```
RecipeSelectUnit("Filename", "UnitName", Number);
```


引数

FileName

レシピテンプレートファイルの名前。**FileName** 引数は、文字列定数またはレシピテンプレートファイルの名前を含むメッセージ型タグとなります。

UnitName

選択したユニットの名前を書き込む先のメッセージ型タグです。引用符または文字列リテラルのない実際のメッセージ型タグ。

Number

引数に返される最大文字列長。InTouch では、文字列（メッセージ型）タグ変数の最大文字数は **131** 文字です。InTouch タグ変数の最大文字列長を削減していない限り、この引数には **131** を使用します。数値または整数型タグ変数。

例

以下のステートメントを実行すると、[ユニットの選択] ダイアログ ボックスが表示されます。

```
RecipeSelectUnit("c:\recipe\recfile.csv", UnitName, 131);
```

ユニットが選択された後に、その名前が **UnitName** タグ変数に返されます。

レシピファイルからの個別レシピの選択

レシピマネージャには、個別のレシピをレシピファイルから選択するための関数セットが含まれています。これらの関数をスクリプトで使用する場合、ファイルから名前によって、またはファイル内のシーケンス内の次のレシピまたは前のレシピによって、レシピを選択できます。

RecipeSelectRecipe() 関数

RecipeSelectRecipe() 関数は、ランタイム ユーザーがレシピを選択できるように、[レシピの選択] ダイアログ ボックスを開きます。選択したレシピ名はメッセージ型タグに返されます。

カテゴリ

レシピ

構文

```
RecipeSelectRecipe("Filename", "RecipeName", Number);
```

引数

FileName

レシピテンプレートファイルの名前。**FileName** 引数は、文字列定数またはレシピテンプレートファイルの名前を含むメッセージ型タグとなります。

RecipeName

選択したレシピの名前を書き込む先のメッセージ型タグです。引用符なしまたは文字列リテラルのない実際のメッセージ型タグ。

Number

引数に返される最大文字列長。InTouch メッセージ型タグの最大文字数は **131** 文字です。InTouch タグ変数の最大文字列長を削減していない限り、このパラメータには **131** を使用します。数値または整数型タグ変数。

例

以下のステートメントを実行すると、[レシピの選択] ダイアログ ボックスが表示されます。

```
RecipeSelectRecipe("c:\recipe\recfile.csv", RecipeName, 131);
```


ダイアログ ボックスからレシピが選択すると、名前が **RecipeName** タグ変数に返されます。

RecipeSelectNextRecipe() 関数

RecipeSelectNextRecipe() 関数は、レシピ テンプレート ファイルの次のレシピを選択します。

カテゴリ

レシピ

構文

```
RecipeSelectNextRecipe("Filename", "RecipeName", Number);
```

引数

FileName

レシピ テンプレート ファイルの名前。FileName 引数は、文字列定数またはレシピ テンプレート ファイルの名前を含むメッセージ型タグとなります。

RecipeName

開始点として使用するレシピ名（関数の実行前）、および選択したレシピ名（関数の実行後）を格納するメッセージ型タグです。引用符なしまたは文字列リテラルのない実際のメッセージ型タグ。

Number

引数に返される最大文字列長。InTouch では、文字列（メッセージ型）タグ変数の最大文字数は **131** 文字です。InTouch タグ変数の最大文字列長を削減していない限り、このパラメータには **131** を使用します。数値または整数型タグ変数。

例

次のステートメントは、タグ変数 **RecipeName** の現在の値を読み取って、ファイルの次のレシピを返します。**RecipeName** の値が空白であるか、見つからない場合は、ファイルの最初のレシピが返されます。**RecipeName** にファイルの最後のレシピ名が含まれている場合は、変更されないまま返されます。レシピは作成された順番でレシピ テンプレート ファイルに保存されます。

```
RecipeSelectNextRecipe("c:\recipe\recfile.csv",  
RecipeName, 131);
```

RecipeSelectPreviousRecipe() 関数

RecipeSelectPreviousRecipe() 関数は、レシピ テンプレート ファイルに定義されている前のレシピを選択します。

カテゴリ

レシピ

構文

```
RecipeSelectPreviousRecipe("Filename", "RecipeName", Number);
```

引数

FileName

レシピ テンプレート ファイルの名前。FileName 引数は、文字列定数またはレシピ テンプレート ファイルの名前を含むメッセージ型タグとなります。

RecipeName

開始点として使用するレシピ名（関数の実行前）、および選択したレシピ名（関数の実行後）を格納するメッセージ型タグです。引用符なしまたは文字列リテラルのない実際のメッセージ型タグ。

Number

パラメータに返される最大文字列長。InTouch では、メッセージ型タグの最大文字数は 131 文字です。InTouch タグの最大文字列長を削減していない限り、このパラメータには 131 を使用します。数値または整数型タグ。

例

以下のステートメントを実行すると、システムがタグ **RecipeName** の現在の値を読み取り、ファイルの前のレシピ名を返します。返されたこの文字列は、**RecipeName** に保存され、現在の値を上書きします。**RecipeName** の値が空白か、または見つからない場合は、ファイルの最後のレシピが返されます。**RecipeName** に現在ファイルの最初のレシピ名が含まれている場合は、変更されないまま返されます。レシピは作成された順番で保存されます。

```
RecipeSelectPreviousRecipe("c:\recipe\recfile.csv", RecipeName, 131);
```

レシピ スクリプト関数によって返されるエラー メッセージの理解

レシピ アプリケーションは、レシピ関数によって返される診断エラー コードを使用してトラブルシューティングします。このセクションでは、レシピ関数エラー コードのリストを記載し、**RecipeGetMessage()** 関数を使用してエラー コードと関連付けられているメッセージを表示する方法について説明しています。

RecipeLoad() 関数は、正常に行われた場合はアナログ型 **ErrorCode** タグ変数の値を 0 に設定します。**RecipeLoad()** が失敗した場合は、**ErrorCode** タグ変数に特定のエラー状態の番号を設定します。

レシピ関数のエラー コードを取得するには、InTouch アナログ型タグ変数と同等にする必要があります。次の例は、レシピ関数エラー コードを返すためのスクリプト ステートメントを示しています。

```
ErrorCode = RecipeLoad(FileName, UnitName, RecipeName);
```

エラー コード メッセージの表示

各レシピ関数は、関数のエラー状態を表す番号を返します。**RecipeGetMessage()** 関数を InTouch データ変換スクリプトで使用するによって、対応するエラー コードがアナログ型タグに書き込まれ、関連付けられているエラー コードメッセージがメッセージ型タグに書き込まれます。

次のコード例は、データ変換スクリプトを示しています。

```
RecipeGetMessage(ErrorCode, ErrorMessage, 131);
```

このスクリプトは、**ErrorCode** タグの値が変更されるたびに自動的に実行されます。このスクリプトが実行されると、**RecipeGetMessage()** 関数が **ErrorCode** タグの現在の数値を読み取り、その値に関連付けられているメッセージを **ErrorMessage** タグに返します。

以下の表には、発生する可能性のあるエラー コードおよびそれに対応するエラー メッセージとその説明が一覧表示されています。

値	エラー メッセージ	説明
0	成功	呼び出されたレシピ関数は正常に実行されました。
-1	そのようなレシピ テンプレートはありません	指定したレシピ テンプレート ファイルが存在しません。
-2	WindowViewer が起動されていません	別のプログラムによって呼び出されたレシピ関数は、WindowViewer が実行されていないため、実行できません。

値	エラー メッセージ	説明
-3	メモリ不足	現在のアクティビティを完了するために十分なメモリがありません。
-4	レシピ内の行が長すぎます	レシピ テンプレート ファイルの行が最大許容長を超えています。
-5	レシピ ファイル内に切り取られた行があります	レシピ テンプレート ファイル内の行が切り取られています。
-6	このレシピ テンプレート ファイルは無効です	指定したファイルが有効なレシピ テンプレート ファイルではありません。
-7	ユニット名またはレシピ名を入力してください	レシピ テンプレート ファイルからユニット名またはレシピ名が見つかりません。
-8	レシピ テンプレート ファイルでユニットが定義されていません	レシピ ファイルのユニット定義テンプレートにユニットが定義されていません。
-9	レシピ テンプレート ファイル内にレシピ名が見つかりません	指定したレシピがレシピ テンプレート ファイルで定義されていません。
-10	レシピ テンプレート ファイル内にユニット名が見つかりません	指定したユニット名がユニット定義テンプレート ファイルで定義されていません。
-12	アナログ／論理値／メッセージの型を入力してください	レシピ テンプレート ファイルのアイテムに間違ったタイプが入力されました。有効なタイプは、アナログ型、論理型、またはメッセージ型です。
-13	タグのタイプがアナログ／論理値／メッセージと照合しません	指定したタグがアイテム タイプと一致しません。たとえば、レシピアイテムはアナログ型として定義されていて、メッセージ型タグがそのユニットとして定義されています。
-14	不正な論理値です。0 または 1 を入力してください	レシピ テンプレート ファイルの論理型タグに対して間違った値が入力されました。論理型タグの有効な値は 0 または 1 のみです。

値	エラー メッセージ	説明
-15	一時ファイルを開くことができません	空きディスク容量が不十分であるために、一時ファイルを開くことができません。
-16	レシピ テンプレート ファイルを保存中に書き込みエラーが発生しました	レシピ テンプレート ファイルを保存中にエラーが発生しました。
-17	ユーザーを選択できませんでした	ユーザーが [レシピの選択] ダイアログ ボックスでレシピ名でなく、[キャンセル] を選択しました。
-19	レシピ テンプレート は別のアプリケーションで使われています	指定されたレシピ テンプレート ファイルは開いているため、 WindowViewer ではアクセスできません。

RecipeGetMessage() 関数

RecipeGetMessage() 関数は（他のレシピ関数によって返された）エラー番号を取り、そのエラー番号のプレーンテキストのエラー メッセージを返します。

カテゴリ

レシピ

構文

```
RecipeGetMessage(Analog_Tag, Message_Tag, Number);
```

引数

Analog_Tag

エラー メッセージを取得するエラー番号です。

Message_Tag

引用符または文字列リテラルのない実際のメッセージ型タグ。

Number

Number 引数は、**Message_Tag** 引数で返される文字列の最大長さを設定します。デフォルトでは、InTouch メッセージ型タグの最大文字数は **131** 文字に設定されます。InTouch タグ変数ディクショナリの **Message_Tag** の最大文字列長を削減していない限り、このパラメータには **131** を使用します。**Number** 引数は、定数または数字を含む整数型タグ変数です。

例

RecipeGetMessage() 関数を InTouch データ変化 QuickScript で使用することによって、エラー コードが **ErrorCode** タグに書き込まれ、関連付けられているエラー コードメッセージが **ErrorMessage** タグに書き込まれます。

```
Data Change Script Tagname[.field]:ErrorCode
Script body:RecipeGetMessage(ErrorCode, ErrorMessage,131);
```

この QuickScript は、ErrorCode タグ変数の値が変更されるたびに自動的に実行されます。この QuickScript が実行されると、RecipeGetMessage() 関数が **ErrorCode** タグ変数の現在の数値を読み取り、その値に関連付けられているメッセージを **ErrorMessage** タグ変数に返します。

```
ErrorCode = RecipeLoad ("c:\App\recipe.csv","Unit1","cookies");
RecipeGetMessage(ErrorCode, ErrorMessageTag, 131);
```

レシピへのセキュリティの適用

レシピへのアクセスは、レシピをロード、保存、または削除するために必要な最小セキュリティ アクセス レベルを設定するレシピ テンプレート ファイルにアイテム名を定義することによって管理できます。

次のファイル サンプルでは、SecurityLevel アイテム名がアナログ型タグ変数として定義されています。Review ユニットの、このアイテムの SecurityLevel アナログ型タグ変数が含まれます。各レシピは、レシピが Review ユニットにロードされるときに SecurityLevel タグ変数にロードされる値を定義します。

MACHINE.CSV							
	1	2	3	4	5	6	7
1	Item Name	Item Type	Unit	Unit	Recipe	Recipe	Recipe
2	Names		Review	PLC1	Setup1A	Setup2A	Setup3A
3	PARM1	Analog		PARM1	11	21	31
4	PARM2	Analog		PARM2	12	22	99
5	PARM3	Analog		PARM3	13	23	66
6	PARM4	Analog		PARM4	14	24	34
7	PARM5	Analog		PARM5	11	21	31
8	PARM6	Analog		PARM6	12	22	32
9	PARM7	Analog		PARM7	13	23	33
10	PARM8	Analog		PARM8	14	24	34
11	PARM9	Analog		PARM9	15	25	35
12	SecurityLevel	Analog	SecurityLevel		2000	5000	7000

選択したレシピに対してセキュリティ アクセス レベルが無効であるときに表示されるアクセス拒否メッセージを含むウィンドウを作成できます。この操作を実行するには、選択したレシピのセキュリティ レベル値が検証用にロードされるアナログ型タグ変数のみを含むユニットに、選択したレシピがロードされる必要があります。

次に例を示します。

```
RecipeSelectRecipe("c:\recipe\machine.csv",MyRecipe, "131");
```

[レシピの選択] ダイアログ ボックスが表示されます。レシピ名を選択すると、RecipeName タグ変数に返され、スクリプトが実行を続行します。

```
RecipeLoad( "c:\Recipe\Machine.csv", "Review", MyRecipe );
IF SecurityLevel <= $AccessLevel THEN
    Status =RecipeLoad( "c:\Recipe\Machine.csv", "PLC1", MyRecipe );
ELSE
    Show "アクセスが拒否されました";
ENDIF;
```

このスクリプトが実行されると、アクセス レベルが 7000 に等しいかそれより大きい場合、選択したレシピの値は PLC1 ユニットのタグ変数にロードされます。それ以外の場合は、[アクセスが拒否されました] ウィンドウが表示され、レシピは PLC1 にロードされません。

InTouch からの SQL データベースの操作

データベースは、共通の属性またはフィールドを共有するテーブルに情報を保存します。SQL (Structured Query Language) は、クエリ形式の情報にアクセスするために使用する言語です。SQL アク

セス マネージャを使用すると、クエリを含むデータベース テーブルへのアクセス、変更、作成、および削除を行うことができます。

SQL アクセス マネージャは、InTouch と共にインストールできるオプションのプログラムです。SQL アクセス マネージャでは以下のことを行うことができます。

- 複雑なクエリを作成して、実行するこれらのクエリは、動的に構築したり、外部ファイルに保存したりすることができます。また、これらのクエリにはランタイムでクエリに渡されるパラメータを含めることができます。
- データベースでサポートされている SQL ステートメントを実行し、クエリから結果を取得する。SQL アクセス マネージャでストアドプロシージャを使用することもできますが、すべてのストアドプロシージャ関数が完全にサポートされているわけではありません。

SQL 関数は、QuickScript エディタ ダイアログ ボックスの [アドオン] ボタンをクリックすることによって、自動的に InTouch QuickScript に挿入できます。SQL 関数はスクリプトの現在のカーソル位置に自動的に挿入されます。

SQL アクセス マネージャを使用して、バッチ レシピなどのデータを SQL データベースから InTouch アプリケーションへ転送できます。SQL アクセス マネージャは、ランタイム データ、アラーム ステータス、または履歴データを InTouch からデータベースに転送するために使用することもできます。たとえば、マシンサイクルが完了した後で、会社は異なるアプリケーション用に複数のデータセットを保存しておきたい場合があります。SQL データベースは、情報を 1 つまたは複数のサードパーティ アプリケーション間で簡単に転送できるようにする機能を提供しています。SQL アクセス マネージャは、このデータにアクセスして、任意の InTouch アプリケーションで表示できるようにします。

SQL アクセス マネージャは、プログラムおよび一組の SQL 関数で構成されます。SQL アクセス マネージャ プログラムは、データベース カラムを作成して、InTouch タグ変数に関連付けます。データベース カラムと InTouch データベース タグ変数の関連付けプロセスは、バインディングと呼ばれます。InTouch データベース タグ変数をデータベース カラムにバインドすると、SQL アクセス マネージャがデータベースに保存されている InTouch データを直接操作できます。

SQL アクセス マネージャは、データベース フィールド名とその関連付けを SQL.DEF と呼ばれるカンマ区切り変数ファイルに保存します。このファイルは InTouch アプリケーション フォルダに常駐し、SQL アクセス マネージャまたはメモ帳などの任意のテキスト エディタを使用して、表示および修正できます。SQL アクセス マネージャは、InTouch で使用されるデータベースの構造および形式を定義するテーブル テンプレートも作成します。

SQL 関数はスクリプトで使用して、オペレータの入力、タグ変数値の変更に基づいて、または特定の条件セットが存在する場合に自動的に実行できます。これらの関数を使用すると、アクセスするために選択したテーブルのレコードを選択、変更、挿入、または削除できます。たとえば、アラーム条件が存在する場合、アプリケーションはすべての適用可能なデータ ポイントとアラームの状態を保存する SQLInsert() 関数または SQLUpdate() 関数を含むスクリプトを実行できます。

データ ソースの設定

SQL アクセス マネージャは、ODBC 互換アプリケーションであり、利用できる ODBC ドライバまたは OLE DB プロバイダをサポートする任意のデータベースと通信します。

データベースへの通信文字列は以下の方法で設定できます。

- Microsoft ODBC アドミニストレータ プログラムを使用して、ODBC ドライバを SQL アクセス マネージャで使用するよう設定します。
- SQLConnect() 関数を実行して、OLE DB プロバイダに引数値を指定します。詳細については、「[SQL Server データベース アプリケーション](#)」を参照してください。

- データベース接続文字列を外部 UDL ファイルに設定します。

ODBC ドライバを設定するには

- Microsoft ODBC アドミニストレータ プログラムを実行します。
- ドライバまたはデータ ソースを選択して、[新しい名前の追加]、[既定値に設定] または [設定] をクリックします。[ODBC ドライバセットアップ] ダイアログ ボックスが表示されます。

オプション	説明
データ ソース名	データ ソースを識別するユーザー定義の名前
説明	データ ソースのユーザー定義の説明
データベース ディレクトリ	データベース ファイルを含むフォルダを識別します。指定されていない場合は、現在の作業フォルダが使用されます。

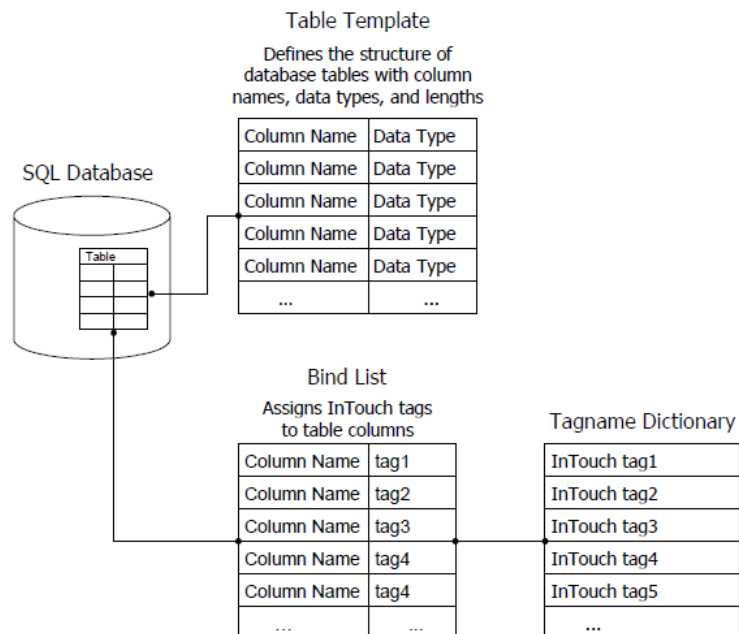
- [OK] をクリックします。

注: ODBC データ ソースを作成すると、ODBC.INI ファイルが Windows ディレクトリに作成されます。ODBC.INI ファイルは手動で編集できます。

データベース カラムへの InTouch タグのマッピング

InTouch タグをデータベース カラムにマップできます。これはバインドリストを使用して実行できます。ほとんどの SQL Access 関数ではバインドリストを使用して、InTouch タグを有効にして SQL データベース テーブルのデータにアクセスできるようにします。

バインドリストはデータベース テーブル カラムを InTouch タグ名ディクショナリのタグに関連付けます。バインドリストには、データベース テーブルの形式を説明するテーブルテンプレートも含まれます。



SQLInsert()、SQLSelect()、または SQLUpdate() 関数を含むスクリプトを実行すると、バインドリストが更新されて、どの InTouch タグが使用されるか、どのテーブル カラムがこれらのタグと関連付けられるかが指定されます。

新しいバインドリストを作成するには

1. [ツール] ペインで **[SQL アクセス マネージャ]** を展開して **[バインドリスト]** をクリックします。
SQL.DEF ファイルの作成を確認するメッセージが表示されます。
2. **[はい]** をクリックして、SQL.DEF ファイルを作成します。
[バインドリストを選択してください] ダイアログ ボックスが表示されます。
3. **[新規]** をクリックします。
[バインドリストの定義] ダイアログ ボックスが表示されます。

4. **[バインドリスト名]** ボックスに、バインドリスト名を入力します。
バインドリスト名は最大 32 文字まで使用できます。
5. バインドリストのタグを定義するには、以下のいずれかを実行します。
 - **[タグ名.フィールド名]** ボックスに、InTouch タグ名を入力します。オプションのタグ ドットフィールドを *tag_name.dotfield_name* の形式で追加できます。
 - **[タグ名]** をダブルクリックして既存のタグを選択します。**[タグの選択]** ダイアログ ボックスが表示されます。リストからタグを選択します。

注記: アプリケーションで使用されていないものの、SQL アクセス バインドリストに指定されている I/O タイプ タグは、WindowViewer が起動するとすぐに有効になります (DAServer に通知)。

6. 以下のいずれかを実行して、タグに追加するドットフィールドを選択します。

- **[タグ名.フィールド名]** ボックスに、タグ名の後にピリオドとドットフィールド名を入力します。
- **[ドット フィールド名]** をクリックします。**[フィールド名の選択]** ダイアログ ボックスが表示されます。タグに追加するドットフィールドをクリックします。

7. **[カラム名]** ボックスに、カラムの名前を入力します。

カラム名の長さは、最大 30 文字までです。カラム名にスペースがある場合は、バインドリスト内およびスクリプトで使用される場合にはカラム名を角括弧で囲みます。次に例を示します。

```
WHERE EXPR= "[Valve ID] = " + text (tagname,"#");
```

8. バインドリストにタグを配置するには、以下のいずれかを実行します。

- 選択したタグをリスト内で1つ上のレベルに移動するには、**[上へ移動]** をクリックします。
- 選択したタグをリスト内で1つ下のレベルに移動するには、**[下へ移動]** をクリックします。

9. **[アイテムの追加]** をクリックして、新しい **[タグ名.フィールド名]** および **[カラム名]** をバインドリストに追加します。

10. **[OK]** をクリックして、新しいバインドリスト設定を保存して、ダイアログ ボックスを閉じます。

バインドリストの **SQL Server 文字列区切り文字の設定**

SQLInsert() 関数および SQLUpdate() 関数は、メッセージ文字列を一重引用符で囲んだデフォルト形式を使用します。ただし、SQL データベースの中には、別のタイプの区切り文字で囲まれたメッセージ文字列を予期しているものがあります。たとえば、Oracle 8.0 では括弧で囲まれた日付文字列を受け取るとを予期しています。この場合は、Delim() 関数を使用する必要があります。

[バインドリストの定義] ダイアログ ボックスの **[カラム名]** フィールドに、カラム名の後で delim() 関数を使用します。キーワード "delim" の後には、以下のように入力する必要があります。

- 左括弧
- 左区切り文字
- システム **[地域と言語のオプション]** で定義されているのリスト区切り文字
- 右区切り文字
- 右括弧

英語システムの例: datestring delim (';')

ドイツ語システムの例: datestring delim (';')

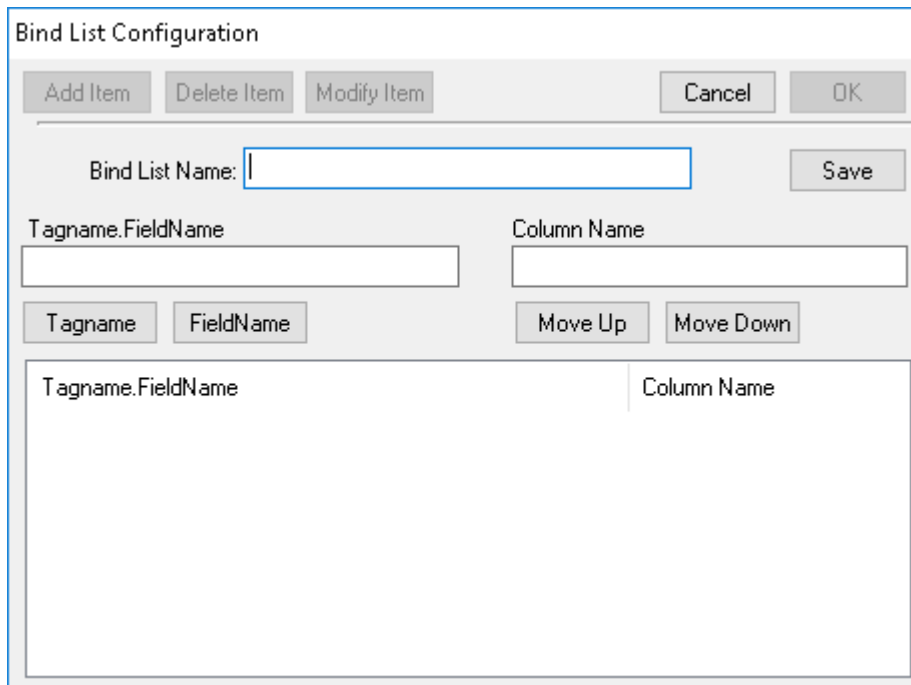
左と右の両方に同じ区切り文字を使用するには、以下の例のように区切り文字を区切り文字なしで括弧内に指定します。

```
datestring delim ('')
```

バインドリストを変更するには

1. **[ツール]** ペインで **[SQL アクセス マネージャ]** を展開して **[バインドリスト]** をクリックします。
SQL.DEF ファイルの作成を確認するメッセージが表示されます。
2. **[はい]** をクリックして、SQL.DEF ファイルを作成します。
[バインドリストを選択してください] ダイアログ ボックスが表示されます。
3. 変更するバインドリスト名を選択して、**[修正]** をクリックします。

「バインドリストの定義」ダイアログボックスが表示されます。



The dialog box is titled "Bind List Configuration". It contains several controls: at the top, three buttons labeled "Add Item", "Delete Item", and "Modify Item" are on the left, and "Cancel" and "OK" are on the right. Below these is a "Bind List Name:" label followed by a text input field and a "Save" button. Underneath, there are two columns of controls. The left column has a "Tagname.FieldName" label above a text input field, and below it are two buttons labeled "Tagname" and "FieldName". The right column has a "Column Name" label above a text input field, and below it are two buttons labeled "Move Up" and "Move Down". At the bottom, there is a large rectangular area with a header row containing "Tagname.FieldName" and "Column Name", followed by several empty rows for data entry.

4. 必要なアイテムを変更します。
5. **[OK]** をクリックして変更を保存し、ダイアログボックスを閉じます。

バインドリストを **Excel** で変更するには

SQL アクセス マネージャは、バインドリストの設定情報およびテーブルテンプレートを **SQL.DEF** ファイルに保存します。このファイルは、カンマ区切り値 (CSV) ファイルの形式となっています。

SQL.DEF ファイルは、Excel などのカンマ区切り値ファイルをサポートする任意の製品を使用して変更できます。

データはファイルで、以下のように表示されます。

```
:BindListName, BindListName
Tagname1.FieldName,ColumnName1
Tagname2.FieldName,ColumnName2
Tagname3.FieldName,ColumnName3
:TableTemplateName, TableTemplateName
ColumnName1,ColumnType,[ColumnLength],Null,Index
ColumnName2,ColumnType,[ColumnLength],Null,Index
ColumnName3,ColumnType,[ColumnLength],Null,Index
```

バインドリストを削除するには

1. **[ツール]** ペインで **[SQL アクセス マネージャ]** を展開して **[バインドリスト]** をクリックします。
SQL.DEF ファイルの作成を確認するメッセージが表示されます。
2. **[はい]** をクリックして、SQL.DEF ファイルを作成します。
[バインドリストを選択してください] ダイアログボックスが表示されます。
3. 削除するバインドリスト名を選択します。

4. [削除] をクリックします。バインドリストの削除を確認するよう促すメッセージが表示されます。
5. [はい] をクリックして、選択したバインドリストを削除します。

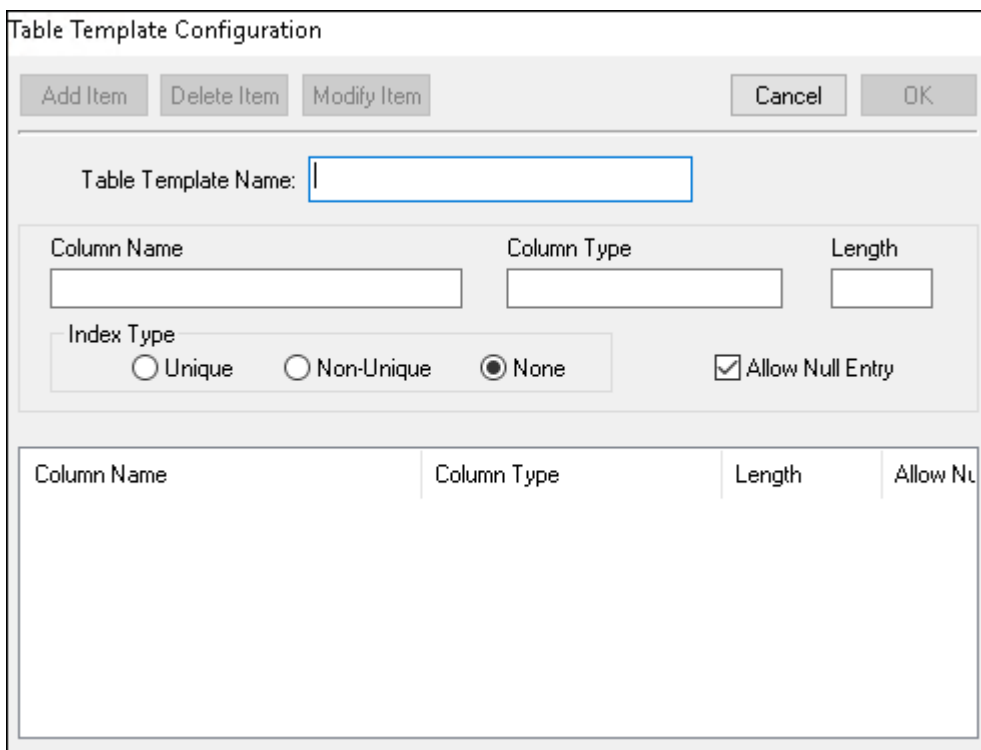
新しいテーブルの構造の定義

テーブルテンプレートは、データベース内に作成する新しいテーブルの構造と形式を定義します。テーブルテンプレートは、SQL.DEF ファイルに保存されます。

新しいテーブルテンプレートを作成するには

1. [ツール] ペインで [SQL アクセス マネージャ] を展開して [テーブルテンプレート] をクリックします。
2. [新規] をクリックします。

[テーブルテンプレートの定義] ダイアログボックスが表示されます。



The dialog box is titled "Table Template Configuration". It contains several controls: "Add Item", "Delete Item", and "Modify Item" buttons on the left; "Cancel" and "OK" buttons on the right. Below these is a "Table Template Name:" label followed by a text input field. Underneath is a section with three input fields: "Column Name", "Column Type", and "Length". Below these is a section for "Index Type" with three radio buttons: "Unique", "Non-Unique", and "None" (which is selected). To the right of the radio buttons is a checkbox labeled "Allow Null Entry" which is checked. At the bottom is a table with four columns: "Column Name", "Column Type", "Length", and "Allow Null Entry".

Column Name	Column Type	Length	Allow Null
-------------	-------------	--------	------------

3. [テーブルテンプレート名] ボックスに、テーブルテンプレートの名前を入力します。
テーブルテンプレート名は、インデックスなしで最大 32 文字まで使用できます。固有であるかどうかに関わらずインデックスを作成している場合、テーブルテンプレート名は 24 文字を超えてはなりません。
4. [カラム名] ボックスに、テーブルテンプレートのカラム名を入力します。
カラム名は最大 30 文字までです。
5. [カラムタイプ] ボックスに、カラムのデータタイプを入力します。データタイプの選択は、使用されているデータベースによって異なります。
6. [インデックスタイプ] 領域で、以下のいずれかのオプションを選択します。
 - 固有: 各カラム値が固有でなければなりません。

- 固有でない: 各カラム値は固有である必要はありません。
- なし: インデックスはありません。

注記: SQLCreateTable() 関数を含むスクリプトを実行する場合、インデックス ファイルが自動的に作成されます。

7. このカラムに Null データを入力できるようにするには、[Null データ入力可] を選択します。

注記: InTouch では、Null データはサポートされていません。

データを挿入するときに、タグに値が入力されていない場合は、タグのタイプによって Null 値が割り当てられます。

データ タイプ	値
論理型	0
整数型	0
メッセージ型	文字のない文字列

8. [アイテムの追加] をクリックして、新しいカラム名、カラム タイプ、長さ、およびインデックス タイプをテーブル テンプレートに追加します。
9. [OK] をクリックして、新しいテーブル テンプレート設定を保存して、ダイアログ ボックスを閉じます。

テーブル テンプレートを変更するには

1. [ツール] ペインで [SQL アクセス マネージャ] を展開して [テーブル テンプレート] をクリックします。
[テーブル テンプレートを選択してください] ダイアログ ボックスが表示されます。
2. 変更するテーブル テンプレート名を選択して、[修正] をクリックします。
[テーブル テンプレートの定義] ダイアログ ボックスが表示されます。
3. 必要なアイテムを変更します。
4. [OK] をクリックして変更を保存し、ダイアログ ボックスを閉じます。

テーブル テンプレートを削除するには

1. [ツール] ペインで [SQL アクセス マネージャ] を展開して [テーブル テンプレート] をクリックします。
[テーブル テンプレートを選択してください] ダイアログ ボックスが表示されます。
2. 削除するテーブル テンプレート名を選択します。
3. [削除] をクリックします。テーブル テンプレートの削除を確認するよう促すメッセージが表示されます。
4. [はい] をクリックします。[テーブル テンプレートの定義] ダイアログ ボックスが再び表示されます。
5. [OK] をクリックして、ダイアログ ボックスを閉じます。

データベース アプリケーションの操作

SQL アクセス マネージャは、Oracle、Microsoft SQL Server、および Microsoft Access データベースをサポートします。各データベースの要件は固有です。このセクションには、各データベースと SQL アクセス マネージャ間の接続を設定する方法を説明する別のセクションが含まれています。

SQL Server データベース アプリケーション

SQLConnect() 関数を InTouch QuickScript で使用して、Microsoft SQL Server データベースに接続します。SQLConnect() 関数は、ユーザーを SQL Server データベースにログオンし、接続を開きます。SQLConnect() 関数によって使用される接続文字列の形式は、以下のとおりです。

```
(SQLConnect(ConnectionId,"<attribute>=<value>;  
<attribute>=<value>;...");
```

ConnectionID 引数は、セッション番号を含む整数型タグ変数です。このセッション番号は、SQL Server データベースへの接続を参照するために、その他のほとんどすべての SQL Access 関数によって使用されます。セッション番号は、それぞれの SQLConnect() 関数呼び出しで 1 つずつ増分します。

次の表は、Microsoft SQL Server によって使用される **SQLConnect()** 関数について説明しています。

属性	値
Provider	SQLOLEDB
Data Source	データベースがインストールされるサーバー名
Initial Catalog	データベース名
User ID	ログオン ID、大文字と小文字の区別あり
Password	パスワード、大文字と小文字の区別あり
"Provider=SQLOLEDB.1;User ID=UserIDStr; Password=PasswordStr;Initial Catalog=DatabaseName;Data Source=ServerName;"	

SQL アクセス マネージャは、4 つのタイプの InTouch タグ変数（論理型、整数型、実数型、メッセージ型）を他の SQL Server データベース データ タイプに関連付けます。

データ タイプ	長さ	範囲	タグ変数タイプ
char 型	8,000 文字	1 ～ 131	メッセージ型
int 型		-2,147,483,648 ～ 2,147,483,647	整数型
浮動小数点 型	15 桁	-1.79E+308 ～ 1.79E+308	実数型

char データ タイプには固定長文字列が含まれています。InTouch メッセージ型タグには char データ タイプが必要です。フィールド長を指定する必要があります。Microsoft SQL Server データベースは、最大 8,000 文字の char フィールドをサポートしています。ただし、InTouch メッセージ型タグは 131 文字に制限されています。メッセージ型タグ値にデータベース フィールドに指定された長さよりも多くの文字が含まれている場合、データベースに挿入されるときに char 文字列は切り捨てられます。

int データ タイプは InTouch 整数型タグ変数を表します。フィールドの長さが指定されていない場合、長さはデータベースのデフォルト値に設定されます。長さが指定されている場合は、Width の形式内にあります。Width は、カラムの最大桁数を決定します。

浮動小数点型のデータ タイプは InTouch 実数型タグ変数を表します。フィールドの長さ設定はデータベースによって固定されます。このデータ タイプのフィールドの長さは必要とされていません。

Microsoft Access データベース アプリケーション

Microsoft Access と通信するには、SQLConnect() 関数を InTouch QuickScript で実行して接続する必要があります。

SQLConnect() 関数は、Microsoft Access データベースに接続するために使用されます。SQLConnect() 関数を含むスクリプトを実行すると、データベース サーバーにログオンし、その他の SQL 関数の実行を許可する接続を開きます。SQLConnect() によって使用される接続文字列の形式は、以下のとおりです。

```
SQLConnect(ConnectionId,"<attribute>=<value>;
<attribute>=<value>;...");
```

DSN は、Microsoft Access によって使用される固有の属性で、Microsoft ODBC Administrator で設定されるデータ ソースの名前を識別します。

```
SQLConnect(ConnectionId,"DSN=MSACC");
```

SQL アクセス マネージャがサポートする有効なデータ タイプは、使用されている ODBC ドライバのバージョンによって異なります。

データ タイプ	長さ	デフォルト	範囲	タグ変数タイプ
テキスト型	255 文字	--	--	メッセージ型
数値型	--	--	--	整数型
数値型	--	--	--	実数

テキストデータ タイプには固定長文字列が含まれ、InTouch メッセージ型タグ変数と共に使用されます。長さを指定する必要があります。Microsoft Access データベースは、最大 255 文字のテキストフィールドをサポートします。InTouch メッセージ型タグ変数は 131 文字に制限されています。メッセージ変数がデータベース フィールドに指定された長さよりも多くの文字を含んでいる場合は、データベースに挿入されたときに文字列は切り捨てられます。Microsoft Access ODBC ドライバは、カラム名に対して最大 17 文字までサポートしています。SQLSetStatement(Select Col1, Col2, ...) を使用しているときにサポートされるカラムの最大数は 40 です。

Oracle データベース アプリケーション

SQL Access および Oracle データベース間に通信を確立するには、SQLConnect() 関数を含むスクリプトを実行することによって接続する必要があります。

Oracle 8.0 データベースと通信するには

1. Oracle OLEDB プロバイダ (MSDAORA.DLL) ファイルが InTouch を実行しているコンピュータにインストールされていることを確認します。このファイルは、InTouch をインストールするときに一緒にインストールされる MDAC によってインストールされます。
2. SQLConnect() 関数を InTouch アクション スクリプトで実行することによって Oracle に接続します。

SQLConnect() 関数によって使用される接続文字列の形式は、以下のとおりです。

```
SQLConnect(ConnectionId,"<attribute>=<value>;  
<attribute>=<value>;...");
```

次の表では、Oracle によって使用される関数属性について説明しています。

属性	値
Provider	MSDAORA
User ID	ユーザー名
Password	パスワード
Data Source	Oracle サーバー マシン名

```
SQLConnect(ConnectionId, "Provider=MSDAORA; Data Source=OracleServer; User ID=SCOTT;  
Password=TIGER;");
```

次の表は、SQL アクセス マネージャが Oracle データベースに対してサポートしている有効なデータ タイプを一覧表示しています。

データ タイプ	長さ	デフォルト	範囲	タグ変数タイプ
char 型	2,000 文字	1 文字		メッセージ型
数値型	38 桁	38 桁		整数型

Oracle 8.0 の日付フィールドに日時をログ記録するには、**delim()** 関数を使用してバインド リストを設定する必要があります。

Oracle 日付フィールドに日付と時間の両方をログ記録するには

1. SQL アクセス マネージャのアプリケーション エクスプローラで、[バインド リスト] をダブルクリックします。[バインド リストの定義] ダイアログ ボックスが表示されます。
2. [タグ変数.フィールド名] ボックスに、使用するタグ変数を入力します。例：DATE_TIME_TAG。
3. [カラム名] ボックスに、Oracle 日付フィールドの名前を入力します。Oracle 8.0 を使用している場合は、**delim()** 関数を使用して区切り文字を指定します。Oracle 9.2 以降を使用している場合は、**delim()** 関数は必要ありません。
4. InTouch アプリケーションで、QuickScript を作成して、現在の日時からの入力データを準備をします。次に例を示します。

```
DATE_TIME_TAG = "TO_DATE(' " + $DateString + " " + StringMid($TimeString,1,8) +  
' ','mm/dd/yy hh24:mi:ss')";
```

QuickScript が WindowViewer で実行された後、以下の形式で日付が表示されます。

```
TO_DATE('08/22/06 23:32:18' ,'mm/dd/yy hh24:mi:ss')
```

InTouch での共通 SQL 操作の実行

InTouch では SQL Access 関数を使用して、データベースに保存されている情報と対話します。これらの SQL Access 関数を使用すると、データベース レコードを選択、変更、挿入、または削除するスクリプトを記述できます。

SQL アクションは同期型です。データベース QuickScript を InTouch アプリケーションから実行すると、関数によって要求されたデータベース アクションが完了するまで、コントロールは InTouch に戻りません。

SQL Access 関数は、関数と関連付けられている引数タイプを説明する句読点基準に準拠しています。引数が引用符に囲まれたスクリプト文字列 ("Arg1") に入力されるとき、正確な文字列が使用されます。引用符が使用されない場合、引数値はタグ名とみなされ、タグの現在の値が引数と関連付けられます。

ほとんどの SQL 関数は結果コードを返します。結果コードがゼロ以外の場合、関数は失敗し、他のアクションを行う必要があります。結果コードは `SQLErrorMsg()` 関数によって使用できます。

InTouch QuickScript エディタを使用して、SQL 関数を QuickScript に挿入します。SQL 関数をスクリプトに挿入する一般手順には以下の手順が含まれます。

SQL 関数をスクリプトに挿入するには

1. InTouch WindowMaker を起動します。
2. QuickScript を QuickScript エディタで開きます。
3. スクリプト内で SQL 関数を挿入する場所にカーソルを置きます。
4. [関数] 領域で、[アドオン] をクリックして、[関数の選択] ダイアログ ボックスを表示します。
5. QuickScript に挿入する SQL 関数をクリックします。スクリプトが更新され、挿入した SQL 関数が表示されます。

SQL Access 関数に関連付けられている引数は、以下のように構成されています。

- *BindList*

SQL.DEF ファイルに定義されているバインドリスト名に対応します。

- *ConnectionID*

ConnectionID 引数は、各データベース接続に **SQLConnect()** 関数によって割り当てられた番号 (ID) を保持するメモリ型整数タグの名前を参照します。

- *ConnectionString*

ConnectionString は、データベース システムと追加のログオン情報を識別します。入力形式は以下のとおりです。

```
"DSN=data source name[;attribute=value  
[;attribute=value]..."
```

Microsoft SQL Server 接続文字列

- SQL Server 用 Microsoft OLE DB プロバイダ (推奨)

```
"Provider=SQLOLEDB.1;User ID=sa; Password=;Initial Catalog=MyDB;Data  
Source=MyServer;"
```

SQL Server 用の OLE DB プロバイダは `sqloledb` です。

- SQL Server 用 Microsoft OLE DB プロバイダ (推奨)

```
"Provider=SQLOLEDB.1;uid=sa;pwd=;Database=MyDB"
```

- ODBC 用 Microsoft OLE DB プロバイダ (SQL Server 用デフォルト プロバイダ MSDASQL 使用) :

```
"DSN=Pubs;UID=sa;PWD=;"
```

- ODBC 用 Microsoft OLE DB プロバイダ (SQL Server 用デフォルト プロバイダ MSDASQL 使用) :

```
"Data Source=Pubs;User ID=sa;Password=;"
```


Oracle 接続文字列

- Oracle 用 Microsoft OLE DB プロバイダ (推奨)

```
"Provider=MSDAORA;Data Source=ServerName;User ID=UserIDStr;  
Password=PasswordStr;"
```

Microsoft Access 接続文字列

Microsoft Jet 用 Microsoft OLE DB プロバイダ (推奨)。Microsoft.Jet.OLEDB.4.0 は、Microsoft Jet 用ネイティブ OLE DB プロバイダ (Microsoft Access Database エンジン) です。

```
"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=d:\DBName.mdb;User  
ID=UserIDStr;Password=PasswordStr;"
```

ODBC 用 Microsoft OLE DB プロバイダ (MS Access 用デフォルト プロバイダ MSDASQL の使用) :

```
"Provider=MSDASQL;DSN=DSNStr;UID=UserName; PWD=PasswordStr;"
```

- *ErrorMsg*

エラーのテキスト説明を含むメッセージ型タグ

- *FileName*

情報を含むファイルの名前

- *MaxLen*

パラメータに関連付けられているカラムの最大サイズ。この引数によって、データが可変長文字かロング可変長文字のどちらの文字タイプであるかが決定されます。**MaxLen** がデータベースによって許可されている最大文字列に等しいかそれより少ない場合、このデータは可変長文字タイプとなります。それより大きい場合、データはロング可変長文字タイプとなります。

- *OrderByExpression*

カラムおよび昇順または降順のソート順序を定義します。ソートに使用できるのはカラム名のみです。式の形式は以下のとおりです。

ColumnName [ASC|DESC]

選択したテーブルをカラム名で昇順にソートするには:

```
"manager ASC"
```

複数カラムでソートするには、式の形式は以下のようになります。

ColumnName [ASC|DESC],

ColumnName [ASC|DESC]

選択したテーブルを 1 つのカラム名 (例: temperature) で昇順に並べ、別のカラム名 (例: time) を降順にソートするには:

```
"temperature ASC, time DESC"
```

- *ParameterNumber*

ステートメント内の実際のパラメータ番号。

- *ParameterType*

指定したパラメータのデータ タイプ。有効な値は以下のとおりです。

タイプ	説明
Char	空白を補った固定長文字列
Var Char	可変長文字列
Decimal	BCD 番号
Integer	4 バイト符号付き整数
Small Integer	2 バイト符号付き整数
Float	4 バイト浮動小数点
Double Precision Float	8 バイト浮動小数点
DateTime	8 バイトの日時値
Date	4 バイトの日時値
Time	4 バイトの日時値
タイプなし	データ タイプなし

- **ParameterValue**

設定する実際の値。

- **Precision**

10 進数値の精度、文字の最大サイズ、または日時値のバイト数。

- **RecordNumber**

取得する実際のレコード番号

- **ResultCode**

ほとんどの SQL 関数が返す整数型変数。ResultCode は、関数が正常に実行されるとゼロ (0)、正常に実行されなかった場合には負の値を返します。

- **Scale**

10 進数値のスケール。この値は、Null に設定するパラメータに対して適用可能な場合にのみ必要です。

- **StatementID**

拡張機能ステートメントを使用している場合、SQL は内部で使用する StatementID を返します。

- **SQLStatement**

実際のステートメント。次に例を示します。

```
ResultCode=SQLSetStatement(ConnectionID, "Select LotNo, LotName from LotInfo");
```

- **TableName**

TableName パラメータには、データベースでアクセスまたは作成するテーブルの名前が含まれます。

- **TemplateName**

TemplateName パラメータは、テーブルを定義する SQL.DEF ファイルのテンプレートの名前です。

- *WhereExpr*

テーブルの行に対する **True** または **False** の条件を定義します。関数はテーブルから **Ture** の条件を持つ行のみを抽出します。式の形式は以下のとおりです。

ColumnName comparison_operator expression

注記: カラムが文字データ タイプの場合、式を一重引用符で囲む必要があります。

次の例は、**Name** カラムに **EmployeeID** という値を含む行をすべて選択します。

```
Name= 'EmployeeID'
```

次の例は、**100 ~ 199** までの部品番号を含むすべての行を選択します。

```
partno>=100 and partno<200
```

次の例は、**temperature** カラムで **350** を超える値があるレコードをすべて選択します。

```
temperature>350
```

データベースの接続と接続解除

SQLConnect() 関数および **SQLDisconnect()** 関数をスクリプトで使用して、**SQL** データベースとの接続または接続解除を行います。

SQLConnect() 関数

SQLConnect() 関数を InTouch QuickScript で使用して、**ConnectionString** 引数で指定されたデータベースに接続できます。

SQLConnect() は、すべての後続 **SQL** 関数でパラメータとして使用される **ConnectionID** 引数に値を返します。**SQLConnect** 関数をスクリプトで使用する前に、アプリケーションでバインドリストを定義する必要があります。

カテゴリ

SQL

構文

```
[ResultCode=]SQLConnect(ConnectionID, "ConnectionString");
```

引数

ConnectionID

各データベース接続に **SQLConnect()** 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

ConnectionString

データベースおよび **SQLConnect()** 関数で使用する追加ログオン情報を識別する文字列

備考

アプリケーション フォルダにバインドリスト (**SQL.DEF** ファイル) が必要です。この関数は、このファイルがないと機能しません。

SQLTrace=1 が **win.ini** ファイルの **[InTouch]** セクションに定義されている場合は、**SQLConnect** が正常に実行されるたびに、**ADO**、プロバイダ、およびデータベースシステムのバージョン情報が **Log Viewer** にログ記録されます。

例

次のステートメントは、**IBM OS/2 Database Manager** と **SAMPLE** という名前のデータベースに接続します。

```
[ResultCode=]SQLConnect(ConnectionID,"DSN=OS2DM;  
DB=SAMPLE");
```

この関数は、すべての後続 SQL 関数でパラメータとして使用される ConnectionID 変数に値が返されます。

```
"DSN=data source name[;attribute=value  
[;attribute=value]..."
```

SQLConnectEx() 関数

SQL 認証で「ユーザー名とパスワード」タイプの資格情報を使用する場合、**SQLConnectEx()** 関数を InTouch QuickScript で使用して、ConnectionString 引数で指定されたデータベースに接続できます。

SQLConnectEx() は、すべての後続 SQL 関数でパラメータとして使用される ConnectionID 引数に値を返します。**SQLConnectEx()** 関数をスクリプトで使用する前に、アプリケーションフォルダでバインドリストを定義する必要があります。

カテゴリ

SQL

構文

```
[ResultCode=]SQLConnectEx(ConnectionID, "ConnectionString", "Credential Name");
```

引数

ConnectionID

各データベース接続に SQLConnectEx() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグの名前

ConnectionString

データベースおよび SQLConnectEx() 関数で使用する追加ログオン情報を識別する文字列

Credential Name

資格情報マネージャに保存された資格情報の名前スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報マネージャから取得されます。

備考

アプリケーションフォルダにバインドリスト (SQL.DEF ファイル) が必要です。この関数は、このファイルがないと機能しません。

SQLTrace=1 が win.ini ファイルの [InTouch] セクションに定義されている場合は、**SQLConnectEx()** が正常に実行されるたびに、ADO、プロバイダ、およびデータベースシステムのバージョン情報が Log Viewer に記録されます。

例

次のステートメントは、IBM OS/2 Database Manager と SAMPLE という名前のデータベースに接続します。

```
[ResultCode=]SQLConnectEx(ConnectionID,"DSN=OS2DM;  
DB=SAMPLE");
```

この関数は、すべての後続 SQL 関数でパラメータとして使用される ConnectionID 変数に値が返されます。

```
"DSN=data source name[;attribute=value  
[;attribute=value]..."
```

SQLDisconnect() 関数

SQLDisconnect() 関数はデータベースとの接続を解除し、**SQLPrepareStatement()** 関数および **SQLInsertPrepare()** 関数用に取得されたすべての未リリース リソースをクリーンアップします。

カテゴリ

SQL

構文

```
[ResultCode=]SQLDisconnect(ConnectionID);
```

引数

ConnectionId

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

参照項目

SQLConnect()

新しいテーブルの作成

SQLCreateTable() 関数を InTouch QuickScript で使用し、指定したテーブル テンプレートからのパラメータを使用してデータベースにテーブルを作成します。

SQLCreateTable() 関数

SQLCreateTable() 関数を InTouch QuickScript で使用し、指定したテーブル テンプレートからのパラメータを使用してデータベースにテーブルを作成します。テーブル テンプレートは、データベース テーブルの構造を含む SQL.DEF ファイルで定義されます。

カテゴリ

SQL

構文

```
[ResultCode=]SQLCreateTable(ConnectionID, TableName, TemplateName);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

TableName

作成するデータベース テーブルの名前

TemplateName

使用するテンプレート定義の名前

例

次の **SQLCreateTable()** 関数の例では、OutputVal テンプレートで定義されたカラム名とデータを使用して、BATCH1 という名前のテーブルを作成します。

```
ResultCode=SQLCreateTable(ConnectionID,"BATCH1",  
"OutputVal");
```

参照項目

SQLConnect()

テーブルの削除

SQLDropTable() 関数を InTouch QuickScript で使用して、データベースからテーブルを削除します。

SQLDropTable() 関数

SQLDropTable() 関数を InTouch QuickScript で使用して、データベースからテーブルを削除します。

SQLDropTable() 関数を含む QuickScript が完了した後は、テーブルは認識されず、どの SQL ステートメントにも応答しません。

カテゴリ

SQL

構文

```
[ResultCode=]SQLDropTable(ConnectionID, TableName);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

TableName

データベースから削除するテーブルの名前

例

次の **SQLDropTable()** 関数の例では、データベースから BATCH1 テーブルを削除します。

```
ResultCode=SQLDropTable(ConnectionID, "BATCH1");
```

参照項目

SQLConnect()

テーブルからのデータの取得

一組の SQL 関数をスクリプトで使用して、データベースからデータを取得し、値を InTouch タグ変数に書き込みます。

- **SQLSelect()** 関数は、テーブルから情報を取得し、この情報をレコードの形式でメモリに作成された一時結果テーブルに配置します。
- **SQLGetRecord()** 関数は、RecordNumber によって指定されたレコードを現在の選択バッファから取得します。
- **SQLNumRows()** 関数は、前の **SQLSelect()** 関数で指定した条件に一致するテーブル行の数を返します。
- **SQLFirst()** 関数は、最後の **SQLSelect()** 関数によって作成された結果テーブルの最初のレコードを取得します。
- **SQLNext()** 関数は、最後の **SQLSelect()** 関数によって作成された結果テーブルの次のレコードを取得します。
- **SQLPrev()** 関数は、論理テーブルの前の行からデータを取得し、その行からの値を InTouch タグ変数にフェッチします。
- **SQLLast()** 関数は、論理テーブルの最後の行を取得し、その行からの値を InTouch タグ変数にフェッチします。

- **SQLEnd()** 関数は、**ConnectionID** と関連付けられている結果テーブルの内容を保存するメモリを解放します。

SQLFirst()、**SQLPrev()**、**SQLNext()**、**SQLLast()**、および **SQLGetRecord()** の関数は、論理テーブルの指定した行からデータを取得し、InTouch タグ変数値として保存します。フィールドが **Null** の場合は、関連付けられている InTouch タグ変数の値が、タグ変数がアナログ型かメッセージ型かによってゼロまたはゼロ長文字列に設定されます。

データベース内の文字列が 131 文字よりも長い場合は、最初の 131 文字のみがデータベースから関連付けられている InTouch メッセージ型タグ変数にコピーされます。

SQLSelect() 関数

SQLSelect() 関数は、テーブルからレコードを取得します。**SQLSelect()** 関数を含むスクリプトが処理されると、取得されたレコードがメモリの一時的結果テーブルに配置されます。これらのレコードは、**SQLFirst()**、**SQLLast()**、**SQLNext()** および **SQLPrev()** の関数を使用してブラウズできます。

重要: **SQLSelect()** 関数を含むスクリプトが完了したときには必ず **SQLEnd()** 関数を呼び出して、結果テーブルで使用されたメモリを解放します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLSelect(ConnectionID,TableName, BindList,WhereExpr,OrderByExpression);
```

引数

ConnectionID

各データベース接続に **SQLConnect()** 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

TableName

アクセスするデータベース テーブルの名前

BindList

使用する InTouch タグ変数とデータベースを定義します。

WhereExpr

テーブルの行に対する **True** または **False** の条件を定義します。**SQLSelect()** 関数は、**WhereExpr** 条件が **Ture** の行のみからデータを抽出します。式の形式は以下のとおりです。

ColumnName comparison_operator expression.

注: 比較が文字列式で行われる場合は、式を一重引用符で囲む必要があります。

次の例は、**name** カラムに **EmployeeID** という値を含む行をすべて選択します。

```
name='EmployeeID'
```

次の例は、100 ~ 199 までの部品番号を含むすべての行を選択します。

```
partno>=100 and partno<200
```

次の例は、**temperature** カラムに 350 を超える値が含まれるすべての行を選択します。

```
temperature>350
```

WhereExpr - Memory message Tag

OrderByExpr - Memory message Tag

Speed_Input - Memory Real - User Input Analog

Serial_Input - Memory Message - User Input String

アナログ型の例

```
WhereExpr = "Speed = " + text
(Speed_Input, "#.##");
```

Speed_Input は数字であるため、WhereExpr 文字列に連結できるようにテキストに変換する必要があります。

文字列の例

```
WhereExpr = "Ser_No = '" +
Serial_input + "'";
```

Serial_Input は文字列であるため、値を WhereExpr = "Ser_No='125gh'" のように一重引用符で囲む必要があります。

Like ステートメントを使用した文字列の例

```
WhereExpr = "Ser_No like '-'"
```

Like 比較演算子を使用するときには、% をワイルドカードとして使用できます。

ブール AND 演算子を使用した文字列およびアナログ型の例

```
WhereExpr = "Ser_No = '" + Serial_input + "'" + " and " + "Speed = " +
text(Speed_Input, "#.##"); OrderByExpr = "";
```

順位が重要でなければ、上記のように null 文字列を使用します。

WhereExpr タグ変数を使用した SQLSelect

```
ResultCode = SQLSelect(Connect_Id, TableName,
BindList,
WhereExpr, OrderByExpr);
Error_msg = SQLErrorMsg( ResultCode );
```

SQLSelect WhereExpr 組み込み関数

```
ResultCode = SQLSelect(Connect_Id, TableName,
BindList,
"Ser_No = '" + Serial_input + "'", OrderByExpr);
Error_msg = SQLErrorMsg( ResultCode );
```

OrderByExpr

テーブルカラム内でデータをソートするための方向を定義します。ソートにはカラム名のみを使用することができ、式は次の形式である必要があります。

ColumnName [ASC|DESC]

次の例は、テーブルを **manager** カラムからのデータによって昇順でソートします。

```
"manager ASC"
```

また、以下のような形式を使用して、複数のカラムでソートすることもできます。

ColumnName [ASC|DESC],

ColumnName [ASC|DESC]

次の例は、選択したテーブルを **temperature** カラムの昇順および **time** カラムの降順でソートします。

```
"temperature ASC,time DESC"
```

例

次のステートメントは、カラム名に **cookie** という値が含まれる **List1** という名前の **BindList** を使用して、**BATCH** テーブルからレコードを選択します。その後、**amount** カラムで昇順に、**sugar** カラムで降順に情報をソートして表示します。

```
ResultCode=SQLSelect(ConnectionID,"BATCH", "List1","type='cookie'", "amount ASC,sugar
DESC");
```

次のステートメントでは、データベースのすべてのデータが選択され、WhereExpr および OrderByExpr の値は指定されません。

```
ResultCode=SQLSelect(ConnectionID,"BATCH", "List1", "", "");
```


参照項目

SQLFirst()、SQLConnect()、SQLLast()、SQLNext()、SQLPrev()、SQLEnd()

SQLGetRecord() 関数

SQLGetRecord() 関数は、RecordNumber によって指定されたレコードを現在の選択バッファから取得します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLGetRecord(ConnectionID, RecordNumber);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

RecordNumber

取得する実際のレコード番号

例

```
ResultCode=SQLGetRecord(ConnectionID,3);
```

参照項目

SQLConnect()

SQLNumRows() 関数

SQLNumRows() 関数は、最後の SQLSelect() 関数で指定した条件に一致する行数を示します。たとえば、WhereExpr 引数を使用して AGE という名前のカラム (AGE は 45 に等しい) を持つ行をすべて選択すると、返される行数は 40 または 4000 となります。これによって、その関数が次回処理されるかが決定されます。

カテゴリ

SQL

構文

```
SQLNumRows(ConnectionID);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

例

次のステートメントは、選択された行数を NumRows 整数型タグ変数に返します。

```
NumRows=SQLNumRows(ConnectionID);
```

参照項目

SQLConnect()

SQLFirst() 関数

SQLFirst() 関数は、直前の SQLSelect() 関数によって作成された結果テーブルの最初のレコードを選択します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLFirst(ConnectionID);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

参照項目

SQLConnect()、SQLSelect()

SQLNext() 関数

SQLNext() 関数は、最後の SQLSelect() 関数によって作成された結果テーブルのシーケンスの次のレコードを選択します。SQLSelect() 関数は、SQLNext() 関数をスクリプトで実行する前に処理する必要があります。

カテゴリ

SQL

構文

```
[ResultCode=]SQLNext(ConnectionID);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

例

```
ResultCode=SQLNext(ConnectionID);
```

参照項目

SQLConnect()、SQLSelect()

SQLPrev() 関数

SQLPrev() 関数は、最後の SQLSelect() 関数によって作成された結果テーブルの前のレコードを選択します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLPrev(ConnectionID);
```

引数

ConnectionID

各データベース接続に `SQLConnect()` 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

備考

このコマンドを使用する前に、`SQLSelect()` 関数を実行する必要があります。

例

```
ResultCode=SQLPrev(ConnectionID);
```

参照項目

`SQLConnect()`、`SQLSelect()`

SQLLast() 関数

`SQLLast()` 関数は、前の `SQLSelect()` 関数によって作成された結果テーブルの最後のレコードを選択します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLLast(ConnectionID);
```

引数

ConnectionID

各データベース接続に `SQLConnect()` 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

例

```
ResultCode=SQLLast(ConnectionID);
```

参照項目

`SQLConnect()`、`SQLSelect()`

SQLEnd() 関数

`SQLEnd()` 関数は、`SQLSelect()` 関数が結果テーブルの内容を保存するために使用しされたメモリを解放した後に実行されます。

カテゴリ

SQL

構文

```
[ResultCode=]SQLEnd(ConnectionID);
```

引数

ConnectionID

各データベース接続に `SQLConnect()` 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

参照項目

`SQLConnect()`、`SQLSelect()`

テーブルへの新しいレコードの書き込み

SQLInsert() 関数を使用して、新しいレコードをデータベースに挿入できます。**SQLInsert()** 関数は、InTouch タグ変数の現在の値を使用して、1 つのレコードをテーブルに挿入します。**SQLInsert()** 関数は、ステートメントを準備、挿入、および終了するワン ステップの操作です。

InTouch メッセージ型タグに関連付けられている文字列が、テーブルの対応するテキスト フィールドの定義されたサイズよりも長い場合は、メッセージ型タグから使用されている文字数がフィールドの定義サイズとなります。

注: InTouch タグ変数に Null は使用できません。バインドリストにフィールドが含まれている場合は、これらの関数を使用したデータベースを更新したり、データベースに Null 値を挿入したりすることはできません。Null 値を許可するように定義されているはずであった、フィールドを含まない INSERT ステートメントに、**SQLExecute** を使用してフィールドに Null 値を挿入できます。

SQL Access は、別々に準備、挿入、およびレコード挿入後にクリーンアップを行う他の 3 つの関数を提供しています。これらの関数を一緒に使用すると、単一の **prepare** および **end** ステートメントを含むスクリプトを記述でき、必要に応じていくつでもレコード挿入ステートメントを追加できます。

SQLInsert() 関数ではなく個別の関数を使用してデータを挿入すると、コンピュータのリソースの使用率を削減できます。

SQLInsert() 関数

SQLInsert() 関数は、指定した BindList のタグ変数の値を使用して、参照されているテーブルに新しいレコードを挿入します。BindList パラメータは、使用される InTouch タグ変数とそれが関連付けられているデータベース カラムを定義します。

ステートメントを準備、挿入、および終了するには、SQLInsert() 関数を使用します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLInsert(ConnectionID, TableName, BindList);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

TableName

アクセスするデータベース テーブル名

BindList

使用される InTouch タグ変数と関連付けられるデータベース カラムを定義します。

例

次のステートメントは、List1 で指定されたタグ変数値を使用して、テーブル ORG に新しいレコードを挿入します。

```
ResultCode=SQLInsert(ConnectionID,"ORG","List1");
```

SQLInsertPrepare() 関数

SQLInsertPrepare() 関数は、関数を実行するたびに Insert ステートメントを作成および準備します。Insert ステートメントは処理されません。StatementID 引数は、ステートメントが処理された後の値を含む整数型タグ変数です。

カテゴリ

SQL

構文

```
[ResultCode=]SQLInsertPrepare (ConnectionID, TableName, BindList, StatementID);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

TableName

アクセスするデータベース テーブルの名前

BindList

使用される InTouch タグ変数と関連付けられるデータベース カラムを定義します。

StatementID

SQLPrepareStatement() 関数が使用されたときに SQL によって返される整数値

参照項目

SQLConnect()、SQLPrepareStatement()

SQLInsertExecute() 関数

SQLInsertExecute() 関数は、SQLInsertPrepare() 関数で指定された、以前に準備された Insert ステートメントを実行します。

SQLInsertExecute() 関数は InTouch タグ変数の現在の値を使用して、1 つの行を前の SQLInsertPrepare() 関数で識別されたテーブルに挿入します。BindList 引数に MS SQL Server テーブル用の Identity キー フィールドが含まれる場合は、SQLInsertExecute() を実行する前に IDENTITY_INSERT オプションを設定する必要があります。

StatementID 引数には、前の SQLInsertPrepare() 関数がスクリプト内で実行されたときに SQL によって返される整数値が含まれます。

カテゴリ

SQL

構文

```
[ResultCode=]SQLInsertExecute(ConnectionID, BindList, StatementID);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

BindList

使用される InTouch タグ変数と関連付けられるデータベース カラムを定義します。

StatementID

SQLPrepareStatement() 関数が使用されたときに SQL によって返される整数値

参照項目

SQLConnect()、SQLPrepareStatement()

SQLInsertEnd() 関数

SQLInsertEnd 関数は、**SQLInsertPrepare** によって作成された **StatementID** 関数に関連付けられているリソースをクリーンアップします。

カテゴリ

SQL

構文

```
[ResultCode=]SQLInsertEnd(ConnectionID, StatementID);
```

引数

ConnectionID

各データベース接続に **SQLConnect()** 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

StatementID

SQLPrepareStatement() 関数が使用されたときに SQL によって返される整数値

例

次の例は、複数の Insert 関数をスクリプトに指定する方法を示しています。

```
ResultCode = SQLSetStatement(ConnectionId, "SET IDENTITY_INSERT Products ON");  
ResultCode = SQLExecute(ConnectionId, "", 0);  
ResultCode = SQLInsertPrepare(ConnectionId, TableName, Bindlist, StatementID);  
ResultCode = SQLInsertExecute(ConnectionId, Bindlist, StatementID);  
ResultCode = SQLInsertEnd(ConnectionId, StatementID);
```

参照項目

SQLConnect()、**SQLPrepareStatement()**

テーブルでの既存のレコードの更新

SQL Access は、InTouch タグ変数からの値を使用してテーブルレコードを更新するための 2 つの関数を提供します。

- **SQLUpdate()**
- **SQLUpdateCurrent()**

SQLUpdate() 関数

SQLUpdate() 関数は、InTouch タグ変数の現在の値を使用して、**WhereExpr** 引数によって設定された条件に一致するテーブルのすべての行を更新します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLUpdate(ConnectionID, TableName, BindList, WhereExpr);
```

引数

ConnectionID

各データベース接続に **SQLConnect()** 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

TableName

アクセスするデータベース テーブルの名前

BindList

使用される InTouch タグ変数と関連付けられるデータベース カラムを定義します。

WhereExpr

テーブルの行に対する True または False の条件を定義します。関数はテーブルから True の条件を持つ行のみを更新します。式の形式は以下のとおりです。

ColumnName comparison_operator expression.

注: カラムが文字データ タイプの場合は、条件式を一重引用符で囲む必要があります。

例

次の例は、name カラムに EmployeeID という値を含む行をすべて選択します。

```
name= 'EmployeeID'
```

次の例は、100 ～ 199 までの部品番号を含むすべての行を選択します。

```
partno>=100 and partno<200
```

次の例は、temperature カラムで 350 を超える値があるすべての行を選択します。

```
temperature>350
```

次のステートメントは、ロット番号が 65 である、BATCH テーブルのすべてのレコードを、BindList "List1" で指定したタグ変数の現在の値に更新します。

```
ResultCode=SQLUpdate(ConnectionID,"BATCH", "List1","lotno=65");
```

注: すべてのレコードが固有であることを確認してください。テーブルに該当するレコードが存在する場合、類似のレコードがすべて更新されます。

参照項目

SQLConnect()

SQLUpdateCurrent() 関数

SQLUpdateCurrent() 関数は、SQLSelect() 関数または SQLExecute() 関数のステートメントで指定されているバインドリストによってテーブルフィールドにマッピングされる InTouch タグ変数を使用して、論理テーブルの現在の行を更新します。現在の行と同一の行が存在する場合、すべての行が更新されます。

最大で 54 の同一レコードを一度に更新できます。SQL Access で更新される同一の行が多数存在する場合、SQLUpdateCurrent() 関数によってエラーが返されます。エラー メッセージは、「Microsoft Cursor Engine:Key column information is insufficient or incorrect.Too many rows were affected by update. (Microsoft カーソルエンジン: キー カラムの情報が不適切か不正です。更新される列が多すぎます。)」という内容となります。

このエラーを回避するには、各行を固有にする固有キー フィールドをテーブルに作成します。SQL Access で使用されるすべてのテーブルが固有のキーを持つことを強くお勧めします。キーがないテーブルの場合は、AutoNumber (Access) 型のフィールドまたは行 Identity (SQL Server) として使用される整数型フィールドをプライマリ キーとして使用して、SQLUpdateCurrent() 関数が一度に 1 つの行のみを更新するようにすることを推奨します。このプライマリ キー フィールドはバインドリストに含まれていない必要はありません。

カテゴリ

SQL

構文

```
[ResultCode=]SQLUpdateCurrent(ConnectionID);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

例

```
ResultCode=SQLUpdateCurrent(ConnectionID);
```

参照項目

SQLConnect()

テーブルからのレコードの削除

データベース テーブルからレコードを削除するには、2 つの SQL 関数を使用できます。

SQL Access は、テーブル レコードを削除するための 2 つの関数を提供します。

- SQLClearTable() はテーブルからレコードを削除します。
- SQLDelete() は、指定した条件に一致するテーブルからレコードを削除します。

SQLClearTable() 関数

SQLClearTable() 関数は、テーブルからすべてのレコードを削除します。テーブルはデータベースから削除されません。

カテゴリ

SQL

構文

```
[ResultCode=]SQLClearTable(ConnectionID, "TableName");
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

TableName

すべてのレコードがクリアされるテーブルの名前

例

次の例では、SQLClearTable() 関数が、BATCH1 テーブルからすべてのレコードをクリアします。

```
ResultCode=SQLClearTable(ConnectionID,"BATCH1");
```

参照項目

SQLConnect()、SQLClearStatement()

SQLDelete() 関数

SQLDelete() 関数は、*WhereExpr* 引数で指定された条件に一致するすべてのレコードをテーブルから削除します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLDelete(ConnectionID, TableName, WhereExpr);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

TableName

WhereExpr 引数で指定された条件に一致するレコードがクリアされるテーブルの名前

WhereExpr

テーブルの行に対する True または False の条件を定義します。SQLDelete() 関数は、WhereExpr 条件が True の行レコードのみを削除します。式の形式は以下のとおりです。

ColumnName comparison_operator expression

注: SQLDelete() 関数に Null の WhereExpr 引数を含めることはできません。

例

次のステートメントでは、ロット番号が 65 に等しい BATCH1 テーブルのすべてのレコードが削除されます。

```
ResultCode=SQLDelete(ConnectionID,"BATCH1", "lotno=65");
```

注: カラムが文字データ タイプの場合は、条件式を "MachineID='AG_LX7_2'" のように一重引用符で囲む必要があります。

参照項目

SQLConnect()

パラメータ化ステートメントの実行

動的クエリを作成するには、SQLSetStatement() 関数および SQLAppendStatement() 関数を使用します。SQLSetStatement() 関数は、新しい SQL ステートメントを開始します。これはストアードプロシージャの名前を含む任意の有効な SQL ステートメントとなります。SQLAppendStatement() 関数は文字列の内容を使用して SQL ステートメントを続行します。

SQLSetStatement() 関数

SQLSetStatement() 関数は、SQLStatement の内容を使用して、確立された接続 ConnectionID で SQL ステートメントバッファを開始します。ConnectionID につき SQL ステートメントバッファは 1 つです。エラーは関数の戻りとして返されます。

カテゴリ

SQL

構文

```
[ResultCode=]SQLSetStatement(ConnectionID, SQLStatement);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

SQLStatement

実際の SQL ステートメントについては、以下の例を参照してください。

例

```
ResultCode=SQLSetStatement(ConnectionID,"Select LotNo, LotName from LotInfo");
```

次の例では、StatementID がゼロに設定され、ステートメントを実行する前に、ステートメントが SQLPrepare(Connect_Id, StatementID) を呼び出す必要はありません。StatementID が、この選択を適切に終了する SQLPrepare によって作成されていないため、SQLClearStatement() 関数の代わりに SQLEnd() 関数を使用します。

```
SQLSetStatement( Connect_Id, "Select Speed, Ser_No from tablename where Ser_No =' " +  
Serial_input + "'");  
SQLExecute(Connect_Id,0);
```

次の例では、StatementID が SQLPrepareStatement() 関数によって作成され、SQLExecute() 関数で使用されます。この SELECT ステートメントを終了するには、SQLClearStatement() 関数を使用して、リソースとハンドルを解放します。

```
SQLSetStatement( Connect_Id, "Select Speed, Ser_No from tablename where Ser_No =' " +  
Serial_input + "'");  
SQLPrepareStatement(Connect_Id,StatementID);  
SQLExecute(Connect_Id,StatementID);  
SQLSetStatement( Connect_Id, "Select Speed, Ser_No from tablename where Ser_No =' " +  
Serial_input + "'");  
SQLPrepareStatement(Connect_Id,StatementID);  
SQLExecute(Connect_Id,StatementID);
```

参照項目

SQLConnect()

SQLAppendStatement() 関数

SQLAppendStatement() 関数は、文字列の内容を使用して SQL ステートメントを続行します。関数呼び出し中にエラーが発生したかどうかは戻り値によって示されます。

InTouch タグ変数は最大 131 文字の文字列をサポートできます。一般的に、SQLAppendStatement() 関数を使用して、追加文字列をステートメントに連結します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLAppendStatement(ConnectionID, "SQLStatement");
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

SQLStatement

付加する実際のステートメント

例

```
ResultCode=SQLAppendStatement(ConnectionID, "where tablename.columnname=TR-773-01");
```

参照項目

SQLConnect()、SQLClearStatement()

ステートメントの作成またはファイルからの既存のステートメントのロード

クエリは他のサードパーティ データベース ツールから作成でき、SQL Access を使用してクエリを実行できます。最初に、サードパーティ データベース ツールから作成された .SQL クエリ ファイルから SQL ステートメントをロードする必要があります。

```
ResultCode = SQLLoadStatement (ConnectionID, "c:\myappdir\lotquery.sql");
```

SQLLoadStatement() 関数を使用して SQL クエリをロードします。これでステートメントを実行する準備ができました。

SQLLoadStatement() 関数

SQLLoadStatement() 関数は、ファイルから SQL ステートメントを読み取ります。

ステートメントは 1 つのファイルにつき 1 つです。ただし、SQLAppendStatement() 関数は、SQLPrepareStatement() 関数または SQLExecute() 関数が呼び出されていない場合に、ステートメントに何かを付加するために使用できます。

カテゴリ

SQL

構文

```
[ResultCode=]SQLLoadStatement(ConnectionID, FileName);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

FileName

SQL ステートメントを含むファイルの名前

備考

ステートメントをロードしてステートメント ハンドルを取得した後に、SQLPrepareStatement() 関数を使用してステートメントの実行を準備してください。

例

SQL.txt ファイルには次の SQL ステートメントが含まれます。

```
Select ColumnName from TableName where ColumnName>100;
```

SQLLoadStatement() 関数は、ファイルからステートメントをロードします。

```
ResultCode=SQLLoadStatement(ConnectionID,  
"C:\SQL.txt")
```

参照項目

SQLConnect()、SQLAppendStatement()、SQLExecute()、SQLPrepareStatement

ステートメントの準備

以下の関数を使用すると、必要なパラメータ化ステートメントを作成し、パラメータに 1 つずつ動的に入力できます。たとえば、以下の関数を使用すると、ファイルで汎用ステートメントを使用して、SQLLoadStatement() 関数を使ってそれをロードし、SQLPrepareStatement() 関数を使用してステートメント ID を取得するよう準備してから、ステートメントのパラメータに入力することができます。

- SQLPrepareStatement()

- **SQLSetParamChar()**
- **SQLSetParamDate()**
- **SQLSetParamDateTime()**
- **SQLSetParamDecimal()**
- **SQLSetParamFloat()**
- **SQLSetParamInt()**
- **SQLSetParamLong()**
- **SQLSetParamNull()**
- **SQLSetParamTime()**
- **SQLClearParam()**
- **SQLClearStatement()**

SQL ステートメントでパラメータ代入を実行するには、後続のパラメータを指定する SQL ステートメントに "?" を配置します。ステートメントが準備され、パラメータがステートメントに設定されて、ステートメントが実行されます。

SQLPrepareStatement() 関数

SQLPrepareStatement() 関数は、SQL ステートメントの実行準備を行います。ステートメントが実行されるのではなく、ステートメントをアクティブにして、パラメータ値を設定できるようにします。

カテゴリ

SQL

構文

```
SQLPrepareStatement(ConnectionId, StatementID)
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

備考

デフォルト ステートメントを準備し、StatementID (1、2、3、...) を返します。この準備は、SQLSetParam{Type} 関数を使用して、設定する必要のあるパラメータを持つステートメントに対して役立ちます。

ステートメント パラメータの設定

SQL アクセス マネージャは、SQL ステートメントに含まれているパラメータに割り当てられている値を変更するための一組の関数を提供しています。

SQLSetParamChar() 関数

SQLSetParmChar() 関数は、スクリプトで使用して、指定した文字列に指定したパラメータの値を設定します。この関数は、実行前に複数回呼び出すことができ、その結果パラメータ値は送信されたすべての値の連続として設定できます。長さが 0 (ゼロ) の場合は無視されます。

カテゴリ

SQL

構文

```
SQLSetParamChar(StatementID, ParameterNumber, ParameterValue, MaxLength);
```

引数

StatementID

SQLPrepareStatement() 関数を使用したときに SQL によって返される整数値

ParameterNumber

ステートメントのパラメータ番号

ParameterValue

パラメータ値として設定される値

MaxLength

このパラメータが関連付けられているカラムの最大幅。この設定によって、パラメータが可変長文字であるかロング可変長文字のどちらのタイプであるかが決定されます。**MaxLength** がデータベースによって許可されている最大文字列に等しいまたはそれより少ない場合、このパラメータは可変長文字タイプとなります。大きい場合は、ロング可変長文字タイプとなります。

参照項目

SQLPrepareStatement()

SQLSetParamDate() 関数

SQLSetParamDate() 関数は、パラメータの値を指定した日付に設定します。

カテゴリ

SQL

構文

```
SQLSetParamDate(StatementID, ParameterNumber, "Value");
```

引数

StatementID

クエリー内で SQL ステートメントを識別する整数値

ParameterNumber

StatementID 引数によって識別される SQL ステートメントでパラメータを識別する整数値

Value

二重引用符で囲まれるリテラルとしてパラメータに割り当てられる日付、または値が日付であるタグ変数の名前。日付に割り当てられる時間は 12:00:00 am です。

例

この例では、NewDate タグ変数に関連付けられている日付に 3 番目のステートメントの 2 番目のパラメータを設定します。

```
SQLSetParamDate(3, 2, NewDate);
```

参照項目

SQLPrepareStatement()

SQLSetParamDateTime() 関数

SQLSetParamDateTime() 関数は、パラメータの値を指定した日時に設定します。

カテゴリ

SQL

構文

```
SQLSetParamDateTime(StatementID, ParameterNumber, Value, Precision);
```

引数

StatementID

クエリー内で SQL ステートメントを識別する整数値

ParameterNumber

StatementID 引数によって識別される SQL ステートメントでパラメータを識別する整数値

Value

ParameterNumber 引数によって識別されるパラメータに割り当てられる日付と時間

Precision

パラメータの値として割り当てられる日時値の文字数を指定する整数

参照項目

SQLPrepareStatement()

SQLSetParamDecimal() 関数

SQLSetParamDecimal() 関数は、パラメータの値を 10 進数に設定します。

カテゴリ

SQL

構文

```
SQLSetParamDecimal(StatementID, ParameterNumber, Value, Precision, Scale);
```

引数

StatementID

クエリー内で SQL ステートメントを識別する整数値

ParameterNumber

StatementID 引数によって識別される SQL ステートメントでパラメータを識別する整数値

Value

Value は、文字列、または 10 進数（123.456）を表す InTouch メッセージ型タグ、または InTouch メモリ実数型タグとなります。

パラメータの精度を確実にするために、実数型タグの代わりにメッセージ型タグを使用することを推奨します。ただし、Value は浮動小数点値（たとえば、DAServer から受け取る実数値）の場合、関数は動作を続行します。ただし、浮動小数点表現の制限により、高精度は保証されない可能性があります。

Precision

数値の桁の合計数を指定する整数

Scale

小数点以下の桁数を指定する整数

例

この例では、3 番目の SQL ステートメントの 2 番目のパラメータを 123.456 に設定します。精度は 6 桁で、スケールは小数点以下の 3 桁です。

```
SQLSetParamFloat(3, 2, 123.456, 6, 3);
```

参照項目

SQLPrepareStatement()

SQLSetParamFloat() 関数

SQLSetParamFloat() 関数は、パラメータの値を 64 ビット符号付き浮動小数点値に設定します。

カテゴリ

SQL

構文

```
SQLSetParamFloat(StatementID, ParameterNumber, Value);
```

引数

StatementID

クエリー内で SQL ステートメントを識別する整数値

ParameterNumber

StatementID 引数によって識別される SQL ステートメントでパラメータを識別する整数値

Value

指定したパラメータの値として割り当てる 64 ビット符号付き浮動小数点数

例

この例では、3 番目の SQL ステートメントの 2 番目のパラメータを -5 に設定します。

```
SQLSetParamFloat(3, 2, -5);
```

参照項目

SQLPrepareStatement()

SQLSetParamInt() 関数

SQLSetParamInt() 関数は、パラメータの値を 16 ビット符号付き整数に設定します。

カテゴリ

SQL

構文

```
SQLSetParamInt(StatementID, ParameterNumber, Value);
```

引数

StatementID

クエリー内で SQL ステートメントを識別する整数値

ParameterNumber

StatementID 引数によって識別される SQL ステートメントでパラメータを識別する整数値

Value

指定したパラメータの値として割り当てる 16 ビット符号付き整数

例

この例では、3 番目の SQL ステートメントの 2 番目のパラメータを -5 に設定します。

```
SQLSetParamInt(3, 2, -5);
```

参照項目

SQLPrepareStatement()

SQLSetParamLong() 関数

SQLSetParamLong() 関数は、パラメータの値を 32 ビット符号付きアナログ数に設定します。

カテゴリ

SQL

構文

```
SQLSetParamLong(StatementID, ParameterNumber, Value);
```

引数

StatementID

クエリー内で SQL ステートメントを識別する整数値

ParameterNumber

StatementID 引数によって識別される SQL ステートメントでパラメータを識別する整数値

Value

指定したパラメータの値として割り当てる 32 ビット符号付きアナログ数

例

この例では、最初のステートメントの 3 番目のパラメータを 2.1e9 に設定します。

```
SQLSetParamLong(1, 3, 2.1e9);
```

参照項目

SQLPrepareStatement()

SQLSetParamNull() 関数

SQLSetParamNull() 関数は、SQL ステートメント内の指定したパラメータを Null に設定します。

カテゴリ

SQL

構文

```
SQLSetParamNull(StatementID, ParameterNumber, ParameterType, Precision, Scale)
```

引数

StatementID

クエリー内で SQL ステートメントを識別する整数値

ParameterNumber

StatementID 引数によって識別される SQL ステートメントでパラメータを識別する整数値

ParameterType

ParameterNumber 引数で指定されたパラメータに関連付けられているデータ タイプを指定する整数値

ParameterType 引数は、次の値に関連付けることができます。

0: 文字列型

- 1: 日付/時刻
- 2: 整数型
- 3: 浮動小数点数
- 4: 10 進数

Precision

パラメータ データ タイプに関連付けられているデータの精度

Scale

10 進数値のスケール。この値は、Null に設定するパラメータに対して適用可能な場合にのみ必要です。

備考

Null 値との比較は、SQL Server の ANSI_NULLS オプションによって管理されます。SQL Server 7.0 では、このオプションはクエリー実行時ではなく、オブジェクト作成時に解決されます。ストアードプロシージャが SQL Server 7.0 で作成される場合、このオプションはデフォルトで ON であり、"WHERE MyField = NULL" などの句は常に NULL (FALSE) を返し、この句を使用する SELECT ステートメントから行は返されません。

比較 = または <> が TRUE または FALSE を返すには、ストアードプロシージャを作成するときにオプションを OFF に設定する必要があります。ANSI_NULLS が OFF に設定されていない場合、SQLSetParamNull() は期待どおりに動作しません。この場合、Null 値に対する比較は "WHERE MyField IS NULL" または "WHERE MyField IS NOT NULL" の構文を使用する必要があります。

例

このトランザクションセットは、ProductName が Null ではない Products テーブル内のすべての行を返します。

```
SET ANSI_NULLS OFF
GO
CREATE PROCEDURE sp_TestNotNull @ProductParam varchar(255)
AS SELECT * FROM Products WHERE ProductName <> @ProductParam
GO
SET ANSI_NULLS ON
GO
```

InTouch は、以下の SQL Access スクリプトを実行できます。

```
ResultCode = SQLSetStatement(ConnectionId, "sp_TestNotNull");
ResultCode = SQLPrepareStatement(ConnectionId, StatementID);
ResultCode = SQLSetParamNull(StatementID, 1, 0, 0, 0);
ResultCode = SQLExecute(ConnectionId, BindList, StatementID);
ResultCode = SQLFirst(ConnectionId);
ResultCode = SQLClearStatement(ConnectionId, StatementID);
```

参照項目

SQLPrepareStatement()

SQLSetParamTime() 関数

SQLSetParamTime() 関数は、指定した時間パラメータを指定した文字列に設定します。

カテゴリ

SQL

構文

```
SQLSetParamTime(StatementID, ParameterNumber, Value)
```

引数

StatementID

クエリー内で SQL ステートメントを識別する整数値

ParameterNumber

StatementID 引数によって識別される SQL ステートメントの実際のパラメータ数

Value

設定する実際の値。**ParameterNumber** 引数で指定されたパラメータを時間値に設定します。関数を実行しているコンピュータからの現在の日付は指定した時間に含まれます。

例

この例では、4 番目の SQL ステートメントの 2 番目のパラメータを 10:00AM に設定します。

```
ResultCode=SQLSetParamTime(1, 3, "10:00:00 AM");
```

参照項目

SQLPrepareStatement()

ステートメントパラメータのクリア

SQLClearParam() 関数は、指定したパラメータの値をクリアします。

SQLClearParam() 関数

SQLClearParam() 関数は、指定したパラメータの値をクリアします。**SQLExecute()** 関数を呼び出してクエリを実行する前に、**SQLSetParamxxx()** 関数の 1 つを再び呼び出してパラメータを再ロードする必要があります。

カテゴリ

SQL

構文

```
ResultCode=]SQLClearParam(StatementID,ParameterNumber);
```

引数

StatementID

SQLPrepareStatement() 関数が実行するときに返される整数値

ParameterNumber

ParameterNumber 引数は、変更する SQL ステートメント内の実際の引数を識別します。**StatementID** に関連付けられている **ParameterNumber** の値を、引数が数値であるか文字列であるかによって、ゼロまたはゼロ長の文字列に設定します。

参照項目

SQLPrepareStatement()、SQLExecute()

ステートメントの実行

SQLExecute() 関数は、InTouch スクリプト内で使用されると、ランタイム時に SQL クエリを実行できます。

SQLExecute() 関数

SQLExecute 関数は、スクリプト内で SQL クエリを実行します。ステートメントに **SELECT** が含まれている場合、**BindList** 引数は、データベース カラムを InTouch タグにバインドするために使用するバインドリストの名前を指定します。バインドリストが **Null** の場合、タグの関連付けは行われません。

カテゴリ

SQL

構文

```
SQLExecute(ConnectionID, BindList, StatementID);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグの名前

BindList

BindList 引数は、ゼロ長文字列にすることができます。StatementID が、行を返すクエリーと関連付けられている場合、論理テーブルは SQLExecute() の結果によって更新されます。実際のバインドリストが指定されている場合、結果は BindList 引数と関連付けられます。ゼロ長バインドリストは、StatementID が行を返すクエリーと関連付けられていないことが事前に分かっている場合に役立ちます。

StatementID

SQLPrepareStatement() 関数が使用されたときに SQL によって返される整数値

備考

エラーは関数の戻りとして返されます。ステートメントが準備されている場合、その状態から返されたステートメントハンドルを渡す必要があります。ステートメントが準備されていない場合、ステートメントハンドルはゼロである必要があります。

注記: 準備されていないステートメントに対する SQLExecute() 関数は、一度のみ呼び出すことができます。ステートメントが準備されていれば、何度でも呼び出すことができます。

デフォルトステートメントは接続 ID と関連付けられています。テキスト SQL ステートメント (SELECT、INSERT、DELETE、または UPDATE)、MS Access のクエリーの名前 (パラメータあり/なし)、または MS SQL Server のストアドプロシージャの名前 (パラメータあり/なし) のいずれかになります。

デフォルトステートメントは、SQLLoadStatement()、SQLSetStatement()、および SQLAppendStatement() の関数によって変更されます。デフォルトステートメントは、StatementID = 0 が指定されるたびに SQLExecute() によって使用されます。

例

この例では、lotquery.sql ファイルから SQL ステートメントをロードし、SELECT ステートメントの結果をバインドリストで指定された InTouch タグに配置します。

```
ResultCode = SQLLoadStatement (ConnectionID, "c:\myappdir\lotquery.sql");  
ResultCode = SQLExecute(ConnectionID, "BindList", 0);  
ResultCode = SQLNext (ConnectionID);
```

この SQLSetStatement() 関数は、複雑なクエリーおよび 131 文字を超える文字列式に使用する必要があります。文字列式が 131 文字を超える場合は、SQLAppend() 関数を使用します。

```
SQLSetStatement(ConnectionID, "Select Speed, Ser_No from tablename where Ser_No =' " +  
Serial_input + "'");  
SQLExecute(ConnectionID, "BindList", 0);
```

前の例では、StatementID 引数がゼロに設定され、ステートメントを実行する前に、SQLPrepareStatement(Connection_Id, StatementID) を呼び出す必要はありません。

StatementID はこの SELECT を適切に終了するために SQLPrepare によって作成されていないため、**SQLClearStatement() 関数**の代わりに SQLEnd() 関数を使用します。

```
SQLSetStatement(Connection_Id, "Select Speed, Ser_No from tablename where Ser_No =' " +  
Serial_input + "'");  
SQLPrepareStatement(Connection_Id, StatementID);  
SQLExecute(Connection_Id, StatementID);
```

上の例では、StatementID が SQLPrepareStatement 関数呼び出しによって作成され、SQLExecute 関数によって使用されます。この SELECT ステートメントを終了するには、スクリプト内で SQLClearStatement() 関数呼び出しを使用して、リソースと StatementID を解放します。

SQLExecute() 関数は、いくつかのストアードプロシージャをサポートしています。たとえば、次の選択ステートメントを含む "LotInfoProc" という名前のデータベース サーバーでストアードプロシージャを作成すると仮定します。 "Select LotNo, LotName from LotInfo".

使用しているデータベースのタイプに基づいてストアードプロシージャを実行する InTouch QuickScript を記述します。次の例は、SQL Server データベースに対してストアードプロシージャを実行するためのスクリプト ステートメントを示しています。

```
ResultCode = SQLSetStatement (ConnectionID,"LotInfoProc");  
ResultCode = SQLExecute(ConnectionID, "BindList", 0);  
ResultCode = SQLNext (ConnectionID);  
{Get results of Select}
```

次の例は、Oracle データベースに対してストアードプロシージャを実行するためのスクリプト ステートメントを示しています。

```
ResultCode = SQLSetStatement (ConnectionID, "{CALL LotInfoProc}");  
ResultCode = SQLExecute(ConnectionID, "BindList", 0);  
ResultCode = SQLNext (ConnectionID);  
{Get results of Select}
```

参照項目

SQLConnect()、SQLPrepareStatement()

占有されているリソースの解放

SQLClearStatement 関数は、*StatementID* によって指定されているステートメントに関連付けられているデータベース リソースを解放します。

SQLClearStatement() 関数

SQLClearStatement() 関数は、*StatementID* 引数で指定されたステートメントに関連付けられているデータベース リソースを解放します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLClearStatement(ConnectionID, StatementID);
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

StatementID

SQLPrepareStatement() 関数が使用されたときに SQL によって返される整数値

参照項目

SQLConnect()、SQLPrepareStatement()

トランザクションセットの操作

SQL Access には、データベースのレコードを変更、挿入、更新、または削除する一組のトランザクション関数が含まれています。一般に、これらのトランザクションはスクリプト内にトランザクションセットの形式でグループ化されています。トランザクションセットは一度に実行されます。

SQLTransact() 関数

SQLTransact() 関数は、トランザクションセットと呼ばれる SQL ステートメント グループの開始を定義します。トランザクションセットは、単一のトランザクションとして処理されます。**SQLTransact()** 関数が実行されると、後続のすべての操作は **SQLCommit()** 関数が正常に実行されるまではデータベースに対して実行されません。

カテゴリ

SQL

構文

```
[ResultCode=]SQLTransact(ConnectionID)
```

引数

ConnectionID

各データベース接続に SQLConnect() 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

例

この例では、トランザクションセットに 3 つの Insert ステートメントが含まれています。

```
ResultCode = SQLTransact(ConnectionID);  
ResultCode = SQLInsertPrepare(ConnectionID, TableName, BindList, StatementID);  
ResultCode = SQLInsertExecute(ConnectionID, BindList, StatementID);  
ResultCode = SQLInsertExecute(ConnectionID, BindList, StatementID);  
ResultCode = SQLInsertExecute(ConnectionID, BindList, StatementID);  
ResultCode = SQLInsertEnd(ConnectionID, StatementID);  
ResultCode = SQLCommit(ConnectionID);
```

参照項目

SQLCommit()、SQLRollback()

SQLCommit() 関数

SQLCommit() 関数は、トランザクションセットの終了を定義します。**SQLTransact()** 関数が実行されると、トランザクションセット内のすべての SQL ステートメントは **SQLCommit()** 関数が正常に実行されるまではデータベースに対して実行されません。

注: SQLCommit() 関数を含む QuickScript を記述するときは、慎重に行ってください。トランザクションセット内の SQL ステートメントの数によって処理時間は増加します。

カテゴリ

SQL

構文

```
[ResultCode=]SQLCommit(ConnectionID)
```

引数

ConnectionID

各データベース接続に **SQLConnect()** 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

例

このサンプル スクリプトには、3 つのデータベース挿入を行うトランザクションセットが含まれています。

```
ResultCode = SQLTransact(ConnectionID);  
ResultCode = SQLInsertPrepare(ConnectionID, TableName, BindList, StatementID);  
ResultCode = SQLInsertExecute(ConnectionID, BindList, StatementID);  
ResultCode = SQLInsertExecute(ConnectionID, BindList, StatementID);  
ResultCode = SQLInsertEnd(ConnectionID, StatementID);  
ResultCode = SQLCommit(ConnectionID);
```

参照項目

SQLRollback()、**SQLTransact()**、**SQLCommit()**

SQLRollback() 関数

SQLRollback() 関数は、最後のトランザクションセットを反転またはロールバックします。トランザクションセットは、**SQLTransact()** 関数と **SQLCommit()** 関数の間で発行されるコマンドグループです。

トランザクションセットは、単一のトランザクションとして処理されます。**SQLTransact()** 関数が実行されると、後続のすべての操作はデータベースに対して実行されません。**SQLCommit()** 関数が実行された後で、データベースに対するクエリー変更が行われます。**SQLRollback()** 関数は、**SQLCommit()** 関数の前に実行される場合にトランザクションセットをロールバックします。

カテゴリ

SQL

構文

```
[ResultCode=]SQLRollback(ConnectionID)
```

引数

ConnectionID

各データベース接続に **SQLConnect()** 関数によって割り当てられた番号 (ID) を保持するメモリ整数型タグ変数の名前

例

この例は、スクリプト内の **SQLTransact** 関数の前にデータベース値をロールバックします。

```
ResultCode =SQLTransact(ConnectionID);  
ResultCode = SQLInsertPrepare(ConnectionID, TableName, BindList, StatementID);  
ResultCode = SQLInsertExecute(ConnectionID, BindList, StatementID);  
ResultCode = SQLInsertEnd(ConnectionID, StatementID);  
ResultCode =SQLRollback(ConnectionID);
```

参照項目

SQLCommit()、**SQLTransact()**

ランタイムで [ODBC データソース アドミニストレータ] ダイアログ ボックスを開く

SQLManageDSN() 関数を使用して、InTouch アプリケーションの実行中に Microsoft ODBC Manager を実行します。

SQLManageDSN() 関数

SQLManageDSN 関数は、InTouch アプリケーションの実行中に Microsoft ODBC Manager セットアップ プログラムを実行します。**SQLManageDSN()** は、スクリプト内で使用して、SQL Server または Access データベースのデータ ソース名を追加、削除、および変更できます。

カテゴリ

SQL

構文

```
SQLManageDSN(ConnectionId)
```

引数

ConnectionId

ConnectionId は使用されません。これは、古いバージョンの SQL Access と下位互換性を保持しています。したがって、任意の数字を関数に渡すことができます。**Microsoft ODBC Manager** を開くために関数を実行する前に、データベース接続を確立する必要はありません。

例

```
SQLManageDSN(0);
```

SQL エラー メッセージの理解

このセクションでは、**SQL Access** 関数を使用する InTouch アプリケーションのトラブルシューティング方法について説明しています。最初のセクションでは、**SQLErrorMsg()** 関数について説明し、SQL 結果コードのテーブルおよびそれに対応するエラー メッセージが記載されています。2 番目のセクションは、特定のデータベース エラー メッセージを含むテーブルが含まれています。

SQLErrorMsg() 関数

すべての SQL 関数は、トラブルシューティングに使用できる結果コードを返します。**SQLErrorMsg()** 関数は、結果コードに関連付けられているエラー メッセージを返し、それを InTouch メッセージ型タグの値として割り当てます。

カテゴリ

SQL

構文

```
ErrorMsg = SQLErrorMsg(ResultCode);
```

引数

ResultCode

前の SQL 関数によって返される整数値。**SQLErrorMsg()** 関数は、InTouch メッセージ型タグの値を結果コードに関連付けられているメッセージに設定します。結果コードに関連付けられているエラー メッセージについては、[「SQL エラー メッセージの理解」](#)を参照してください。

備考

ドキュメントに記載されていない結果コードについては、データベースのマニュアルを参照してください。また、その他のエラー メッセージについては、**Log Viewer** を参照してください。

win.ini ファイルの [InTouch] セクションの下に定義されている SQLTrace=1 フラグは、SQL Access スクリプトのデバッグに役立ちます。

例

この例では、SQL アクセス マネージャの結果コードに関連付けられているエラー メッセージを ErrorMessage 型タグに割り当てます。

```
ErrorMessage = SQLErrorMsg(ResulItCode)
```

参照項目

SQLConnect()

SQL アクセス マネージャの結果コードとメッセージ

以下の表には、いくつかの一般的な SQL Access 結果コードと、それに対応するエラー メッセージおよび説明が一覧表示されています。

結果コード	エラー メッセージ	説明
0	エラーは発生していません	SQL 関数がエラーを生じることなく正常に実行されました。
-1	<データベース プロバイダからのメッセージ>	ベンダー データベースからの特定のエラー メッセージ
-2	空白のステートメントは実行できません	SQLExecute(ConnectionId, BindList, 0) が、事前に空白でないステートメントを持つ SQLSetStatement または SQLLoadStatement を呼び出さことなく実行されます。
-4	返される値は Null です	データベースから読み取られる整数値または実数値が Null です。これは単に Log Viewer に送信される警告メッセージです。
-5	フェッチできる行がありません	テーブルの最後のレコードに到達しました。
-7	無効なパラメータ ID です	SQLSetParamChar()、SQLSetParamDate()、SQLSetParamDateTime()、SQLSetParamDecimal()、SQLSetParamFloat()、SQLSetParamInt()、SQLSetParamLong()、SQLSetParamNull()、または SQLSetParamTime() の関数が無効なパラメータ ID で呼び出されています。
-8	無効なパラメータ リストです	無効なパラメータ リストの例: 1, 2, 3, 5 (4 が見つかりません)
-9	NULL パラメータに対する無効なタイプです	SQLSetParamNull 関数が無効な Type 引数値で呼び出されています。

結果コード	エラー メッセージ	説明
-10	パラメータのデータ タイプは変更できません	SQLSetParamChar()、SQLSetParamDate()、SQLSetParamDateTime()、SQLSetParamDecimal()、SQLSetParamFloat()、SQLSetParamInt()、SQLSetParamLong()、SQLSetParamNull()、または SQLSetParamTime() の関数が既存のパラメータに対して異なるタイプで呼び出されています。
-11	ステートメントが正しく実行された後にパラメータを追加することはできません。	SQLSetParamChar()、SQLSetParamDate()、SQLSetParamDateTime()、SQLSetParamDecimal()、SQLSetParamFloat()、SQLSetParamInt()、SQLSetParamLong()、SQLSetParamNull()、または SQLSetParamTime() の関数が、ステートメントが正常に実行された後に新しいパラメータ ID に対して呼び出されています。
-12	無効な日付/時刻形式です	たとえば、SQLSetParamTime()、SQLInsertExecute()、または SQLUpdateCurrent() の実行時に無効な日付/時刻形式が検出されました。
-13	無効な 10 進数形式です	たとえば、SQLSetParamDecimal()、SQLInsertExecute()、または SQLUpdateCurrent() の実行時に無効な 10 進数形式が検出されました。
-14	無効な通貨形式です	たとえば、SQLInsertExecute() または SQLUpdateCurrent() の実行時に無効な通貨形式が検出されました。
-15	この操作に対する無効なステートメント タイプです	SQLInsertEnd が SQLPrepareStatement() によって作成されたステートメント ID に対して呼び出されているか、SQLClearStatement() が SQLInsertPrepare() によって作成されたステートメント ID に対して呼び出されています。
-1001	メモリ不足です	この関数を実行するために十分なメモリがありません。
-1002	無効な接続	<i>ConnectionId</i> 引数に渡された値が有効ではありません。
-1003	バインドリストが見つかりません	指定したバインドリスト名が存在しません。
-1004	テンプレートが見つかりません	指定したテーブルテンプレート名が存在しません。

結果コード	エラー メッセージ	説明
-1005	内部エラー	内部エラーが発生しました。テクニカル サポートに連絡してください。
-1006	文字列が Null です	警告 - データベースから読み取られた文字列が Null です。これは単に Log Viewer に送信される警告メッセージです。
-1007	文字列が切れています	警告 - データベースから読み取られた文字列が 131 文字よりも長く、選択したときに切れています。この警告は、Log Viewer に送信されます。
-1008	場所を示す文字列がありません	Delete に Where 句がありません。
-1009	接続に失敗	データベースへの接続の失敗に関する詳細情報については、Log Viewer を確認してください。
-1010	接続文字列の DB= portion で指定されたデータベースは存在しません	指定したデータベースが存在しません。
-1011	行が選択されていません	SQLSelect() 関数または SQLExecute() 関数を実行せずに SQLNumRows()、SQLFirst()、SQLNext()、SQLLast()、または SQLPrev() 関数が呼び出されています。
-1013	読み込むファイルがありません	SQLLoadStatement() 関数が、見つからないファイル名で呼び出されています。

ベンダー データベースからのエラー メッセージが ResultCode -1 を返します。SQL Access 関数 ResultCode は常に -1 ですが、メッセージがデータベース プロバイダから正確にコピーされます。

Oracle データベースを使用しているときに発生するエラー メッセージの場合、特定のエラー メッセージおよび解決方法については Oracle Server のマニュアルを参照してください。

以下の表には、Microsoft SQL Server または Access データベースを使用しているときに発生する可能性がある一般的なエラー メッセージが一覧表示されています。

エラー メッセージ	解決方法
一度に複数のステートメントをアクティブにすることはできません	SQLSelect() 関数を呼び出した後で、SQL コマンドを実行しようとしています。SQLEnd() を実行して SQLSelect() からリソースを解放するか、2 番目のステートメントに対して別の ConnectionId を使用します。

エラー メッセージ	解決方法
コマンドを処理するために使用できる十分なメモリがありません	クライアント ワークステーションの再起動を試行します。
無効なオブジェクト名テーブル名です	テーブル名が使用しているデータベースに存在しません。DB=database という名で試行します。

特定のエラー メッセージおよび解決方法については、Microsoft SQL Server のマニュアルを参照してください。

予約語リスト

このセクションでは、SQL Access バインドリスト、テーブルテンプレート、および ODBC インターフェイスでの使用から除外されているキーワードを一覧表示しています。

予約キーワードがバインドリストまたはテーブルテンプレート内のカラム名として使用されている場合、Log Viewer にエラー メッセージが表示されます。エラーのタイプは、使用されている ODBC ドライバとキーワードが検出される場所によって異なります。たとえば、最も一般的なエラーの一つは、DATE および TIME をバインドリストまたはテーブルテンプレートのカラム名に使用することです。このエラーを回避するには、たとえば "aDATE" および "aTIME" など少し異なる名前を使用します。

予約キーワードは、InTouch SQL Access によって使用される SQL (Structured Query Language) を定義します。キーワードは、使用されている特定の ODBC ドライバによっても認識されます。SQL コマンドを正しく解釈できない場合、SQL アクセス マネージャは Log Viewer で表示できるエラー メッセージを生成します。

次のアルファベット順のリストは、SQL Access および ODBC の予約キーワードを示しています。

ABSOLUTE	ADA	追加
ALL	ALLOCATE	ALTER
AND	ANY	ARE
AS	ASC	ASSERTION
AT	AUTHORIZATION	AVG
BEGIN	BETWEEN	BIT
BIT_LENGTH	BY	CASCADE
CASCADE	CASE	CAST
CATALOG	CHAR	CHAR_LENGTH
CHARACTER	CHARACTER_LENGTH	CHECK
CLOSE COALESCE	COBOL	COLLATE
COLLATION	COLUMN	COMMIT
CONNECT	CONNECTION	CONSTRAINT

CONSTRAINTS	CONTINUE	CONVERT
CORRESPONDING	COUNT	CREATE
CURRENT	CURRENT_DATE	CURRENT_TIME
CURRENT_TIMESTAMP	CURSOR	DATE
DAY	DEALLOCATE	DEC
DECIMAL	DECLARE	DEFERRABLE
DEFERRED	削除	DESC
DESCRIBE	DESCRIPTOR	DIAGNOSTICS
DICTIONARY	DISCONNECT	DISPLACEMENT
DISTINCT	DOMAIN	DOUBLE
DROP	ELSE	END
ESCAPE	EXCEPT	EXCEPTION
EXEC	EXECUTE	EXISTS
EXTERNAL	EXTRACT	FALSE
FETCH	FIRST	FLOAT
FOR FOREIGN	FORTRAN	FOUND
FROM FULL	GET	GLOBAL
GO	GOTO	GRANT
GROUP	HAVING	HOURL
IDENTITY	IGNORE	IMMEDIATE
IN	INCLUDE	INDEX
INDICATOR	INITIALLY	INNER
INPUT	INSENSITIVE	挿入
INTEGER	INTERSECT	INTERVALL
INTO	IS	ISOLATION
JOIN	KEY	LANGUAGE
LAST	LEFT	LEVEL
LIKE	LOCAL	LOWER
MATCH	MAX	MIN
MINUTE	MODULE	MONTH
MUMPS	NAMES	NATIONAL

NCHAR	NEXT	NONE
NOT	NULL	NULLIF
NUMERIC	OCTET_LENGTH	OF
OFF	ON	ONLY
OPEN	OPRN	OPTION
OR	ORDER	OUTER
OUTPUT	OVERLAPS	PARTIAL
PASCAL	PLI	POSITION
PRECISION	PREPARE	PRESERVE
PRIMARY	PRIOR	PRIVILEGES
PROCEDURE	PUBLIC	RESTRICT
REVOKE	RIGHT	ROLLBACK
ROWS	SCHEMA	SCROLL
SECOND	SECTION	選択
SEQUENCE	SET	SIZE
SMALLINT	SOME	SQL
SQLCA	SQLCODE	SQLERROR
SQLSTATE	SQLWARNING	SUBSTRING
SUM	SYSTEM	TABLE
TEMPORARY	THEN	TIME
TIMESTAMP	TIMEZONE_HOUR	TIMEZONE_MINU
TO	TRANSACTION	TRANSLATE
TRANSLATION	TRUE	UNION
UNIQUE	UNKNOWN	UPDATE
UPPER	USAGE	USING
VALUE	VALUES	VARCHAR
VARING	VIEW	WHEN
WHENEVER	WHERE	WITH
WORK	YEAR	

16 ペン トренд ウィザードの使用

InTouch ウィザードを使用して、最大 **16** 個のタグ変数からのデータを表示できるリアルタイム トренд および履歴トレンドを作成できます。**16** ペン トрендは、InTouch のインストール時にインストールできる補足コンポーネントです。

16 ペン トренд ウィザードは、他の InTouch チャート ウィザードと同じ様に設定できます。**16** ペン トренд ウィザードを使用すると、以下のトレンドプロパティを設定できます。

- 各トレンドペンに割り当てられているタグ変数
- トレンドの線の太さと色
- 履歴トレンドの開始日時と終了日時
- リアルタイム トレンドの更新率と時間範囲
- トレンド タグ変数に割り当てられている最小工学単位および最大工学単位
- トレンドの時間軸の主目盛りと小目盛り
- トレンドの縦軸の主目盛りと小目盛り

16 ペン トレンドの作成

トレンドはオプションの **16** ペン トренд ウィザードを WindowMaker から選択することによって作成できます。

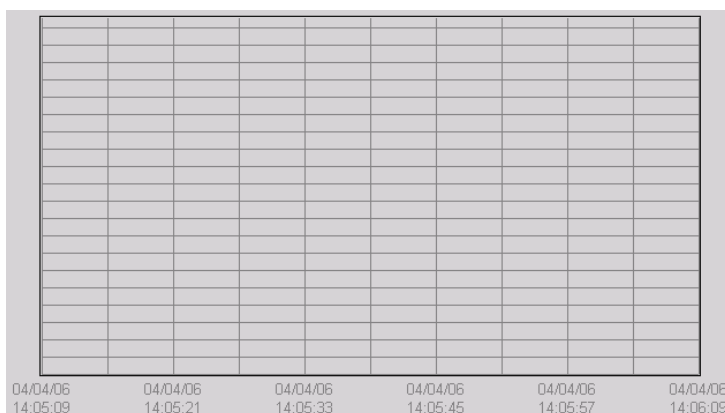
16 ペン リアルタイム トレンドまたは履歴トレンドを作成するには

1. **16** ペン トレンドを配置するウィンドウを WindowMaker で開きます。
2. [描画] メニューの [挿入] グループで [ウィザード] をクリックします。
[ウィザードの選択] ダイアログ ボックスが表示されます。
3. ウィザードのリストから [トレンド] を選択します。
[ウィザードの選択] ダイアログ ボックスの右ペインに、トレンド ウィザードのアイコンのセットが表示されます。
4. [**16** ペン トレンド] ウィザードを選択し、[OK] をクリックします。



[ウィザードの選択] ダイアログ ボックスが閉じて目的のウィンドウが再度表示されます。

5. ウィンドウ内をクリックして、**16** ペン トレンドを配置します。
ウィザードによって、**16** ペン トレンドテンプレートがウィンドウに配置されます。



6. 16 ペントレンドテンプレートをダブルクリックして、[PenTrend コントロール] ダイアログボックスを開きます。

PenTrend Control

Object Name:

Time Axis Format

Major Divisions: ☐

Minor Divisions: ☐

Update Rate (Sec):

Span (Sec):

Value Axis Format

Major Divisions: ☐

Minor Divisions: ☐

Chart

Background: ☐

Border: ☐

Trend Type

☐ Historical

☒ Realtime

Options

☐ Enable runtime configuration

Done

Cancel

	Color	Tagname	EU Text	Min EU	Max EU	Min Scale	Max Scale	Dec.Pos.	Width
1			???	0	100	0	1	1	1
2			???	0	100	0	1	1	1
3			???	0	100	0	1	1	1
4			???	0	100	0	1	1	1
5			???	0	100	0	1	1	1
6			???	0	100	0	1	1	1
7			???	0	100	0	1	1	1
8			???	0	100	0	1	1	1
9			???	0	100	0	1	1	1
10			???	0	100	0	1	1	1
11			???	0	100	0	1	1	1
12			???	0	100	0	1	1	1
13			???	0	100	0	1	1	1
14			???	0	100	0	1	1	1
15			???	0	100	0	1	1	1
16			???	0	100	0	1	1	1

7. [トレンドタイプ] 領域で、作成するトレンドのタイプとして [履歴] または [リアルタイム] を選択します。

Trend Type

☐ Historical

☒ Realtime

Options

☒ Enable runtime configuration

[PenTrend コントロール] ダイアログボックスには、選択したトレンドのタイプに基づいて適切な時間および更新オプションが自動的に表示されます。

8. [オプション] 領域で、[ランタイム構成を有効にする] オプションをオンまたはオフにします。

このオプションをオンにすると、16 ペントレンドの実行中にオペレータはプロパティを変更できます。

トレンドグラフに表示するタグの設定

16 ペントレンドウィザードを使用すると、タグ変数をトレンドペンに割り当てることができます。16 ペントレンドウィザードには、トレンドに表示されるタグ変数プロパティを指定する一組の列が含まれます。これらの列では、タグ名ディクショナリからタグに割り当てられるデフォルトのプロパティ値が使用されます。これらの割り当てられたタグ変数値は、トレンドを設定するときに他の値を指定することによって上書きできます。

16 ペントレンドタグ変数を設定するには

1. 16 ペントレンドテンプレートを含むウィンドウを開きます。
2. 16 ペントレンドをダブルクリックします。[PenTrend コントロール] ダイアログボックスが表示され、トレンドペンに関連付けられているタグ変数を指定するためのグリッド領域が下部付近に表示されます。

Color	Tagname	EU Text	Min EU	Max EU	Min Scale	Max Scale	Disc.Pos	Width
1	InPumpPress	°C	0	200	0	1	1	1
2	OutPumpPress	???	0	100	0	1	1	1
3		???	0	100	0	1	1	1
4		???	0	100	0	1	1	1
5		???	0	100	0	1	1	1
6		???	0	100	0	1	1	1
7		???	0	100	0	1	1	1
8		???	0	100	0	1	1	1
9		???	0	100	0	1	1	1
10		???	0	100	0	1	1	1
11		???	0	100	0	1	1	1
12		???	0	100	0	1	1	1
13		???	0	100	0	1	1	1
14		???	0	100	0	1	1	1
15		???	0	100	0	1	1	1
16		???	0	100	0	1	1	1

3. [オブジェクト名] ボックスで、16 ペントレンドに名前を割り当てます。

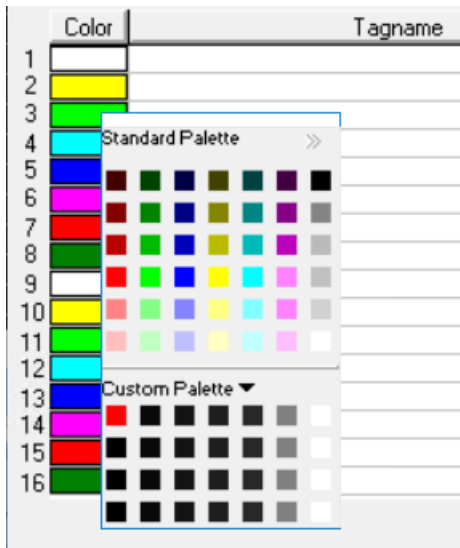
デフォルトの名前は PenTrend_1 で、新しいトレンドを作成するたびにこの名前が増分します。

4. [タグ名] ボックスに、グリッドの左に一覧表示されているペン番号に関連付けられているタグ変数の名前を入力します。

[タグ名] の下にあるセル内をダブルクリックすると、[タグを選択してください] ダイアログボックスが表示されます。タグを [タグを選択してください] ダイアログボックスから選択することによって、ペンにタグを割り当てることができます。

注記: タグ変数を削除するには、タグ変数名を含む [Tagname] を選択して、キーボードのスペースバーを押します。

5. [色] 列でカラーボックスをクリックして、カラーパレットを開きます。ペンの色を選択します。



6. **[EU テキスト]** カラムに、ランタイムでそれぞれのペンのペン軸のヘッダテキストとして最初に使用するテキストを入力します。

このテキストは、ペンがアクティブに設定されているときの軸テキストです。**[EU テキスト]** カラムには、初めはタグ名ディクショナリからのタグの工学単位が割り当てられています。これらの 16 ペントレンド用のデフォルトの工学単位を上書きできます。

7. **[工学値最小値]** カラムに、ペンに割り当てる最小工学単位値を入力します。

[工学値最小値] カラムには、初めはタグ名ディクショナリからのタグの最小工学単位値が表示されます。16 ペントレンドにのみ適用する別の最小工学単位値を割り当てることができます。

8. **[工学値最大値]** カラムに、ペンに割り当てる最大工学単位値を入力します。

[工学値最大値] カラムには、初めはタグ名ディクショナリからのタグの最大工学単位値が表示されます。16 ペントレンドにのみ適用する別の最大工学単位値を割り当てることができます。

注記: 最小/最大工学単位は、履歴トレンドデータを表示するために非常に重要です。履歴トレンドは、工学単位スケールの 0 ～100 % を表示します。

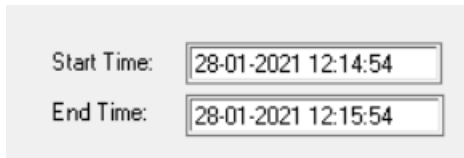
1. **[最小スケール]** カラムに、それぞれの工学単位スケールの最小ペン軸グリッドを計算するためにランタイムで最初に使用するパーセンテージを入力します。
2. **[最大スケール]** カラムに、それぞれの工学単位スケールの最大ペン軸グリッドを計算するためにランタイムで最初に使用するパーセンテージを入力します。
3. **[小数点以下の桁数]** カラムに、ペン軸グリッドをラベル付けする場合に、ランタイム時に最初に使用する小数点以下の桁数を入力します。
4. **[幅]** カラムで、トレンドに表示されるデータ値を描画するペン線の幅をピクセル単位で選択します。
5. 次の手順に進み、16 ペントレンドのトレンド時間と更新率を更新します。

トレンドの時間範囲と更新レートの設定

[PenTrend コントロール] ダイアログボックスには、リアルタイムまたは履歴の 16 ペントレンドのどちらを作成しているかに基づいて異なるオプションが表示されます。履歴トレンドに対しては時間範囲、リアルタイムトレンドに対しては更新レートを設定できます。

16 ペン履歴トレンドの時間範囲を設定するには

1. ウィンドウで 16 ペン履歴トレンドをダブルクリックします。[PenTrend コントロール] ダイアログボックスが表示され、トレンドの開始日時および終了日時を設定するためのオプションが表示されます。



Start Time: 28-01-2021 12:14:54
End Time: 28-01-2021 12:15:54

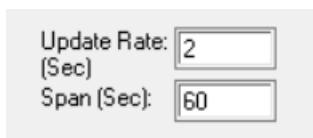
2. 履歴トレンドの開始日時および終了日時を設定します。

開始日時および終了日時には、次の形式を使用します。

DD-MM-YYYY HH:MM:SS AM/PM

16 ペンリアルタイムトレンドの更新レートを設定するには

1. ウィンドウ内の 16 ペンリアルタイムトレンドオブジェクトをダブルクリックします。[PenTrend コントロール] ダイアログボックスが表示され、リアルタイムトレンドの更新レートと範囲を設定するためのオプションが表示されます。



Update Rate (Sec): 2
Span (Sec): 60

2. [Update Rate] ボックスに、履歴トレンドのリフレッシュ間隔を秒数で入力します。
3. [Span] ボックスに、トレンドに表示されるリアルタイム間隔の秒数を入力します。

トレンド表示オプションの設定

16 ペントレンドウィザードを使用して、トレンドの視覚的外観を設定できます。

16 ペントレンドの表示オプションを設定するには

1. WindowMaker で 16 ペントレンドをダブルクリックします。
[PenTrend コントロール] ダイアログボックスが表示されます。
2. [時間軸の形式] 領域で、主目盛りの数を [主目盛り] に入力します。このオプションは、トレンドの横軸に主目盛りを設定します。
3. 主目盛りの線に別の色を割り当てる場合は、[主目盛り] の右にあるカラーボックスをクリックしてカラーパレットを開き、色を選択します。それ以外の場合は、この手順を省略して主目盛りの線にデフォルトで割り当てられている色を受け入れます。
4. [小目盛り] ボックスに、トレンドの横軸に表示される小目盛りの数を入力します。
5. 小目盛りの線の色を選択します。
6. [値軸の形式] 領域で、主目盛りの数を [主目盛り] に入力します。このオプションでは、縦の値軸に表示される主目盛り数が設定されます。
7. 主目盛りの色を設定します。
8. [小目盛り] ボックスに、トレンドの縦軸に表示される小目盛りの数を入力します。

9. 小目盛りの色を設定します。
10. [チャート] 領域で、トレンドの背景色および境界色を選択します。
11. [完了] をクリックして、16 ペントレンドに行われた設定変更を保存します。

ランタイムでのトレンド設定の変更

[PenTrend コントロール] ダイアログボックスで [ランタイム構成を有効にする] オプションが選択されている場合、アプリケーションが実行している間、オペレータは 16 ペントレンドの一部の特性を変更できます。

ランタイムでの 16 ペントレンドに対する変更は永続的なものではありません。オペレータが WindowViewer を閉じ、再びアプリケーションウィンドウを開始した場合、16 ペントレンドでは WindowMaker で元々定義されている設定が保持されます。次の図は、16 ペントレンドを実行中にクリックしたときに表示される [PenTrend コントロール] ダイアログボックスを示しています。

PenTrend Control

Object Name:

Time Axis Format

Major Divisions:

Minor Divisions:

Update Rate: (Sec)

Span (Sec):

Done

Cancel

Value Axis Format

Major Divisions:

Minor Divisions:

Chart

Background:

Border:

Trend Type

☐ Historical

☒ Realtime

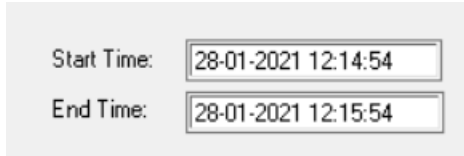
Options

☒ Enable runtime configuration

Color	Tagname	EU Text	Min EU	Max EU	Min Scale	Max Scale	Dec.Pos.	Width
1	\$Language	N/A	0	100	0	1	1	1
2	\$Language	N/A	0	100	0	1	1	1
3	???	0	100	0	1	1	1	1
4	???	0	100	0	1	1	1	1
5	???	0	100	0	1	1	1	1
6	???	0	100	0	1	1	1	1
7	???	0	100	0	1	1	1	1
8	???	0	100	0	1	1	1	1
9	???	0	100	0	1	1	1	1
10	???	0	100	0	1	1	1	1
11	???	0	100	0	1	1	1	1
12	???	0	100	0	1	1	1	1
13	???	0	100	0	1	1	1	1
14	???	0	100	0	1	1	1	1
15	???	0	100	0	1	1	1	1
16	???	0	100	0	1	1	1	1


ランタイムでは、以下の内容を変更できます。

- トрендペンに割り当てられているタグ変数または式
- トрендタグ変数または式の特性
- トレンドのタイプ（履歴またはリアルタイム）
- 履歴トレンドの日時範囲



Start Time: 28-01-2021 12:14:54
End Time: 28-01-2021 12:15:54

- リアルタイムトレンドの更新レートおよび間隔



Update Rate: 2 (Sec)
Span (Sec): 60

【完了】をクリックした後で、現在の WindowViewer アプリケーションセッションの間、トレンドでは設定変更が保持されます。

スクリプトの使用による 16 ペントレンドウィザードのコントロール

QuickScript で一組の関数を使用すると、ランタイムで 16 ペントレンドオブジェクトを管理できます。たとえば、チャートにペンを接続したり、チャートに新しいイベントを追加したり、グリッドを削除または再描画したり、スクータを削除または再描画したりすることができます。

ptGetTrendType() 関数

ptGetTrendType() 関数はスクリプトで使用して、16 ペントレンドの現在のモードが履歴データまたはリアルタイムデータのどちらであることを示す値を返すことができます。

カテゴリ

ペントレンド

構文

```
ptGetTrendType(TrendName);
```

引数

TrendName

トレンドの名前。TrendName は、文字列定数またはメッセージ型タグ変数である必要があります。

戻り値

トレンドタイプを返します。

0 = 履歴トレンド

1 = リアルタイムスクロールなし

2 = リアルタイムトレンド

例

以下の例では、PumpPress トレンドが履歴データまたはリアルタイムデータのどちらを表示しているかを示す値を返しています。

```
ptGetTrendType("PumpPress");
```

ptLoadTrendCfg() 関数

ptLoadTrendCfg() 関数はスクリプトで使用して、ファイルからトレンド設定値をロードできます。

カテゴリ

ペントレンド

構文

```
ptGetTrendCfg(TrendName,FileName);
```

引数

TrendName

トレンドオブジェクトの名前。**TrendName** に割り当てられる値は、文字列定数またはメッセージ型タグである必要があります。

FileName

設定ファイルの名前。設定ファイルへのフォルダパスは **FileName** 引数に含める必要があります。

例

TankFarm トレンドは、C:\TrendCfg.txt ファイルからの値で設定されます。

```
ptLoadTrendCfg("TankFarm","C:\TrendCfg.txt");
```

ptPanCurrentPen() 関数

ptPanCurrentPen() 関数はスクリプトで使用して、16 ペントレンドのペンを縦の値軸で上方または下方にスクロールできます。縦スクロールは引数値として指定されるトレンドの主ユニットまたは小ユニットの単位数によって決定されます。

カテゴリ

ペントレンド

構文

```
ptPanCurrentPen(TrendName,MajorUnits, MinorUnits);
```

引数

TrendName

トレンドオブジェクトの名前。**TrendName** は、文字列定数またはメッセージ型タグ変数である必要があります。

MajorUnits

主目盛りで定義されたユニット数によってスクロールする乗数。負の数は縦軸の下方へのスクロールを示しています。

MinorUnits

小目盛りで定義されたユニット数によって追加スクロールするための乗数。負の数は縦軸の下方へのスクロールを示しています。

例

この例では、ペンを主目盛りで上に 1 つスクロールします。

```
ptPanCurrentPen("TrendName", 1, 0);
```

この例では、ペンを小目盛りで半分上にスクロールします。

```
ptPanCurrentPen("TrendName", 0, 0.5);
```

この例では、ペンを主目盛りで 2 つ、小目盛りで半分下にスクロールします。

```
ptPanCurrentPen("TrendName", -2, -0.5);
```

この例では、主目盛りで 1 つ上に、小目盛りで 2 つ下にスクロールします。

```
ptPanCurrentPen("TrendName", 1, -2);
```

ptPanTime() 関数

ptPanTime() 関数はスクリプトで使用して、16 ペン トレンドのペンを指定したトレンドの主ユニットまたは小ユニット数に基づいて、横軸で左または右にスクロールできます。

カテゴリ

ペン トレンド

構文

```
ptPanTime(TrendName, MajorUnits, MinorUnits);
```

引数

TrendName

トレンド オブジェクトの名前。TrendName は、文字列定数またはメッセージ型タグ変数である必要があります。

MajorUnits

横軸の主目盛りの数によってスクロールする乗数。負の数はトレンドで左にスクロールすることを示しています。

MinorUnits

小目盛りで定義されたユニット数によって追加スクロールするための乗数。負の数はトレンドで左にスクロールすることを示しています。

備考

開発時に [PenTrend Control] ダイアログ ボックスで指定した [Major Division] および [Minor Division] の設定は、スクロール量が計算される基本となります。トレンドの時間範囲が 120 秒で、主目盛りの値が 10、および小目盛りの値が 2 の場合、主目盛りを 12 秒ごと、小目盛りを 6 秒ごとにスクロールするトレンドとなります。ptPanTime("TrendName",1,0.5) 関数では、時間軸が $1 \times 12 + 0.5 \times 6 = 15$ 秒でスクロールされます。

例

この例では、ペンを主目盛りで 1 つ横軸で右にスクロールします。

```
ptPanTime("TrendName", 1, 0);
```

この例では、ペンを小目盛りで 0.5 ほど横軸で右にスクロールします。

```
ptPanTime("TrendName", 0, 0.5);
```

この例では、ペンを主目盛りで 2.5 ほど横軸で左にスクロールします。

```
ptPanTime("TrendName", -2, -0.5);
```

この例では、ペンを主目盛りで 1 つ右に、小目盛りで 2 つ左にスクロールします。

```
ptPanTime("TrendName", 1, -2);
```

ptPauseTrend() 関数

ptPauseTrend() 関数はスクリプトで使用して、16 ペン トレンドがグラフを更新することを一時的に停止できます。トレンドは、ptPauseTrend を値 0 で再度呼び出すまで停止したままです。

カテゴリ

ペン トレンド

構文

```
ptPauseTrend(TrendName, Value);
```

引数

TrendName

トレンドオブジェクトの名前。**TrendName** は、文字列定数またはメッセージ型タグ変数である必要があります。

Value

1 はトレンド更新を一時停止します。0 はトレンド更新を再開します。

例

この例では、**Value** 引数が 1 の間、16 ペントレンドへのさらなる更新が一時停止します。

```
ptPauseTrend ("TrendName",1);
```

ptSaveTrendCfg() 関数

ptSaveTrendCfg() 関数はスクリプトで使用して、トレンドの現在の設定値をファイルに保存できます。

カテゴリ

ペントレンド

構文

```
ptSaveTrendCfg(TrendName,FileName);
```

引数

TrendName

トレンドオブジェクトの名前。文字列定数またはメッセージ型タグのいずれかである必要があります。

FileName

トレンドの設定値を保存するファイルの名前。設定ファイルへのフォルダパスは **FileName** 引数で指定できます。

例

ptSaveTrendCfg() 関数は、PumpTrend 16 ペントレンドからの値を C:\Config.txt ファイルに保存します。

```
ptSaveTrendCfg ("PumpTrend", "C:\Config.txt")
```

ptSetCurrentPen() 関数

ptSetCurrentPen() 関数はスクリプトで使用して、割り当てられた数によってペンを選択して、ペン軸を管理できます。

カテゴリ

ペントレンド

構文

```
ptSetCurrentPen(TrendName, PenNum);
```

引数

TrendName

トレンドの名前。文字列定数またはメッセージ型タグ変数のいずれかである必要があります。

PenNum

現在のトレンドペンとして割り当てられているペンの数 (1 ~ 16)。

例

ptSetCurrentPen() 関数はペン 2 を PumpPress トレンドの現在のペンとして割り当てます。

```
ptSetCurrentPen("PumpPress",2);
```

ptSetPen() 関数

ptSetPen() 関数はスクリプトで使用して、トレンド ペンにタグ変数を割り当てることができます。

カテゴリ

ペン トレンド

構文

```
ptSetPen(TrendName, PenNum, TagName);
```

引数

TrendName

トレンド オブジェクトの名前。文字列定数またはメッセージ型タグ変数である必要があります。

PenNum

現在のトレンド ペンとして割り当てられているペンの数。

TagName

トレンド ペンとして割り当てられているタグ変数の名前。

例

ptSetPen() 関数は **PumpInP** タグ変数を PumpPress トレンドのペン 2 に割り当てます。

```
ptSetPen ("PumpPress",2,"PumpInP");
```

ptSetPenEx() 関数

ptSetPenEx() 関数はスクリプトで使用して、特定のトレンド ペンにタグ変数を割り当て、タグ変数ディクショナリで指定されたタグ変数の設定値を上書きできます。

カテゴリ

ペン トレンド

構文

```
ptSetPenEx(TrendName, PenNum, TagName, minEu, maxEU, minPercent, maxPercent, Decimal, EU);
```

引数

TrendName

トレンド オブジェクトの名前。文字列定数またはメッセージ型タグ変数のいずれかである必要があります。

PenNum

現在のトレンド ペンとして割り当てられているペンの数。

TagName

トレンド ペンとして割り当てられているタグ変数の名前。

minEU

指定したタグ変数の最小工学単位値。

maxEU

指定したタグ変数の最大工学単位値。

minPercent

それぞれの工学単位スケールの最小ペン軸グリッドを計算するためにランタイムで最初に使用するパーセンテージ。

maxPercent

それぞれの工学単位スケールの最大ペン軸グリッドを計算するためにランタイムで最初に使用するパーセンテージ。

Decimal

トレンドのタグ変数値の小数点精度。

EU

タグ変数の工学単位のラベル。

例

ptSetPenEx() 関数は **PumpInP** タグ変数を **PumpPress**トレンドのペン 2 に割り当てます。タグ変数の工学単位範囲は 0 ~ 1500 の間で設定され、ユニットは PSI です。グリッドのパーセンテージ範囲は 0 ~ 1 で、小数点精度は 2 に設定されています。

```
ptSetPenEx ("PumpPress", 2, "PumpInP", 0, 1500, 0, 1, 2, "PSI");
```

ptSetTimeAxis() 関数

ptSetTimeAxis() 関数はスクリプトで使用して、トレンドの開始日時および終了日時を設定できます。

カテゴリ

ペントレンド

構文

```
ptSetTimeAxis(TrendName, StartDateTime, EndDateTime);
```

引数**TrendName**

トレンドオブジェクトの名前。文字列定数またはメッセージ型タグのいずれかである必要があります。

StartDateTime

トレンドを開始する日時。開始日時の形式は dd/mm/yyyy hh:mm:ss AM/PM です。

EndDateTime

トレンドを終了する日時。終了日時の形式は dd/mm/yyyy hh:mm:ss AM/PM です。

例

ptSetTimeAxis() 関数は、トレンドの開始日時および終了日時を 2007 年 5 月 22 日月曜日の 8:30 から開始する 25 時間単位で設定します。

```
ptSetTimeAxis ("PumpPress", "05/22/2007 08:30:00 AM", "05/23/2007 09:30:00 AM");
```

ptSetTimeAxisToCurrent() 関数

ptSetTimeAxisToCurrent() 関数はスクリプトで使用して、現在のチャートの範囲およびチャートの終了時間を計算できます。

カテゴリ

ペントレンド

構文

```
ptSetTimeAxisToCurrent(TrendName);
```

引数

TrendName

トレンドオブジェクトの名前。***TrendName*** は、文字列定数またはメッセージ型タグ変数である必要があります。

例

ptSetTimeAxisToCurrent() 関数は、PumpPressトレンドの終了日時を現在の日時に設定します。

```
ptSetTimeAxisToCurrent("PumpPress");
```

ptSetTrend() 関数

ptSetTrend() 関数はスクリプトで使用して、16 ペントレンドへの更新を一時停止または再開します。

カテゴリ

ペントレンド

構文

```
ptSetTrend(TrendName, EnableUpdates);
```

引数

TrendName

トレンドオブジェクトの名前。文字列定数またはメッセージ型タグ変数のいずれかである必要があります。

EnableUpdates

1 はトレンドの更新を開始します。0 はトレンド更新を停止します。

例

ptSetTrend() 関数は PumpPressトレンドを更新します。

```
ptSetTrend("PumpPress",1);
```

ptSetTrendType() 関数

ptSetTrendType() 関数はスクリプトで使用して、トレンドが履歴データまたはリアルタイムデータのどちらを表示しているのかを指定できます。

カテゴリ

ペントレンド

構文

```
ptSetTrendType(TrendName, TrendType);
```

引数

TrendName

トレンドオブジェクトの名前。文字列定数またはメッセージ型タグ変数のいずれかである必要があります。

TrendType

1 は履歴トレンドを示します。2 はリアルタイムトレンドを示します。

例

ptSetTrendtype() 関数は PumpPressトレンドがリアルタイムデータを表示するように指定します。

```
ptSetTrendType("PumpPress",2);
```

ptZoomCurrentPen() 関数

ptZoomCurrentPen() 関数はスクリプトで使用して、トレンドの Y 軸に表示される値範囲を変更できます。トレンドの縦軸の範囲は、指定したズーム率によって増減できます。

カテゴリ

ペントレンド

構文

```
ptZoomCurrentPen(TrendName,ZoomFactor);
```

引数

TrendName

トレンドオブジェクトの名前。文字列定数またはメッセージ型タグ変数のいずれかである必要があります。

ZoomFactor

1.0 よりも大きい数値を割り当てると、現在の範囲限界をズーム因数で乗算することによって、トレンドの値範囲が増加します。1.0 よりも小さいズーム因数を割り当てると、トレンドの縦軸に表示される値範囲が減少します。

備考

ズーム率は、現在のペンの Y 軸の既存の範囲に適用されます。たとえば、トレンドの Y 軸が -50 ~ 50 の範囲で始まり、2.0 の比率でズームすると、新しい範囲は -100 ~ 100 となります。2.0 の比率でもう一度ズームすると、新しい範囲は -200 ~ 200 となります。ズーム率は元の Y 軸範囲ではなく、現在有効である範囲に適用されます。

ズーム率は各トレンドのペンを実行している間維持されます。ptSetCurrentPen() 関数を使用して 1 つのペンから別のペンに切り替える場合、Y 軸の値範囲は選択したペンに対して現在のスケールを反映します。

例

ptZoomCurrentPen 関数は、トレンドの縦 Y 軸に表示される「PumpPress」というトレンドにある現在のタグ変数の Y 軸範囲を 2 倍にします。

```
ptZoomCurrentPen("PumpPress",2);
```

ptZoomTime() 関数

ptZoomTime() 関数はスクリプトで使用して、トレンドの横軸に表示される時間範囲を変更できます。

カテゴリ

ペントレンド

構文

```
ptZoomTime(TrendName,Zoom);
```

引数

TrendName

トレンドオブジェクトの名前。文字列定数またはメッセージ型タグ変数のいずれかである必要があります。

Zoom

1.0 よりも大きい数値を割り当てると、トレンドの横軸に表示される期間が増加します。1.0 よりも小さい数値を割り当てると、横軸に表示される期間が減少します。

例

ptZoomTime() 関数は、トレンドの横軸に表示される期間を 17 パーセント増加させます。

```
ptZoomTime("PenTrend_1", 1.17);
```

ptZoomTime() 関数は、トレンドの横軸に表示される期間を 50 パーセント削減させます。たとえば、**ptZoomTime()** 関数は、元の時間範囲が 1 時間に設定されている場合は、トレンドの期間を 30 分に短縮します。

```
ptZoomTime("PenTrend_1", 0.5);
```

Symbol Factory

Symbol Factory には、InTouch アプリケーション ウィンドウで視覚要素として使用できる 4,000 を超える産業用シンボルコレクションが含まれています。Symbol Factory は、InTouch HMI のインストール時にインストールできる補足コンポーネントです。

注記: 産業用グラフィック エディタを使用して、Application Server と相互作用する InTouch アプリケーション用の視覚要素を作成します。グラフィック エディタを使用して、InTouch HMI とは別のアプリケーション用にインテリジェントな視覚要素を作成することもできます。産業用グラフィック エディタについては、Application Server のマニュアルを参照してください。

AVEVA では、この製品に関する一切の保証は提供していません。問題が発生した場合は、グローバルテクニカルサポートに連絡してください。新しいユーティリティまたはアプリケーションをインストールまたは使用する前に、常にアプリケーションおよびデータをバックアップすることを強くお勧めします。

シンボル タイプ

Symbol Factory には以下の 4 つのタイプのウィザードが含まれます。

- ピクチャ ウィザード
- ビットマップ ウィザード
- テクスチャ ウィザード
- InTouch オブジェクト

ピクチャ ウィザード

Symbol Factory ピクチャ ウィザードは、装置図またはフロー図のベクトルベースの画像です。アプリケーションを作成するときに、以下の手順でピクチャ ウィザード画像を変更できます。

- アニメーションを画像に割り当てる
- 画像を横または縦に回転させる
- 画像の縦横の視点を変更する
- 画像を軸上で回転させる
- 画像の塗りつぶし色およびパターンを変更する
- 画像の線のサイズ、パターン、および色を変更する

ビットマップ ウィザード

ビットマップ ウィザードは、ウィンドウ アイコンまたはテキストブロックなどのビットマップ画像です。アプリケーションを作成するときに、以下の手順でピクチャ ビットマップ ウィザード画像を変更できます。

- アニメーションをビットマップに割り当てる
- 画像を横または縦に回転させる
- ビットマップの縦横の長さを変更する
- ビットマップ境界の周囲に境界線または影を配置する
- ビットマップ イメージを軸上で 90 度ずつ回転させる
- 透明色を定義する
- ビットマップ内の最大 3 色を他の色に置換する

テクスチャ ウィザード

テクスチャ ウィザードはビットマップ ウィザードと似ていますが、連続パターンを形成するためにサイズ変更できる点が異なります。テクスチャ ウィザードは一般的に、ウィンドウの背景の作成またはグラフィック オブジェクトの塗りつぶしとして使用されます。テクスチャ ウィザードは、**Symbol Factory** テクスチャ カテゴリから選択できます。

InTouch オブジェクト

InTouch オブジェクトは、**Symbol Factory** で現状のまま保存される InTouch セルまたはウィザードです。

InTouch オブジェクトを **Symbol Factory** から **WindowMaker** に貼り付けた後は、**Symbol Factory** では編集できません。

WindowMaker でオブジェクトをダブルクリックすると、オブジェクトがセルの場合は **［タグ変数の変更］** ダイアログ ボックスが表示され、個別のグラフィック オブジェクトに対してはアニメーション リンク選択ダイアログ ボックスが表示されます。

Symbol Factory の使用

Symbol Factory ウィザードの使用方法は、その他のウィザードと似ています。ウィザードを選択してウィンドウに配置し、設定します。

簡単な使用方法

Symbol Factory の使用に慣れている場合は、以下のヒントを参照して簡単に使用を開始できます。

- ウィザードのオプションを設定するには、ウィンドウでウィザードをダブルクリックし、**［Reichard Software による Symbol Factory］** ダイアログ ボックスの **［オプション］** をクリックします。
- ウィザードを別のカテゴリにコピーするには、サムネイル画像を **［カテゴリ］** ウィンドウにドラッグして別のカテゴリにドロップします。移動するには、**Shift** キーを押します。
- ウィザードの説明を編集するには、ウィザードのサムネイルを右クリックします。カテゴリの説明を編集するには、カテゴリを右クリックします。
- ウィザードを削除するには、ウィザードのサムネイルを右クリックし、**［シンボルの削除］** をクリックします。
- **Symbol Factory** は、開発者グループがネットワーク間で同じウィザードのライブラリを使用および提供できるように設定できます。

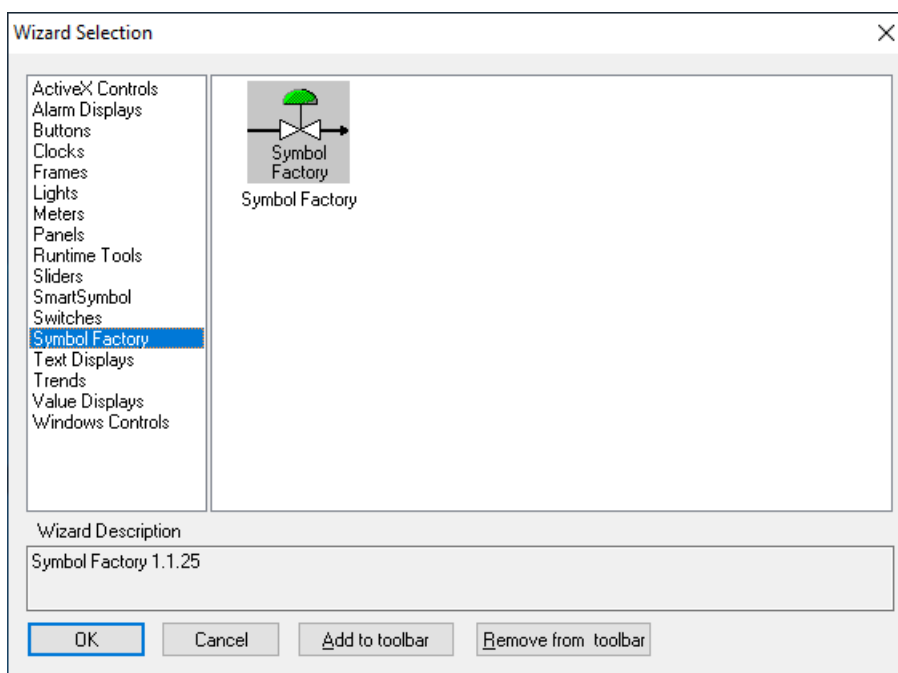
- 特定のカテゴリのすべてのウィザードは、.cat という拡張子の付いたファイルで保存されます。
Symbol Factory カテゴリ ファイルは、通常は c:\program files\wonderware\intouch\symfac フォルダに配置されます。このファイルは別のコンピュータの c:\program files\wonderware\intouch\symfac フォルダにコピーできます。

ウィンドウへの Symbol Factory ウィザードの配置

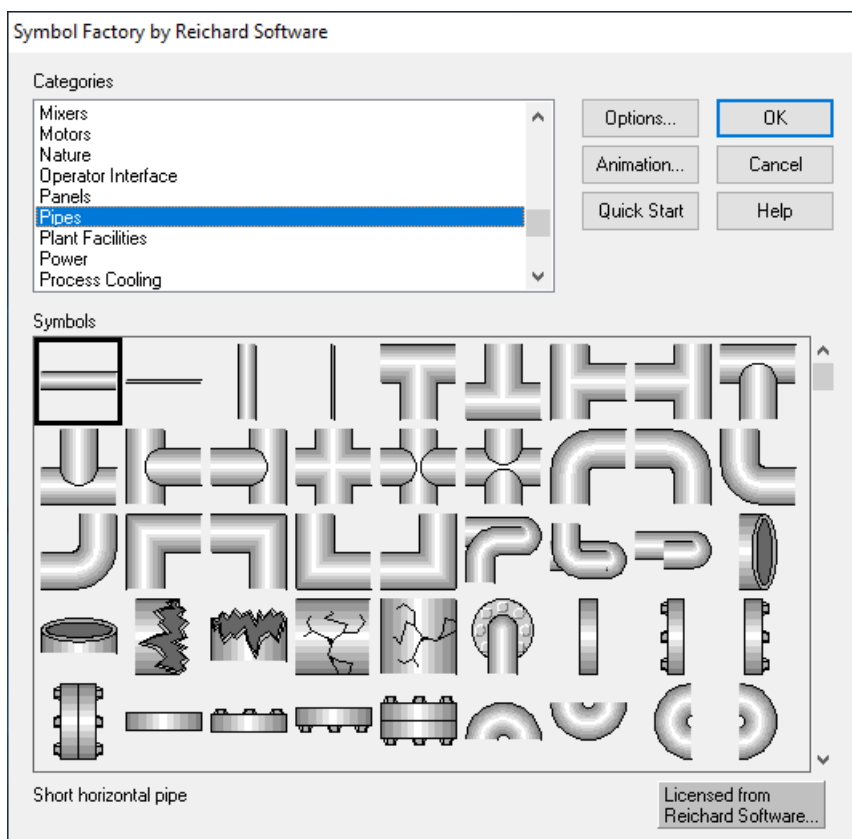
Symbol Factory ウィザードは、他のウィザードを配置するのと同様にウィンドウに配置します。

ウィンドウに Symbol Factory ウィザードを配置するには

1. WindowMaker でアプリケーションを開きます。
2. [描画] メニューの [挿入] グループで [ウィザード] をクリックします。
[ウィザードの選択] ダイアログ ボックスが表示されます。



3. 左ペインに表示されているウィザードのリストで、[Symbol Factory] をクリックします。
4. 表示領域で Symbol Factory ウィザードを選択して、[OK] をクリックします。
5. ウィンドウ内をクリックして、ウィザードを配置します。
[Reichard Software による Symbol Factory] ダイアログ ボックスが表示されます。



6. [カテゴリ] リストから、カテゴリを選択します。[Symbol] ウィンドウに、選択したカテゴリのウィザードが表示されます。
7. 配置するウィザードを選択して、[OK] をクリックします。

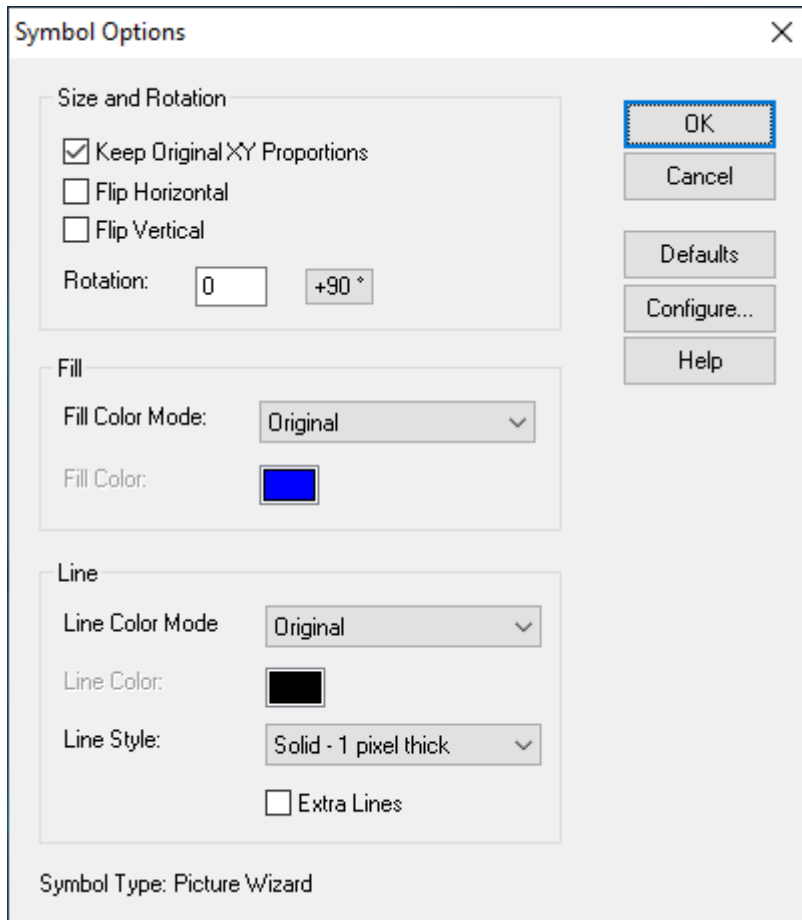
シンボル オプションの設定

ウィザードオプションはウィザードによって異なります。色を変更すると、各ピクセルがスキャンされ、変更される場合があるので、ビットマップおよびテクスチャの描画速度に影響が与えられます。

ウィザードオプションを設定するには

1. Symbol Factory ウィザードを含むウィンドウを開きます。
2. ウィザードをダブルクリックします。
[Reichard Software による Symbol Factory] ダイアログ ボックスが表示されます。
3. ウィザードを選択したままで、[Options] をクリックします。

[シンボル オプション] ダイアログ ボックスが表示されます。[シンボル オプション] ダイアログ ボックスに表示される画像プロパティは、編集するために選択したウィザードのタイプによって異なります。以下の例は、ピクチャ ウィザードシンボルを選択した場合に使用できるオプションを示しています。



ヒント: **[Symbol Factory の設定]** ダイアログ ボックスで **[マウスの右ボタンの代替ボタンを有効にする]** オプションが選択されている場合は、**[シンボルの編集]** ボタンが表示され、ボタンをクリックすると選択したウィザードを設定できます。

4. **[サイズと回転]** 領域で、以下のいずれかを実行します。
 - ウィザードのオリジナルの縦横比を保持するには、**[元の XY 比率を維持]** チェック ボックスをオンにします。
 - ウィザードを左右反転するには、**[左右反転]** チェック ボックスをオンにします。
 - ウィザードを上下反転するには、**[上下反転]** チェック ボックスをオンにします。
 - **[回転タイプ]** ボックスで、ウィザードを回転する角度の数を入力します。ピクチャ ウィザードは任意の角度に回転できます。ビットマップ ウィザードおよびテキスト ウィザードは、90 度の倍数 (0、90、180、または 270) でのみ回転させることができます。ボタンをクリックすると、自動的に回転角度が 90 度ずつ増えます。
5. ピクチャ ウィザードを設定している場合は、**[線]** 領域および **[塗りつぶし]** 領域で、以下のいずれかを実行します。
 - **[塗りつぶしの色モード]** リストで、塗りつぶしタイプをクリックします。**[塗りつぶしの色]** ボックスをダブルクリックして、カラーパレットを開きます。
 - **[線の色モード]** リストで、線の色をクリックします。**[線の色]** ボックスをダブルクリックして、カラーパレットにアクセスします。

- **[線スタイル]** リストで、線のスタイルをクリックします。
 - **[余分な線]** チェック ボックスをオンにして、ウィザード内の斜めの境界線に線を追加します。
6. ビットマップ ウィザードまたはテキスト ウィザードを設定している場合は、**[効果]** 領域および **[色の変更]** 領域で、以下のいずれかを実行します。
- ウィザードの周囲に黒い境界線を作成するには、**[境界線を含める]** チェック ボックスをオンにします。
 - ウィザードの後ろに暗い灰色の影を作成するには、**[影を含める]** チェック ボックスをオンにします。
 - それぞれのカラー ボックスをクリックすると、カラー パレットが開き、ウィザード内の色を変更できます。
7. **[OK]** をクリックします。

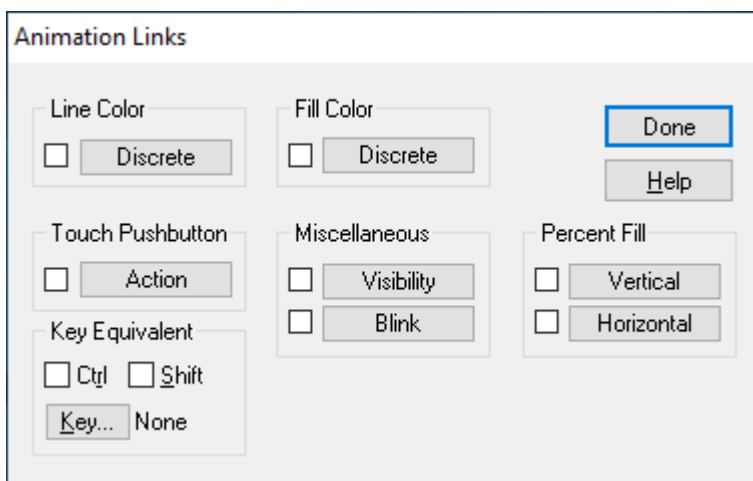
ウィザードのアニメーション化

すべての **Symbol Factory** ウィザードはアニメーション化できます。**Symbol Factory** は、最も一般的なアニメーション リンクへのアクセスを提供します。

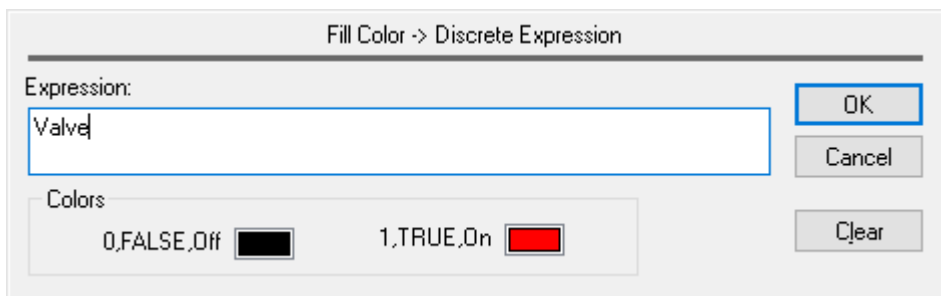
別のタイプのアニメーションが必要な場合は、ウィザードを分解して、標準の InTouch アニメーション リンクを使用してアニメーション化します。

ウィザードをアニメーション化するには

1. **Symbol Factory** でウィザードを選択するか、すでにウィンドウに貼り付けられている場合は、ウィザードをダブルクリックします。
[Reichard Software による Symbol Factory] ダイアログ ボックスが表示されます。
2. **[アニメーション]** をクリックします。
[アニメーション リンク] ダイアログ ボックスが表示されます。



3. 各アニメーション リンクのタイプのボタンをクリックして、選択したウィザードに適用します。
式ダイアログ ボックスが表示されます。



4. [式] ウィンドウに式を入力します。

ウィンドウでダブルクリックして、[タグの選択] ダイアログ ボックスを開きます。既存のタグを使用する場合は、式でタグをダブルクリックして**タグ名ディクショナリ**を開いて、タグ定義を参照できます。

未定義のタグを使用する場合は、式ダイアログ ボックスを閉じるときに、定義するよう促すメッセージが表示されます。

5. アニメーション リンクのタイプの詳細を設定します。

6. [OK] をクリックします。

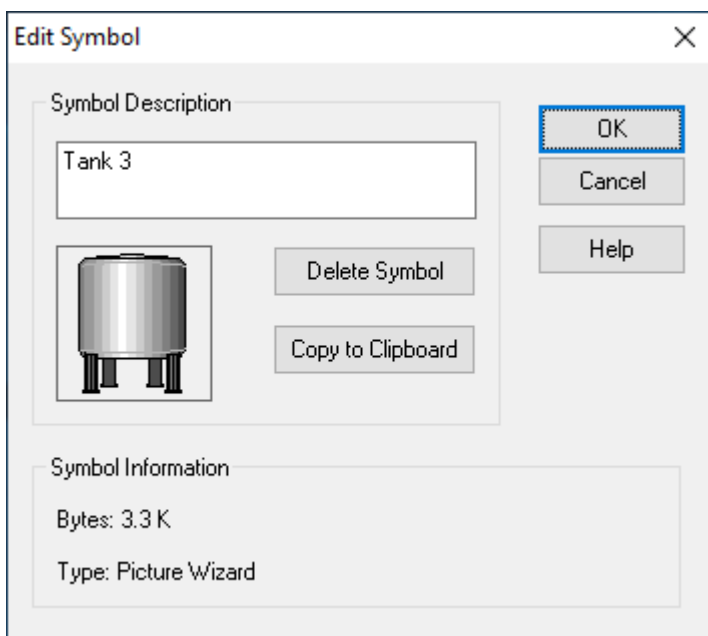
シンボルの編集

ウィザード ツールヒントの説明を変更すること、ウィザードを削除すること、またはウィザードを Windows クリップボードへコピーすることができます。

ウィザードを編集するには

1. [Reichard Software による Symbol Factory] ダイアログ ボックスで、ウィザードを含むカテゴリを選択します。
2. ウィザードを右クリックします。

[シンボルの編集] ダイアログ ボックスが表示されます。



3. ウィザードを編集します。以下のいずれかを実行します。

- [シンボル説明] ボックスに、ツールヒント テキストを入力します。説明の最大文字数は 80 文字です。
- ウィザードを削除するには、[シンボルの削除] をクリックします。
- ウィザードを Windows クリップボードにコピーするには、[クリップボードにコピー] をクリックします。ウィザードがピクチャ ウィザードの場合は、Windows メタファイルとしてコピーされます。ウィザードがビットマップ ウィザードまたはテキスト ウィザードの場合は、Windows ビットマップとしてコピーされます。

4. [OK] をクリックします。

編集のためのウィザードの分解

Symbol Factory ウィザードは、分解することによって個々に編集できます。ただし、ウィザードを分解した後は、ウィザードのプロパティは失われます。誤ってウィザードを分解した場合、元に戻すツールを使用して再アセンブルできます。

ウィザードを分解するには

1. WindowMaker を開きます。
2. [アニメーション] メニューの [セル] グループで [分解] をクリックします。

ネットワークでのシンボルのカテゴリの共有

複数の開発者がネットワーク上でウィザードのカテゴリ ファイルを使用および提供できるように Symbol Factory を設定することができます。

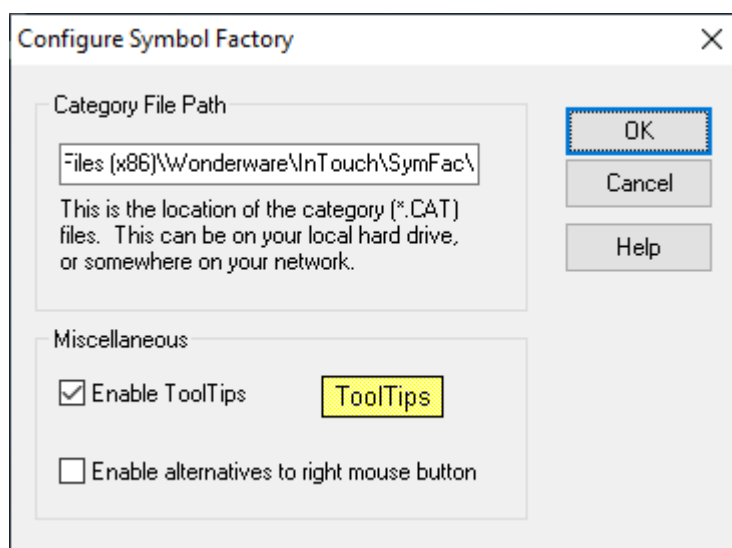
カテゴリ ファイルをネットワーク フォルダに移動するには

1. [Symbol Factory] ダイアログ ボックスで、[オプション] をクリックします。

[シンボル オプション] ダイアログ ボックスが表示されます。

2. [設定] をクリックします。

[Symbol Factory の設定] ダイアログ ボックスが表示されます。



3. [カテゴリ ファイルパス] ボックスに、カテゴリ ファイルが保存されているネットワーク フォルダ へのフルパスを入力します。
4. [OK] をクリックします。

カテゴリの読み取り専用設定

ウィザードファイルをネットワーク フォルダに保存する場合は、他のユーザーがウィザードを移動したり、名前を変更したりすることを防ぐためにカテゴリを読み取り専用にすることをお勧めします。

カテゴリを読み取り専用にするには

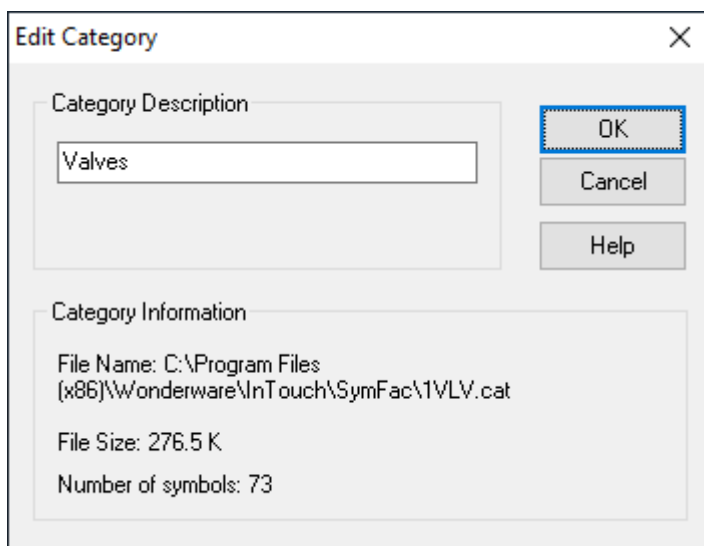
- ファイルを Windows エクスプローラで読み取り専用に設定します。

カテゴリ プロパティの表示

カテゴリ パス、ファイルサイズ、およびウィザードの数を参照できます。

カテゴリのプロパティを表示するには

1. [Symbol Factory] ダイアログ ボックスで、[カテゴリ] リスト内のカテゴリを右クリックします。
[カテゴリの編集] ダイアログ ボックスが表示されます。



2. [カテゴリ説明] ボックスに、カテゴリの新しい説明を入力して [OK] をクリックします。説明の最大文字数は 40 文字です。
3. [カテゴリ情報] 領域で、プロパティを表示します。

カテゴリ情報	説明
ファイル名	カテゴリ (.cat) ファイルパス。デフォルトでは、このパスは C:\Program Files(x86)\Wonderware\InTouch\SymFac です。
ファイルサイズ	カテゴリ ファイルのサイズ (キロバイト単位)。

カテゴリ情報	説明
シンボルの数	カテゴリに含まれるウィザードの合計数。最大値は 32,767 です。

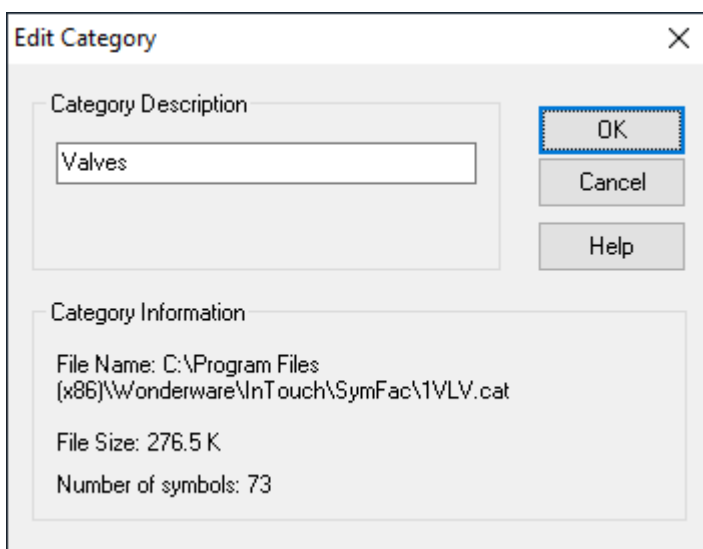
4. **[OK]** をクリックします。

既存カテゴリの編集

カテゴリ名のみ編集できます。

既存のカテゴリを編集するには

1. **[Symbol Factory]** ダイアログ ボックスで、**[カテゴリ]** リスト内のカテゴリを右クリックします。
[カテゴリの編集] ダイアログ ボックスが表示されます。



2. **[カテゴリ説明]** ボックスに、カテゴリの新しい説明を入力して **[OK]** をクリックします。説明の最大文字数は 40 文字です。
3. **[OK]** をクリックします。

カテゴリの削除

Windows エクスプローラを使用して、カテゴリのファイル名を指定することによって、カテゴリ (.cat) ファイルを削除します。

ヒント : カテゴリ ファイル名は **[Edit Category]** ダイアログ ボックスで確認できます。

Symbol Factory の設定

Symbol Factory を設定する場合、以下の項目を指定できます。

- ウィザードを選択するときにツールヒントを表示するかどうか
- **[Reichard Software による Symbol Factory]** ダイアログ ボックスに追加オプションを表示するかどうか。デフォルトでは、カテゴリおよびウィザードの説明を編集するには、アイテムを右クリックする必要があります。

- カテゴリ（.cat）ファイルの場所。各カテゴリのすべてのウィザードのすべてのデータは、1つのカテゴリ ファイルに保存されます。パフォーマンス上の理由から、このパスには .cat ファイルのみを含める必要があります。ウィザードを他の開発者と共有するには、このパスをネットワーク フォルダに設定します。「[ネットワークでのシンボルのカテゴリの共有](#)」を参照してください。

注意: ローカルの InTouch アプリケーション フォルダにはカテゴリ ファイルを配置しないでください。カテゴリファイルは、次の場所に保存してください。C:\Program Files\Wonderware\intouch\symfac

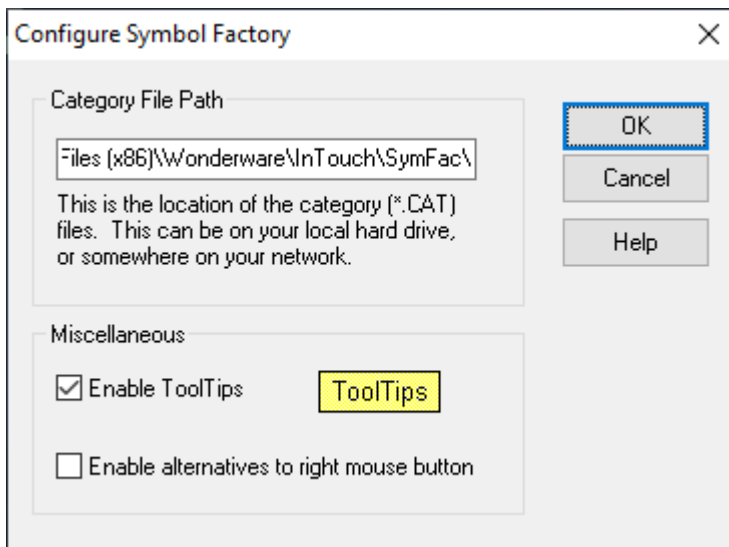
Symbol Factory を設定するには

1. [Symbol Factory] ダイアログ ボックスで、[オプション] をクリックします。

[シンボル オプション] ダイアログ ボックスが表示されます。

2. [設定] をクリックします。

[Symbol Factory の設定] ダイアログ ボックスが表示されます。



3. Symbol Factory を設定します。以下の手順を実行します。

- [カテゴリ ファイル パス] ボックスに、Symbol Factory カテゴリ（.cat）ファイルを保存する場所を入力します。
- ツールヒントを [Reichard Software による Symbol Factory] ダイアログ ボックスのウィザードに対して表示する場合は、[ツールヒントを有効にする] チェック ボックスをオンにします。
- ボタンを [Reichard Software による Symbol Factory] ダイアログ ボックスに追加する場合は、[マウスの右ボタンの代替ボタンを有効にする] チェック ボックスをオンにします。これらのボタンは、カテゴリおよびウィザードを編集するために右マウス ボタンの代わりに使用できます。

カテゴリの編集

選択したカテゴリの [カテゴリの編集] ダイアログ ボックスを表示します。

シンボルの編集

選択したウィザードの [シンボルの編集] ダイアログ ボックスを表示します。

4. [OK] をクリックします。

トラブルシューティング

誤って **Symbol Factory** ウィザードをアンインストールしてしまった場合は、もう一度インストールする必要があります。ウィザードのインストールの詳細については、『*AVEVA™ InTouch HMI アプリケーション開発ガイド*』の「[ウィザード](#)」を参照してください。

誤ってウィザードを削除した後で、そのウィザードを元に戻す場合は、そのウィザードを取得する必要があります。

削除したウィザードをカテゴリから取得するには

1. ファイルの名前を `~cat.bak` から `temp.cat` に変更します。
2. **Symbol Factory** を実行して、削除したウィザードが元に戻っているかどうかを確認します。ウィザードを元のカテゴリに移動して、`temp.cat` ファイルを削除します。
3. 上記の手順で取得できない場合は、**Ctrl** キーを押しながら削除したウィザードのカテゴリを右クリックします。この操作により、カテゴリ ファイルが圧縮され、新しいバックアップ `~cat.bak` が作成されます。

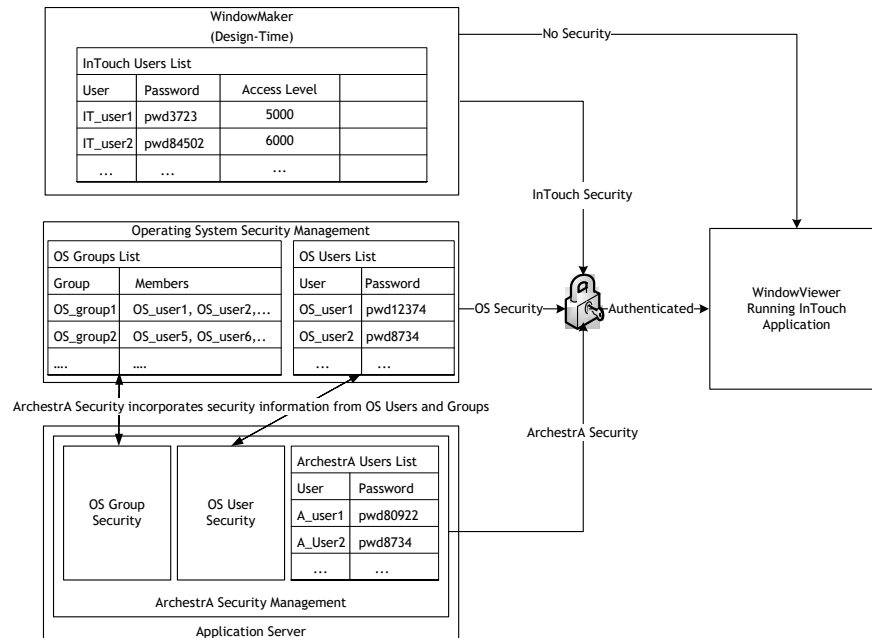
削除したウィザードが見つかるまで前の手順を繰り返します。

InTouch のセキュリティ保護

InTouch アプリケーションは、以下を使用して保護できます。

- 従来の InTouch ベースのセキュリティ
- オペレーティング システム ベースのセキュリティ
- ArchestrA ベースのセキュリティ

以下の図は、3 つのタイプのセキュリティの関係を示しています。



InTouch セキュリティ機能

InTouch アプリケーションの実行中に保護するには、以下の操作を実行できます。

- 無操作のタイムアウト時間を設定する
- キーをロックする
- メニューを非表示にする

無操作タイムアウトの設定

無操作オペレータを **WindowViewer** アプリケーションから自動的にログオフするように **InTouch** を設定できます。オペレータは無操作のためにログオフされた場合、後で再びログオンする必要があります。自動無操作ログオフ期間を設定することによって、オペレータがワークステーションを離れる場合、**InTouch** アプリケーションへの不正アクセスを防ぐことができます。

タイマーは、オペレータが実行中の **InTouch** アプリケーションと対話していない時間を測定します。タイマーは、オペレータがマウスまたはその他の入力デバイスを使用してデータを入力するたびにリセットされます。タイマーで設定した時間が経過した場合、ユーザーは自動的にログオフされます。

注記: 無操作タイマーは、**Active-X** コントロールおよび **OLE** オートメーション コントロールに対してはリセットされません。

オペレータの自動ログオフは2段階のステップで行われます。

1. **WindowViewer** は、オペレータの無操作時間が指定された警告期間を超えると、**\$InactivityWarning** システム タグを 1 に設定します。条件 **QuickScript** で **\$InactivityWarning** タグを使用して、オペレータに保留中の無操作ログオフについて警告するウィンドウを表示できます。オペレータは、指定したタイムアウト時間が発生する前に応答することによって、ログオンした状態を維持できます。オペレータが何らかの操作を行うと、**\$InactivityWarning** タグと無操作タイマーがゼロにリセットされます。
2. オペレータが無操作警告の後で応答しなかった場合、タイムアウト時間に達すると、**\$InactivityTimeout** システム タグが 1 に設定されます。**\$InactivityTimeout** が 1 になると、**WindowViewer** はログオンしているオペレータ名を予約名 **None** に設定して、**\$AccessLevel** セキュリティ タグを 0 に設定します。ユーザーは自動的にログオフされます。

タイムアウト機能は警告機能とは無関係に使用できます。

無操作タイムアウトを設定するには

1. **WindowMaker** を開きます。
2. **[ファイル]** メニューで、**[設定]** をポイントし、**[WindowViewer]** をクリックします。
WindowViewer の設定画面が表示されます。
3. **[無操作時間の設定 (秒)]** 領域で、警告とタイムアウトの値を設定します。以下の手順を実行します。
 - **[警告]** ボックスに、**\$InactivityWarning** タグが 1 に設定されるまでの秒数を入力します。
 - **[タイムアウト]** ボックスに、**\$InactivityTimeout** タグが 1 に設定されてユーザーが自動的にログオフされるまでの時間を秒で入力します。



4. **[OK]** をクリックします。
5. 無操作警告時間が経過した後に「警告 - ログオフが保留中です」という名前のウィンドウを表示するには、条件として "**\$InactivityWarning**" を使用し、スクリプト本文が以下のような条件スクリプトを作成します。

```
Show "ログオフが保留中です";
```
6. 無操作タイムアウト時間が経過した後に「ログオフしました」という名前のウィンドウを表示するには、"**\$InactivityTimeout**" を条件とし、スクリプト本文が以下のような条件スクリプトを作成します。

```
Show "ログオフしました";
```

\$InactivityTimeout システム タグ変数

無操作に設定された時間が経過したことを示します。

カテゴリ

セキュリティ

使用法

\$InactivityTimeout

備考

無操作時間タイマーが経過すると 1 に設定されます。ログオフ時間の設定の詳細については、「[無操作タイムアウトの設定](#)」を参照してください。

注意：無操作時間タイマーは、Active-X コントロール、OLE コントロール、およびオートメーション コントロールに対してはリセットされません。

データタイプ

論理型（読み取り専用）

参照項目

\$InactivityWarning

例

以下の例は「True の時」条件スクリプトです。

```
If $InactivityTimeout == 1 THEN
    Show "ログオフしました";
ENDIF
```

参照項目

\$InactivityWarning

\$InactivityWarning システム タグ変数

ログオフが発生しようとしていることをユーザーに警告するために設定された時間が経過したことを示します。

カテゴリ

セキュリティ

使用法

\$InactivityWarning

備考

無操作警告時間が経過すると 1 に設定されます。無操作時間タイマーは、マウス クリックまたはキーボード操作によってのみリセットされます。ログオフ警告の設定の詳細については、「[無操作タイムアウトの設定](#)」を参照してください。

注意：無操作時間タイマーは、ActiveX コントロール、OLE オートメーション コントロール、および SPC ウィザードに対してはリセットされません。

データタイプ

論理型（読み取り専用）

例

以下の例は「True の時」条件スクリプトです。

```
If $InactivityWarning == 1 THEN
    Show "Logoff Pending";
ENDIF;
```

参照項目

\$InactivityTimeOut

システム キーのロック

InTouch アプリケーションを実行しているコンピュータでシステム キーを無効にすることによって、標準の Windows 機能へのオペレータ アクセスを制限できます。たとえば、オペレータが Windows の CTRL+ALT+DEL キーの組み合わせを使用して、[タスク マネージャ] ダイアログ ボックスを表示することを防ぐことができます。システム キーを無効にすることによって、オペレータが InTouch HMI から別の Windows アプリケーションに切り替えることを防ぎます。

WindowViewer には、InTouch アプリケーションが起動するときにシステム キーのデフォルト状態を設定するキー フィルタ オプションがあります。キー フィルタは、アクティブである場合にシステム キーを無効にします。

さまざまな InTouch ユーザーによって完了される予測されるタスクに基づき、システム キーを無効にします。オペレータに対して、ほとんどのファンクション キーを無効にする必要があります。管理者は InTouch タスクに対してファンクション キーが引き続き必要となります。

WindowViewer にログオンするユーザーのアクセス レベルに基づき、システム キーを有効または無効にするスクリプトを記述できます。スクリプトに EnableDisableKeys() 関数を使用して、Windows のファンクション キーを選択的に有効または無効にします。

キー フィルタを有効にするには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[WindowViewer] をクリックします。
WindowViewer の設定画面が表示されます。
3. [ウィンドウ] タブをクリックします。

The screenshot shows the 'Window' tab of the 'Preferences' dialog. The 'Target resolution' section has a dropdown set to 'Screen Resolution', a checked box for 'Show target resolution boundary canvas', and input fields for 'Width: 1536 Pixels', 'Height: 864 Pixels', and 'Aspect ratio: 16 : 9'. The 'Window' section includes a checked box for 'Hide Titlebar', a text field for 'Titlebar' containing 'InTouch - WindowViewer', and a checked box for 'Show application director'. Below this are checked boxes for 'Control menu', 'Minimize box', 'Maximize box', and 'Size controls'. The 'Menus' section has a checked box for 'Menu bar' and a list of sub-menus: 'File' (checked), 'Logic' (checked), 'Debug' (unchecked), 'Special' (checked), and 'Security' (checked). Under 'File' is 'WindowMaker' (checked). Under 'Special' are 'Start uninit conversations' (checked), 'Reinitialize I/O' (checked), 'Reinitialize all' (checked), 'Select...' (checked), 'Restart historical log' (checked), 'Stop historical logging' (checked), and 'Tag viewer' (unchecked). Under 'Security' are 'Log on' (checked), 'Change password' (checked), 'Configure users' (checked), 'Log off' (checked), and 'Language' (checked). The 'Miscellaneous' section at the bottom has checkboxes for 'Impossible to close' (unchecked), 'Disable ALT key' (unchecked), 'Disable ESC key' (unchecked), 'Disable WIN key' (unchecked), 'Allow CTRL-Break to stop scripts' (checked), 'Hide cursor' (unchecked), 'Always maximize' (unchecked), and 'Enable fast switch' (checked).

1. [その他の設定] 領域で、WindowViewer システム キーを無効にします。以下の手順を実行します。
 - [高速切り替えを有効にする] チェック ボックスをオフにして、WindowViewer からユーザーを WindowMaker に切り替えるための [開発] ボタンを削除します。
 - [ALT キーの使用を禁止] チェック ボックスをオンにして、InTouch アプリケーションを実行しているコンピュータで ALT キーを無効にします。
 - [WIN キーの使用を禁止] チェック ボックスをオンにして、InTouch アプリケーションを実行しているコンピュータで WIN キーを無効にします。
 - [ESC キーの使用を禁止] チェック ボックスをオンにして、InTouch アプリケーションを実行しているコンピュータで ESC キーを無効にします。
2. [OK] をクリックします。
3. WindowViewer が InTouch アプリケーションの実行を開始するときに実行するスクリプトを記述します。

スクリプトには、**WindowViewer** にログオンするユーザーのアクセス レベルに基づき、キーを動的にロックまたはアンロックするステートメントを含める必要があります。

スクリプトに **EnableDisableKeys()** 関数を含め、**ALT** キー、**ESC** キー、および **WIN** キーを有効または無効にします。**EnableDisableKeys()** 関数は、その引数の論理値に基づきシステム キーを有効または無効にします。

```
EnableDisableKeys(AltKey, EscKey, WinKey);
```

引数値 **1** の引数値を指定すると、キー フィルタが有効になり、キーが無効になります。

EnableDisableKeys() 関数

Alt キー、**Esc** キー、および **Windows** キーのキー フィルタを有効または無効にします。

カテゴリ

表示

構文

```
EnableDisableKeys(AltKey, EscKey, WinKey);
```

パラメータ

AltKey

Alt キーのキー フィルタを有効または無効にする整数型 :

1 = フィルタの有効化 (**Alt** キーを無効化)

0 = フィルタの無効化 (**Alt** キーを有効化)

EscKey

Esc キーのキー フィルタを有効または無効にする整数型 :

1 = フィルタの有効化 (**Esc** キーの無効化)

0 = フィルタの無効化 (**Esc** キーの有効化)

WinKey

Windows キーのキー フィルタを有効または無効にする整数型 :

1 = フィルタの有効化 (**Win** キーの無効化)

0 = フィルタの無効化 (**Win** キーの有効化)

備考

Alt キーを無効にすると、**Win+L** キーの組み合わせも無効になります (**Windows** デスクトップをロックする場合)。**Win+L** は、**Alt** キーを含むキーの別の組み合わせのショートカットです。したがって、**Windows** デスクトップのロックで **Alt** キーを無効にすると、ショートカットも無効になります。

Esc キーを無効にすると、そのすべてのアクションが無効になります。

例

```
EnableDisableKeys(0,0,0); // 3 つのキーをすべて有効にします
```

```
EnableDisableKeys(1,1,1); // 3 つのキーをすべて無効にします
```

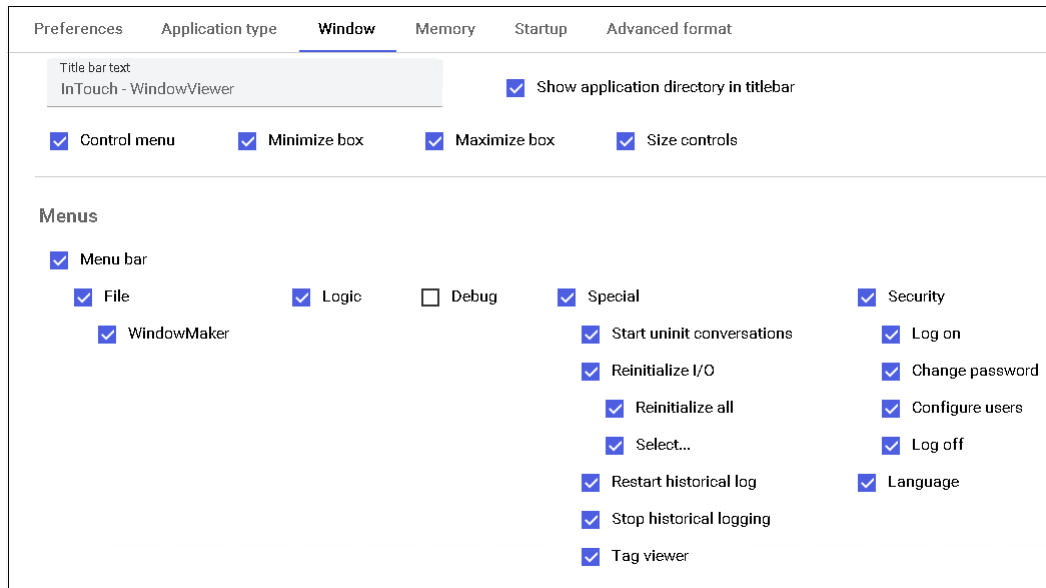
```
EnableDisableKeys(0,0,1); // Alt キー、Esc キーを有効にし、Windows キーを無効にします
```

ランタイム時のメニュー項目の非表示

InTouch アプリケーションが実行している間に **WindowViewer** メニューとコマンドを非表示にすることによって、それらのメニューやコマンドへのオペレータ アクセスを制限します。

ランタイム時にメニュー項目を非表示にするには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[WindowViewer] をクリックします。
WindowViewer の設定画面が表示されます。
3. [ウィンドウ] タブをクリックします。



4. [メニューの設定] 領域で、オペレータに対して表示する WindowViewer メニューとコマンドを選択します。以下の手順を実行します。
 - [WindowMaker] チェック ボックスをオフにして、[WindowMaker] コマンドを WindowViewer の [ファイル] メニューから使用できないようにします。このオプションをオフにしても、WindowMaker への高速切り替えには影響しません。
 - [ロジック] チェック ボックスをオフにして、QuickScript を起動および停止するコマンドを含む WindowViewer の [ロジック] メニューを非表示にします。

注記: \$LogicRunning システム タグを使用すると、オペレータがすべての QuickScript を開始および停止できるようにすることができます。

[Ctrl-Break でスクリプトの停止] オプションを選択すると、[ロジック] メニューの表示の有無に関わらず、オペレータはすべての QuickScript の実行を停止できます。

現在実行中の非同期クイック関数は停止できません。ただし、オペレータが新しい非同期クイック関数を開始することを防ぐことはできます。

- アプリケーションをテストしている場合、[デバッグ] チェック ボックスをオンにします。それ以外の場合、[デバッグ] チェック ボックスをオフにして、ランタイム中は [デバッグ] メニューを非表示にします。
- [システム] メニュー項目をオフにすると、オペレータがロギングや I/O 接続のような継続的な InTouch 関数を停止することを防ぐことができます。
- [セキュリティ] チェック ボックスをオフにすると、オペレータがセキュリティ関連のオプションを変更することを防ぐことができます。

5. [ウィンドウ コントロール] 領域で、WindowViewer からオペレータに対して利用可能とするウィンドウ コントロールを選択します。これらのオプションは、InTouch アプリケーションを実行しているウィンドウに影響を与えます。以下の手順を実行します。
 - [コントロール メニュー] チェック ボックスをオフにすると、ウィンドウを閉じるコントロール、および最小化、最大化、またサイズ変更するコントロールを非表示にできます。
 - [アイコン化ボタン] チェック ボックスをオフにすると、オペレータがウィンドウを最小化することを防ぐことができます。
 - [最大化ボタン] チェック ボックスをオフにすると、オペレータがウィンドウを最大化することを防ぐことができます。
 - [サイズ コントロール] チェック ボックスをオフにすると、オペレータがウィンドウをサイズ変更することを防ぐことができます。
6. [タイトル] バー領域で、InTouch アプリケーションを実行しているウィンドウのタイトルバーを設定します。以下の手順を実行します。
 - [タイトルバーのテキスト] ボックスに、WindowViewer タイトルバーに表示するタイトルを入力します。
 - [アプリケーションのディレクトリの表示] チェック ボックスをオンにして、InTouch アプリケーションのフォルダへのパスをタイトルバーに表示します。
 - [タイトルバーを隠す] チェック ボックスをオンにして、ウィンドウのタイトルバーを非表示にします。
7. [その他の設定] 領域で、以下の手順を実行します。
 - [WindowViewer の強制終了不可] チェック ボックスをオンにして、オペレータが InTouch アプリケーションを実行している WindowViewer ウィンドウを閉じることを防ぎます。このオプションを選択すると、ウィンドウの [閉じる] ボタンが無効になります。
[閉じる] ボタンを非表示にする場合、[ウィンドウ コントロール] 領域で [コントロール メニュー] チェック ボックスをオフにします。
 - [CTRL-Break でスクリプトの停止] チェック ボックスをオフにすると、オペレータが QuickScript の停止を有効にする CTRL + BREAK キーの組み合わせが無効となります。

注記: 現在実行中の非同期クイック関数は停止できません。ただし、新しい非同期クイック関数が実行されることを防ぐことができます。

- [カーソルの非表示] チェック ボックスをオンにすると、ランタイム中にマウス ポインタが非表示になります。この機能は、タッチスクリーン用のアプリケーションを設計している場合に役立ちます。
 - [WindowViewer を常時最大表示] チェック ボックスをオンにすると、InTouch アプリケーションを実行しているウィンドウがオペレータの画面で最大表示されます。
1. [OK] をクリックします。
 2. WindowViewer を再起動して、変更を適用します。

認証ベースおよび承認ベースのセキュリティ

InTouch セキュリティは、アプリケーションを使用しようとしているユーザーが有効なユーザーとして認識されるかどうかを決定する 2 段階プロセスの最初の段階です。2 番目の段階では、認証されたユーザーに与える InTouch 権限を決定します。

認証と承認の比較

認証は、ユーザーの識別を確認するプロセスです。一般に、オペレータは InTouch アプリケーションを使用する前にユーザー名とパスワードを入力して自分自身を認証します。セキュリティの 3 つのタイプはすべて、認証プロセスの一部としてログインプロセス中にユーザーのアカウント情報を確認します。

承認は、認証されたユーザーが要求されたリソースにアクセスできるかどうかを判断するプロセスです。一般に、InTouch 機能へのアクセスはグループ内または割り当てられたアクセス レベルのユーザーのメンバーシップに基づいて与えられます。

さまざまな認証セキュリティ モード

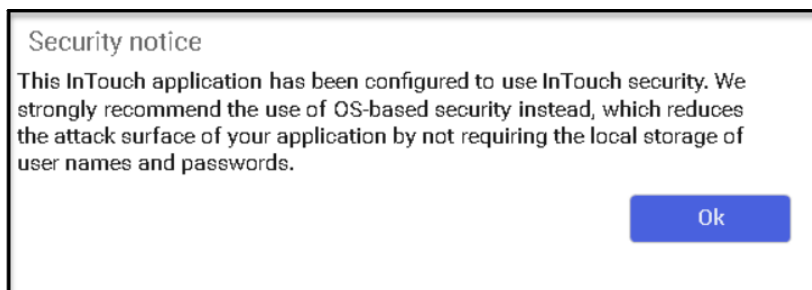
InTouch セキュリティのすべてのタイプは、ログオン プロセス中、ユーザー名とパスワードの組み合わせを使用してユーザーを認証します。それぞれのタイプのセキュリティでは、認証プロセス中に異なるメカニズムを使用して、ユーザー名とパスワードを確認します。

InTouch ベースのセキュリティの使用

アプリケーションへのセキュリティの適用はオプションです。デフォルトでは、InTouch アプリケーションにセキュリティは適用されていません。しかし、対象となる機能を内部タグにリンクすることによって、操作者が実行できる機能を制限できます。さらに、アプリケーションにセキュリティを確立するとき、アラームとイベントを InTouch HMI にログオンしているオペレーターに関連付ける監査記録を作成できます。

InTouch にセキュリティ タイプを設定するには

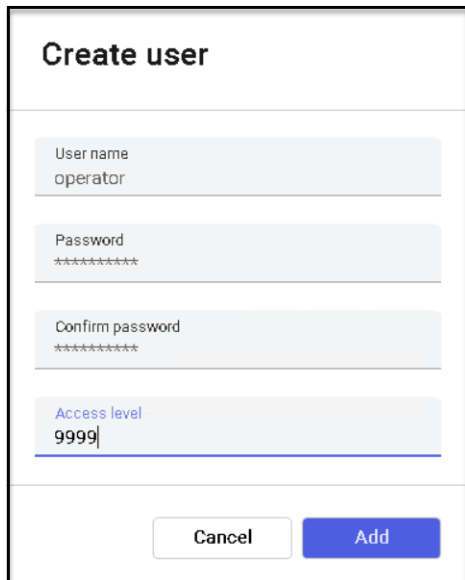
1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[セキュリティ] をクリックします。
セキュリティ設定画面が表示されます。
3. セキュリティ タイプを [InTouch] として選択します。



セキュリティ タイプを InTouch として追加すると、[ユーザーの設定] セクションが有効になります。セキュリティは、InTouch アプリケーションにログオンするためにユーザー名とパスワードを入力することによって、オペレータが自分自身を認証する過程に基づきます。各オペレータに対してユーザー名、パスワード、およびアクセス レベルを割り当てる必要があります。

ユーザーを設定するには

1. [追加] アイコンをクリックします (Alt+A)。
[ユーザーを作成] ダイアログが表示されます。

A screenshot of the 'Create user' dialog box. It has a title bar 'Create user'. Below the title bar are four input fields: 'User name' with the text 'operator', 'Password' with masked characters '*****', 'Confirm password' with masked characters '*****', and 'Access level' with the text '9999'. At the bottom are two buttons: 'Cancel' and 'Add'.

2. [ユーザー名] ボックスに、オペレータに割り当てる名前を入力します。
3. [パスワード] ボックスに、最大 29 文字までのオペレータ パスワードを入力します。
4. [パスワードの確認] ボックスに、同じパスワードを再度入力します。
5. [アクセス レベル] ボックスに、オペレータのアクセス レベル（最低 0 ～最高 9999）を入力します。

新しいユーザー名をセキュリティ リストに追加して **WindowMaker** または **WindowViewer** を再起動すると、デフォルトのユーザー名はアクセス レベル **0** の **None** に自動的にリセットされるので、**WindowMaker** と **WindowViewer** の両方で [ユーザーの設定] コマンドへのアクセスが防止されます。ただし、**Administrator** アカウトとパスワードは維持され、引き続き使用できます。

オペレータがアプリケーションにログオンすると、保護された機能へのアクセスは、機能にリンクされた内部セキュリティ タグで指定された値に対してそのオペレータのパスワードとアクセス レベルが照合された後で許可されます。

スタンドアロン アプリケーションの場合、管理者特権を持つユーザーだけが **InTouch WindowMaker** でアプリケーションを開くこと、および編集することができます。管理特権を持たないユーザーが **InTouch WindowMaker** を起動しようとする、その処理を続行するには管理特権が必要であることを通知するエラー ダイアログが表示されます。管理特権を持たないユーザーは、マネージド アプリケーション用の IDE から **WindowMaker** を起動できます。

オペレーティング システム ベースのセキュリティの使用

オペレーティング システム ベースの認証方法では、**Windows** オペレーティング システムのアカウント ポリシーの一部が継承され、その他のポリシーは **InTouch HMI** から施行されます。最大有効期間、最低有効期間、最低文字数などのパスワード ポリシーは、オペレーティング システムによって施行されます。

インストール中に使用されたユーザー名は、オペレーティング システムの一部として動作します。これらの基準を施行するには、Windows ドメインに希望するアカウント ポリシーが設定されている必要があります。無操作タイムアウト時間は InTouch HMI で施行されます。

オペレーティング システム ベースの認証方法では、Windows ネットワーク ドメインまたはワークグループに関連付けられているユーザー リストからユーザー名を選択できます。各ユーザー名には、特定のアクティビティに対するユーザーの認証を決定するアクセス レベルが割り当てられます。オペレーティング システムはパスワードを内部で管理するため、InTouch HMI ではアプリケーションをホストするノードに対するパスワードは保存されません。

オペレーティング システム ベースのセキュリティでは、InTouch AddPermission() スクリプト関数を使用して、ユーザー リストおよびそれらのユーザーに対応するアクセス レベルを定義して、管理します。AddPermission() 呼び出しの実行後に作成されるこのリストは、ディスクに書き込まれます。ユーザーの認証の詳細が含まれるファイルは、NAD クライアント ノードにコピーされません。

オペレータは、WindowViewer の [システム] メニューで ([システム] メニューが表示されている場合)、[セキュリティ] から [ログオン] メニュー コマンドを実行することによってアプリケーションにログオンできます。または、内部セキュリティ タグにリンクされているタッチ入力オブジェクトを使用してカスタムのログオン ウィンドウを作成することもできます。

アプリケーションにセキュリティを確立するために使用するコマンドは、WindowMaker および WindowViewer の両方の [セキュリティ] に配置されています。セキュリティ コマンドは、アプリケーションに対するログオンおよびログオフ、パスワードの変更、有効なユーザー名の設定、パスワード、およびアクセス レベルのリストの設定に使用されます。

たとえば、ログオンしているオペレータのアクセス レベルを 2000 より大きく指定して、ウィンドウへのアクセス、オブジェクトの表示などを制御できます。

ArchestrA ベースのセキュリティの使用

ArchestrA セキュリティを使用するようにノードを設定する場合、InTouch HMI はログオンやログオフ操作に対して、Application Server からのメソッドとダイアログ ボックスを使用します。ユーザーは Application Server の Galaxy レポジトリ ノードで設定されます。詳細については、Application Server のマニュアルを参照してください。

ArchestrA セキュリティを使用すると、ユーザーを簡単に定義して、実行を許可された操作を割り当てることができます。ユーザーがオートメーション オブジェクトを使用して実行できる操作の観点から、セキュリティ上の許可を定義します。基本的な手順は、以下のとおりです。

1. セキュリティ モデルを定義します。
2. 保護のセキュリティ モデルに従って、オートメーション オブジェクトを構成します。
3. セキュリティ モデルに従って、ユーザーを定義します。

システム管理者は、対応するユーザー プロファイルを作成して、システム ユーザーを定義します。システム管理者は、セキュリティ モデルで事前定義されたユーザー役割のリストから選択することによって、1 つまたは複数の役割を各ユーザーに割り当てます。

InTouch を ArchestrA ベースのセキュリティと同時に使用した場合、パスワードの最大文字数は 31 です。

InTouchView ユーザーは通常、パスワードベースのログオンによって認証されます。

認証にスマートカードを使用する

スマートカードは、集積回路が内蔵されているポケットサイズのカードです。カードは、プライベートキーおよびパブリックキーの証明書を含むデータを安全に保管します。カードの所有者は個人識別番号（PIN）によって認証され、カード上の特定のデータにのみアクセスできるよう承認されます。

InTouch アプリケーションは、ユーザー認証でスマートカードをサポートするように設定することができます。アプリケーションにユーザー名、パスワード、およびドメインを要求させる代わりに、スマートカード証明書と、関連する PIN 番号を認証に使用できます。また、スマートカードの代わりに、名前、パスワード、およびドメイン名でログオンすることもできます。

ログオンやセキュリティ書き込み/確認書き込みのようなユーザー認証を必要とする操作でも、スマートカード認証を利用することができます。詳細については、[「セキュリティ書き込みおよび確認書き込みの使用」](#)を参照してください。

スマートカード認証の設定

スマートカード認証を設定するには、以下の手順を実行する必要があります。

- InTouch アプリケーションを InTouch OS または ArchestrA OS セキュリティのいずれかを使用するように設定します。ArchestrA セキュリティは、ユーザーベースまたはグループベースのいずれかにすることができます。ArchestrA セキュリティは System Platform IDE を使用して設定します。詳細については、IDE のマニュアルを参照してください。
- WindowViewer コンピュータをネットワークの適切なドメインに参加させます。
- WindowMaker で、InTouch アプリケーション用のスマートカード認証を有効にします。詳細については、[「WindowMaker でスマートカード認証を有効にする」](#)を参照してください。
- スマートカードを、使用するドメインに合わせて設定します。
- WindowViewer コンピュータにカードのドライバをインストールします。スマートカードと専用のドライバはハードウェアに特化しています。スマートカードリーダーのインストールおよび設定の詳細については、リーダーに付属しているマニュアルを参照してください。
- スマートカードリーダーを WindowViewer コンピュータの適切なポートに接続します。手順については、スマートカードに付属するマニュアルを参照してください。
 - 複数のユーザーが関与する確認書き込みの機能を実施するには、複数のスマートカードリーダーが必要になります。
 - ターミナルサーバーと RDP クライアントでスマートカードを使用するには、スマートカード認証を有効にするため、スマートカードリーダーをクライアントシステムに接続しておく必要があります。RDP を使用するターミナルサーバーにスマートカードを接続するには、RDP クライアントの接続設定で、[ローカルデバイスとリソース] の下の [スマートカード] オプションを有効にしてあることを確認してください。

WindowMaker でスマートカード認証を有効にする

スマートカードリーダーを認証に使用するには、WindowMaker でスマートカード認証を有効にしておく必要があります。

スマートカードオプションを設定するには

1. WindowMaker を開きます。
2. [ファイル] メニューの [設定] をクリックし、[セキュリティ] をクリックします。

3. 使用可能なオプション（[なし]、[InTouch]、[OS]、および [Archestra]）からセキュリティ タイプを選択します。
 - [Archestra] をクリックする場合には、事前に IDE を使用して Archestra OS セキュリティ（OS ユーザーベースまたは OS グループベース）を設定しておいてください。
 - [OS] または [Archestra] のセキュリティ タイプを選択した場合、[スマート カードの認証] チェック ボックスが有効になります。スマート カード認証を有効にするには、このチェック ボックスをオンにします。

スマート カードによるログオン

スマート カードを使用して InTouch WindowViewer にログオンすることができます。スマート カードで InTouch アプリケーションにログオンするには、スマート カードによる認証が有効になっているアプリケーションが必要です。

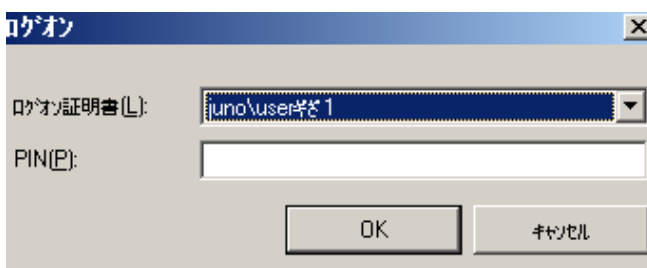
スマート カードは、ドメインで設定された証明書を少なくとも 1 つ含んでいる必要があります。WindowViewer を実行しているコンピュータにスマート カードリーダーを接続する必要があります。使用するスマート カードの PIN を入力することが求められます。

ログオンしようとしたときにリーダーでスマート カードが検出されなかった場合には、代わりにユーザー名とパスワードで認証するように求められます。

セキュリティ 書き込みおよび確認データ書き込みのための認証では、スマート カードを使用することができます。詳細については、「[セキュリティ書き込みおよび確認書き込みの使用](#)」を参照してください。

スマート カードを使用してログオンするには

1. WindowViewer を実行します。
2. スマート カードを挿入します（まだ挿入されていない場合）。
3. [システム] メニューで、[セキュリティ] をポイントして、[ログオン] をクリックします。
[ログオン] ダイアログ ボックスが表示されます。



スマート カードを挿入していた場合には、ログオン証明書（ドメインとユーザー名）がダイアログ ボックスに表示されます。

スマート カードログオン ダイアログ ボックスは、WindowViewer 内のスクリプトから LogonCurrentUser() 関数または PostLogonDialog() 関数を実行した場合にも表示されます。これらの関数は InTouch スクリプトのみで使用可能です。Archestra クライアント スクリプトでは使用できません。

4. [PIN] ボックスに、スマート カードの PIN を入力します。
スマート カードが利用できない場合には、システムはユーザー ID とパスワードでログオンするように要求します。
5. [OK] をクリックします。これで WindowViewer にログオンできます。

注記: スマートカードユーザーとしてログオンしたら、カードをスマートカードリーダーに挿入した状態にしておく必要があります。スマートカードをリーダーから取り外すと、ログオフされます。

セキュリティ書き込みおよび確認書き込みの使用

InTouch アプリケーションを設定して、オペレータが、特定のセキュリティ分類に設定されている Galaxy 属性にデータを書き込むようにすることができます。

- 「セキュリティ書き込み」分類では、書き込み処理を完了させるために、ランタイムのオペレータに資格情報を要求します。
- 「確認書き込み」分類では、2つの署名が要求されます。オペレータは、適切な資格情報を提供すればデータを書き込むことができますが、書き込み処理を完了するには、追加の確認者による承認が要求されます。

セキュリティ書き込みおよび確認書き込みでは、次のことが必要になります。

- Galaxy に対してセキュリティが有効にされている必要があります。
- InTouch アプリケーションに対して Archestra セキュリティが有効にされている必要があります。
- ランタイム オペレータは Galaxy 内で設定された適切な権限を持っている必要があります。
 - オペレータは、セキュリティ書き込みまたは確認書き込みのいずれかを行うには、「操作属性変更可能」の権限を持っている必要があります。
 - 確認書き込みの確認を行うには、確認者は「確認書き込み可能」の権限を持つユーザーである必要があります。

現在誰が InTouch アプリケーションのランタイム ユーザーとしてログインしているかには関わりなく、セキュリティ書き込みまたは確認書き込みでは、常にユーザー認証が必要とされます。オペレータとしてログオンしていない場合でも、セキュリティ書き込みまたは確認書き込みセキュリティ分類の設定された属性を変更することはできます。この操作はログオンしているユーザーのセッションには影響を与えません。

重要: セキュリティ設定が有効になっていて、Application Server バージョン 3.5 に移行された Galaxy の場合、「操作属性変更可能」の権限の設定は、「確認書き込み可能」の属性にコピーされます。Application Server 3.5 移行では、Galaxy の「確認書き込み可能」の権限の設定は、デフォルトで無効にされています。

InTouch Tag Viewer では、ランタイム ユーザーが行えるのは、Application Server 属性への参照を持つ間接タグへの書き込みだけです。

セキュリティ書き込みおよび確認データ書き込みのための認証では、スマートカードを使用することができます。詳細については、「[セキュリティ書き込みおよび確認書き込みの使用](#)」を参照してください。

セキュリティ書き込みの実行

セキュリティ書き込みセキュリティ分類が設定されている Application Server Galaxy 属性の値を変更しようとする場合には、有効なセキュリティ アカウント（ドメイン名、ユーザー名、およびパスワード）またはスマートカードのいずれかを使用して、自分自身を認証する必要があります。スマートカードのオプションは、WindowViewer コンピュータにスマートカードリーダーが接続されている場合にのみ使用できます。

セキュリティ書き込みを行う場合、オペレータは Galaxy 内で「操作属性変更可能」の権限を持っている必要があります。

オペレータのセキュリティ書き込みの認証は、現在ログオンしているユーザーのセッションには影響しません。オペレータがスマート カードでログオンしていた場合には、自分自身を再認証する必要があります。

セキュリティ書き込みを実行するには

1. セキュリティ書き込みセキュリティ分類が設定された属性の値を変更することを試みます。[セキュリティ書き込み] ダイアログボックスが表示されます。スマート カードが利用できない場合には、[モード] ボタンは無効になります。



2. 事前定義された [コメント] リストから選択するか、[コメント] テキスト ボックスにコメントを入力して、書き込み操作に対するコメントを追加します。コメントは 200 文字までです。

SignedWrite() スクリプト関数を使えば、コメントのリストを事前定義できます。または、[コメント] テキスト ボックスに新しいコメントを入力することができます。事前定義されたコメント リストは、SignedWrite() スクリプト関数を使用する場合にのみアクセスできます。

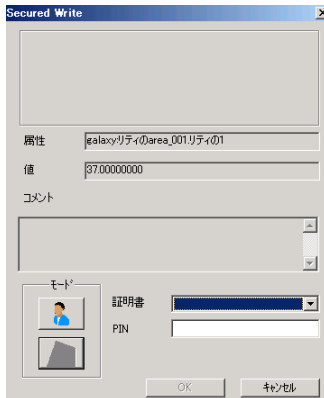
3. ネットワーク ユーザー アカウントを使用して認証する場合には、ユーザー アカウントのオプションが表示されます。

以下の手順を実行します。

- a. [ユーザー名] ボックスに、ユーザー名を入力します。デフォルトでは現在ログオンしているユーザーの名前が表示されます。現在ログオンしているユーザーがいない場合、空白のボックスが表示されます。
- b. [パスワード] ボックスに、ユーザー名に関連付けられているパスワードを入力します。
- c. [ドメイン] ボックスに、ドメイン名を入力します。
- d. [OK] をクリックします。



4. スマート カードを使用して認証する場合には、スマート カードのオプションが表示されます。



スマート カードを使用して認証するには、以下の手順を実行します。

- a. **【証明書】** リストで、自分のスマート カード証明書を選択します。証明書のリストは、ドメイン名／ユーザー名の形式で表示されます。証明書のリストが表示されるには、その時点でコンピュータに接続されているリーダーにスマート カードが挿入されている必要があります。デフォルトでは現在ログオンしているユーザーの証明書が表示されます。**【セキュリティ書き込み】** ダイアログ ボックスが開いているときにカードの抜き差しを行った場合、証明書のリストは自動的に更新されます。
- b. **【PIN】** ボックスに、スマート カードの PIN を入力します。
- c. **【OK】** をクリックします。

スマート カードがある場合でも、名前、パスワード、およびドメイン名で認証を行う場合には、**【モード】** ボタンをクリックします。手順 3 に移動します。

確認書き込みの実行

確認書き込みセキュリティ分類が設定されている **Application Server Galaxy** 属性の値を変更しようとする場合には、有効なセキュリティ アカウント（ドメイン名、ユーザー名、およびパスワード）またはスマート カードのいずれかを使用して、自分自身を認証する必要があります。この書き込みは、別の人によって確認される必要があります。

- 確認書き込みを実行するには、オペレータは「操作属性変更可能」の操作権限を持っている必要があります。
- 確認書き込みの確認を行うには、確認者も「確認書き込み可能」の権限を持つユーザーである必要があります。

オペレータの確認書き込みの認証は、現在ログオンしているユーザーのセッションには影響しません。

スマート カードのオプションは、**WindowViewer** コンピュータにスマート カードリーダーが接続されている場合にのみ使用できます。オペレータとしてログインする場合、確認者としてログインする場合、またはその両方としてログインする場合でもスマート カードを使用できますが、オペレータと確認者は別の人である必要があります。オペレータがスマート カードでログオンしていた場合には、自分自身を再認証する必要があります。

以下のオプションがあります。

- 2 台のスマート カードリーダーと 2 枚のスマート カードを使用します。
- スマート カードリーダーが 1 台しか利用できない場合には、オペレータまたは確認者のために、1 台のスマート カードリーダーと 1 枚のスマート カードを使用することができます。オペレータが

証明書番号と PIN を使用してログオンしている場合には、確認者はユーザー名とパスワードを使用してログオンする必要があります。その逆も可能です。

- オペレータと確認者の両方がユーザー名とパスワードを使用して認証することもできます。

確認書き込みを実行するには

1. **【確認書き込み】** セキュリティ分類が設定された属性の値を変更することを試みます。

【確認書き込み】 ダイアログボックスが表示されます。スマートカードが利用できない場合には、**【モード】** ボタンは無効になります。

2. 事前定義された **【コメント】** リストから選択するか、**【コメント】** テキストボックスに自分自身のコメントを入力して、書き込み操作に対するコメントを追加します。コメントは **200** 文字までです。

事前定義されたコメントリストは、**SignedWrite()** スクリプト関数を使用する場合にのみアクセスできます。

3. ネットワーク ユーザー アカウントを使用して認証する場合には、ユーザー アカウントのオプションが表示されます。

以下の手順を実行します。

- a. **【ユーザー名】** ボックスに、ユーザー名を入力します。デフォルトでは現在ログオンしているユーザーの名前が表示されます。現在ログオンしているユーザーがいない場合、空白のボックスが表示されます。
- b. **【パスワード】** ボックスに、ユーザー名に関連付けられているパスワードを入力します。
- c. **【ドメイン】** ボックスに、ドメイン名を入力します。
- d. **【OK】** をクリックします。
- e. 代わりにスマートカードを使用して認証するには、**【証明書】** ボタンをクリックします。手順 4 に移動します。

4. スマートカードを使用して認証する場合には、スマートカードのオプションが表示されます。

スマート カードを使用して認証するには、以下の手順を実行します。

- a. **【証明書】** リストで、自分のスマート カード証明書を選択します。証明書のリストは、ドメイン名／ユーザー名の形式で表示されます。証明書のリストが表示されるには、その時点でコンピュータに接続されているリーダーにスマート カードが挿入されている必要があります。現在ログオンしているユーザーがスマート カードを使用していた場合、そのユーザーの証明書がデフォルトで表示されます。**【セキュリティ書き込み】** ダイアログ ボックスが開いているときにカードの抜き差しを行った場合、証明書のリストは自動的に更新されます。
- b. **【PIN】** ボックスに、スマート カードの PIN を入力します。
- c. **【OK】** をクリックします。
- d. スマート カードがある場合でも、名前、パスワード、およびドメイン名で認証を行う場合には、**【モード】** ボタンをクリックします。手順 3 に移動します。

【セキュリティ書き込み】 / 【確認書き込み】 ダイアログ ボックスのカスタマイズ



SignedWrite() スクリプト関数を使用すれば、**【セキュリティ書き込み】** または **【確認書き込み】** ダイアログ ボックスの以下の点を設定できます。

- 理由メッセージの表示
- **【コメント】** リストの事前設定



- **【コメント】** テキスト ボックスの編集を許可する

SignedWrite() 関数の詳細、および構文、パラメータ、詳細例などを始めとする使用方法については、『産業用グラフィック エディタ ユーザー ガイド』を参照してください。

SignedWrite() 関数をランタイム時に操作する

SignedWrite() 関数を使えば、セキュリティ書き込みまたは確認書き込みの署名を必要とする属性に、直接値を割り当てることができます。

セキュリティ書き込みまたは確認書き込みセキュリティ分類が設定されている値を設定して、それを変更すると、**【セキュリティ書き込み】** または **【確認書き込み】** ダイアログ ボックスが表示されます。**【セキュリティ書き込み】** または **【確認書き込み】** ダイアログ ボックスに表示される内容は、値の変更方法に応じて変わります。

- 値を SignedWrite() 関数を使用して変更する場合、関数からのパラメータ設定に基づいて、**【セキュリティ書き込み】** または **【確認書き込み】** ダイアログ ボックスにオプションが表示されます。
- 値をユーザー操作によって変更する場合、理由メッセージのエリアにフィールド属性の説明が表示されます（存在する場合）。属性がフィールド属性でない場合、または説明がない場合には、理由メッセージのエリアにその属性が属している ApplicationObject の説明が表示されます。事前定義された **【コメント】** リストは利用できません。

InTouch WindowViewer で属性の値を変更する場合、**【セキュリティ書き込み】** または **【確認書き込み】** ダイアログ ボックスに表示される理由を説明するメッセージを参照することができます。ダイアログ ボックスには、属性の名前と、その属性に書き込まれる新しい値が表示されます。

注意：理由についての説明および事前定義された「コメント」リストとテキストボックスは、InTouch WindowViewer を使用している場合にのみ、「セキュリティ書き込み」または「確認書き込み」ダイアログボックスに表示されます。Tag Viewer では表示されません。

ユーザーの管理と承認レベルの設定

InTouch HMI を使用する必要のあるユーザーのグループに対してセキュリティを実装するには、以下を実行する必要があります。

- 各ユーザーにユーザー名とパスワード認証アカウント情報を割り当てます。
- 各ユーザーに InTouch 承認レベル（アクセス レベル）を割り当てます。

InTouch セキュリティ認証と承認の設定

各オペレータに対して、ユーザー名、パスワード、およびアクセス レベルを割り当てる必要があります。

None と **Administrator** 名前は予約されており、**Administrator** のパスワードのみを変更できます（デフォルトは wonderware です）。アプリケーションのユーザー名を設定した後で、**Administrator** パスワードを変更します。**Administrator** のデフォルトのアクセス レベル（9999）は最高の値であり、「ユーザーの設定」コマンドを含むすべての InTouch 機能にアクセスできます。

「ユーザー入力 - 論理型」ボタンを \$ConfigureUsers タグにリンクさせて、アクセス レベルが 9000 以上の認証されたオペレータが「ユーザーの設定」ダイアログボックスにアクセスして、セキュリティユーザー名リストを編集することを許可できます。オペレータがこのボタンをクリックすると、\$ConfigureUsers タグの値が 1 に設定され、「ユーザーの設定」ダイアログボックスが表示されます。オペレータがダイアログボックスを閉じると、システムによって値は 0 にリセットされます。これは書き込み専用のみを対象としたシステム論理型タグです。

注記：\$ConfigureUsers タグは、セキュリティタイプが InTouch に設定されている場合のみ機能します。ArcheStrA ベースおよびオペレーティングシステムベースのセキュリティでは機能しません。

アプリケーションに対してオペレータのセキュリティを設定するには

1. WindowMaker を開きます。
2. 「ファイル」メニューの「設定」をクリックし、「セキュリティ」をクリックします。
セキュリティ設定画面が表示されます。
3. セキュリティタイプを「InTouch」として選択します。
「ログオン」ダイアログが表示されます。
4. ユーザー名とパスワードを入力します。

注記：InTouch のオペレータを設定できるのは、管理者アカウントを持つユーザーか 9000 以上のアクセス レベルを持つユーザーだけです。

「ユーザーの設定」セクションが有効になります。

5. 「追加」アイコンをクリックします。
「ユーザーを作成」ダイアログが表示されます。

Create user

User name
operator

Password

Confirm password

Access level
9999

Cancel Add

6. セキュリティ アカウントを追加するには、以下の手順を実行します。
 - a. [ユーザー名] ボックスに、オペレータに割り当てる名前を入力します。
 - b. [パスワード] ボックスに、最大 29 文字までのオペレータ パスワードを入力します。
 - c. [アクセス レベル] ボックスに、オペレータのアクセス レベル（0～9999）を入力します。
 - d. [追加] をクリックして、ユーザー名を InTouch セキュリティ リストに追加します。
7. ユーザー名を変更するには、[ユーザーの設定] グリッドからエントリを選択します。
8. 必要な変更を行い、[更新] をクリックします。
9. ユーザー名を変更するには、[ユーザーの設定] グリッドからエントリを選択し、[削除] をクリックします。
10. [OK] をクリックします。

InTouch オペレータ パスワードの変更

オペレータは、WindowMaker または WindowViewer から自分のパスワードを変更できます。

WindowMaker でオペレータ パスワードを変更するには

1. [ファイル] メニューの [設定] をクリックし、[セキュリティ] をクリックします。
セキュリティ設定画面が表示されます。
2. ユーザーのセキュリティ タイプを選択します。
パスワードを変更できるのは管理者または 9000 以上のアクセス レベルを持つユーザーだけなので、サインインが要求されます。
3. ユーザー名とパスワードを入力します。
4. [ユーザーの設定] リストで、パスワードを変更するユーザー アカウントを選択し、[編集] をクリックします。
[ユーザーを編集] 画面が表示されます。

Edit user

User name
Operator01

Password

Confirm password

Access level
458

Cancel Save

5. [パスワード] フィールドにパスワードを入力します。
6. [パスワードの確認] フィールドにパスワードを再度入力して確認します。
7. [保存] をクリックします。

WindowViewer でオペレータ パスワードを変更するには

1. WindowViewer を起動します。
2. [システム] メニューの [セキュリティ] をクリックして、[パスワードの変更] をクリックします。
[パスワードの変更] ダイアログ ボックスが表示されます。

Change Password

Old Password:

OK

New Password:

Cancel

Verify New Password:

3. パスワードを設定します。以下の手順を実行します。
 - [旧パスワード] ボックスに、現在のパスワードを入力します。
 - [新パスワード] ボックスに、新しいパスワードを入力します。
 - [新パスワードの確認] ボックスに、新しいパスワードを再入力します。

4. [OK] をクリックします。

論理値ボタンを使用してオペレータ パスワードを変更するには

WindowViewer で [システム] メニューを表示しない場合、論理値ボタンを作成して、それを \$ChangePassword 内部タグにリンクできます。\$ChangePassword タグの値を 1 に設定すると、[パスワードの変更] ダイアログ ボックスが表示されます。オペレータはそのパスワードを変更できます。オペレータがダイアログ ボックスを閉じると、システムによって \$ChangePassword の値は 0 にリセットされます。これは書き込み専用のみを対象としたシステム論理型タグです。

オペレーティング システム ベースの認証と承認の設定

オペレーティング システム ベースのセキュリティでは、承認済み Windows ユーザー グループのリストから InTouch を認証します。ローカル コンピュータまたは Active Directory サーバーのいずれかで、Windows ユーザー グループを作成します。Windows ユーザーを特定のグループに追加することによって、グループに関連付ける必要があります。ユーザー グループの作成の詳細については、Windows オペレーティング システムのマニュアルを参照してください。

スクリプトの AddPermission() 関数を使用して、InTouch アクセス レベルを Windows グループに割り当てます。AddPermission() 関数は、ユーザーがログオンする準備が整っている場合、WindowViewer で承認済みユーザー グループが認識されるように、アプリケーションの起動時に通常は呼び出されます。

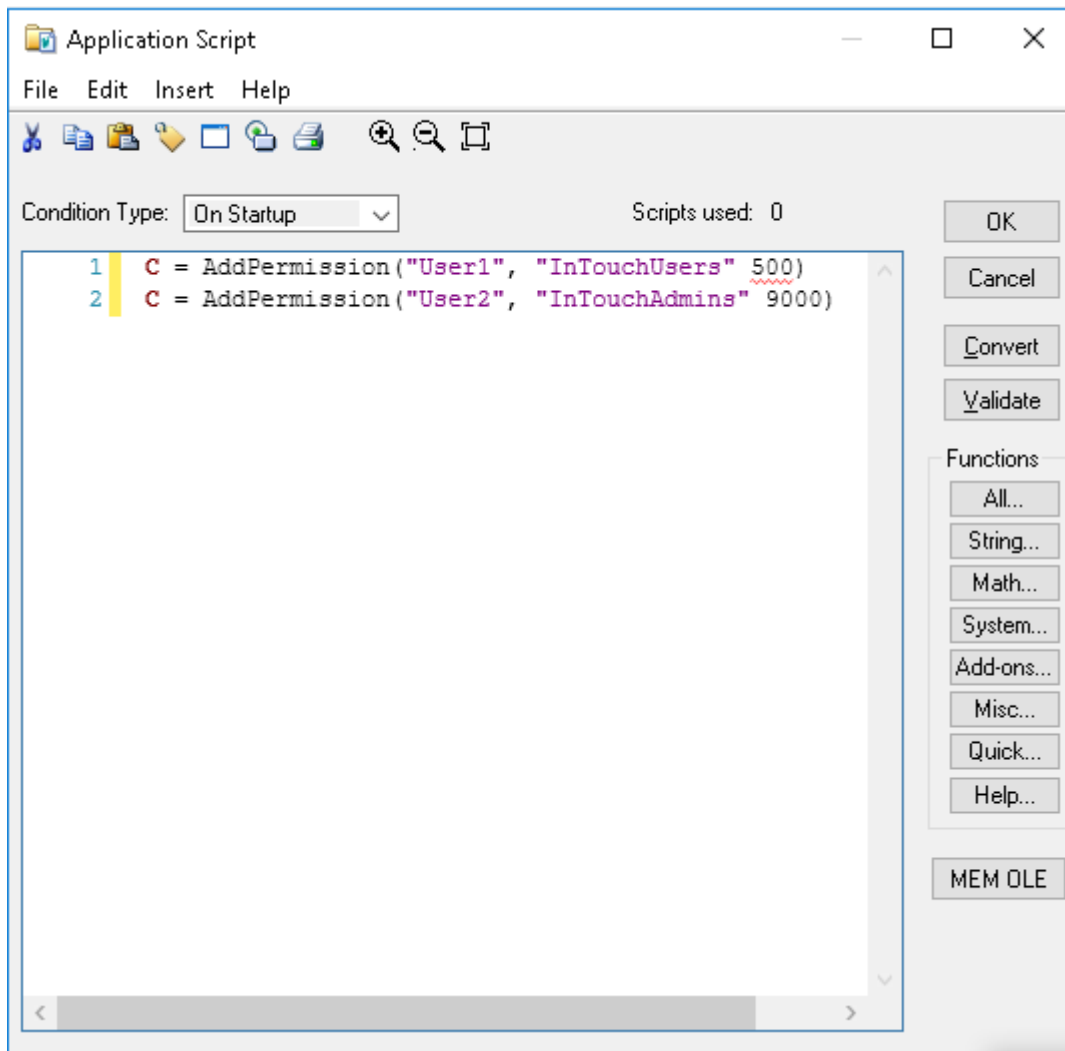
一般に、InTouch アプリケーションを作成した後ですぐに、オペレーティング システム ベースのセキュリティを指定します。

オペレーティング システムの認証と内部 InTouch 承認を使用するように InTouch アプリケーションを設定した後、[システム] メニューの [セキュリティ] で、[パスワードの変更]、[ログオン]、[ユーザーの設定]、および [ログオフ] コマンドは使用できない状態となります。

オペレーティング システム ベースのセキュリティを設定してアクセス レベルを設定するには

1. [スクリプト] ペインで [アプリケーション] をクリックします。

[アプリケーション スクリプト] ダイアログ ボックスが表示されます。



2. [実行条件] リストで [起動時] をクリックします。
3. AddPermission() 関数を使用して、グループ名と対応するアクセス レベルを指定します。
AddPermission() の引数はオペレーティング システム（またはドメイン）、グループ名、およびアクセス レベルです。
4. [OK] をクリックします。

Archestra ベースのセキュリティの設定

Archestra セキュリティ システムはグローバル機能であり、Galaxy データベースの全オブジェクトに適用されます。ユーザーおよび Galaxy のオブジェクトと機能の関係に基づいたシステムです。このシステムは、セキュリティの役割（設定、システム管理、ランタイム権限）およびセキュリティ グループに基づいています。これにより、特定のセキュリティ役割のランタイム権限がオブジェクト レベルで決定されます。セキュリティ システムの設定は、IDE（Integrated Development Environment）で実行され、そのエディタを通して全オブジェクトに適用されます。

Archestra 認証を使用するように InTouch アプリケーションを設定した後で、WindowMaker の [パスワードの変更]、[ログオン]、[ユーザーの設定]、および [ログオフ] コマンドは使用できなくなります。

ArchestraA ベースのセキュリティを設定するには

1. WindowMaker でウィンドウを開きます。
2. [ファイル] メニューの [設定] をクリックし、[セキュリティ] をクリックします。
セキュリティ設定画面が表示されます。
3. セキュリティ タイプのオプションから [ArchestraA] を選択します。

AddPermission() 関数

特定の InTouch アクセス レベルをローカル システムまたはドメインの指定したユーザー グループに割り当てます。AddPermission() 関数が呼び出された後でそのグループに属すユーザーが InTouch HMI にログオンするとき、そのユーザーは指定されたアクセス レベルを受け取ります。

カテゴリ

セキュリティ

構文

```
DiscreteTag=AddPermission( "Domain", "Group", AccessLevel);
```

引数

ドメイン

グループが配置されているドメインまたはローカル コンピュータの名前

Group

Windows ユーザー グループ

AccessLevel

指定したグループに関連付ける InTouch アクセス レベル

備考

オペレーティング システムのセキュリティの場合のみ有効です。この関数が呼び出されると、指定したドメインまたはワークグループの指定したグループの存在を確認します。成功した場合、TRUE が返され、指定したアクセス レベルが後続のユーザー ログオンのグループに関連付けられます。その他すべての場合（つまり、引数に対して無効な値が指定されている場合）、FALSE が返されます。

この関数は通常、アプリケーションの起動時に実行されるように設定されています。現在ログオンしているユーザーには影響は与えられません。AddPermission() が正常に呼び出された後でログオンするユーザーのみがグループに関連付けられているアクセス レベルを受け取ります。

例

```
DiscreteTag=AddPermission( "corporate_hq", "InTouchAdmins", 9000);  
DiscreteTag=AddPermission( "johns01", "InTouchUsers", 5000);
```

Operations Control 接続エクスペリエンス

AddPermission() メソッドが Operations Control 接続エクスペリエンスで受け入れるのは 2 つのパラメータだけです。

- AVEVA Connect グループ
- アクセス レベル

Operations Control 接続エクスペリエンスの AddPermission() のスクリプト関数を以下に示します。

```
DiscreteTag=AddPermission("", "AVEVA Connect group", AccessLevel);
```


ランタイムユーザーが **CONNECT** の複数のグループのメンバーである場合、アクセスレベルは最上位のアクセスレベルを持つグループによって決定されます。

参照項目

PostLogonDialog()、InvisibleVerifyCredentials()、IsAssignedRole()、AttemptInvisibleLogon()、QueryGroupMembership()

ChangePassword() 関数

ログオンしているオペレータがパスワードを変更することを許可する **【パスワードの変更】** ダイアログボックスを表示します。

カテゴリ

セキュリティ

構文

```
[Result=]ChangePassword();
```

戻り値

以下の整数値のいずれかを返します。

0 = **【キャンセル】** がクリックされました。

1 = **【OK】** がクリックされました。

備考

オペレータがタッチ スクリーンを使用している場合、オペレータは英数字キーボードを使用して、新しいパスワードを入力できます。

例

以下のスクリプトはボタンに配置したり、条件スクリプトまたはデータ変化スクリプトから呼び出したりすることができます。

```
Errmsg=ChangePassword();
```

\$AccessLevel システム タグ

現在ログインしているユーザーのアクセス レベルを定義します。

カテゴリ

セキュリティ

使用法

\$AccessLevel

備考

このタグの値は、InTouch HMI で現在ログインしているユーザーのセキュリティ プロファイルに割り当てられているアクセス レベルによって決まります。このプロファイルには、WindowViewer の **【ユーザーの設定】** メニューを使用してアクセスできます。

\$AccessLevel の実際の数字は WindowViewer に対して何の意味も持ちません。ただし、9000 またはそれより大きい値は管理特権を示し、WindowViewer の **【セキュリティ】** メニューを有効にします。システム内のセキュリティをより詳細にカスタマイズするには、\$AccessLevel システム タグを使用します。

データ タイプ

整数型（読み取り専用）

有効値

0 ～ 9999

例

以下のステートメントは、ログオンしているユーザーのアクセス レベルに基づいて表示されるボタンなどのオブジェクトを作成するための表示リンクに対して使用されます。

```
$AccessLevel >= 2000;  
{オブジェクトは、$AccessLevel に基づく式を持つ「無効化」リンクを関連付けることができます。}  
$AccessLevel < 5411;  
IF $AccessLevel <=500 THEN  
Show "アクセス不可"; {アクセスを拒否するポップアップ ウィンドウ}  
ELSE  
Show "アクセス可"; {アクセスを許可するポップアップ ウィンドウ}  
ENDIF;
```

参照項目

\$Operator、\$OperatorEntered、\$PasswordEntered、\$ConfigureUsers

\$ChangePassword システム タグ変数

[パスワードの変更] ダイアログ ボックスを表示します。

カテゴリ

セキュリティ

使用法

\$ChangePassword

備考

[パスワードの変更] ダイアログ ボックスを表示するには、この値を 1 に設定します。ダイアログ ボックスが閉じられると、\$ChangePassword システム タグ変数の値は 0 にリセットされます。このシステム タグ変数を 1 以外の値に設定すると、結果は定義されません。

データタイプ

論理型（書き込み専用）

有効値

1

例

[パスワードの変更] ダイアログ ボックスを開く論理値押しボタンを作成できます。セット オプションが選択された状態の、単一の論理値押しボタン リンクを押しボタンに割り当てます。ボタンが押されると、\$ChangePassword システム タグ変数が 1 に設定され、[パスワードの変更] ダイアログ ボックスが表示されます。

参照項目

\$AccessLevel、\$OperatorEntered、\$PasswordEntered、\$Operator、\$ConfigureUsers

\$ConfigureUsers システム タグ関数

[ユーザーの設定] ダイアログ ボックスを表示します。

カテゴリ

セキュリティ

使用法

\$ConfigureUsers

備考

この関数は InTouch セキュリティでのみ動作します。

[ユーザーの設定] ダイアログ ボックスを開くには、値を **1** に設定します。

ダイアログ ボックスが閉じられると、このシステム タグ変数の値は **0** にリセットされます。このシステム タグ変数を **1** 以外の値に設定すると、結果は定義されません。

このダイアログ ボックスを表示するには、ユーザーは **9000** 以上の \$AccessLevel を持つ必要があります。

データタイプ

論理型（書き込み専用）

有効値

1

例

[ユーザーの設定] ダイアログ ボックスを開く論理値押しボタンを作成できます。セット オプションが選択された状態の、単一の論理値押しボタン リンクを押しボタンに割り当てます。ボタンが押されると、\$ConfigureUsers システム タグ変数が **1** に設定され、[ユーザーの設定] ダイアログ ボックスが表示されます。

参照項目

\$Operator、\$OperatorEntered、\$ChangePassword、\$PasswordEntered、\$AccessLevel

ログオンとログオフ

InTouch アプリケーションに対するログオンやログオフは、アプリケーションを保護するために使用されるセキュリティのタイプによって異なります。

InTouch によりセキュリティが保護されたアプリケーションへのログオン

ログオン情報が間違っていて入力されたり、無効である場合、ログオン試行が失敗したことを示すメッセージが表示されます。

ログオンが正常に行われた場合、\$AccessLevel システム タグは InTouch セキュリティ ユーザー リストでユーザーに関連付けられた事前定義の値に設定されます。

注記: PostLogonDialog() 関数を使用して、[ログオン] ダイアログ ボックスを表示することもできます。詳細については、「[PostLogonDialog\(\) 関数](#)」を参照してください。

アプリケーションにログオンするには

1. [ファイル] メニューの [設定] をクリックし、[セキュリティ] をクリックします。

2. 関連するセキュリティ タイプを選択します。

[ログオン] ダイアログが表示されます。

3. [名前] ボックスに、ユーザー名を入力します。
4. [パスワード] ボックスに、パスワードを入力します。
5. [サインイン] をクリックします。

オペレーティング システムによりセキュリティが確保されたアプリケーションへのログオン

ユーザーが InTouch アプリケーションにログオンする際、以下を要求するダイアログ ボックスが表示されます。

- ユーザー名
- パスワード
- ドメインまたはローカル コンピュータ名

ドメインとユーザー名の組み合わせはオペレーティング システムに渡され、ユーザーのアカウント情報が認証されます。オペレーティング システムのキャッシュを有効または有効にしない状態で、ログオンが試行されます。ユーザーが（ネットワーク障害などの原因で）キャッシュなしでログオンできなくてもキャッシュが有効な状態で以前に認証されている場合、ローカルの InTouch キャッシュからユーザーの氏名とアクセス レベルが取得されます。

セキュリティ チェックがすべて正常に完了すると、ユーザーは InTouch HMI にログオンしていると見なされ、関連するデータ構造（たとえば、\$Operator など）が更新されます。それ以外の場合は、エラー メッセージが表示されます。

オペレータが今まで正常にログオンしたことがなく、ドメインが使用できない場合、ログオン試行は失敗します。InTouch HMI はシステム イベントをエラー ログにログ記録します。

パスワードの期限が切れている場合、エラー メッセージが表示されます。オペレータが [OK] をクリックすると、[パスワードの変更] ダイアログ ボックスが表示され、オペレータはパスワードを変更して、新しいパスワードを使用して再びログオンを試行できます。

ArchestrA によりセキュリティが保護されたアプリケーションへのログオン

ArchestrA によりセキュリティが保護された InTouch アプリケーションには通常、有効なユーザー名とパスワードを入力してログオンし、ログオフします。

InTouch アプリケーションが ArchestrA セキュリティ「なし」に設定されている場合、デフォルトユーザーのログオン アカウント情報が使用され、オペレータはログオンするように要求されません。以下の手順では、システムが「Galaxy」、「OS User based」、または「OS Group based」など ArchestrA 認証モードに対して設定されていると仮定しています。

ログオンするには

1. ArchestrA によりセキュリティが保護された InTouch アプリケーションを起動します。ログイン ダイアログ ボックスが表示されます。
2. 有効なユーザー名とパスワードを入力します。システムで認証されないと、再びログオンするよう促すメッセージが表示されます。

システムによってログオン アカウント情報が認証されると、セキュリティ モデルでユーザーに関連付けられている役割または許可に応じて、今後のすべての操作へのアクセスが許可されます。

InTouch アプリケーションからのログオフ

オペレータは作業が完了した後で、InTouch アプリケーションからログオフします。オペレータによる操作なしで一定時間が経過した場合、オペレータを自動的にログオフするようにアプリケーションを設定することもできます。詳細については、「[無操作タイムアウトの設定](#)」を参照してください。

アプリケーションからログオフするには

1. [ファイル] メニューで、[資格情報] をクリックします。
2. InTouch HMI WindowMaker ユーザー セクションで [サインアウト] をクリックします。

注記: コンフィグレータの [ライセンス モード] タブで [Operations Control 接続エクスペリエンス] が選択されている場合、[資格情報] オプションは使用できません。

カスタム ログオン ウィンドウの作成

[システム] メニューが WindowViewer で表示されていない場合、オペレータがアプリケーションにログオンするためのカスタム ログオン ウィンドウを作成できます。

カスタム ログオン ウィンドウを作成するには

- \$OperatorEntered、\$PasswordEntered、および \$OperatorDomainEntered システム タグ変数をデータ入力オブジェクトにリンクするか、スクリプトで使用して、ユーザー名、パスワード、およびドメイン名を設定します。これらのタグ変数は、書き込み操作のみを対象とした内部メッセージ型タグ変数です。

\$OperatorDomainEntered タグ変数は、セキュリティ モードがオペレーティングシステム ベースの場合のみ必要とされます。それ以外の場合、このタグ変数は無視されます。セキュリティ モードがオペレーティングシステム ベースで、\$OperatorDomainEntered が null の場合、ローカル コンピュータを指していると考えられます。

値が \$PasswordEntered システム タグ変数に書き込まれると、\$OperatorEntered、\$PasswordEntered、および \$OperatorDomainEntered システムタグ変数の値を使用してログオン試行が実行されます。値が \$OperatorEntered または \$OperatorDomainEntered システム タグ変数にのみ書き込まれた場合、ログオンは実行されません。

エントリが有効である場合、\$AccessLevel と \$Operator の内部タグ変数は（セキュリティ ユーザー リストで設定した）定義済みの値に設定されます。

[データ入力 - 論理値] ボタンを \$ConfigureUsers タグ変数にリンクさせて、アクセス レベルが 9000 以上の認証されたオペレータが [ユーザーの設定] ダイアログ ボックスにアクセスして、セキュリティ ユーザー リストを編集することを許可できます。オペレータがこのボタンをクリックすると、\$ConfigureUsers タグ変数の値が 1 に設定され、[ユーザーの設定] ダイアログ ボックスが表示されます。オペレータがダイアログ ボックスを閉じると、システムによって値は 0 にリセットされます。これは書き込み専用のみを対象としたシステム論理型タグ変数です。

注意：\$ConfigureUsers タグ変数は、セキュリティ タイプが InTouch に設定されている場合のみ機能します。ArchestrA ベースのセキュリティに対しては機能しません。

PostLogonDialog() 関数

InTouch の [ログオン] ダイアログ ボックスを表示し、TRUE を返します。

カテゴリ

セキュリティ

構文

```
DiscreteTag=PostLogonDialog();
```

例

```
DiscreteTag=PostLogonDialog();
```

参照項目

InvisibleVerifyCredentials()、AttemptInvisibleLogon()、IsAssignedRole()、QueryGroupMembership()、AddPermission()

LogonCurrentUser() 関数

現在 Windows オペレーティング システムにログオンしているユーザー アカウントで InTouch にログオンします。

- OS セキュリティが設定された InTouch の場合：ユーザーは WindowViewer にログオンします。
- ArchestraA セキュリティが設定された InTouch の場合：ユーザーは Archestra OS ユーザーベースまたは OS グループベース セキュリティのメンバーである必要があります。
- ArchestraA OS ユーザーベースまたは OS グループベースのセキュリティが設定された InTouch およびスマート カードの資格情報で設定されたユーザー アカウントの場合：ユーザーはスマート カードのアカウント情報を使用してログオンします。スマート カードがリーダーから取り外されている場合、ユーザーはログオフされます。

カテゴリ

セキュリティ

構文

```
IntegerResult = LogonCurrentUser();
```

戻り値

ログオンが失敗すると -1 を返し、\$Operator、\$OperatorName、\$OperatorDomain、および \$AccessLevel に割り当てられる値に変更はありません。

備考

この関数は InTouch スクリプトのみで使用可能です。Archestra クライアント スクリプトでは使用できません。

例

```
IntegerResult = LogonCurrentUser();
```

参照項目

PostLogonDialog()、InvisibleVerifyCredentials()、IsAssignedRole()、AttemptInvisibleLogon()、QueryGroupMembership()、AddPermission()

Logoff() 関数

InTouch アプリケーションからユーザーをログオフします。

カテゴリ

セキュリティ（書き込み専用）

構文

```
DiscreteTag = LogOff();
```

備考

現在ログオンしているユーザーをログオフし、現在のユーザーのステータスをデフォルトの **none** に設定します。

例

```
DiscreteTag = LogOff();
```

参照項目

PostLogonDialog()、InvisibleVerifyCredentials()、IsAssignedRole()、AttemptInvisibleLogon()、QueryGroupMembership()、AddPermission()

AttemptInvisibleLogon() 関数

AttemptInvisibleLogon() 関数はスクリプトで使用して、指定したアカウント情報でユーザーを InTouch にログオンできます。ユーザーはパスワードまたはユーザー ID を入力する必要はありません。

カテゴリ

セキュリティ

構文

```
DiscreteTag=AttemptInvisibleLogon( "UserId", "Password", "Domain" );
```

引数

UserId

有効なユーザー アカウント名

パスワード

ユーザーのパスワード

Domain

ユーザーが所属するローカル コンピュータ、ワークグループ、またはドメインの名前。この引数は、現在のセキュリティ モードがオペレーティング システム ベースの場合のみ適用されます。

戻り値

認証が正常に行われると、**TRUE** を返します。それ以外の場合は、**FALSE** を返します。

備考

指定したアカウント情報を使用して、InTouch HMI へのログオンが試行されます。

- ログオン試行が正常に行われると、**TRUE** が返され、**\$OperatorDomain**、**\$OperatorName**、**\$AccessLevel**、および **\$Operator** のシステム タグ変数が適宜更新されます。
- ログオン試行に失敗した場合、**FALSE** が返され、現在ログオンしているユーザー（存在する場合）がそのまま現在のユーザーとなります。

Domain 引数は、オペレーティング システム ベースのセキュリティの場合のみ有効です。Archestra セキュリティ モードが使用中で、Archestra セキュリティが代わりにオペレーティング システム ベースのセキュリティを使用している場合、**UserId** 引数にはドメイン名またはコンピュータ名を使用した完全修飾ユーザー名が含まれます。

例

オペレーティング システム ベースのセキュリティの場合：

```
DiscreteTag=AttemptInvisibleLogon( "UserId", "Password", "Domain" );
```

InTouch ベースまたは ArchestrA ベースのセキュリティの場合：

```
DiscreteTag=AttemptInvisibleLogon("UserId", "Password", "" );
```

参照項目

PostLogonDialog()、InvisibleVerifyCredentials()、IsAssignedRole()、QueryGroupMembership()、AddPermission()

AttemptInvisibleLogonEx() 関数

AttemptInvisibleLogonEx() 関数はスクリプトで使用して、資格情報マネージャに保存された資格情報でユーザーを InTouch にログオンできます。ユーザーはパスワードまたはユーザー ID を入力する必要はありません。

カテゴリ

セキュリティ

構文

```
DiscreteTag=AttemptInvisibleLogonEx("Credential Name");
```

引数

Credential Name

資格情報マネージャに保存された資格情報の名前スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報マネージャから取得されます。

戻り値

認証が正常に行われると、TRUE を返します。それ以外の場合は、FALSE を返します。

備考

資格情報マネージャに保存された資格情報を使用して InTouch HMI へのログオンが試行されます。スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報マネージャから取得されます。

- ログオン試行が正常に行われると、TRUE が返され、\$OperatorDomain、\$OperatorName、\$AccessLevel、および \$Operator のシステム タグ変数が適宜更新されます。
- ログオン試行に失敗した場合、FALSE が返され、現在ログオンしているユーザー（存在する場合）がそのまま現在のユーザーとなります。

Domain 引数は、オペレーティング システム ベースのセキュリティの場合のみ有効です。ArchestrA セキュリティ モードが使用中で、ArchestrA セキュリティが代わりにオペレーティング システム ベースのセキュリティを使用している場合、**UserId** 引数にはドメイン名またはコンピュータ名を使用した完全修飾ユーザー名が含まれます。

例

オペレーティング システム ベースのセキュリティの場合：

```
DiscreteTag=AttemptInvisibleLogonEx("TestCredentialName01");
```

参照項目

PostLogonDialog(), InvisibleVerifyCredentials(), IsAssignedRole(), QueryGroupMembership(), AddPermission()

\$OperatorEntered システム タグ変数

有効なユーザー名を入力するために使用します。

カテゴリ

セキュリティ

使用法

\$OperatorEntered

備考

このタグ変数を使用して、カスタム ログオン ウィンドウを作成できます。タッチ入力オブジェクトや QuickScript をこのタグ変数にリンクして、ログオンのユーザー名を設定できます。

注意 : \$OperatorEntered システム タグ変数が有効である場合、\$AccessLevel や \$Operator のシステム タグ変数は事前定義された値に設定されます。

データタイプ

メッセージ型（書き込み専用）

参照項目

\$AccessLevel、\$Operator、\$PasswordEntered、\$ChangePassword、\$ConfigureUsers

\$PasswordEntered システム タグ変数

有効なパスワードを入力するために使用します。

カテゴリ

セキュリティ

使用法

\$PasswordEntered

備考

\$PasswordEntered システム タグ変数は常に空の文字列として読み取られます。このシステム タグ変数に関連付けられた表示リンクは常に空白です。タグ変数は常に空の文字列を返すため、データ変化スクリプトはこのタグ変数をトリガオフしません。このタグ変数を使用して、カスタム ログオン ウィンドウを作成できます。タッチ入力オブジェクトや QuickScript をこのタグ変数にリンクして、ユーザーのパスワードを設定できます。

注意 : \$PasswordEntered が有効である場合、\$AccessLevel や \$Operator のシステム タグ変数は事前定義された値に設定されます。

データタイプ

メッセージ型（書き込み専用）

参照項目

\$AccessLevel、\$Operator、\$OperatorEntered、\$ChangePassword、\$ConfigureUsers

\$OperatorDomainEntered システム タグ変数

ドメイン名はオペレータによって入力されます。

カテゴリ

セキュリティ

備考

\$PasswordEntered タグ変数に変更されると、ダイアログ ボックスを表示しない状態でログオンが試行されます。このログオン試行では、入力ユーザー名として \$*Entered タグ変数が、ドメイン名として \$OperatorDomainEntered の文字列値が使用されます (現在のモードがオペレーティング システム ベースのセキュリティの場合のみ使用)。モードがオペレーティング システム ベースでない場合、このタグ変数は無視されます。

データタイプ

文字列型

例

```
$OperatorEntered == "john";  
$OperatorDomainEntered == "Corporate_HQ";  
$PasswordEntered == "password";
```

参照項目

\$Operator

オペレータまたはアクセス レベルに基づく機能の有効化と無効化

アプリケーションのセキュリティを実装した後で、\$AccessLevel および \$Operator セキュリティ タグ変数を、ボタン、アニメーションリンクの式、または QuickScript で使用すると、ログオンしているオペレータが特定のアプリケーション機能を実行することが許可されるかどうかを制御できます。

たとえば、ログオンしているユーザーのアクセス レベルに基づいてオブジェクトを表示させるには、以下のステートメントを表示アニメーションリンクの式に使用します。

```
$AccessLevel >= 2000;
```

または、IF ステートメントによってスクリプトをバインドできます。

```
IF $Operator == "DayShift" THEN  
    Show "Control Panel Window";  
    {DayShift Operator に対してのみ実行するその他のライン}  
ENDIF;
```

無効化アニメーションリンクを使用して、内部セキュリティ タグ変数の値に基づいてオブジェクトのタッチリンク機能を制御することもできます。次に例を示します。

Object Disabled -> Discrete Value

Expression:
\$AccessLevel == 0 OR \$Operator == "none"

Disabled State
☒ On ☐ Off

OK Cancel Clear

この式を使用することによって、ログオンしているユーザーがいない場合、オブジェクトやボタンが変更されないように保護されます。

InvisibleVerifyCredentials() 関数

InvisibleVerifyCredentials() 関数は同期 QuickScript で使用して、ユーザーを InTouch HMI にログオンすることなく特定のユーザーのアカウント情報を確認できます。

カテゴリ

セキュリティ

構文

```
AnalogTag=InvisibleVerifyCredentials( "UserId", "Password", "Domain" );
```

引数

UserId

ローカル コンピュータ、ワークグループ、またはドメインの一部である Windows オペレーティング システムのユーザー アカウント名

Password

アカウントのパスワード

Domain

アカウントの Windows ドメイン

備考

指定したユーザー、パスワード、およびドメインが有効である場合、そのユーザーに関連付けられた該当するアクセス レベルが整数として返されます。それ以外の場合は、-1 が返されます。

注意 : InvisibleVerifyCredentials() 関数は同期 QuickScript から実行する必要があります。非同期 QuickScript から実行されると、関数は常に -1 を返します。

この関数によって、現在ログオンしているユーザーは変更されません。Domain 引数は、オペレーティング システム ベースのセキュリティの場合のみ有効です。Archestra セキュリティ モードが使用中で、Archestra セキュリティが代わりにオペレーティング システム ベースのセキュリティを使用している場合、UserId 引数にはドメイン名またはコンピュータ名を使用した完全修飾ユーザー名が含まれる必要があります。

例

```
AnalogTag=InvisibleVerifyCredentials( "john", "Password", "corporate_hq" );
```

参照項目

PostLogonDialog()、AttemptInvisibleLogon()、IsAssignedRole()、QueryGroupMembership()、AddPermission()

InvisibleVerifyCredentialsEx() 関数

InvisibleVerifyCredentialsEx() 関数は同期 QuickScript で使用して、ユーザーを InTouch HMI にログオンすることなく特定のユーザーの資格情報を確認できます。

カテゴリ

セキュリティ

構文

```
AnalogTag=InvisibleVerifyCredentialsEx("Credential Name");
```

引数

Credential Name

資格情報マネージャに保存された資格情報の名前スタンドアロン InTouch アプリケーションの場合、資格情報はアプリケーション マネージャから取得されます。マネージド InTouch アプリケーションの場合、資格情報は Application Server の資格情報マネージャから取得されます。

備考

指定したユーザー、パスワード、およびドメインが有効である場合、そのユーザーに関連付けられた該当するアクセス レベルが整数として返されます。それ以外の場合は、-1 が返されます。

注記: InvisibleVerifyCredentialsEx() 関数は同期 QuickScript から実行する必要があります。非同期 QuickScript から実行されると、関数は常に -1 を返します。

この関数によって、現在ログオンしているユーザーは変更されません。Domain 引数は、オペレーティング システム ベースのセキュリティの場合のみ有効です。Archestra セキュリティ モードが使用中で、Archestra セキュリティが代わりにオペレーティング システム ベースのセキュリティを使用している場合、UserId 引数にはドメイン名またはコンピュータ名を使用した完全修飾ユーザー名が含まれる必要があります。

例

```
AnalogTag=InvisibleVerifyCredentialsEx("john", "Password", "corporate_hq");
```

参照項目

PostLogonDialog()、AttemptInvisibleLogon()、IsAssignedRole()、QueryGroupMembership()、AddPermission()

現在ログオンしているオペレータに関する情報の取得

監査は、すべてのセキュリティ システムでのプライマリ機能です。一組のシステム タグ変数を使用して、InTouch アプリケーションにログオンしているユーザー、ユーザーのログオン元であるドメイン、およびログオンがいつ試行されたかなどの情報を特定できます。

GetAccountStatus() 関数

ユーザーのパスワードが期限切れとなるまでの日数を返します。

カテゴリ

セキュリティ

構文

```
Result=GetAccountStatus(Domain, UserID);
```

引数

Domain

ユーザー アカウントが配置されているドメインまたはローカル コンピュータの名前

UserID

ローカル コンピュータ、ワークグループ、またはドメインの一部である Windows ユーザー アカウント名

戻り値

この関数は以下の値も返します。

Result	説明
-1	アカウント パスワードが期限切れです
-2	アカウント パスワードは期限切れとなりません
-3	アカウントがロックアウトされています
-4	アカウントが無効です
-5	アカウント情報が失敗しました

備考

オペレーティング システム ベースのセキュリティでのみこの関数を使用します。ArchestrA セキュリティ モードではこの関数を使用しないでください。

GetAccountStatus() 関数が ArchestrA セキュリティで使用されると、スクリプトはアカウント情報をドメイン コントローラから直接取得しようと試みます。ArchestrA Galaxy が同じドメインでオペレーティング システム セキュリティを使用している限り、この処理は機能します。

例

```
Status = GetAccountStatus("Corporate_HQ","Operator");
```

IsAssignedRole() 関数

現在ログオンしているユーザーが指定したユーザー役割のメンバであるかどうかを判断します。ArchestrA セキュリティに対してのみ適用されます。

カテゴリ

セキュリティ

構文

```
DiscreteTag=IsAssignedRole( "RoLeName" );
```

引数

RoleName

Application Server ユーザーに関連付けられている役割

備考

ArchestrA セキュリティ モードのみに対して有効で、現在ログオンしているユーザーに適用されます。ユーザーが現在ログオンしており、Galaxy IDE で **RoleName** 役割が割り当てられている場合、TRUE が返されます。それ以外の場合は、FALSE が返されます。

例

```
DiscreteTag=IsAssignedRole( "Administrators" );
```

参照項目

AttemptInvisibleLogon()、PostLogonDialog()、InvisibleVerifyCredentials()、QueryGroupMembership()、AddPermission()

QueryGroupMembership() 関数

現在ログオンしているユーザーが指定したユーザー グループのメンバであるかどうかを判断します。オペレーティング システム セキュリティに対してのみ適用されます。

カテゴリ

セキュリティ

構文

```
DiscreteTag=QueryGroupMembership( "Domain", "Group" );
```

引数

Domain

グループが配置されているドメインまたはローカル コンピュータの名前

Group

グループの名前

備考

オペレーティング システム セキュリティ モードに対してのみ有効で、現在ログオンしているユーザーに適用されます。ユーザーが現在ログオンしており、ドメインに配置されているグループの一部である場合、TRUE が返されます。それ以外の場合は、FALSE が返されます。

QueryGroupMembership() 関数はオペレーティング システム ベースのセキュリティで動作し、ArchestrA セキュリティがオペレーティング システム ベースのセキュリティに設定されている場合のみ ArchestrA セキュリティで動作します。

例

```
DiscreteTag=QueryGroupMembership( "corporate_hq", "InTouchAdmins" );  
DiscreteTag=QueryGroupMembership( "Domain", "Group" );
```

参照項目

PostLogonDialog()、InvisibleVerifyCredentials()、IsAssignedRole()、AttemptInvisibleLogon()、AddPermission()

\$OperatorName システム タグ変数

オペレーティング システム ベースまたは ArchestrA の認証が使用されており、誰かがログオンし、まだログオフしていない場合、オペレータのフル ネームが含まれます。それ以外の場合、ログオンしたユーザーの名前が含まれます（\$Operator タグ変数の内容と同じ）。

カテゴリ

セキュリティ

データタイプ

文字列（読み取り専用）

例

```
IF $OperatorName <> "" THEN  
    {デフォルト値を設定}  
ENDIF;
```

参照項目

\$Operator

\$OperatorDomain システム タグ変数

使用されているセキュリティのタイプに基づいて、異なる値が含まれます。

- オペレーティング システム ベースのセキュリティが選択され、オペレータが正常にログオンしている場合、\$OperatorDomain タグ変数にはログオン時に指定されたドメインまたはノード名が含まれます。
- ArchestrA セキュリティが選択され、ユーザーがログオンしている場合、\$OperatorDoamin には "ArchestrA" が含まれます。
- セキュリティが選択されている場合、\$OperatorDomain タグ変数には文字列 "InTouch" が含まれます。
- 「なし」が選択されている場合、空白の文字列 ("") となります。

カテゴリ

セキュリティ

データ タイプ

文字列型

例

```
IF $OperatorDomain == "PRODUCTION" THEN
    {Allow change to setpoint}
ELSE
    {Change denied}
ENDIF;
```

参照項目

\$Operator

\$Operator システム タグ変数

ログオンしているユーザーのログオン名が含まれます。

カテゴリ

セキュリティ

データタイプ

文字列型

\$VerifiedUserName システム タグ変数

InvisibleVerifyCredentials() 関数の呼び出しが正常に行われ、セキュリティ モードがオペレーティング システム ベースまたは ArchestrA Application Server ベースに設定されている場合、確認されたユーザーのフル ネームが含まれます。呼び出しが失敗した場合、システム タグ変数は null に設定されます。

カテゴリ

セキュリティ

使用法

\$VerifiedUserName

備考

\$VerifiedUserName システム タグ変数が変更されると (InvisibleVerifyCredentials() 関数が呼び出されるときの)、イベントが生成されます。

データタイプ

メッセージ型 (読み取り専用)

有効値

ユーザーのフルネーム

例

```
Tag = InvisibleVerifyCredentials( "john","password", "Plant_Floor");{呼び出しが正常に行われた場合、$VerifiedUserName は "John Smith" に設定され、オペレータ イベントが生成されます。上記の呼び出しが正常に行われなかった場合、$VerifiedUserName は "" に設定されます。}
```

参照項目

InvisibleVerifyCredentials()、\$OperatorName、\$Operator

セキュリティ システム タグと関数の概要

以下の表には、それぞれのセキュリティ モードで利用できるセキュリティ システム タグと関数が示されています。

	InTouch セキュリティ	オペレーティング システム セキュリティ	ArchestrA セキュリティ
\$AccessLevel	使用可能	使用可能	使用可能
\$ChangePassword	使用可能	使用可能	使用可能
\$ConfigureUsers	使用可能	使用不可	使用不可
\$InactivityTimeout	使用可能	使用可能	使用可能
\$InactivityWarning	使用可能	使用可能	使用可能
\$Operator	使用可能	使用可能	使用可能
\$OperatorDomain	使用不可	使用可能	使用可能*
\$OperatorDomainEntered	使用不可	使用可能	使用可能*
\$OperatorEntered	使用可能	使用可能	使用可能
\$OperatorName	使用可能	使用可能	使用可能
\$PasswordEntered	使用可能	使用可能	使用可能
\$VerifiedUserName	使用不可	使用可能	使用可能
AddPermission()	使用不可	使用可能	使用不可
AttemptInvisibleLogon()	使用可能	使用可能	使用可能

	InTouch セキュリティ	オペレーティング システム セキュリティ	ArchestrA セキュリティ
AttemptInvisibleLogonEx()	使用可能	使用可能	使用可能
ChangePassword()	使用可能	使用不可	使用不可
EnableDisableKeys()	使用可能	使用可能	使用可能
GetAccountStatus()	使用不可	使用可能	使用可能*
InvisibleVerifyCredentials()	使用不可	使用可能	使用可能*
InvisibleVerifyCredentialsEx()	使用不可	使用可能	使用可能*
IsAssignedRole()	使用不可	使用不可	使用可能
Logoff()	使用可能	使用可能	使用可能
LogonCurrentUser()	使用不可	使用可能	使用可能*
PostLogonDialog()	使用可能	使用可能	使用可能
QueryGroupMembership()	使用不可	使用可能	使用可能*

* ArchestrA セキュリティが OS ユーザーまたはグループ ベースの場合

管理者以外のユーザーに許可されたアプリケーション マネージャの操作

管理者以外のユーザーが実行できる操作は、管理者に許可された操作のサブセットです。管理者以外のユーザーによる基本的な操作へのアクセスを制限することによってセキュリティの脅威を防止できます。以下の表には、InTouch HMI アプリケーション マネージャの各タブで実行できる操作が一覧表示されています。

[InTouch] タブ

変換処理	管理者以外
新規アプリケーション	不可
WindowMaker の起動	不可
WindowViewer の起動	可
DBLoadDBLoad	不可
DBDumpDBDump	不可
アプリケーションの削除	不可
アプリケーション名の変更	不可
アプリケーションのプロパティ	不可
テンプレートとしてのエクスポートする	不可

変換処理	管理者以外
アプリケーションのインポート	不可
OPC UA サーバー設定	可
アプリケーションの検索	可
IoT 用にエクスポート	可
AVEVA Connect へのアップロード	可
エッジデバイスのユーザーの設定	不可
AVEVA Insight データ ソースとしてのタグ データのパブリッシュ	可
ノードのプロパティ	
アプリ開発 > サービスとしての WindowViewer でのアプリケーションの起動	不可
アプリ開発 > ネットワーク アプリケーション開発の有効化	可
解決策	可
メモリ設定	不可
パフォーマンス	不可
[最新の情報に更新]	可
表示	可
AVEVA Connect へのログイン	可
サムネイルの変更	可
アプリケーション フォルダを開く	可

[Web Client] タブ

変換処理	管理者以外
Web Client の有効化	不可
Web Client の起動	可
Web Client の設定	
グラフィックのリフレッシュ レート	不可
アラームのリフレッシュ レート	不可
Web Client サイト名	不可
ヘッダーを表示	不可

変換処理	管理者以外
ナビゲーションバーを有効化	不可
匿名アクセスを許可	不可
AIM 登録設定	
AIM サーバーを認証サーバーとして使用する	不可
Identity Server	該当なし
ユーザー名	不可
Password	不可
セキュア ゲートウェイ	不可
Web サイトへの産業用グラフィックの埋め込みを許可する	不可

アプリケーション フォルダへの役割ベースのセキュリティの適用

役割ベースのセキュリティをアプリケーション フォルダに適用すると、承認されたユーザーは、アプリケーション フォルダ内のリソースにアクセスしてアプリケーションを編集できます。役割ベースのセキュリティが適用されたアプリケーションはセキュア アプリケーションと呼ばれます。役割ベースのセキュリティが適用されていないアプリケーションは非セキュア アプリケーションと呼ばれます。既存のアプリケーションをインポートまたは変更するときでもアプリケーション フォルダを保護できます。

セキュリティ ユーザー役割の定義

InTouch HMI がインストールされている場合、以下の InTouch ユーザー グループがローカル ユーザー グループに自動的に追加されます。

- **InTouch 開発者** - このグループのユーザーには InTouch アプリケーション ルート フォルダのフル コントロールが付与されます。アプリケーション全体への読み取り/書き込み権限でアプリケーションを編集および管理できます。
- **InTouch オペレータ** - このグループのユーザーには InTouch アプリケーション ルート フォルダの制限付きコントロールが付与されます。アプリケーションを実行できます。ほとんどのファイルへの読み取り専用アクセスがあり、いくつかのファイルには読み取り/書き込み権限が必要になります。

InTouch アプリケーションへのアクセス制限は、'InTouchDevelopers' と 'InTouchOperators' グループのユーザーに対してのみ適用できます。

ドメイン グループの割り当て

ドメイン グループを 'InTouchDevelopers' または 'InTouchOperators' グループに割り当てることができます。ドメイン グループに属するユーザーには、開発者またはオペレータ レベルのアクセスが付与されます。

ローカル グループを 'InTouchDevelopers' または 'InTouchOperators' グループに割り当てることはできません。ユーザーを 'InTouchDevelopers' または 'InTouchOperators' グループに追加することが推奨されます。

InTouch アプリケーション フォルダのセキュリティを有効にするには

InTouch アプリケーション フォルダのセキュリティは、次のいずれかの方法で有効にすることができます。

オプション 1：アプリケーション マネージャからアプリケーション フォルダのセキュリティを有効にする

1. 管理者としてアプリケーション マネージャを起動します。
2. [ツール] メニューの [ツール] タブで [セキュリティ] をクリックします。
[セキュリティ] 画面が表示されます。
3. ['InTouchDevelopers' および 'InTouchOperators' グループのユーザーに対してスタンドアロン InTouch アプリケーションへのアクセスを制限する] チェック ボックスをオンにします。

オプション 2：Configuration.ini ファイルからアプリケーション フォルダのセキュリティを有効にする

1. Configuration.ini ファイル (C:\ProgramData\Wonderware\InTouch) を見つけます。
2. メモ帳などのテキスト エディタを使用してファイルを開きます。
3. セキュリティを有効にするには、SecureApplicationFolder の値を 1 に設定します。
セキュリティを無効にするには、SecureApplicationFolder の値を 0 に設定します。

ユーザーとグループを追加するには

1. 管理者としてアプリケーション マネージャを起動します。
2. [ツール] メニューの [ツール] タブで [セキュリティ] をクリックします。
[セキュリティ] 画面が表示されます。
3. [追加] (+) アイコンをクリックします。
[ユーザー、コンピュータ、またはグループの選択] ダイアログが表示されます。
4. ユーザー名またはグループ名を入力して [OK] をクリックします。

注：ユーザーとグループは、オブジェクト名、タイプ、または位置情報で検索できます。[名前の確認] をクリックして、選択したオブジェクトが AD 名またはグループに解決されることを確認します。

入力したユーザーまたはグループが [ユーザーとグループ] グリッドに表示されます。

5. ユーザー/グループを InTouch 開発者として割り当てるには、[InTouchDevelopers] 列のチェック ボックスをオンにします。
6. ユーザー/グループを InTouch オペレータとして割り当てるには、[InTouchOperators] 列のチェック ボックスをオンにします。
7. [保存] をクリックします。

ユーザーとグループを削除するには

1. [ユーザーとグループ] グリッドで、削除するエントリを選択します。
2. [削除] (-) アイコンをクリックします。
選択したユーザー/グループが削除されます。

マネージド InTouch アプリケーションおよび NAD InTouch アプリケーションのローカル作業ディレクトリの完全性チェック

この機能を使用して、配置されたアプリケーションとローカル作業ディレクトリ内のアプリケーションを比較できます。マネージドおよび NAD アプリケーション用の WindowViewer を起動すると、配置済みアプリケーションとローカル作業ディレクトリ内のアプリケーションの間でファイルタイムスタンプとファイルコンテンツハッシュの比較が実行されます。比較は特定のファイルタイプ（.dlls、.exe、.vedef など）に制限されています。ファイルが一致しない場合、WindowViewer は配置済みアプリケーションをローカル作業ディレクトリにコピーしてアプリケーションを実行します。

完全性チェックを有効にするには

1. アプリケーションマネージャで、[ツール]>[ノードのプロパティ]>[セキュリティ]に移動します。
2. **[Enable local working directory integrity check for Managed and NAD InTouch Applications]**チェックボックスをオンにします。

注：このオプションは**[Limit access to Standalone InTouch applications to users in InTouchDevelopers and InTouchOperators groups]**オプションと連動していません。これらは同じノード上で両方または個別に有効にできます。

3. **[OK]** をクリックします。

Node properties

App development

Resolution

Memory settings

Performance

Security

☒ Enable local working directory integrity check for Managed and NAD InTouch Application.

☒ Standalone InTouch application folder inherits permissions of parent folder.

☐ Limit access to all standalone InTouch applications to users in InTouchDevelopers and InTouchOperators groups.

☐ Limit access to specific standalone InTouch applications to users in InTouchDevelopers and InTouchOperators groups.

Cancel

Ok

注：コピー処理によって WindowViewer の起動に遅延が発生する場合があります。ノード上でネットワークアプリケーション開発（NAD）が有効になっている場合、セキュリティモードのオプションは無効になります。

マネージド InTouch アプリケーションの動作

次のいずれかに当てはまる場合は比較後にファイルが一致しません。

- 配置済みアプリケーションに新しいバージョンがある。

- ローカル作業ディレクトリ内のアプリケーションファイルの内容が変更されている。

どちらのシナリオでも、WindowViewer は配置済みアプリケーションをローカル作業ディレクトリにコピーしてアプリケーションを実行します。これが原因で起動時間の遅延が発生します。

アプリケーションの InTouch.ini ファイルには次の 2 つの追加設定があります。

1. 名前：NewDefaultLocalWorkingDirectory

- 2023R2 以降の新規アプリケーションの場合、値は 1 になります。
- 2023R2 以降の移行済みアプリケーションの場合、値は 0 になります。

2. 名前：VIEWEDMANAGEDDIALOG

[ファイル]>[設定]>[WindowViewer]>[マネージドアプリケーション]に移動すると、値は 1 に設定されます。これは、ローカル作業ディレクトリが更新されたという通知が 1 回のみ表示されるためです。

通知：

新しいマネージドアプリケーションの場合は、既定のローカル作業ディレクトリが%LOCALAPPDATA%\Archestra\ManagedApp に更新されたという内容の通知になります。

移行済みマネージドアプリケーションの場合は、推奨されるローカル作業ディレクトリが%LOCALAPPDATA%\Archestra\ManagedApp であるという内容の通知になります。

既定の作業ディレクトリを復元するには：

マネージドアプリケーションの WindowViewer で、[ファイル]>[設定]>[WindowViewer]>[マネージドアプリケーション]に移動し、ページ下部にある[Restore Default]をクリックします。

[Local working directory]フィールドが次の内容に更新されます。

%LOCALAPPDATA%\Archestra\ManagedApp

NAD InTouch アプリケーションの動作

次のいずれかに当てはまる場合は比較後にファイルが一致しません。

- NAD マスターが更新されている。
- ローカル作業ディレクトリ内のアプリケーションファイルの内容が変更されている。

どちらのシナリオでも、WindowViewer は NAD マスターアプリケーションをローカル作業ディレクトリにコピーしてアプリケーションを実行します。

SP 2023 R2 より前のノードで NAD が以前から有効になっており、SP 2023 R2 以降にアップグレードされた場合、ローカル作業ディレクトリパスは変更されません。新しいファイルパスを推奨する通知が表示されます。

System Platform 2023 R2 以降の新規インストールにおける、NAD の既定のローカル作業ディレクトリパスは以下のとおりです。

C:\Users*USER*\AppData\Local\NAD

付録 AJ InTouch HMI のセキュリティの管理

このセクションでは、InTouch HMI のさまざまなセキュリティ ガイドラインとセキュリティ設定について説明します。

セキュリティに関する全体的な考察

このセクションに記載されている情報をレビューする前に、以下のチェックリストを使用して、ICS および組織に該当するセキュリティ領域が計画でカバーされていることを確認してください。

セキュリティ領域	参照先
ホストへの物理および仮想アクセス	ホストのセキュリティに関する一般的なガイドライン
Windows の最新のセキュリティ修正プログラムの適用	Windows の更新
ウィルスおよびマルウェアからのホストの保護	Scanning the Host
ホスト上のコンテンツへのアクセス	ホスト上のアプリケーションおよびコンテンツの保護
ネットワークの保護	ネットワークの保護
サービスおよびポートの設定	ネットワーク サービスおよびポートの管理
クライアント/サーバー通信のセキュリティ	クライアントとホストの間の通信のセキュリティ
ユーザーおよびグループの管理	認証および承認によるシステムのセキュリティ
非常時の計画	緊急時対応計画

セキュリティ機能に関するヘルプ トピックのリストについては、このセクションの末尾にある表を参照してください。

はじめに

この付録では、AVEVA ソフトウェア製品を産業用制御システム（ICS）として安全に配置する方法の概要について説明します。

この付録の内容は包括的なものではなく、詳細な手順は含まれていません。ここでは、システムのセキュリティのチェックリストとして使用できる基本的な概念と推奨事項を説明します。このガイドに含まれる特定の項目に関する詳細については、該当する項目の公式なドキュメントを参照してください。たとえば、ウィルス防止ソフトウェアの詳細については、そのソフトウェアのドキュメントを参照してください。

サイト ネットワークおよび ICS ソフトウェアのセキュリティに関する AVEVA のアプローチは、以下の原則に基づいています。

- 管理と技術の両面からセキュリティを考える
- IT と ICS の両面からセキュリティ対策を行う
- 複数のネットワーク、システム、およびソフトウェア セキュリティ レイヤーを設計および構築する
- 業界基準、規制基準、および国際標準を考慮に入れる
- 検出および緩和措置によってセキュリティ違反を防止する

これらの原則は、以下のセキュリティの推奨事項を実装することによって実現しています。

- 以下のコンポーネントの活用によるセキュリティ違反の防止:
 - ファイアウォール
 - ネットワーク ベースの侵入防止/検出
 - ホスト ベースの侵入防止/検出
- IT と工場ネットワークの分離
- 明確に定義および周知された変更管理ポリシーの実装（ファイアウォール設定の変更など）。

注記: ICS ソフトウェアのセキュリティに関して、米国商務省から提供されているガイドラインに従うことを強くお勧めします。ドキュメント『産業用制御システム（ICS）のセキュリティガイド』[NIST 特別刊行物 800-82 改訂 2] (<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>) には、ICS、一般的なシステム トポロジ、セキュリティの脅威と脆弱性、およびセキュリティ対策を実装するための推奨事項に関する詳細情報が記載されています。

ホストのセキュリティ

産業用の制御の重要性を考えると、ICS ソフトウェアだけではなく、以下の要素もセキュリティで保護する必要があります。

- ICS ソフトウェアが実行するホスト
- ICS ソフトウェアが接続されているネットワーク
- ICS ソフトウェアで使用されているハードウェア

注記: 「ホスト」は、ICS ソフトウェアがインストールされて実行している Windows コンピュータまたは Windows Embedded デバイスです。

ホストのセキュリティについては、次のような要因を考慮に入れる必要があります。

- ホストへのアクセス
- 最新の Windows 更新の履歴追跡および適用
- ウィルスおよびマルウェアからのホスト コンピュータの保護
- ホスト上のアプリケーションおよびコンテンツの保護

以降のセクションでは、これらの各要素について説明します。

ホストのセキュリティに関する一般的なガイドライン

ここでは、ホストのセキュリティに関するいくつかのガイドラインを示します。

- ICS ソフトウェアのインストールには管理特権のあるアカウントを使用し、ICS ソフトウェアの実行には管理特権のないアカウントを使用します。

- ICS の設定を一部のユーザーにのみ制限します。
- ICS ソフトウェアを **Windows** サービスとして実行することを検討します（このオプションが使用可能な場合）。ICS ソフトウェアがサービスとして実行する場合、低い特権の仮想サービスアカウントとして実行してください。
- ホストを設定して恒久的な場所に設置したら、承認された人員（システム管理者、アプリケーションエンジニア、ランタイム オペレーターなど）だけがホストを使用できるよう、ホストに対する物理的なアクセスとリモートアクセスを制限します。
- 外部ストレージデバイスとしてデータを転送できる物理ポート（USB やメモリ カードなど）は無効にするか、取り外すことを検討してください。

Windows の更新

ホスト上の **Windows** オペレーティング システムが **Microsoft** の「メインストリーム サポート」の対象であること（現在、**Microsoft** が保守し、更新プログラムをリリースしているバージョンであること）を確認してください。古いバージョンの **Windows** は **Microsoft** の「拡張サポート」の対象で、現在、積極的な保守は行われておらず、通知なしで脆弱になる可能性があります。**Windows** のバージョンおよびサポートのレベルの詳細については、**Microsoft Web** サイトの **Windows** ライフサイクルのファクトシートを参照してください。

Microsoft Windows Server Update Services (WSUS) で **Microsoft** 製品の更新を自動化して、ネットワーク上のコンピュータの更新プログラムの管理および配布を行います。**WSUS** の詳細については、**Microsoft Web** サイトの「**Windows Server Update Services**」を参照してください。ホストで **WSUS** サーバーへの接続を確立できない場合は、ホストがプライベートネットワークにある可能性があります。その場合、手動で更新プログラムを適用する手順を作成するか、更新の頻度が低い **LTSC (Long-Term Servicing Channel)** バージョンの **Windows** にオペレーティング システムを変更することを検討してください。

さらに、**AVEVA ICS** ソフトウェアは **Microsoft Update** との互換性に関してテストされていて、その結果は [Security Central サイト](#) で公開されています。このサイトでは、セキュリティ アドバイザリおよびセキュリティ情報も公開されています。

Windows の更新がバックグラウンドで実行している場合、別のソフトウェア プロセスに悪影響が生じる可能性があります。したがって、更新は計画されたシャットダウン期間中にのみ実行するようスケジュールすることが重要です。

☰

All ▾ Search...

🔍 ⓘ

Security Central

Microsoft Security Updates Reports

Product Cyber Security Updates

Policy & Guidelines

Select Product Line:

Wonderware ▾

Archive



Export

Posted	Report ▾	Status	MS Security	Description	Microsoft KB/OS
May 10, 2022	WW22-057	In Testing	Release Notes	Microsoft Office (KB5002199, KB5002204, KB5002187, KB...	View
May 10, 2022	WW22-056	In Testing	Release Notes	SQL Server Cumulative Updates (KB5011644)	View
May 10, 2022	WW22-055	In Testing	Release Notes	Security Only Update for .Net Framework (KB5013839, KB...	View
May 10, 2022	WW22-054	In Testing	Release Notes	Security and Quality Rollup for .Net Framework (KB50139...	View
May 10, 2022	WW22-	In Testing	Release Notes	Monthly Rollup for Windows (KB5014011, KB5014017)	View

ICS ソフトウェアの更新

推奨されるすべてのパッチおよびホットフィックスがホスト上の ICS ソフトウェアにインストールされていることを確認してください。

いくつかの AVEVA アプリケーションでは更新が定期的にリリースされます。これらの更新にはセキュリティ関連の修正が含まれることがあるので、可能な限り早急に適用してください。

注記: AVEVA ソフトウェア製品の Technology Matrix (<https://gcsresource.aveva.com/TechnologyMatrix/Home/Index>) が AVEVA の Global Customer Support (GCS) グループから公開されています。このマトリックスには、ソフトウェア製品の互換性がテストされた Windows オペレーティングシステムのバージョンが一覧表示されています。さらに、ソフトウェアの互換性のあるランタイム、ブラウザ、および仮想環境も一覧表示されています。同じコンピュータにインストールできるその他の製品のリスト、およびこのソフトウェアが通信できるその他の製品のリストも含まれています。

Scanning the Host

ウィルス防止およびマルウェア防止ソフトウェアとファイル整合チェック ソフトウェアの両方を使用して、ホストを定期的にスキャンします。

Windows には、デフォルトで Windows Defender が含まれていますが、多くの種類のマルウェアのスキャンやその他の機能を実行できる追加のソフトウェアをインストールして使用することもできます。その場合は、ソフトウェアの提供元が信頼できる会社であることを確認してください。また、オペレーティングシステムと同様に、ホストがソフトウェアの更新サービスにアクセスできない場合は、手動で更

新プログラムを適用する手順を作成してください。手動更新の手順を作成する場合、ネットワークまたはサイト上のすべてのデバイスを対象としてください。古いデバイスが 1 つ存在するだけでも、ネットワークまたはサイト全体が脆弱になります。

ホスト上のアプリケーションおよびコンテンツの保護

ホスト上のアプリケーションおよびコンテンツを保護するには

- **Windows** ファイアウォールを有効にして、ICS ソフトウェアで使用していないすべてのポートを閉じます。ポートの使用の詳細については、「[ネットワーク サービスおよびポートの管理](#)」を参照してください。
- リモートデスクトップやファイル共有などの **Windows** の機能を無効化し、ゲームやソーシャルメディアなどの不要なプログラムを削除します。
- ホスト上のファイル、データベース、レジストリ、およびその他のリソースへのアクセスを制限します。
- モバイルコンピュータのハードドライブまたは安全な施設内にないコンピュータのハードドライブの場合、**Windows BitLocker** を使用して暗号化します。しかし、**BitLocker** を使用するとコンピュータのパフォーマンスに影響が生じることがあります。
- モバイルデバイスに機密データを格納することは避けて、サーバー クラス ストレージ (SAN) インフラストラクチャを使用することを検討してください。
- アプリケーションが **SQL** サーバーにデータを保存する場合、**Windows** 認証は、**SQL** 認証よりも優れたアプリケーションセキュリティを提供できます。この理由から、**Windows** 認証から **SQL** 認証に切り替えると、**Windows** 認証を使用することを推奨するポップアップダイアログが表示されます。この警告を無視して **SQL** 認証を使用する場合は、**[OK]** をクリックします。同様のメッセージが **OCMC (SMC) Log Viewer** に記録されます。

AVEVA は、**Windows** オペレーティング システム上に構築されたセキュリティを活用して、暗号化キーを保管および管理しています。暗号化キーは、暗号化ストアと呼ばれるローカル ストレージに保管されます。**Windows** の暗号化ストアの詳細については、**Microsoft** のマニュアルの「証明書ストア」(<https://docs.microsoft.com/ja-jp/windows-hardware/drivers/install/certificate-stores>) を参照してください。

データ保護のフェーズ

データは異なる 3 つの層に存在し、各フェーズに対して保護を提供する必要があります。

- 保存時
- 転送中
- 使用中

保存時のデータ

保存時のデータは、現在使用またはアクセスされていないデータです。このようなデータの例には、ハードドライブ、ラップトップ、フラッシュドライブ、**RAID** アレイ、ネットワーク アタッチドストレージ (**NAS**)、ストレージエリア ネットワーク (**SAN**) に保存されたデータ、および何らかの方法でアーカイブ/保存されているデータがあります。保存時のデータ保護は、デバイスやネットワークに保存されている非アクティブなデータを保護することを目的としています。保存時のデータを保護するためには、機密ファイルを保存する前に暗号化するか、ストレージ ドライブそのものを暗号化することができます。ドライブ全体を暗号化する場合、**Windows** のコントロールパネルから呼び出すことのできる **BitLocker** ドライブ暗号化を使用できます。

SCADA および ICS システムのコンテキストでは、保存時のデータには、保存された設定データ、履歴データ、バックアップ、およびその他の静的データが含まれます。保存期間（長期または短期）は、この保存時のデータの分類には影響しません。保存時のデータ保護は、データがこの状態にある場合に該当しますが、この状態は恒久的な状態ではありません。

適切な承認権限を設定して、承認されていないユーザーがデータを表示することを防止する必要があります。個々のデータ要素を別々の場所に保存するなどの手段も検討してください。たとえば、企業で承認されたオフラインバックアップを使用すると、詐欺などの犯罪に利用されるだけの情報が取得されるリスクを削減できます。オフラインバックアップは、ランサムウェアの脅威に対する最良の緩和策です。

転送時のデータ

転送時のデータ（移動時のデータ）は、1つの場所から別の場所にアクティブに移送しているデータです。

SCADA および ICS システムのコンテキストでは、ランタイム ノードへのプロジェクトの配置に加えて、プロセス変数、VTQ データ、および実行中の運用システムの複数のノード間で送信されるその他のデータの送信が含まれます。これには、アラートとアラームが含まれます。

転送時のデータ保護は、移動中のデータを保護することです。以下の例が含まれます。

- ネットワーク内のノードとノードの間
- ネットワークとネットワークの間
- インターネット経由のアクセス
- ローカルストレージデバイスからクラウドストレージデバイスへの送信

データがどこを移動するかに関係なく、移動中のデータのセキュリティは低いと考えられることが多いので、転送時のデータの効果的なデータ保護対策は重要です。セキュリティのベストプラクティスは、HTTPS プロトコルを使用するすべての通信に TLS 1.2 暗号化を使用することです。

使用中のデータ

使用中のデータは、ローカルまたはリモートで現在処理またはアクセスされているデータを指します。一般的に、これは、アプリケーションおよびユーザー（複数の異なるコンピュータ、モバイルデバイス、リモートターミナル、またはその他のデバイスを使用する複数のユーザー）によるアクセスと処理を目的にデータをメモリ（RAM）に配置することになります。使用中のデータは、攻撃に対して特に脆弱です。使用中のデータを保護するには、暗号化、ユーザー認証、および ID 管理を強くお勧めします。

SCADA および ICS システムのコンテキストでは、使用中のデータ（Historian によってアクティブに使用されるデータおよびランタイム ノードに配置されるデータ）は、データベースに適用できます。セキュアな転送チャンネルで保護する必要があります。

SQL Server での暗号化の設定

SQL Server の暗号化接続を有効にすることが推奨されます。SQL Server データベース エンジンのインスタンスの暗号化接続を有効にし、SQL Server 構成マネージャーを使用して証明書を指定します。サーバー コンピュータには、証明書がプロビジョニングされている必要があります。サーバー コンピュータ上で証明書をプロビジョニングするには、証明書を Windows にインポートします。証明書のルート証明機関を信用するようクライアント マシンをセットアップする必要があります。

SQL Server は Transport Layer Security (TLS) を使用して、SQL Server のインスタンスとクライアント アプリケーションの間のネットワークで転送されるデータを暗号化できます。TLS 暗号化はプロトコル レイヤー内で実行され、サポートされるすべての SQL Server クライアントで使用できます。

TLS 暗号化を有効にすると、SQL Server のインスタンスとアプリケーションの間のネットワークで転送されるデータのセキュリティを強化することができます。しかし、TLS を使用して SQL Server とクライアント アプリケーションの間のすべてのトラフィックを暗号化すると、以下に示す追加プロセスが必要になります。

- 接続時に追加のネットワーク ラウンドトリップが必要です。
- アプリケーションから SQL Server のインスタンスに送信されるパケットはクライアント TLS スタックによって暗号化され、サーバー TLS スタックによって解読される必要があります。
- アプリケーションから SQL Server のインスタンスに送信されるパケットはサーバー TLS スタックによって暗号化され、クライアント TLS スタックによって解読される必要があります。

証明書の要件

SQL Server で TLS 証明書をロードするには、以下の条件を満たす必要があります。

- 証明書は、ローカル コンピュータの証明書ストアまたは現在のユーザーの証明書ストアのいずれかに存在する必要があります。
- SQL Server のサービス アカウントに TLS 証明書にアクセスするために必要な権限が必要です。
- 現在のシステム時間が証明書の **【有効期限の開始】** プロパティよりも後で、証明書の **【有効期限の終了】** プロパティよりも前である必要があります。

サーバーへのインストール

SQL Server 2019 (15.x) では、証明書管理は SQL Server 構成マネージャーに統合されています。SQL Server 2019 (15.x) の SQL Server 構成マネージャーは、以前のバージョンの SQL Server で使用できます。

SQL Server 2012 (11.x) から SQL Server 2017 (14.x) を使用し、SQL Server 2019 (15.x) の SQL Server 構成マネージャーが使用できない場合は、以下の手順に従います。

1. **【スタート】** メニューの **【ファイル名を指定して実行】** をクリックし、**【名前】** ボックスに **MMC** と入力して **【OK】** をクリックします。
2. MMC コンソールの **【ファイル】** メニューから **【スナップインの追加と削除】** をクリックします。
3. **【スナップインの追加と削除】** ダイアログ ボックスで **【追加】** をクリックします。
4. **【スタンドアロン スナップインの追加】** ダイアログ ボックスで **【証明書】** をクリックし、**【追加】** をクリックします。
5. **【証明書スナップイン】** ダイアログ ボックスで **【コンピューター アカウント】** をクリックし、**【完了】** をクリックします。
6. **【スナップインの追加とスタンドアロン】** ダイアログ ボックスで **【閉じる】** をクリックします。
7. **【スナップインの追加と削除】** ダイアログ ボックスで **【OK】** をクリックします。
8. 証明書スナップインで **【証明書】** を展開し、**【個人】** を展開します。
9. **【証明書】** を右クリックし、**【すべてのタスク】** をポイントして **【インポート】** の順にクリックします。

10. インポートした証明書を右クリックし、[すべてのタスク] をポイントして [秘密キーの管理] をクリックします。[セキュリティ] ダイアログボックスで、SQL Server のサービス アカウントで使用されているユーザー アカウントに読み取り許可を追加します。
11. 証明書のインポート ウィザードを完了し、証明書をコンピュータに追加して MMC コンソールを閉じます。コンピュータへの証明書の追加の詳細については、Windows のドキュメントを参照してください。

サーバー証明書のエクスポート

サーバー証明書をエクスポートするには:

1. 証明書スナップインから [証明書] / [個人] フォルダで証明書を見つけます。
2. [証明書] を右クリックし、[すべてのタスク] をポイントして [エクスポート] の順にクリックします。
3. 証明書のエクスポート ウィザードを完了し、証明書ファイルを適切な場所に配置します。

サーバーの設定

暗号化接続を強制するようにサーバーを設定します。SQL Server で暗号化を強制する証明書の読み取り許可が SQL Server のサービス アカウントに必要です。特権のないサービス アカウントの場合、読み取り許可を証明書に追加する必要があります。そうしないと、SQL Server サービスを再起動できないことがあります。

サーバーを設定するには:

1. SQL Server 構成マネージャーで [SQL Server ネットワーク構成] を展開し、[<サーバー インスタンス> のプロトコル] を右クリックして [プロパティ] を選択します。
2. [<インスタンス名> のプロトコルのプロパティ] ダイアログボックスの [証明書] タブで、[証明書] ドロップダウンボックスから目的の証明書を選択し、[OK] をクリックします。
3. [フラグ] タブの [暗号化を強制] ボックスで [はい] を選択し、[OK] をクリックしてダイアログボックスを閉じます。
4. SQL Server サービスを再起動します。

クライアントの設定

暗号化接続を要求するようにクライアントを設定します。

1. 元の証明書、またはエクスポートした証明書をクライアント コンピュータにコピーします。
2. クライアント コンピュータで、証明書スナップインを使用して、ルート証明書またはエクスポートした証明書ファイルをインストールします。
3. SQL Server 構成マネージャーを使用して、[SQL Server ネイティブ クライアント構成] を右クリックして [プロパティ] をクリックします。
4. [フラグ] ページの [プロトコル暗号化を強制] ボックスで [はい] をクリックします。

注記: このトピックのセクションは、Microsoft のドキュメントから派生しています。詳細については、Microsoft ドキュメントの「データベース エンジンへの暗号化接続の有効化」を参照してください。

ネットワークの保護

通常、ホスト コンピュータには何らかのネットワーク アクセスがあり、ICS デバイスが完全なスタンドアロンデバイスとして実行するケースは少なくなっています。ホストはネットワークを使用して、その他の ICS コンポーネント（コントローラ、センサー、データベース、リモート クライアントなど）やピアツーピアの関係のその他のホストと通信することがあります。ネットワークを使用して、開発または監視コンピュータから複数の ICS デバイスを管理することも考えられます。

ホストでネットワーク アクセスを使用する場合は、ネットワークへの接続方法を決定する必要があります。近年では、ビジネスや産業ユーザーの間でも有線ネットワーク（Ethernet）からワイヤレス ネットワーク（Wi-Fi）への移行が進んでいます。ネットワークにアクセスするユーザーまたはデバイスを物理的に制御できなくなるので、ICS ネットワークでは Wi-Fi の使用は推奨されません。ワイヤレス アクセス ポイント（WAP）の範囲内にあるコンピュータやデバイスであれば、ネットワークへのアクセスを試みることができます。また、ネットワークがセキュリティで保護されている場合でも、侵入者によってネットワーク トラフィックの傍受および分析が行われて脆弱性が発見される可能性があります。

ICS ネットワークで Wi-Fi を使用する場合は、暗号化（WPA/WPA2 など）、強力なパスワード、および承認済み MAC アドレスのリストなどのアクセス制御機能を WAP で有効にしてください。SSID（サービス セット識別子）のブロードキャストを無効にして Wi-Fi ネットワークを「隠す」ことは避けてください。この方法では、傍受および分析される可能性のあるネットワーク トラフィックの量が増えます。

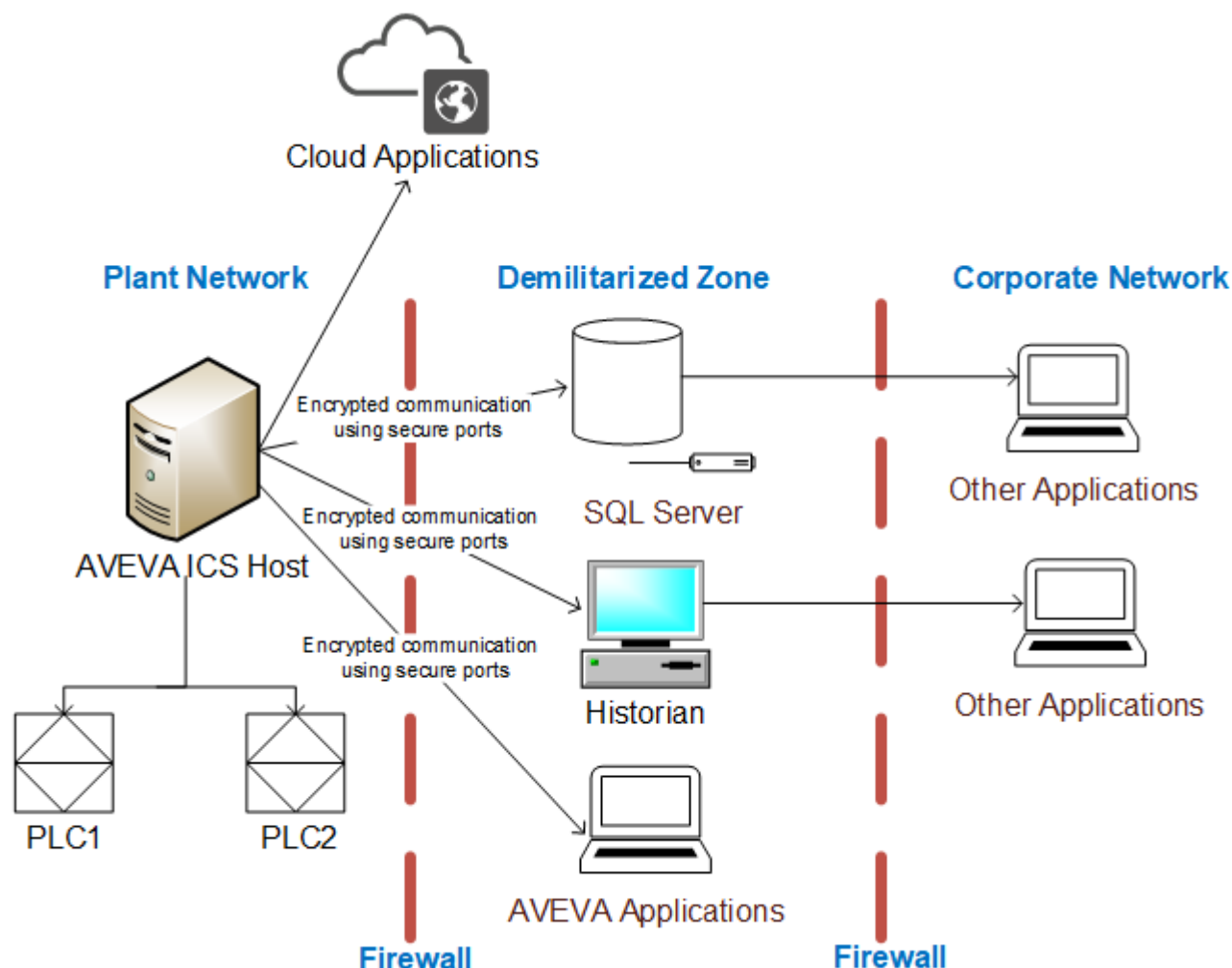
ICS ネットワークのセグメント分割

ICS ネットワーク自体は、その他の企業ネットワークから物理的に分離することや論理的にセグメント分割することができます。物理的に分離されたネットワークが最もセキュアです。ネットワーク ハードウェア、すべてのコンピュータとデバイスが他のネットワークに物理的に接続されていない閉じたネットワークから接続されている場合、侵入者は物理的な場所にアクセスしない限り、ネットワークにアクセスすることはできません。

一方、論理的にセグメント分割されたネットワークはその他の企業ネットワークおよびパブリック ネットワークに物理的に接続されていますが、ICS ネットワーク トラフィックは、さまざまな手段で他のネットワーク トラフィックから分離されています。以下のような手段が使用されます。

- 一方向ゲートウェイを使用する
- 企業ネットワークと ICS ネットワークの間にネットワーク トラフィックが直接移動することを防止するファイアウォールを備えた DMZ（非武装地帯）ネットワーク アーキテクチャを実装する
- 企業ネットワークと ICS ネットワークでの別々の認証メカニズムと資格情報を使用する
- ICS では、複数のレイヤーを含むネットワーク トポロジを使用して、最もクリティカルな通信が最もセキュアで信頼性の高いレイヤーで行われるようにしてください。

サンプルの配置トポロジを以下に示します。



どのような場合でも、パブリック ネットワークから ICS ネットワークとデバイスへの直接アクセスが行われないようにしてください。ICS の一部をアクセス可能にする場合（ブラウザやスマートフォンから Web 対応の HMI 画面を表示する場合など）、必要なトラフィックを ICS ネットワークと外部に接続されたサーバーの間で安全に渡す機能を ICS ソフトウェアに実装してください。

ネットワーク サービスおよびポートの管理

ネットワーク ポートはオペレーティング システムの通信のエンドポイントです。この用語はハードウェア デバイスでも使用されていますが、ソフトウェアの場合は、特定のプロセスまたはサービスの種類を識別する論理構造です。したがって、ネットワーク ポートは、USB、メモリ カード、有線ネットワーク接続などのハードウェア ポートとは概念的に異なります。

コンピュータとデバイスは、さまざまなネットワーク ポートで通信することによって、さまざまなネットワーク サービスに同時にアクセスできます。各ネットワーク サービスまたは通信プロトコルには、ポート番号が関連付けられています。一部のポート番号は国際標準で指定されていて、ユニバーサルに認識されています。その他のポートはソフトウェアによって指定されているので、ほとんどの場合、その他のソフトウェアやサービスと競合する際はソフトウェア設定で変更できます。

ファイアウォールは、これらのネットワーク ポートの通信を許可または拒否することによってネットワーク トラフィックを制御します。ポートが開いている場合、通信が許可されます。ポートが閉じている

場合は通信が拒否されます。ネットワークの各レイヤー（個々のコンピュータやデバイスのオペレーティング システムからネットワーク内のトラフィックを管理するルータやネットワーク間のトラフィックを管理するゲートウェアまで）には、独自のファイアウォールがあります。

ICS ソフトウェアのドキュメントには、ソフトウェアで一般的に使用されるネットワーク ポートの一覧が記載されているはずですが、ICS の本質を考慮すると、通常、この一覧には、サーバーとクライアント間の通信を行う **Web**、電子メール、ファイル転送、外部データベース、デバイス ドライバ、および ICS ソフトウェアが含まれます。ICS で実際に使用されるネットワーク ポートだけを開くようにファイアウォールを設定します。使用されていないすべてのサービスを無効化し、使用されていないすべてのポートを閉じます。

クライアントとホストの間の通信のセキュリティ

ほとんどのサーバー/クライアント アプリケーションと同様に、ICS ソフトウェアは、サーバーとクライアントの間で送信されるメッセージが同じネットワーク上の他のステーションから読み取られることのないよう、サーバーとクライアントの間でのセキュアな通信をサポートする必要があります。これは、ネットワークへの不正アクセスを防止するためのネットワーク セキュリティとは異なることに注意してください。

この種の通信は、サーバー/クライアントのメッセージを暗号化するためにトランスポート層セキュリティ (TLS) を使用するので、「暗号化されたチャネル」と呼ばれることがあります。最新バージョンの標準は TLS 1.3 (2018 年 8 月リリース) ですが、まだ一般的に使用されてはいません。一般的な最新バージョンの標準は TLS 1.2 (2008 年 8 月リリース) です。以前の SSL (Secure Sockets Layer) は今でも古いアプリケーションで使用されていますが、TLS は SSL に置き換わる標準です。

証明書

TLS および SSL は証明書とキーを使用して、サーバーとクライアントの間で送信されるメッセージに電子的に「署名」します。サーバーとクライアントの間の通信が確立すると、接続を行った側から名前、ネットワーク アドレス、組織、物理的位置などを識別する証明書が提示されます。接続された側では、その証明書を受け入れるか拒否するかが決定されます。証明書が受け入れられた場合、同じ証明書で暗号化されたメッセージが受け入れられ、暗号化されたメッセージは関連付けられたキーで解読されます。

このような通信を設定する場合、次のいずれかを選択する必要があります。

- 自己署名証明書を使用する
- パブリック認証機関 (CA) によって署名された証明書を使用する
- ドメイン発行の証明書、または Microsoft Active Directory Certificate Service (AD CS) のようなシステムを使用するプライベート認証局によって署名された証明書を使用する

自己署名証明書は、それを処理するアプリケーションによって発行および署名されます。自己署名証明書は作成と管理が容易ですが、サーバーとクライアントの両方を管理している場合にのみセキュアなので、証明する制御メカニズムは両方にインストールされます。

一方、CA によって署名された証明書は若干異なり、取得するコストがかかりますが、サーバーとクライアントの両方を制御する必要がないので、自己署名証明書よりも柔軟です。CA によって署名された証明書を使用するようサーバーを設定する場合、証明書は CA を認識するクライアントによって受け入れられます。

ドメイン発行の証明書は、一般的に IT 部門によって管理される内部証明書です。この証明書は、Active Directory 証明機関によって発行および検証されます。ドメイン発行の証明書は無料ですぐに発行できます。

CA によって署名された証明書およびドメイン発行の証明書は定期的な間隔で更新する必要があります。

ICS ソフトウェアで暗号化されたチャネルの機能を有効にして自己署名証明書を管理する方法の詳細については、該当するソフトウェアのドキュメントを参照してください。しかし、CA によって署名された証明書を取得してその他の証明書を署名する手順の説明は、一般的な ICS ソフトウェア ドキュメントには含まれていません。

注: 一般的に、暗号化された通信と暗号化されていない通信は、それぞれ別のネットワーク ポートを使用します。

クラウド ベースのシステム

ICS ソフトウェアがクラウド ベースのソリューションにアクセスする可能性や、クラウド上でホストされている可能性を考慮する必要があります。クラウド ベースのアクセスおよびホスティングに関連するリスクを軽減することが重要です。

クラウド ベースのソリューションへのアクセス

現在、多くの AVEVA アプリケーションがクラウドを介して提供されているので、これらのアプリケーションに ICS ソフトウェアを接続する必要がある場合があります。クラウド ベースのアプリケーションにアクセスする際の主なリスクとして不正アクセスが挙げられます。ICS ソフトウェアをクラウドに接続する場合、セキュアな方法で接続する必要があります。また、TLS (トランスポート層セキュリティ) などのセキュアなプロトコルを使用する必要があります。

データの整合性を常に維持することが重要です。データ分類を使用して、機密のデータと公開できるデータを識別します。格納および転送されるデータを保護するために、コンピュータ、ストレージ、およびネットワーキングをセキュリティで保護します。クラウド サービス プロバイダ (CSP) と協力して、ユーザーの設定、アクセス レベルの割り当て、およびアクセスの監視と制御を行います。CSP のビルが物理的に安全であること、および不正アクセスから保護されていることを確認してください。

クラウド ベースの ICS ソフトウェア

クラウド プロバイダによる ICS ソフトウェアのホスティングには柔軟性、スケーラビリティ、および可用性などのメリットがありますが、組織の評判の失墜につながる可能性のあるハッキングなどのセキュリティ リスクが伴います。したがって、ICS ソフトウェアをクラウド上で提供する前にセキュリティ戦略を実装することが重要です。クラウド上の ICS ソフトウェアを保護するには、以下の点を考慮する必要があります。

- 認証、監視、およびサポートのメカニズムを実装してアクセスポイントを保護する。
- クラウド ベースの一元的なセキュリティ対策を実装する (TLS を使用した通信の暗号化など)。

注記: NIST サイバーセキュリティ フレームワーク (<https://www.nist.gov/cyberframework>) で追加情報を参照することが推奨されます。

認証および承認によるシステムのセキュリティ

一般的に、ICS ソフトウェアは多くのシステムで構成されていて、各システムには、さまざまなユーザー (エンジニア、オペレータ、マネージャなど) がアクセスします。各タイプのユーザーに必要なアクセスのレベルは異なるので、システムをセキュリティで保護するには、ユーザー認証および承認を管理する必要があります。

認証

認証は、ユーザー／システムの ID を検証するプロセスです。認証は以下の方法で管理できます。

- ICS ソフトウェア（アプリケーションアカウントを使用します）
- Windows アカウント（単一のコンピュータのローカルアカウント）
- 認証システム（詳細については次のセクションを参照してください）

ICS ソフトウェアではユーザーおよび役割を管理できますが、従業員と役割が変化した場合、多くのユーザーアカウントを管理する作業は面倒で複雑になることがあります。そのため、Windows アカウントを使用することが一般的に推奨されます。

認証システム

Active Directory や LDAP（Lightweight Directory Access Protocol）などの認証システム（認証サーバー）は、すべてのシステムアカウントおよび個々のユーザーアカウントを一元管理するためのリポジトリです。認証プロトコルは、認証を要求するユーザーまたはサーバーと認証サーバーの間のすべての通信で使用されます。

認証システムを使用するとスケーラビリティが向上しますが、運用のサイズと複雑さに応じて以下の要因を考慮する必要があります。

- 認証サーバーには高度なセキュリティが必要です。
- 認証サーバーシステムでは、すべてのシステムアカウントを管理するための単一のシステムが作成されます。したがって、認証サーバーシステムは常時アクセス可能である必要があります。非常時に中断を最小限に抑えるために、冗長性を考慮に入れる必要があります。
- ユーザー資格情報のキャッシュは、最近認証されたユーザーだけに制限します。
- 認証プロトコルをサポートするネットワークは正常な認証を行うために高い信頼性とセキュリティが必要です。

PingID などの追加アプリケーションを使用して 2 要素認証を実装することも検討してください。

承認

承認は、認証済みのユーザー、システム（HMI、現場デバイス、および SCADA サーバー）、およびネットワーク（リモートサイトの LAN）にアクセスルールを適用することによって、ユーザーに正しいレベルの権限を提供するプロセスです。

Windows でのユーザーとグループの管理

セキュリティを設定する際、以下の点を考慮してください。

- 単一のアプリケーションにローカルな設定を行う。
- 複数のアプリケーションで設定を共有する。
- 設定はネットワークドメインの一部として管理する（Active Directory を使用するなど）。一般的に、このオプションでは、ネットワーク、ホスト、および ICS ソフトウェアで同じユーザーアカウントを使用できます。Active Directory を使用すると、以下のようなメリットがあります。
 - ユーザーおよびグループデータの一元化されたリポジトリとして、セキュリティポリシーおよび手順を効果的に実装できます。
 - ユーザーが識別および認証された後にすべてのネットワークリソースへのアクセスの単一ポイントが提供されます。

ユーザーとグループを管理するには:

- 最初に各グループの特定の役割を定義し、次に役割に適したグループ権限を設定します。
- グループは重複させることが可能ですが、グループは明確に分離して、必要に応じてユーザーを複数のグループに割り当てることが推奨されます。
- ICS ソフトウェアのデフォルトユーザー（guest など）のパスワードを設定または変更します。
- 厳密なパスワードポリシーを定義して、ユーザーに強力なパスワードを作成することを強制します。定期的なパスワード更新を必須とします。

ICS ソフトウェアでのユーザーおよびグループの管理

ICS ソフトウェアには、ソフトウェアを使用できるユーザーおよびユーザーの権限を制御するセキュリティ システムが組み込まれています。

各ユーザーには、ICS システム内で許可される操作を決定する権限が割り当てられます。権限は、アカウント ベース、または役割に基づくグループ ベースで管理できます。管理が大幅に簡素化されるので、グループ（役割）ベースのアクセス制御が推奨されます。ユーザーの役割は組織のニーズに応じて変更することができ、必要な場合は 1 人のユーザーに複数の役割を割り当てることができます。

各ユーザーには、一意のユーザー名と強力なパスワードを含むユーザー アカウントが設定されます。ユーザー アカウントは、1 つまたは複数のグループに割り当てることができます。

アカウントには、そのユーザーの職務を遂行するために必要な最低の権限を割り当てます。Windows 管理者権限のあるアカウントは最小限にして、ソフトウェアのインストールと構成にのみ使用します。同様に、SQL Server のシステム管理者権限のあるアカウントも最小限にして、ソフトウェアのインストールと構成にのみ使用します。

ほとんどの場合、ICS ソフトウェアでは、製品内の役割を Windows グループに関連付けることができます。役割の定義と割り当てを行う際は、以下の点を考慮してください。

- 役割は、職務に必要な最低の権限を割り当てるように定義します。
- 権限の割り当てをシンプルにするために、役割は単一の目的に制限します。
- ユーザーには複数の役割を割り当てることができます。

緊急時対応計画

非常事態を引き起こすようなインシデントを想定に入れる必要があります。損失を最小限に抑えてシステムを保護するには、インシデントをすばやく検出し、タイムリーな方法で対応できるような戦略を作成することが重要です。火災や洪水、そしてハードウェアやソフトウェア コンポーネントの障害などのインシデントから生じる非常事態を考慮する必要があります。ランサムウェアなどのサイバー攻撃も考慮に入れる必要があります。

さまざまな障害や事態に対処できるように緊急時対応計画を策定することが推奨されます。従業員のトレーニングを行い、緊急時対応計画に内容を周知します。

緊急時対応計画の一環として、中央のサイトから物理的に分離され、レプリケーション能力を備えたサイトを確立することが重要です。この方法では、中央のサイトで火災や洪水などの災害が発生した場合に運用システムの整合性が保証されます。レプリケーション能力には別のセットの同一ハードウェアに加えて、ソフトウェア設定とキー状態情報を中央のサイトから復旧サイトに定期的にコピーする機能が含まれます。各復旧シナリオは一意なので、通信機器、ハードウェア、およびソフトウェアの設定に関してシステム統合のエキスパートに相談することが重要です。

システムに格納されたデータの保護も最重要項目です。フルバックアップおよび増分バックアップを定期的に行う必要があります。バックアップは、バックアップデータからの復旧テストを行って検証する必要があります。ランサムウェアなどのサイバー攻撃から保護するために、バックアップはオフラインで保存する必要があります。

また、緊急時対応計画に似た事業継続計画および障害復旧計画も必要です。これらの計画については、以降のセクションで簡単に説明します。

監査とログ

ICS ソフトウェアのセキュリティ実装の一環として、さまざまなシステムおよびネットワークでのアクティビティを監査および記録（ログ）することが重要です。

監査とログへの記録を行うことによって、ICS の現在の状態に関する情報を把握し、予期した通りにシステムが機能していることを確認できます。何らかのアクティビティが発生した場合、アクティビティログを使用してコンピュータ、ユーザー、またはネットワークのアクティビティの原因を追跡できます。監査とログは、問題のトラブルシューティングでも役立ちます。

クラウドベースのソリューションに接続する場合、すべての仮想マシン（VM）を監査してデータの整合性を確認してください。

事業継続計画

事業継続計画は、システムの中断が発生した場合に運用の維持または再確立を行うための戦略を策定するものです。中断は、自然災害（水害や地震など）、意図的または偶発的な人的イベント（放火、オペレータのミス、停電など）、またはシステム障害によって発生することがあります。

中断によって生じる潜在的な ICS アプリケーションの障害の時間に応じて、短期的な障害に対処するための運用回復計画と長期的な障害に対処する障害復旧計画を作成する必要があります。リスクレベルの高いデータ収集および制御システムが存在する運用サイトに物理的なセキュリティを導入することも重要です。事業継続計画では、システムのシステムおよびデータ復旧手順を規定する必要があります。復旧手順を文書化した後、復旧手順をテストする必要があります。システム設定データおよび製品または生産データのバックアップの検証も考慮に入れる必要があります。手順は定期的にレビューする必要があります。

クラウドベースのソリューションにアクセスする場合、システムが常時使用可能であることを確認してください。災害が発生した場合、新しい物理的場所にサービスを切り替えてサービスの提供を継続する必要があります。

障害復旧計画

障害復旧計画（DRP）は、障害が発生した場合に IT インフラストラクチャを保護および復旧する手順です。これには、障害の前、最中、そして後の手順が含まれます。障害は、自然、環境、または人的（意図的または偶発的）な原因で発生することがあります。

DRP は ICS の継続的な可用性に不可欠で、以下の要素を含む必要があります。

- DRP を開始するとき（イベントの期間と重大性に基づきます）。
- 外部接続が確立されるまで ICS を手動で運用するための詳細な方法。
- 各手順の責任者。
- データを安全にバックアップおよび復元するためのプロセス。以下の要素をカバーする必要があります。

- 冗長性を構築するための要件。
- ファイルのバックアップ手順。
- バックアップの頻度。
- フルバックアップおよび増分バックアップのストレージメカニズム。
- インストールメディア、ライセンスキー、および設定情報の安全なストレージ。
- バックアップを実行、テスト、保守、および復元する担当者のリスト。
- ICS への物理および仮想アクセスを持つ担当者のリスト。
- ICS のコンポーネントに関する詳細な設定情報。
- DRP のテストスケジュール。

まとめ

セキュリティの問題は ICS ソフトウェアおよびインフラストラクチャに対する深刻な脅威となります。したがって、すべての組織は以下の点に留意する必要があります。

- セキュリティの問題を防止するための予防措置を取る。
- 潜在的な問題を識別する。
- 問題が発生した場合はタイムリーな方法で検出する。
- 問題に対処し、サービスの中断を最小限に抑えると共に最大限の可用性を保証する。

具体的には以下のような対策を取ります。

- コンピュータとネットワークをセキュリティで保護する。
- ユーザーとグループを認証および承認する。
- 不都合なイベントや予期しないイベントが発生した場合に回復できるよう非常時計画を実装する。

追加の詳細および推奨事項については、ドキュメント『産業用制御システム（ICS）のセキュリティガイド』[NIST 特別刊行物 800-82 改訂 2] (<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>) を参照してください。

InTouch HMI のセキュリティ設定

InTouch HMI で設定する必要のあるセキュリティ領域、および該当する手順を示すセクションの詳細を以下の表に示します。

セキュリティ領域	このガイド内のトピック	サマリ
ホスト上のアプリケーションおよびコンテンツの保護	無操作タイムアウトの設定	操作を行っていないオペレータを WindowViewer アプリケーションから自動的にログオフするように InTouch を設定します。

セキュリティ領域	このガイド内のトピック	サマリ
認証および承認によるシステムのセキュリティ	認証ベースおよび承認ベースのセキュリティ	ユーザーは InTouch アプリケーションを使用する前に認証を受ける必要があります。 InTouch では、認証されたユーザーが特定の機能を使用できるかどうかを検証されます。
	仮想アカウントの使用	仮想アカウントを使用すると、アラーム関数にアクセスする際のセキュリティを強化することができます。『InTouch HMI アラームとイベント ガイド』の「 仮想アカウントの使用 」を参照してください。
ホスト上のアプリケーションおよびコンテンツの保護	システム キーのロック	InTouch アプリケーションを実行しているコンピュータでシステム キーを無効にすることによって、標準の Windows 機能へのオペレータ アクセスを制限します。
認証および承認によるシステムのセキュリティ		
ICS ソフトウェアでのユーザーとグループの管理	InTouch ベースのセキュリティの使用	対象となる機能を内部タグにリンクすることによって、オペレータが実行できる機能を制限します。
Windows でのユーザーとグループの管理	オペレーティング システム ベースのセキュリティの使用	Windows オペレーティング システムからユーザー/グループ アカウント ポリシーを継承します。
アプリケーションデータ フォルダの保護	アプリケーション フォルダへの役割ベースのセキュリティの適用	ユーザーの役割とユーザー グループを定義して、アプリケーションデータ フォルダへのアクセスを管理できます。