Roboticsware



Factory Automation シリーズ

アクションマニュアル

RWMN-5-21-920

ようこそ **4**
本マニュアルについて 4
1章　アクションの概要 **5**
アクションの位置付けと目的 5
アクションの動作 6
アクション一覧 7
各パッケージで利用可能なアクション 8
2章　アクションリファレンス **9**
ヒストリカルデータサーバアクション 9
ロガーアクション 13
スクリプト Ver1 アクション 19
スクリプト Ver2 アクション 21
ブリッジアクション 25
タグレポートアクション 26
バッチ転送（タグ→テキスト）アクション 28
バッチ転送（テキスト→タグ）アクション 31
バッチ転送（タグ→ODBC）アクション 32
バッチ転送（ODBC→タグ）アクション 35
メール送信アクション 38
ファイル操作アクション 44
ファイルバックアップアクション 46
FTP 転送アクション 47
SQL 実行（ODBC）アクション 48
ダイアルアップアクション 49
外部アプリケーション実行アクション 50
スクリーンショットアクション 51
シャットダウンアクション 52
時刻同期アクション 53
サマリアクション（CSV/ODBC） 54
日報月報年報アクション 55
アラームサーバーアクション 61
ファイルサーバーアクション 65
クエリーサーバーアクション 67
3章　スクリプト VER2 アクションについて **68**
概要 68
設定手順 68
3－1　基本的なスクリプトの実行方法について（SAMPLE01） 70
概要 70
解説 71
動作確認 74
3－2　スクリプト VER2 の DB 連携について（SAMPLE02） 75
概要 75
解説 76
動作確認 78

３－３　スクリプト VER2 でヒストリカルデータを取得（SAMPLE03）..... 79
概要..... 79
解説..... 79
動作確認..... 82

４章　実践編..... 83

４－１　定周期で計算を行う..... 84
４－２　ＰＬＣ間のデータ共有（データ転送）..... 86
４－３　WEB 経由での簡易モニタリング..... 88
４－４　PLC の大量データを一括出力する（テキスト出力）..... 95
４－５　PLC の大量データを一括出力する（DB 出力）..... 98
４－６　PLC へ大量データの一括書き込みを行う（テキスト経由）..... 101
４－７　PLC へ大量データの一括書き込みを行う（DB 経由）..... 104
４－８　任意のタイミングでメール送信を行う..... 108
４－９　アラームメール通知..... 111
４－10　ファイルを別のフォルダにコピーする..... 114
４－11　ファイルバックアップ..... 116
４－12　ファイルを FTP 転送する..... 118
４－13　SQL を実行して DB のレコードを追加/削除する..... 120
４－14　ダイアルアップ接続／切断を行う..... 123
４－15　外部アプリケーションの実行..... 127
４－16　画面全体のスクリーンショットを取得..... 131
４－17　SERVER アプリケーションを終了させる..... 133
４－18　PC と PLC の時刻を同期させる..... 135

APPENDIX：　アクション共通の設定..... 138

ODBC 設定..... 138
名前フォーマット..... 140
相対時刻フォーマット..... 141

ようこそ

本マニュアルについて

本書「アクションマニュアル」は、本パッケージを使用する上での基礎となるものです。
ユーザーズマニュアルと併せてお読みいただくことを推奨いたします。

1章　アクションの概要

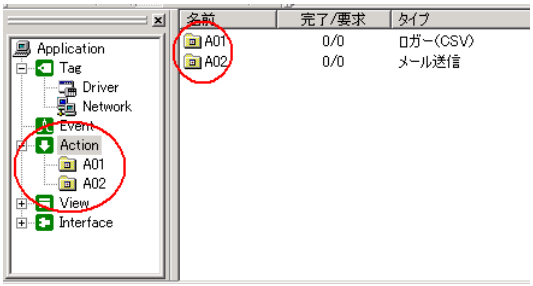
アクションの位置付けと目的

本パッケージは、タグ、イベント、アクション、ビュー、インターフェースの５つの大きな概念から成り立ちます。本マニュアルは「アクション」について記述されています。

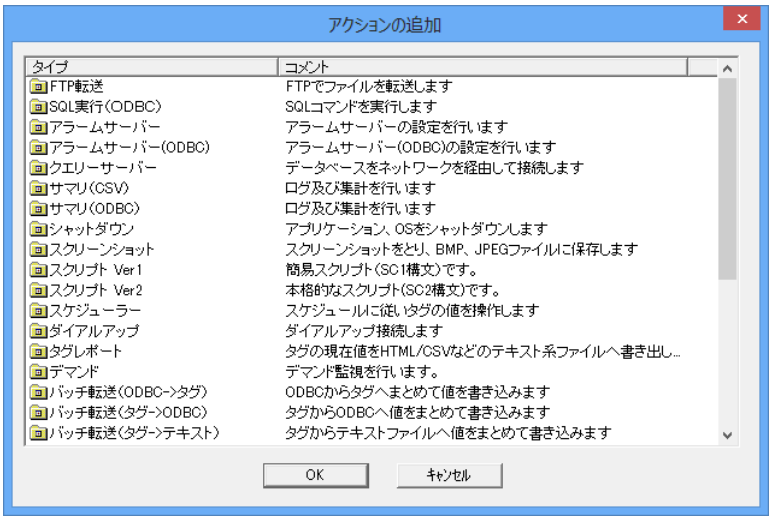
	タグ	PLC との通信の設定です。
	イベント	アクションを起動するトリガの設定です。
	アクション	ロガー、メールなどのアクションの設定です。
	ビュー	タグリスト、タグモニタなどのビュー（画面）の設定です。
	インターフェース	OPC/DDE/IPLink などの、外部アプリケーションから接続するためのインターフェースの設定です。

ロギング、アラーム監視、レポート集計（日報・月報・年報）、メール送信などの各種機能は、アクションによって実行されます。そして、ほとんどのアクションは「イベント」をトリガとして実行されます。

アクションの追加は、画面左側ツリーの「アクション」アイコンから行います。

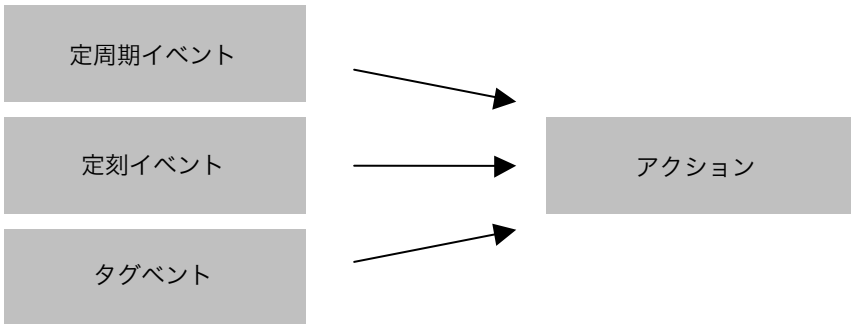


利用可能なアクションの種類は、パッケージにより異なります。

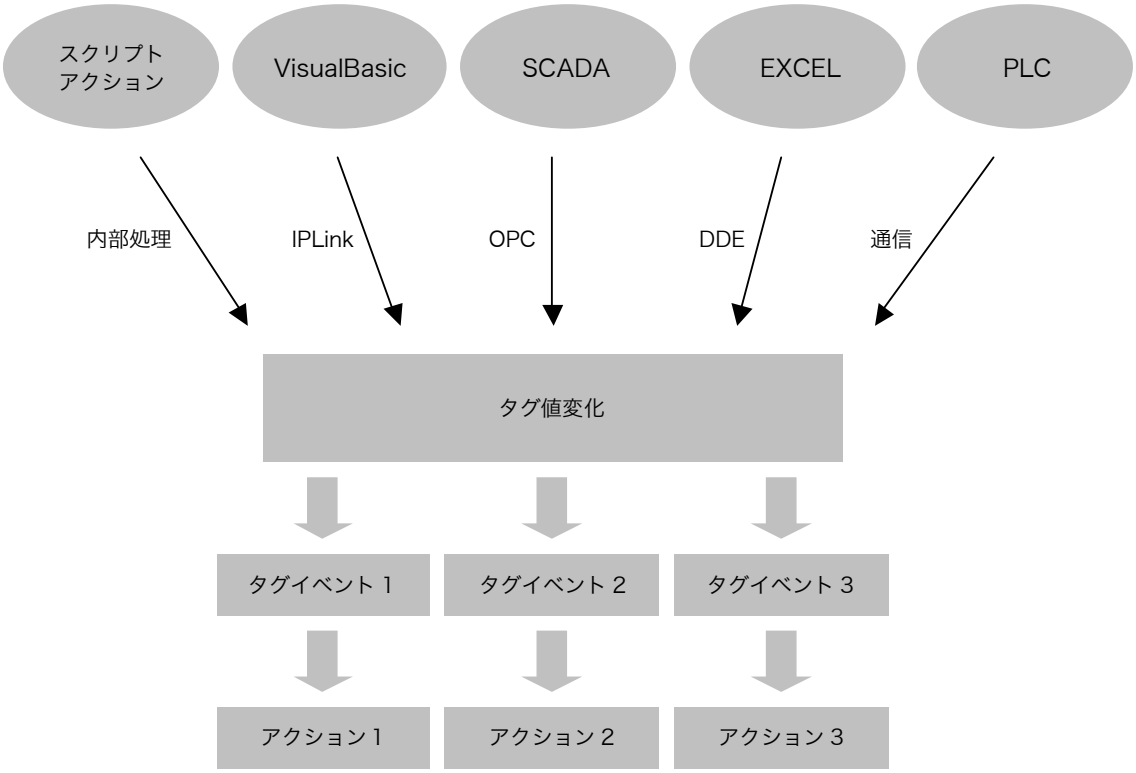


アクションの動作

アクションはイベントをトリガとして実行されます。1 つのアクションは複数のイベントから実行することができます。複数のイベントが同時に発生した場合、呼び出された順番に実行されます。



タグイベントを応用すると、PLC や外部アプリケーションからの信号値をトリガとしてアクションを実行させることができます。



アクションとイベントの関係については「イベントマニュアル」を参照下さい。

アクション一覧

アクション	サーバ機能	概要
ヒストリカルデータサーバ(CSV)	○	CSV ファイルとリンクするヒストリカルデータサーバを定義します。
ヒストリカルデータサーバ(ODBC)	○	ODBC ファイルとリンクするヒストリカルデータサーバを定義します。
ロガー(CSV)	○	CSV ファイルへロギングを行います。
ロガー(ODBC)	○	ODBC データベースへロギングを行います。
スクリプト Ver1		簡易スクリプト（SC1 構文）です。
スクリプト Ver2		本格的なスクリプト（SC2 構文）です。
タグブリッジ		タグからタグへ値を転送します。
タグレポート		タグの現在値を HTML/CSV などのテキスト系ファイルへ書き出します。
バッチ転送（タグ→テキスト）		タグからテキストファイルへ値をまとめて書き込みます。
バッチ転送（タグ←テキスト）		テキストファイルからタグへ値をまとめて書き込みます。
バッチ転送（タグ→ODBC）		タグから ODBC データベースへ値をまとめて書き込みます。
バッチ転送（タグ←ODBC）		ODBC データベースからタグへ値をまとめて書き込みます。
メール送信		メールを送信します。
ファイル操作		ファイルやフォルダのコピー、削除、移動を行います。
ファイルバックアップ		ファイルのバックアップを行います。
FTP 転送		FTP でファイルを転送します。
SQL 実行		SQL コマンドを実行します。
ダイアルアップ		ダイアルアップ接続を行います。
外部アプリケーション実行		外部アプリケーションを実行します。
スクリーンショット		スクリーンショットをとり、BMP、JPEG ファイルに保存します。
時刻同期		PC と PLC の時刻の同期をとります。
シャットダウン		アプリケーションや OS をシャットダウンします。
サマリ（CSV）	○	CSV ファイルへログ及び集計を行います。
サマリ（ODBC）	○	ODBC データベースへログ及び集計を行います。
日報・月報・年報	○	日報、月報、年報を作成します。
アラームサーバー	○	アラームを監視します。
アラームサーバー（ODBC）	○	アラームを監視します（ODBC データベースを使用する）。
ファイルサーバー	○	ネットワーク経由でファイルを転送します。
クエリーサーバー	○	ネットワーク経由でデータベースに接続します。



上記の「サーバ機能」とは、FA-Panel のクライアント環境である「Panel Browser」から接続することができるサーバ機能を持ったアクションを意味します。

各パッケージで利用可能なアクション

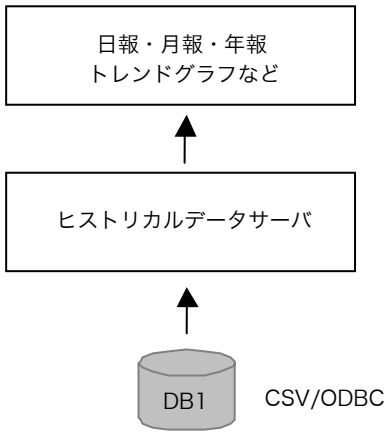
FA シリーズでは、パッケージの種類により利用可能なアクションが異なります。
下表は、各パッケージで使用可能なアクションの対応表です。

アクション	FA-Panel	FA-Studio	FA-Logger	FA-Report	FA-Trend
ヒストリカルデータサーバ(CSV)	○	○	○	○	○
ヒストリカルデータサーバ(ODBC)	○	○	○	○	○
ロガー(CSV)	○	○	○		
ロガー(ODBC)	○	○	○		
スクリプト Ver1	○	○	○		
スクリプト Ver2	○	○			
タグブリッジ	○	○			
タグレポート	○	○			
バッチ転送（タグ→テキスト）	○	○			
バッチ転送（タグ←テキスト）	○	○			
バッチ転送（タグ→ODBC）	○	○			
バッチ転送（タグ←ODBC）	○	○			
メール送信	○	○		○	○
ファイル操作	○	○			
ファイルバックアップ	○	○			
FTP 転送	○	○		○	○
SQL 実行	○	○			
ダイアルアップ	○	○			
外部アプリケーション実行	○	○			
スクリーンショット	○	○			○
時刻同期	○	○	○		
シャットダウン	○	○			
サマリ（CSV）	○				
サマリ（ODBC）	○				
日報・月報・年報	○	○		○	
アラームサーバー	○				
アラームサーバー（ODBC）	○				
ファイルサーバー	○				
クエリーサーバー	○				

2章　アクションリファレンス

ヒストリカルデータサーバアクション

ヒストリカルデータサーバアクションは、ヒストリカルデータか格納されているデータベースに接続してヒストリカルデータを取得するためのアクションです。データベースは CSV 及び ODBC に対応しています。このアクションは「日報・月報・年報アクション」「ログデータグラフ」などと組み合わせて使用されます。また、FA-Panel においては、外部アプリケーションによってデータベースに蓄積された履歴データをトレンドグラフに表示する場合などに利用されます。



ヒストリカルデータとは？

ヒストリカルデータとは、日付・時刻フィールドを含む時系列に並んでいる履歴データの事です。日付・時刻フィールド以外に MS フィールド（ミリ秒）を定義することもできます。MS フィールドは ms 単位（1 秒=1000ms）のデータを扱う場合に必要です。

(MS フィールドなしの場合)

時刻	DATA00	DATA01	・・・	DATA98	DATA99
2003/1/1 00:00:00	23	0	・・・	2	75
2003/1/1 00:00:01	44	1	・・・	3	34
・・・	・・・	・・・	・・・	・・・	・・・
2003/1/1 23:59:59	54	0	・・・	4	66

(MS フィールド有りの場合)

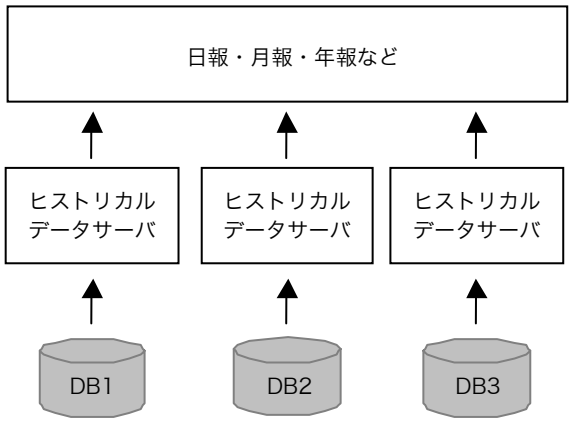
時刻	ms	DATA00	DATA01	・・・	DATA98	DATA99
2003/1/1 00:00:00	100	23	0	・・・	2	75
2003/1/1 00:00:00	200	44	1	・・・	3	34
・・・	・・・	・・・	・・・	・・・	・・・	・・・
2003/1/1 23:59:59	100	54	0	・・・	4	66

※1 秒以内のサンプリングレートのデータを扱う場合には MS フィールドを定義して下さい。



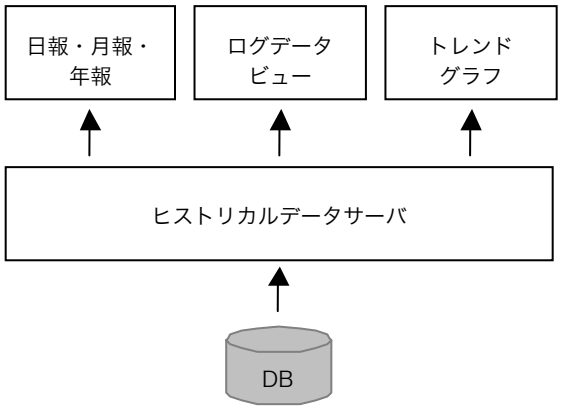
ログデータグラフなどに MS フィールドが定義されたヒストリカルデータサーバを連結すると、ms 単位での細かなグラフの表示が行われます。

複数の異なるデータベースに接続して履歴データを取得したい場合、各 DB に対してヒストリカルデータサーバを定義します。

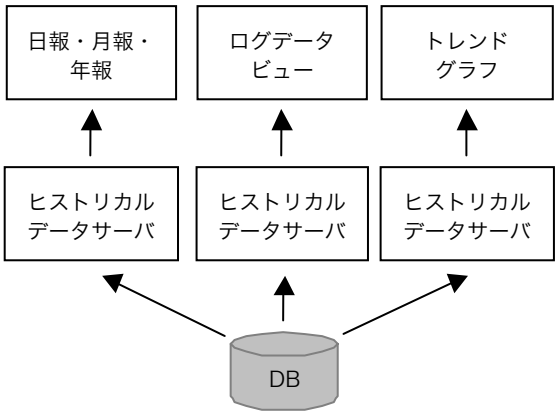


1 つのヒストリカルデータサーバは、複数のアクションやビューから共有することができます。但し、ヒストリカルデータサーバに対する要求が複数同時に発生した場合には、要求を受け付けた順番に処理が行われるため、後続の処理が待たされる可能性があります。このような場合には、各アクションやビュー毎にヒストリカルデータサーバを個別に定義することで並列処理が行われ、高速な処理が行えます。

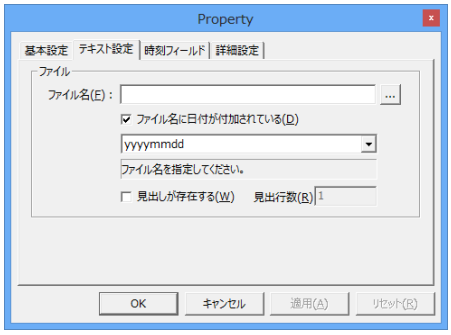
同時に要求が来た場合、順番に処理が待たされる



サーバを分けると並列的に処理される



アクション仕様	
パラメータ	なし
ダイアログ	■テキスト設定（CSV）



▽ファイル名に日付が付加されている

CSV ファイルの場合、日付ごとにファイルを分割する方法と、分割しない方法の２種類に対応しています。日付ごとにファイルが分割されている場合は、上記プロパティの「ファイル名に日付が付加されている」をチェックして下さい。

・データが１つのファイルに含まれる場合

LOG.csv

時刻	DATA00	・・・	DATA99
2003/1/1 00:00:00	23	・・・	75
2003/1/1 00:00:01	44	・・・	34
・・・	・・・	・・・	・・・
2003/1/2 00:00:00	43	・・・	55
2003/1/2 00:00:01	53	・・・	74
・・・	・・・	・・・	・・・

・日付ごとにデータファイルが分かれる場合

LOG20030101.csv

時刻	DATA00	・・・	DATA99
2003/1/1 00:00:00	23	・・・	75
・・・	・・・	・・・	・・・
2003/1/1 23:59:59	54	・・・	66

LOG20030102.csv

時刻	DATA00	・・・	DATA99
2003/1/2 00:00:00	43	・・・	55
・・・	・・・	・・・	・・・
2003/1/2 23:59:59	67	・・・	86

ファイル名は末尾に「年月日」を付加されている必要があります。

LOG<u>20030101</u>.csv

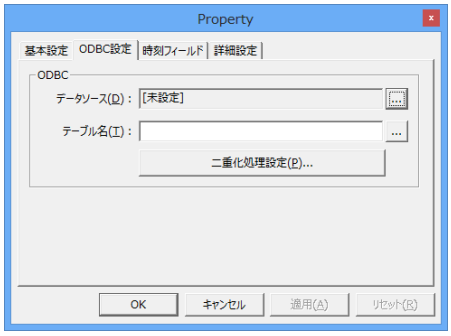
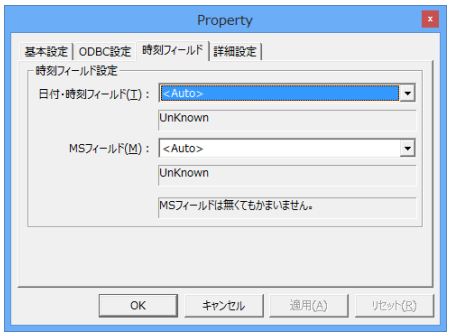
[yyyymmdd]

▽見出しが存在する / 見出行数

「見出しが存在する」をチェックすると、指定した行数を見出し行とみなします。

時刻	DATA00	・・・	DATA99
2003/1/1 00:00:00	23	・・・	75
・・・	・・・	・・・	・・・
2003/1/1 23:59:59	54	・・・	66

見出しは無視される

アクション仕様	
ダイアログ	■ODBC 設定  詳細は本書文末の「アクション共通の設定」「ODBC 設定」を参照ください。 ■時刻フィールド  「日付・時刻フィールド」は自動的に判別されるため、通常は指定する必要はありません。この設定は、データ項目に「日付・時刻フィールド」が２つ以上含まれる場合に、どちらのフィールドを時系列として使用するのかを明示したい場合のみ指定して下さい。また、MS フィールドは、データベースにミリ秒のフィールドが存在するときのみ指定して下さい。MS フィールドを含めると、ヒストリカルトレンドグラフ上にミリ秒単位でグラフ表示が行えます。

ロガーアクション

ロガーアクションは、PLC から収集したデータを CSV や ODBC データベースへ履歴データとして保存するためのアクションです。ロガーアクションについての詳しい説明は、別マニュアル「ロガーアクション詳細マニュアル」を参照してください。

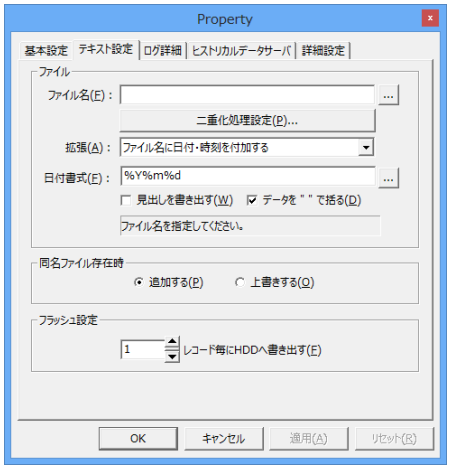
アクション仕様

パラメータ

パラメータ文字	動作
SETFILENAME	ファイル名を設定するときに使用します。通常のロギングではパラメータは指定しません。詳しくはロガーアクション詳細マニュアルを参照して下さい。
MAINTENANCE	ロガーのメンテナンスを行います（ログファイルの自動削除を任意のタイミングで行います）。

ダイアログ

■テキスト設定（ロガーCSV）



▽ファイル名

ログを保存するファイル名を指定します。

▽拡張

ログ保存ファイル名の拡張設定です。

・なし

ファイル名で指定したそのままの名前で保存します。

・ファイル名に日付・時刻を付加する

ファイル名の後ろに「日付書式」で指定した日付を付加します。%Y%m%d と指定すると 1 日毎にファイルが作成されます（例 DB20050101.csv）。また%Y%m と指定すると月ごとに、%Y%m%d%h と指定すると 1 時間ごとにファイルが作成されます。

・ファイル名にタグ値を付加する

ファイル名にタグの値を付加します。詳しくはロガーアクション詳細マニュアルを参照下さい。

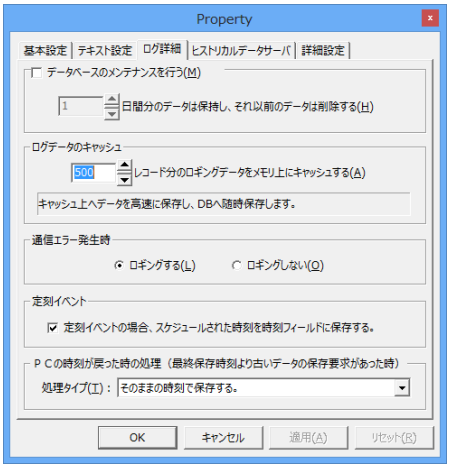
▽見出しを書き出す

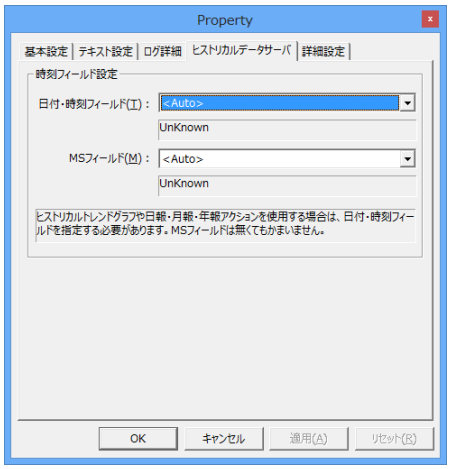
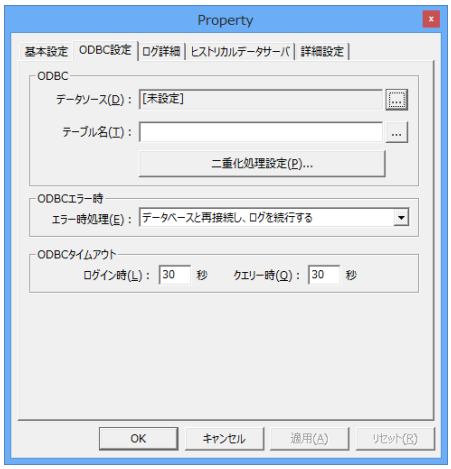
ファイルの 1 行目に必ずファイルの見出しを書き出します。見出しに使われる文字列は、フィールド毎に設定したフィールド名です。

▽同名ファイル存在時

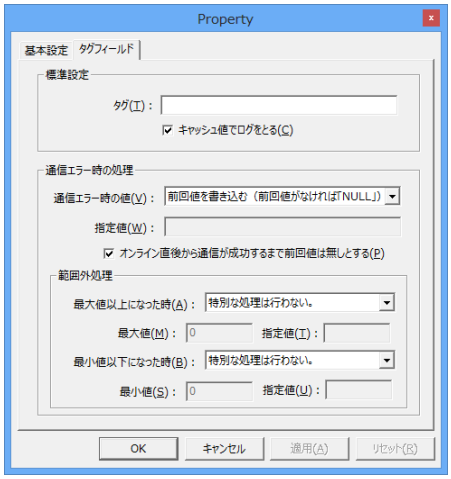
・追加する - 同名ファイルに追加書き込みします。通常はこの設定にします。

・上書きする - 古いファイルを削除し、新しいファイルを作成し新規に保存します。

アクション仕様	
ダイアログ	▽フラッシュ設定 指定した行数ごとにログデータをハードディスクに書き込みます。 ■ログ詳細  ▽データベースのメンテナンスを行う 詳しくはロガーアクション詳細マニュアル「DB のメンテナンス（自動削除）」を参照してください。 ▽ログデータのキャッシュ ログデータの最大キャッシュ数。キャッシュを利用することにより、正確なタイムスタンプのログが得られます。 ▽通信エラー発生時 通信エラーが発生したときのログの処理方法です。詳しくはロガーアクション詳細マニュアル「通信エラー発生時の動作設定」を参照してください。 ▽定刻イベントの場合、スケジュールされた時刻を時刻フィールドに保存する 定刻イベントでデータログを行う場合、（たとえタイミングがずれたとしても）必ず定刻イベントのスケジュールと同じ時刻で時刻フィールドに対しログを行いたい場合、チェックを入れてください。 ▽ＰＣの時刻が戻った時の処理（最終保存時刻より古いデータ保存要求があった時） ＰＣの時刻合わせ等で時刻が戻された場合の処理です。詳しくはロガーアクション詳細マニュアル「ＰＣの時刻が戻った時の処理」を参照してください。

アクション仕様	
ダイアログ	■ヒストリカルデータサーバ  ロガーアクションはヒストリカルデータサーバ機能を持っています。「ヒストリカルデータサーバ」タブで設定する項目は自動的に判別されるため、通常は指定する必要はありません。この設定は、データ項目に「日付・時刻フィールド」が２つ以上含まれる場合に、どちらのフィールドを時系列として使用するのかを明示したい場合のみ指定して下さい。また、MS フィールドは、データベースにミリ秒のフィールドが存在するときのみ指定して下さい。MS フィールドを含めると、ヒストリカルトレンドグラフ上にミリ秒単位でグラフ表示が行えます。 ■ODBC 設定（ロガーODBC）  ▽データソース・テーブル名 「アクション共通の設定」「ODBC 設定」を参照ください。 ▽ODBC エラー時 ログを続行したい場合は、通常「データベースと再接続し、ログを続行する」を選択してください。

■タグフィールド



タグの値を保存したいとき、タグフィールドを選択します。

▽標準設定

・タグ

　取得を行いたいタグ名（例 U01.F01.T01）です。

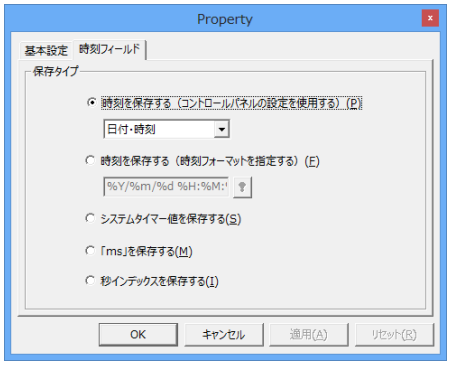
・キャッシュ値でログをとる

　詳しくはロガーアクションマニュアルの「ログの詳細フロー」を参照してください。

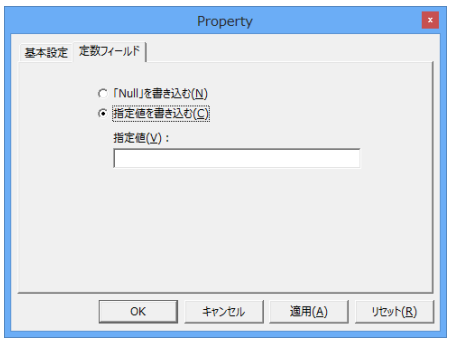
▽通信エラー時の処理

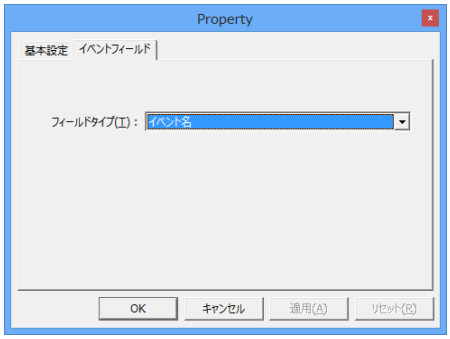
詳しくはロガーアクションマニュアルの「通信エラー発生時の動作設定」を参照してください。

■時刻フィールド



時刻の情報を保存したいとき、時刻フィールドを選択します。

アクション仕様	
ダイアログ	▽コントロールパネルの設定を使用 OS の標準設定に従い時刻を作成します。0 秒のとき秒が省略されます。0 秒の時でも省略したくない時は、「フォーマットを指定する」を選択して「%Y%m%d %H%M%S」としてください。 ▽フォーマットを指定する 時刻のフォーマットを指定します。 ▽システムタイマーを使用する 1ms 単位で増加するシステムタイマーの値（0～4294967295 の値）です。4294967295 を超えると 0 に戻ります。 ▽「ms」を使用する ms 単位で時刻を取得したい場合に選択します。値は 0～999 でログされます。通常の秒単位の時刻フィールドと ms のフィールド（計 2 つのフィールド）を組み合わせて使用すると、トレンドグラフに ms 単位で表示できます。 ▽秒インデックスを保存する 同一秒内で記録されたデータにインデックスを付加する機能です。 ■定数フィールド  指定した値や空文字を保存したいとき、定数フィールドを選択します。 ▽NULL を書き込む 何も書き込みません。 ▽指定値を書き込む 指定された文字を書き込みます。

アクション仕様	
ダイアログ	■イベントフィールド  発生したイベントの情報を保存したいときイベントフィールドを選択します。 ▽フィールドタイプ

イベント名	イベント名。　例）E01
タグ名	タグ名。　例）U01.F01.T01
タグコメント	タグコメント。タグのプロパティ「基本設定」タブで設定します。
メッセージ	メッセージ。イベントのプロパティ「イベント属性」タブで設定します。
値	タグイベントの場合イベントが発生した時のタグ値。
値（数値型のみ保存）	タグイベントの場合イベントが発生した時のタグ値。数値型のタグのみ保存します。
値（文字型のみ保存）	タグイベントの場合イベントが発生した時のタグ値。文字列型のタグのみ保存します。
値品質	タグイベントの場合イベントが発生した時のタグの値の品質。通常は「GOOD」です。PLC と通信ができないと「BAD」になります。まだ 1 度も通信が行われていない状態だと「UNCERTAIN」になります。データベース（ODBC）での保存では、数値型の DB フィールドに対して保存をする場合、数値（0-BAD 1-GOOD 2-UNCERTAIN）で保存されます。
カテゴリ	メッセージ。イベントのプロパティ「イベント属性」タブで設定します。
条件名	条件名。イベントのプロパティ「タグイベント」タブで設定します。
サブ条件名	サブ条件名。イベントのプロパティ「タグイベント」タブで設定します。
重要度	重要度。イベントのプロパティ「イベント属性」タブで設定します。

スクリプト Ver1 アクション

スクリプト Ver1 アクションでは SC1 構文を用いたスクリプトを実行することができます。

スクリプト Ver1 は+ - * / < > などの VisualBasic 互換の演算子を利用し、タグ同士の演算を行う簡易的なスクリプトです。手軽に使用できることを目的としており、if 文（条件分岐）や for 文（繰り返し処理）などは使用できません。より高度な処理を行いたい場合はスクリプト Ver2 アクションをご使用下さい。

下記はスクリプトの例です。

```
U01.F01.T01 = 1
U01.F01.T01 = U01.F01.T02 * (U01.F01.T03 + 1)
U01.F01.T01 = (U01.F01.T02 And U01.F01.T03) Or (U01.F01.T04 And U01.F01.T05)
```

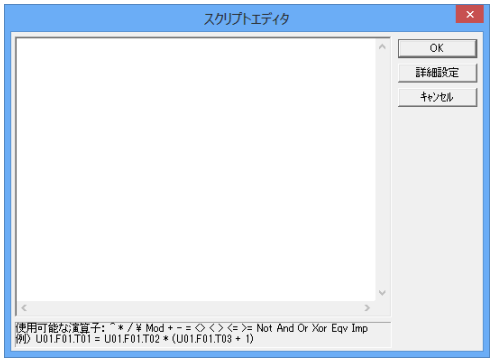
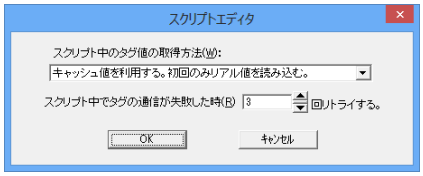
このアクションを使用したサンプルについては、本マニュアルの「実践編」「定周期で計算を行う」項目を参照してください。



スクリプトの構文については、「スクリプトマニュアル」「スクリプト Ver1（SC1 構文)」を参照して下さい。



Ver4 ではタグスクリプトアクションと呼ばれていましたが、Ver5 からスクリプトの構文体系を整理するため、スクリプト Ver1 アクションと名称が変わりました。

アクション仕様	
パラメータ	なし
ダイアログ	■スクリプトエディタ  式はスクリプトエディタから記述します。複数行の式を記述することもできます。スクリプトはノートパッドやワードなどのテキストエディタで編集したものをコピー＆ペーストできます。 ■スクリプトエディタ（詳細設定）  ▽スクリプト中のタグ値の取得方法 スクリプトの右辺に含まれるタグの値の取得方法です。 ・キャッシュ値を利用する。 キャッシュ値を利用すると、定周期通信によって更新されたキャッシュの値が計算に使用されます。キャッシュ値を用いるため、計算は即座に行われます。 ・キャッシュ値を利用する。初回のみリアル値を読み込む。 上記同様にキャッシュ値を利用しますが、初回のみリアル値（PLC デバイス値）を読み込みます。オンライン直後に定周期通信が１度も実行されていない状態でも、初回にリアル値を読み込むため、値が取得できない状態にはなりえません。初回以外はキャッシュ値を使用するため、以降の計算は即座に行われます。 ・１つの式ごとにリアル値を取得する。 １つの式ごとにリアル値（PLC デバイス値）を読み込みます。従って、処理速度は最も遅くなりますが、１つの式ごとに値を PLC から取得して計算結果を PLC へ書き込むため、PLC の値との同期が確実に保てます。 ・全式で必要なリアル値をまとめて取得する。 全式で必要なリアル値をまとめて読み込みます。まとめて値を取得するため、１つの式ごとに値を取得するよりも処理が速くなります。 ▽スクリプト中で通信が失敗したとき X 回リトライする 通信が失敗したときのリトライ回数を指定します。

スクリプト Ver2 アクション

スクリプト Ver2 アクションでは SC2 構文を用いたスクリプトを実行することができます。
SC2 構文については、「スクリプトマニュアル」「スクリプト Ver2 (SC2 構文)」を参照して下さい。

スクリプト Ver2 は、if や switch 文などの条件文や、繰り返し処理を行うための for 文などの高度なスクリプトを記述することができます。また、値変化を受け取る OnValueChange などのイベント機能も搭載しています。

スクリプトはテキストファイルに記述します。アクションのプロパティでスクリプトファイルのパスを指定します。定周期イベント/.定刻イベント/タグイベントでスクリプトを実行する周期やタイミングを設定します。

このアクションのサンプルについは、本マニュアルの「スクリプト Ver2 アクションについて」項目を参照してください。

1）構文について
スクリプトの構文については、「スクリプトマニュアル」「スクリプト Ver2 (SC2 構文)」を参照して下さい。

2）オブジェクトの階層構造
オブジェクトは階層構造を持っています。階層構造とは親子の関係のことです。スクリプト Ver1 アクションでは root オブジェクトと Math オブジェクトの 2 階層しかありません。オブジェクトはメソッド、イベントを持っています。



3）ルートオブジェクトで利用できるメソッド及びイベント

ルートオブジェクトでは、下記のメソッドおよびイベントが利用できます。各メソッド及びイベントの詳細は「コントロールリファレンスマニュアル」に記述されています（「コントロールリファレンスマニュアル」については後述するヒントを参照して下さい）。

<table>
<tr><th colspan="2">ルートオブジェクトで利用できるメソッド及びイベント</th></tr>
<tr><td>共通メソッド
共通イベント</td><td>「コントロールリファレンスマニュアル」の「共通メソッド」に掲載されているメソッド及びイベントが利用できます。各メソッドの仕様についてはマニュアルを参照して下さい。

※「コントロールリファレンスマニュアル」でサポート対象「スクリプトアクション」に○が付いているメソッドやイベントが利用できます。</td></tr>
<tr><td>ルートメソッド</td><td>「コントロールリファレンスマニュアル」の「ルートメソッド」に掲載されているメソッドが利用できます。各メソッドの仕様についてはマニュアルを参照して下さい。

※「コントロールリファレンスマニュアル」でサポート対象「スクリプトアクション」に○が付いているメソッドが利用できます。</td></tr>
<tr><td>new 演算子</td><td>new 演算子が使用できます。new 演算子はオブジェクトの生成時に利用されます。
new 演算子により下記のオブジェクトをスクリプト内で使用できます。

・DirectQueryDB コントロール（データベースへの接続）
・TextDump コントロール（テキストファイルへの出力）
・HistoricalData コントロール（ヒストリカルデータサーバから時系列データを取得）

new 演算子の使用方法は下記のようになります。
<pre>var objDirectQueryDB;
event OnInitialize()
{
　　// データベースオブジェクト生成
　　objDirectQueryDB = new DirectQueryDB;
}</pre>
<table><tr><td>テキストダンプコントロール
TextDump</td><td>使用できるプロパティ、メソッドは「コントロールリファレンスマニュアル」を参照して下さい。</td></tr><tr><td>ダイレクトクエリーデータベースコントロール
DirectQueryDB</td><td>使用できるプロパティ、メソッドは「コントロールリファレンスマニュアル」を参照して下さい。</td></tr><tr><td>ヒストリカルデータコントロール
HistoricalData</td><td>使用できるプロパティ、メソッドは下記です。
メソッド）
AddField/DelField/ClearField/ReadHistoricalData
プロパティ）
HDSRequestType
プロパティ、メソッドの仕様は「コントロールリファレンスマニュアル」を参照して下さい。</td></tr></table>
※各機能を使用したサンプルを用意しています。詳細については、
本マニュアルの「第 3 章 スクリプト Ver2 について」をご参照ください。</td></tr>
</table>

コントロールリファレンスマニュアルはオンラインマニュアルで作成されています。オンラインマニュアル（ControlReference5.chm）はインストールフォルダ内にあります。オンラインヘルプにて共通メソッド、共通イベント、ルートメソッドの一覧表の「スクリプトアクション」に○が付いているメソッド、イベントが利用できます。


コントロールリファレンス > 1章 基本コントロール > 1-1 共通仕様:

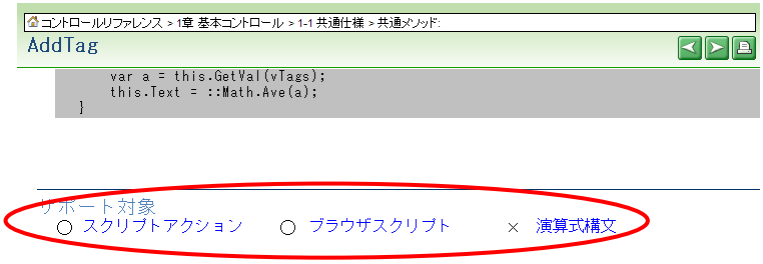
共通メソッド

すべてのコントロールが共通的に持つメソッドのことを、共通メソッドと呼びます。

タグ

メソッド名	説明	スクリプトアクション	ブラウザスクリプト	演算式構文
AddTag	オブジェクト内のスクリプトからアクセスするタグを宣言します（タグを登録する）。これにより、GetVal/ ReadVal/ WriteVal/ ReadRequestRefresh/ ReadRefresh/ WriteRequestRefresh/WriteRefresh でタグ値の入出力が行えるとうになります。また、タグの値が化したときに「OnTagValueChanged」イベントが呼び出されるようになります。	○	○	×
DelTag	タグの登録を解除します。タグの登録はフォームを閉じるときに自動的に解除されるため、特に理由がない限りDelTag を呼び出す必要はありません。	○	○	×
ClearAllTag	すべてのタグの登録を解除します。	○	○	×

また、各メソッドのページで「スクリプトアクション」に○が付いているかどうかでも判断できます。


コントロールリファレンス > 1章 基本コントロール > 1-1 共通仕様 > 共通メソッド:

AddTag

```
    var a = this.GetVal(vTags);
    this.Text = ::Math.Ave(a);
}
```

リポート対象

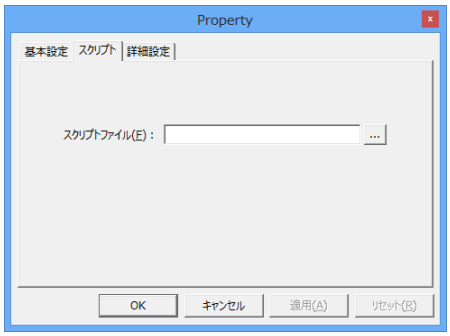
○ スクリプトアクション　○ ブラウザスクリプト　× 演算式構文

使用できるコントロールは下記の３つです。

・ダイレクトクエリーデータベースコントロール（DirectQueryDB）

・テキストダンプコントロール（TextDump）

・ヒストリカルデータコントロール（HistoricalData）

アクション仕様	
パラメータ	なし。定周期イベント/.定刻イベント/タグイベントでスクリプトを実行する周期やタイミングを設定します。イベントによりアクションが実行される毎に、スクリプトの内部処理が行われ、OnValueChanged/OnExpressionValueChanged/OnTimer 等が呼び出されます。 また、OnHeartBeat イベントはイベントによりアクションが実行される毎に必ず呼び出されます。例えば、1 秒の定周期イベントでアクションを呼び出すと設定をすると OnHeartBeat イベントは 1 秒ごとに呼び出されます。また、タイマーについてですが、10 秒の定周期イベントにてアクションを呼び出す設定にした場合、スクリプト中にて SetTimer(0, 5000, -1)で 5 秒タイマーを発生させたとしても、OnTimer イベントは 10 秒毎しか発生しません。タイマーを使用する場合はタイマーより早い周期のイベントでスクリプトアクションを呼び出す必要があります。 詳細は下記のヒント項目を参照してください
ダイアログ	■スクリプト  スクリプトはテキストファイルに記述します。アクションで実行したいスクリプトファイルのパスを指定します。

ヒント

OnHeartBeat イベントはイベントによりアクションが実行される毎に必ず呼び出されます。尚、定周期的な処理を実装する場合には、SetTimer メソッドによる OnTimer イベントの使用を推奨します。SetTimer には、複数のタイマーを同時に起動する、個々の周期を設定できるなどのメリットがあります。例えば、定周期的な処理は OnTimer イベントを使用し、定刻イベントなどでスクリプトを実行したい場合は、OnHeartBeat イベントを使用するといった使い分けをしてください。

ヒント

OnTimer イベントは、ScriptVer2 アクションがイベントにより呼ばれた時に、条件（OnTimer で設定した周期が経過している）を満たしていた場合に実行されます。
例えば、イベントの周期：10 秒　OnTimer の周期：11 秒と指定している場合、最初の実行から 11 秒後に OnTimer の処理がされるのではなく、下記のようにイベントで ScriptVer2 が再び呼ばれた段階で、処理が実行されます。

1：初回実行（0 秒）→OnTimer のカウント開始。
2：イベントで 1 度目の実行（10 秒経過）→周期は 11 秒なので、条件が不成立の為まだ処理はされない。
3：OnTimer の周期経過（11 秒経過）→ScriptVer2 は呼び出されていない為まだ実行されない。
4：イベントで 2 度目の実行（20 秒経過）→OnTimer の設定周期を超えている為、条件が成立し、処理が実行される。

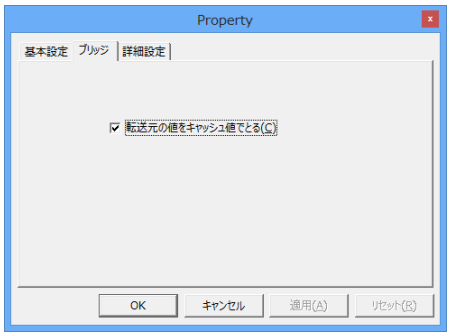
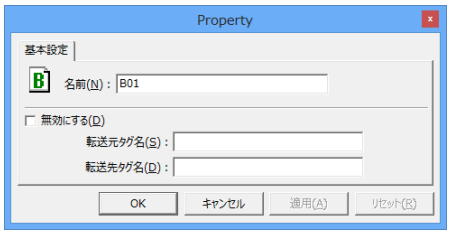
イベント毎に OnTimer の処理を行いたい場合は、イベントの周期よりも短い周期を OnTimer の周期とする必要があります。

ブリッジアクション

タグからタグへデータを転送します。配列にも対応しており、まとまった領域を効率よくデータ転送できます。

このアクションを使用したサンプルについては、本マニュアルの「実践編」「ＰＬＣ間のデータ共有（データ転送）」項目を参照してください。

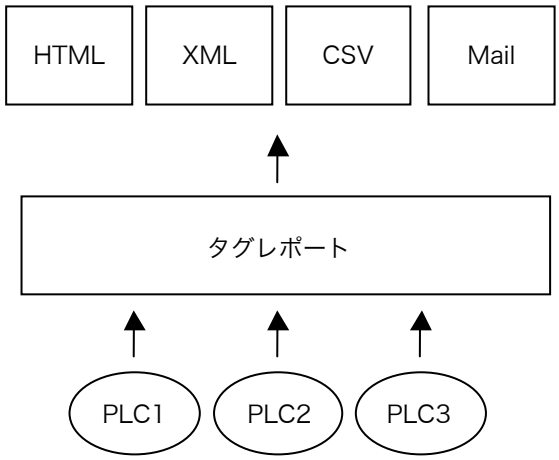
アクション仕様

パラメータ	なし
ダイアログ	■ブリッジ  ▽転送元の値をキャッシュ値でとる 転送元の値をキャッシュから取得する事により、他のアクションや通信に影響を与えず安定した処理が行えます。よって、リアルタイムに転送を行いたい場合などはキャッシュを用いるようにしてください。チェックを入れなかった場合はキャッシュ値を使用せず、ブリッジアクションが実行される度にリアル値を取得するための PLC との通信が行われます。転送があまり頻繁に行われないようなバッチ的な処理の場合はキャッシュ値を使用しない設定が向いています。 ■ブリッジアイテムの基本設定  ▽転送元タグ名・転送先タグ名 タグ値の転送は、転送元タグから転送先タグに対して行われます。双方に配列タグを設定すると、連続した領域を一括で高速に転送できます。尚、まとまった領域を指定しない場合においても、内部的には出来る限り高速な通信が行えるように最適化されています。

大量のタグを設定する場合には、テキストエディタなどで設定ファイル（xml ファイル）を直接開いて編集すると効率的です。

タグレポートアクション

タグレポートアクションは、テキスト系のファイル（HTML,XML,CSV なども含む）へタグの値を埋め込むことができます。



タグレポートアクションの設定では、タグの値を埋め込む元となる「マスターファイル」を用意します。例えば、Web で値をブラウジングしたい場合は、マスターとなる HTML ファイルを用意し、埋め込みたい場所へマーク文字列を記述していきます。マーク文字列とは、タグの値をどの位置に埋め込みたいのかをタグレポートアクションへ知らせるための文字列です。マーク文字列として使用可能な文字には特に制約はありません。ユーザーは自由に文字列を定義して埋め込むことができます。

マーク文字列の例）

[TAG01]
[TAG02]
[TAG03]

マスターファイルの文章例）

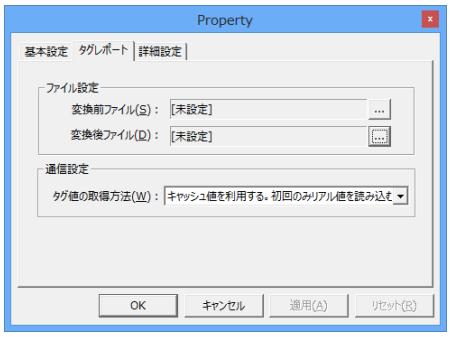
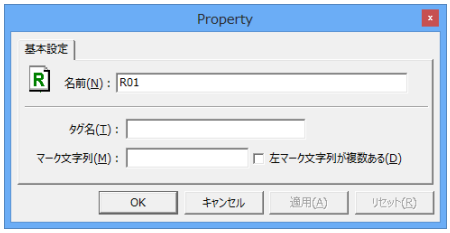
[TAG01]のエラーが発生しました。
ポンプ A の温度は**[TAG02]** ℃です。
ポンプ B の温度は**[TAG03]** ℃です。

アクションによる変換結果）

123 のエラーが発生しました。
ポンプ A の温度は 50.2℃です。
ポンプ B の温度は 48.5℃です。

このアクションを使用したサンプルについては、本マニュアルの「実践編」「WEB 経由での簡易モニタリング」及び「アラームメール通知」項目を参照してください。

テキストファイルであれば、どんな位置にマーク文字列があってもかまいません。例えば、HTML で JavaScript を使用している場合、その関数の引数や変数の初期値としてタグの値を埋め込むことなどもできます。

アクション仕様	
パラメータ	なし
ダイアログ	■タグレポート  ▽変換前ファイル 変換前のマスターファイルを指定します。このファイルを元に埋め込みが行われます。設定の方法は「アクション共通の設定」「名前フォーマット」を参照ください。 ▽変換後ファイル タグの値が埋め込まれた変換後のファイル名を指定します。設定の方法は「アクション共通の設定」「名前フォーマット」を参照ください。 ▽タグ値の取得方法 埋め込み時のタグ値の取得方法について、以下の中から選択します。 ・キャッシュ値を利用する。 ・キャッシュ値を利用する。初回のみリアル値を読み込む。 ・リアル値を利用する。 ■タグレポートアイテム基本設定  ▽タグ名 値を埋め込むタグを指定します。 ▽マーク文字列 マスターファイルに埋め込むマーク文字列を指定します。この文字列がタグ値と置き換えられます。 ▽左マーク文字列が複数ある 上記マーク文字列がレポート内に複数ある場合には、このチェックを入れてください。

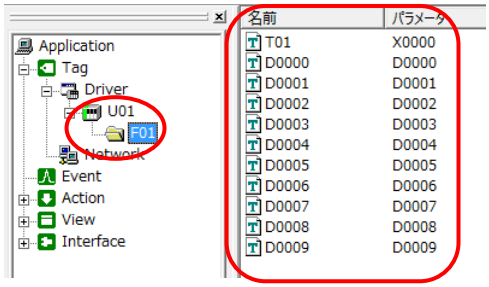
バッチ転送（タグ→テキスト）アクション

タグの情報をテキストファイルへ書き出します。書き出すことができる情報は下記のとおりです。

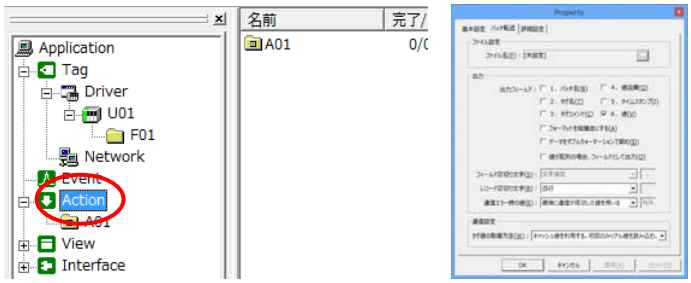
種類	説明
バッチ名	アクションでタグごとに定義した名称です。
タグ名	タグパスです（U01.F01.T01 など）。
タグコメント	タグに定義されたコメントです。
値品質	タグの値の品質です。通常は「GOOD」です。PLC と通信ができないと「BAD」になります。タグの値の取得方法で「キャッシュ値を利用する」を選択し、まだ 1 度も通信が行われていない状態だと「UNCERTAIN」になります。
タイムスタンプ	情報を取得した時刻です。
値	タグの値そのものです。

下記はアクションの設定手順です。ドラッグ＆ドロップにより簡単に定義することができます。

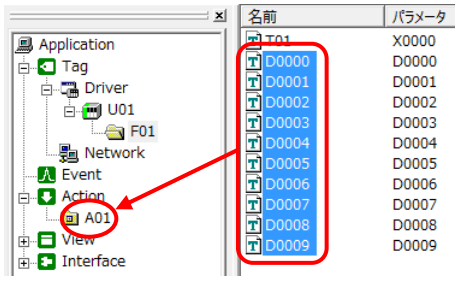
1．タグを定義する



2．バッチ転送（タグ→テキスト）アクションを定義する



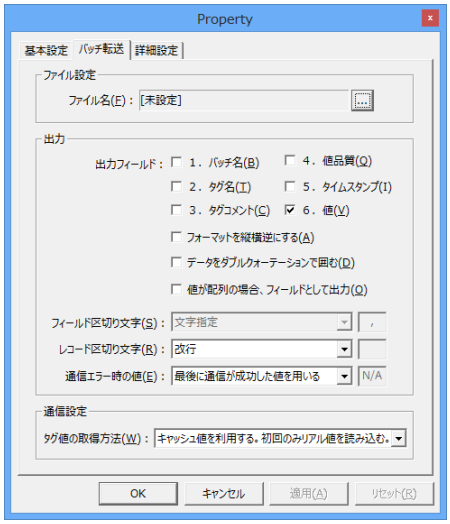
3．バッチ転送（タグ→テキスト）アクションにアイテムを追加する



タグをドラックアンドドロップします。

このアクションを使用したサンプルについては、本マニュアルの「実践編」「PLC の大量データを一括出力する（テキスト出力）」項目を参照してください。

アクション仕様	
パラメータ	なし
ダイアログ	■バッチ転送



▽ファイル名

出力するファイルの名称を指定します。

▽出力フィールド

ここで選択されたフィールドがテキストファイルに出力されます。1～6の順序で保存されます。

▽フォーマットを縦横逆にする

例）

U01.F01.T01	GOOD	123
U01.F01.T02	GOOD	456
・・・	・・・	・・・
U01.F01.T99	GOOD	789

上記を逆にすると下記になります。

U01.F01.T01	U01.F01.T02	・・・	U01.F01.T99
GOOD	GOOD	・・・	GOOD
123	456	・・・	789

▽データをダブルクォーテーションで囲む

テキストファイルへ出力する際に各値が「"」で囲まれます。

▽値が配列の場合、フィールドとして出力

配列タグの場合、配列要素数分の値を「フィールド区切り文字」の指定に従い値を出力します。

▽フィールド区切り文字

テキストに出力される値は、出力フィールドの設定内容により複数の値が出力される場合がありますが、複数の値が出力される場合の区切り文字を指定します。出力フィールドが２つ以上の場合 又は 配列をフィールドとして出力する場合に指定します。

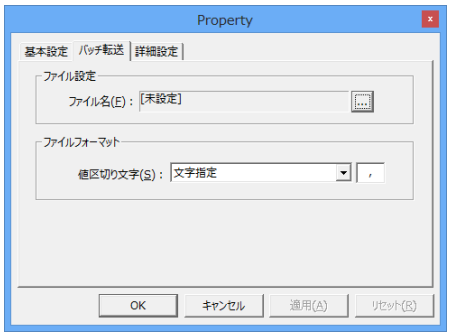
アクション仕様	
ダイアログ	▽レコード区切り文字 ここでは、タグ毎のレコード出力を区切るための区切り文字を指定することができます。尚、上記の「フィールド区切り文字」で設定する文字は、１つのタグ内で複数の値が出力される時に各値を区切ための区切り文字です。 ▽通信エラー時の値 通信が正しく行われなかった場合に、代わりに出力する値を指定します。 ▽タグ値の取得方法 埋め込み時のタグ値の取得方法について、以下の中から選択します。 ・キャッシュ値を利用する。 ・キャッシュ値を利用する。初回のみリアル値を読み込む。 ・リアル値を利用する。

バッチ転送（テキスト→タグ）アクション

テキストファイルを読み込んで値を取得し、取得した値をタグに書き込みます。

このアクションを使用したサンプルについては、本マニュアルの「実践編」「PLC へ大量データの一括書き込みを行う（テキスト経由）」項目を参照してください。

アクション仕様

パラメータ	なし
ダイアログ	■バッチ転送  ▽ファイル名 データが格納されているファイルの名称を指定します。 ▽ファイルフォーマット 上記ファイルのフォーマットです。値の区切り文字を指定します。

バッチ転送（タグ→ODBC）アクション

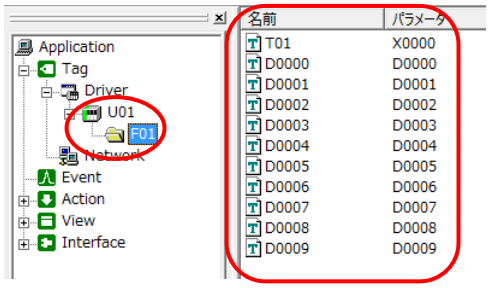
タグの情報を ODBC データベースへ書き出します。

データベースへ書き出すことができる情報は下記のとおりです。

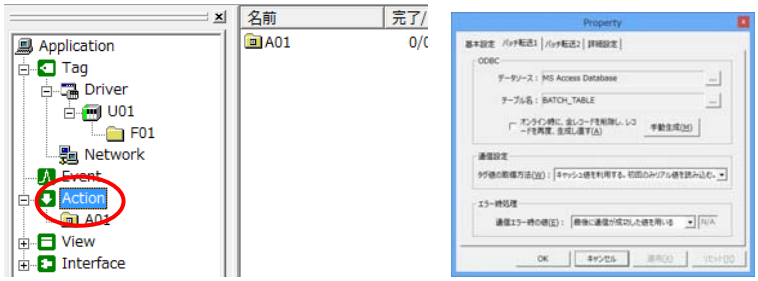
種類	説明
バッチ名	アクションでタグごとに定義した名称です。
タグ名	タグパスです（U01.F01.T01 など）。
タグコメント	タグに定義されたコメントです。
値品質	タグの値の品質です。通常は「GOOD」です。PLC と通信ができないと「BAD」になります。タグの値の取得方法で「キャッシュ値を利用する」を選択し、まだ 1 度も通信が行われていない状態だと「UNCERTAIN」になります。
タイムスタンプ	情報を取得した時刻です。
値	タグの値そのものです。

アクションの設定手順は以下のとおりです。

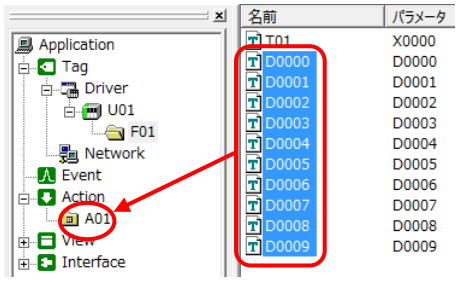
1．タグを定義する



2．バッチ転送（タグ→ODBC）アクションを定義する



3．バッチ転送（タグ→ODBC）アクションにアイテムを追加する



タグをドラックアンドドロップします。

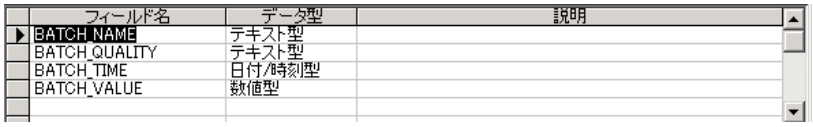
4．データベースを作成する

アクションで定義しているフィールド名に合わせてデータベースのレイアウトを定義します。
例）



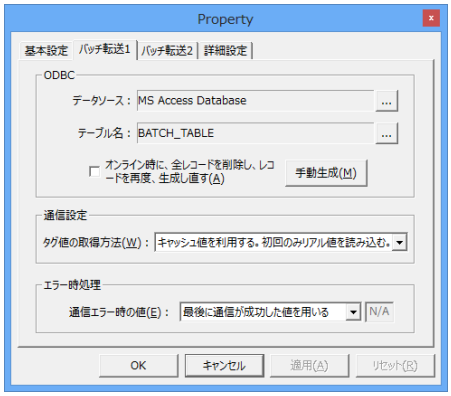
バッチ名フィールド：BATCH_NAME（「バッチ名」フィールドは必須です）。
値品質フィールド：BATCH_QUALITY
タイムスタンプフィールド：BATCH_TIME
値フィールド：BATCH_VALUE

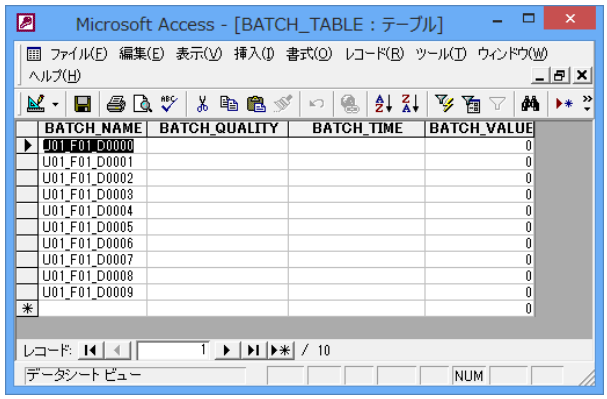
Access でのフィールド設定例：



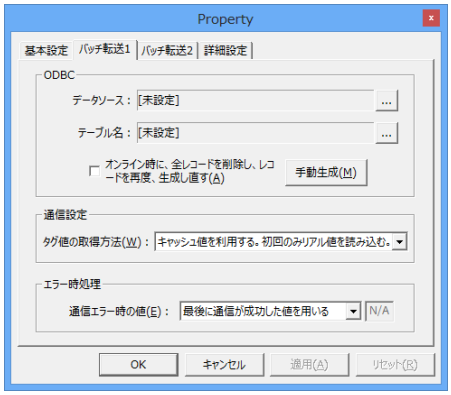
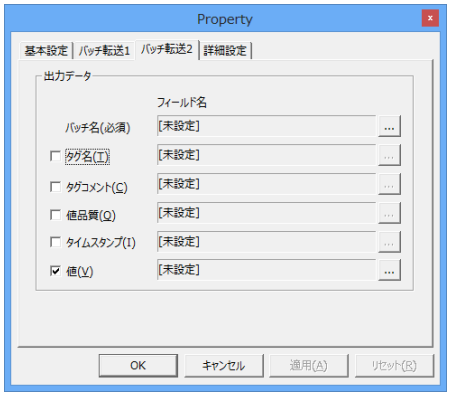
フィールド名	データ型	説明
BATCH_NAME	テキスト型	
BATCH_QUALITY	テキスト型	
BATCH_TIME	日付/時刻型	
BATCH_VALUE	数値型	

5．「手動作成」ボタンを押し、キーとなるバッチ名をデータベースへ書き込む
当アクションはバッチ名をキー項目としてデータベースに値を書き込むため、準備としてこの作業を行う必要があります。右の画面はバッチ名がデータベースに正しく書き込まれた状態です（Access の例）。





このアクションを使用したサンプルについては、本マニュアルの「実践編」「PLC の大量データを一括出力する（DB 出力）」項目を参照してください。

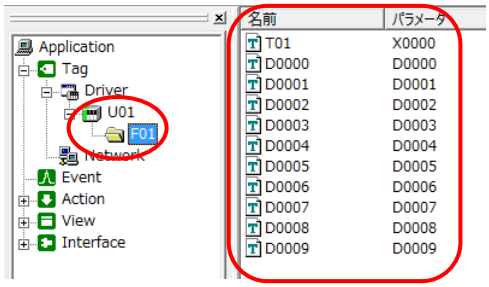
アクション仕様	
パラメータ	なし
ダイアログ	■バッチ転送１  ▽データソース 「アクション共通の設定」「ODBC 設定」を参照ください。 ▽テーブル名 テーブル名を指定します。日付や時刻によりテーブルを切り替えることもできます。 ▽オンライン時に、全レコードを削除し、レコードを再度、生成し直す オンラインにする度に、データベースの全てのレコードを削除し、新たにレコードを生成しなおします。この設定を有効にすると、オンラインの度にデータベースの追加削除が行われるため、通常は無効にしておきます。但し、この設定を有効にした場合、「手動生成」ボタンによりあらかじめレコードを作成しておく必要がなくなります。 ▽タグ値の取得方法 埋め込み時のタグ値の取得方法について、以下の中から選択します。 ・キャッシュ値を利用する。 ・キャッシュ値を利用する。初回のみリアル値を読み込む。 ・リアル値を利用する。 ▽通信エラー時の値 通信が正しく行われなかった場合に、代わりに出力する値を指定します。
	■バッチ転送２  データベースに保存したい出力データを選択します。バッチ名フィールドは必須項目につき、必ずデータベースへフィールドを定義してください。尚、日付や時刻によりフィールドを切り替えることもできます。

バッチ転送（ODBC→タグ）アクション

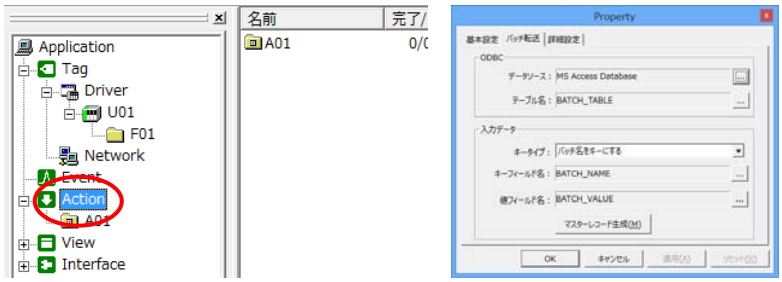
ODBC データベースから値を取得し、取得した値をタグに書き込みます。

アクションの設定手順は以下のとおりです。

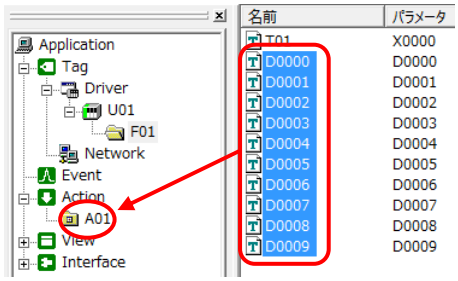
1．タグを定義する



2．バッチ転送（ODBC→タグ）アクションを定義する



3．バッチ転送（ODBC→タグ）アクションにアイテムを追加する

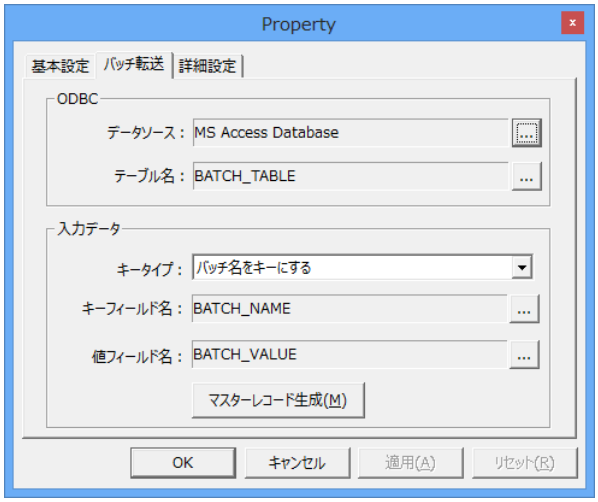


タグをドラックアンドドロップします。

このアクションを使用したサンプルについては、本マニュアルの「実践編」「PLC へ大量データの一括書き込みを行う（DB 経由）」項目を参照してください。

4．データベースを作成する

アクションで定義しているフィールド名に合わせてデータベースのレイアウトを定義します。
例）

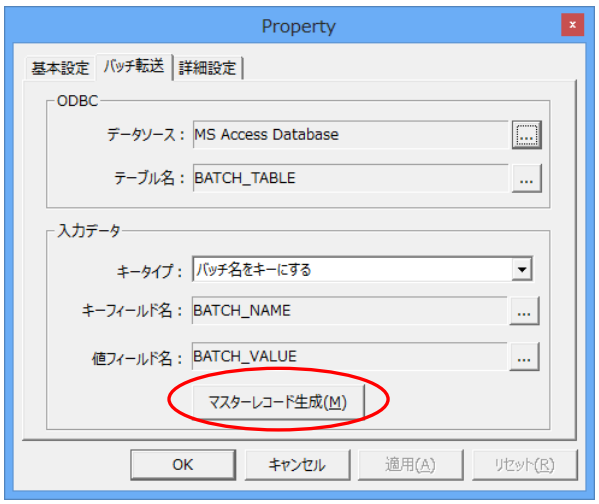


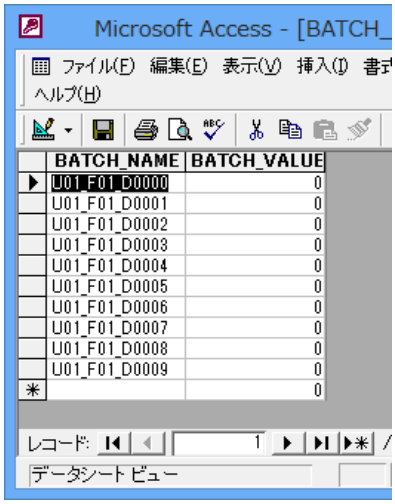
キーフィールド名：BATCH_NAME
値フィールド名：BATCH_VALUE

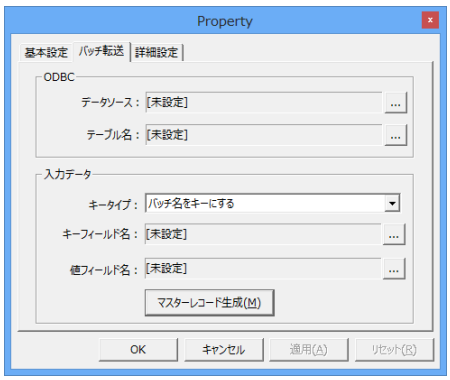
Access でのフィールド設定例：

	フィールド名	データ型	説明
▶	BATCH_NAME	テキスト型	
	BATCH_VALUE	数値型	

5．「マスターレコード生成」ボタンを押し、キーとなるバッチ名をデータベースへ書き込む
当アクションはバッチ名をキー項目としてデータベースから値を取得するため、準備としてこの作業を行う必要があります。右の画面はバッチ名がデータベースに正しく書き込まれた状態です（Access の例）。





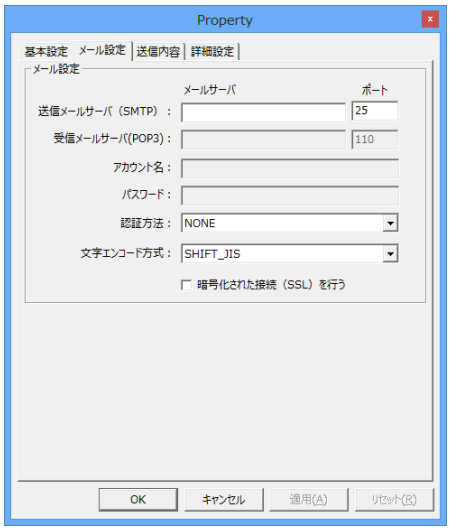
アクション仕様	
パラメータ	なし
ダイアログ	■バッチ転送  ▽データソース 「アクション共通の設定」「ODBC 設定」を参照ください。 ▽テーブル名 テーブル名を指定します。日付や時刻によりテーブルを切り替えることもできます。 ▽キータイプ ・バッチ名をキーにする ・タグ名をキーにする ▽キーフィールド名 キーフィールドの名前 ▽値フィールド名 値を格納するフィールドの名前

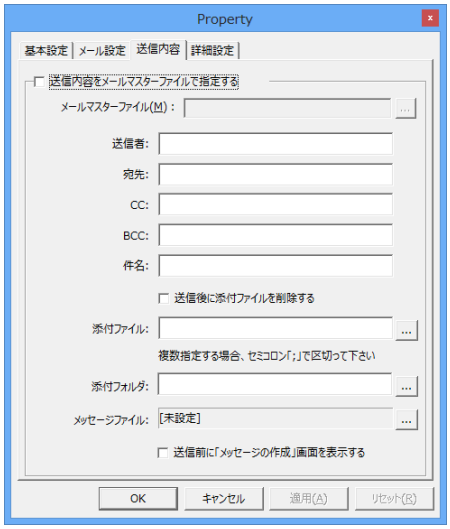
メール送信アクション

メール送信アクションを使うと、メールを送信することができます。

このアクションを使用したサンプルについては、本マニュアルの「実践編」「任意のタイミングでメール送信を行う」及び「アラームメール通知」項目を参照してください。

アクション仕様

パラメータ	1）なにも指定しない場合 メールを送信します。 2）「RELOADMAILMASTER」を指定した場合 メールマスターファイルを再読み込みします。システム運用中にメールマスターが編集された場合にこのパラメータを指定します。 メールの送信先などの情報をシステム運用中にオペレータによって変更・追加を行えるようにするには、まず、VB などの外部アプリケーションや Panel Browser からメールマスターファイルを編集して保存します。続いて RELOADMAILMASTER パラメータを付加したイベント（タグイベントなど）でメール送信アクションを実行すると、編集したメールマスターファイルが再読み込みされます。再読込み後、パラメータ無しでメール送信アクションを実行すると、編集後の内容でメールが送信されます。
ダイアログ	■メール設定  ▽送信メールサーバー(SMTP)： メールを送信するための SMTP サーバのアドレスとポート番号です。

アクション仕様	
ダイアログ	▽受信メールサーバー(POP)： 　認証で利用する POP サーバのアドレスとポート番号です。 ▽アカウント名：アカウント名です。 ▽パスワード：パスワードです。 ▽認証方法：メール送信時の認証方法を選びます。 ▽文字エンコード方式：文字のエンコード方式を選びます。 ▽暗号化された接続（SSL）を行う：SSL 接続したい場合にチェックします。 ■送信内容  ▽送信内容をメールマスターファイルで指定する： 　メールの送信内容を外部テキストファイルで指定することができます。運用中にオペレータが送信先を追加や編集したいときなどはメールマスターファイルを利用します。メールマスターファイルのフォーマットについては次ページを参照してください。 ▽送信者： 　送信者のメールアドレスです。 ▽宛先： 　送信先のメールアドレスです。 ▽CC： 　カーボンコピーです。CC 欄にアドレスを書いて送られたメールの場合、受取人はそのメールが自分以外の誰に送られたかをメッセージヘッダーから読みとることが出来ます。 ▽BCC： 　ブラインドカーボンコピーです。BCC 欄を使って送られたメールの場合は、受取人名がメッセージヘッダーに記録されないため、自分以外の誰か宛てに送られたメールかどうかは分からなくなります。 ▽件名： 　メールのタイトルを指定します。 ▽送信後に添付ファイルを削除する： 　メールの送信後に、添付したファイルを削除します。たとえば日報月報年報アクションで Excel などのサブレポートを特定のフォルダに作成するようにしておき、メール送信アクションで添付ファイルとしてサブレポートファイルを送信後に、ファイルを削除するという流れを自動化できます。

アクション仕様	
ダイアログ	▽添付ファイル： メールに添付するファイルです。複数指定する場合は、セミコロン「;」で区切ります。右の選択ボタンを押すと次々と追加されますので、削除したい場合はエディットボックスから直接削除して下さい。 ▽添付フォルダ： フォルダ内にあるすべてのファイルをメールに添付します。 ▽メッセージファイル： メールのメッセージ部にあたるテキストファイルです。 ▽送信前に「メッセージ作成」画面を表示する： 手動でメールの内容を編集したいときに、送信前にメッセージ作成のための画面を表示します。

メールマスターファイルについて

メールマスターはメール送信先等の情報を定義したテキストファイルです。ノートパッド（メモ帳）などのテキストエディタで作成・編集します。メールマスターのファイル名は自由です。メールマスターのサンプルは「¥¥sampel¥common¥action¥mail¥mailmaster.txt」にあります。

下記はメールマスターのフォーマットです。

```
[MailParam]
DELETEATTACHEDFILE=[0 or 1]

[MailSubject]
タイトル

[MailSender]
送信者

[MailSendToName]
送信先１
送信先２
…
(end)

[MailSendToNameCC]
送信先(CC)1
送信先(CC)2
 …
(end)

[MailSendToNameBCC]
送信先(BCC)1
送信先(BCC)2
 …
(end)

[MailAttachedFile]
添付ファイル１
添付ファイル２
…
(end)

[MailAttachedFolder]
添付ファイルが含まれるフォルダ１
添付ファイルが含まれるフォルダ２
…
(end)

[MailMessage]
メールメッセージ１行目
メールメッセージ２行目
…
 (end)
```

プロパティ	解説

[MailParam]	メール送信のパラメータを記述します。現在設定できるパラメータは下記のみです。
	
	// メール送信後、添付したファイルを削除しない
	DELETEATTACHEDFILE=0
	// メール送信後、添付したファイルを削除する
	DELETEATTACHEDFILE=1
	
	※「=」の前後にスペースを入れないでください。

[MailSubject]	メールのタイトル（件名）を指定します。 例） [MailSubject] Title
[MailSender]	メール送信者を指定します。 例） [MailSender] xxx@xxx.co.jp
[MailSendToName]	メール送信先を指定します。複数のメールアドレスに送信したい場合は、メールアドレスを下記のように列挙します。末尾は必ず(end)で終わらせます。 例） [MailSendToName] xxx1@xxx.co.jp xxx2@xxx.co.jp (end)
[MailSendToNameCC]	メール送信先(CC)を指定します。複数のメールアドレスに送信したい場合は、メールアドレスを下記のように列挙します。末尾は必ず(end)で終わらせます。 例） [MailSendToNameCC] xxx1@xxx.co.jp xxx2@xxx.co.jp (end)

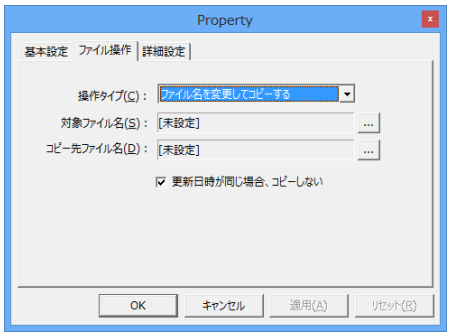
プロパティ	解説
[MailSendToNameBCC]	メール送信先(BCC)を指定します。複数のメールアドレスに送信したい場合は、メールアドレスを下記のように列挙します。末尾は必ず(end)で終わらせます。 例） [MailSendToNameBCC] xxx1@xxx.co.jp xxx2@xxx.co.jp (end)
[MailAttachedFile]	添付したいファイル名を指定します。複数のファイルを添付したい場合は、ファイル名を下記のように列挙します。末尾は必ず(end)で終わらせます。 例） [MailAttachedFile] c:¥xxx1.txt c:¥xxx2.txt (end)
[MailAttachedFolder]	添付したいファイルが含まれるフォルダを指定します。フォルダ内のすべてのファイルが添付されます。複数のフォルダを添付したい場合は、フォルダ名を下記のように列挙します。末尾は必ず(end)で終わらせます。 例） [MailAttachedFolder] c:¥xxx1 c:¥xxx2 (end)
[MailMessage]	メールの文章を指定します。末尾は必ず(end)で終わらせます。 例） [MailMessage] メールの内容１行目。 メールの内容２行目。 メールの内容３行目。 (end)

ファイル操作アクション

ファイル操作アクションでは、ファイルのコピー、移動、削除を行うことが可能です。ファイルやフォルダ名に日付を付加することもできます。

例）
昨日のファイル（ファイル名「ABC[YYYYMMDD].txt」）をコピーする
先月のフォルダを削除する

このアクションを使用したサンプルについては、本マニュアルの「実践編」「ファイルを別のフォルダにコピーする」項目を参照してください。

アクション仕様	
パラメータ	なし
ダイアログ	■ファイル操作  ▽操作タイプ 下記から選択します。 ・ファイル名を変更してコピーする ・ファイルをフォルダへコピーする ・ファイルをフォルダへ移動する ・ファイルを削除する ・フォルダの中身をコピーする ・フォルダの中身を移動する ・フォルダの中身を削除する ・フォルダを作成する ・フォルダを削除する ▽対象 XXX 名 「アクション共通の設定」「名前フォーマット」を参照ください。 ▽コピー先 XXX 名 「アクション共通の設定」「名前フォーマット」を参照ください。



必要なファイルを誤って削除してしまわないように、ファイルやフォルダの削除の指定はくれぐれも慎重に行って下さい。

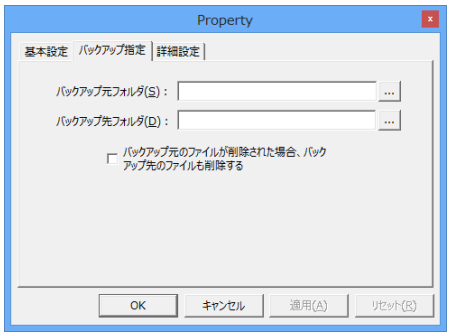


ファイル名に日付を付加するときは、日付フォーマットのところにファイル名の拡張子を指定してください。例えば ABCYYYYMMDD.csv としたいときは、ファイル名に「ABC」を指定し、日付書式に「%Y%m%d.csv」と指定してください。

ファイルバックアップアクション

ファイルバックアップアクションは、指定したフォルダ内のファイルやフォルダをバックアップ先と同じ状態に保持します。

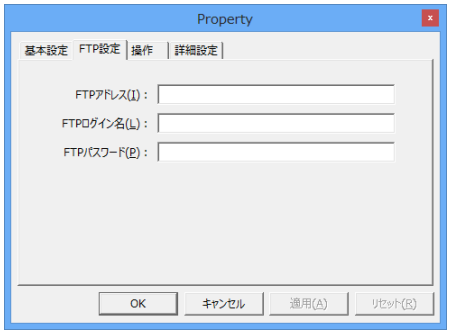
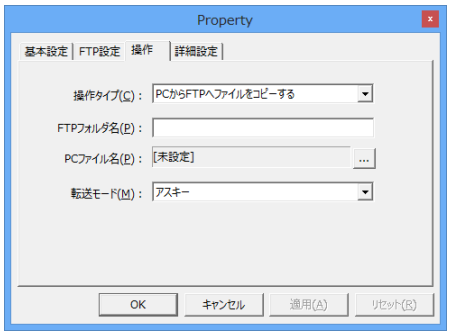
このアクションを使用したサンプルについては、本マニュアルの「実践編」「ファイルバックアップ」項目を参照してください。

アクション仕様	
パラメータ	なし
ダイアログ	■バックアップ指定  ▽バックアップ元のファイルが削除された場合、バックアップ先のファイルも削除する 上記チェックを行わないと、バックアップ元のファイルやフォルダが削除されたときに、バックアップ先のファイルやフォルダを削除しません。

FTP 転送アクション

FTP 転送アクションは FTP を用いてファイルを転送します。

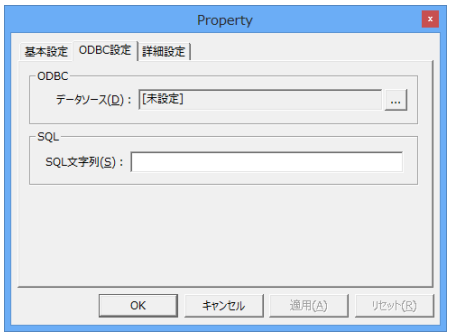
このアクションを使用したサンプルについては、本マニュアルの「実践編」「ファイルを FTP 転送する」項目を参照してください。

アクション仕様	
パラメータ	なし
ダイアログ	■FTP 設定  ▽FTP アドレス：FTP サーバのアドレス ▽FTP ログイン名：ログイン名 ▽FTP パスワード：パスワード ■操作  ▽操作タイプ 下記から選択します。 ・PC から FTP へファイルをコピーする ・PC から FTP へフォルダの内容をコピーする ・FTP から PC へファイルをコピーする ・FTP から PC へフォルダの内容をコピーする ・FTP のファイルを削除する ・FTP のフォルダを削除する ・FTP のフォルダの内容を削除する ▽FTP フォルダ名：FTP のフォルダへのパス ▽PC ファイル名：「アクション共通の設定」「名前フォーマット」を参照ください。 ▽転送モード：アスキーとバイナリーを選択できます。

SQL 実行（ODBC）アクション

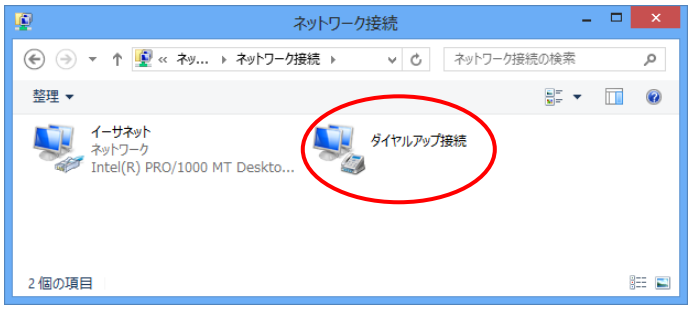
ODBC データベースに対して SQL 文を実行します。

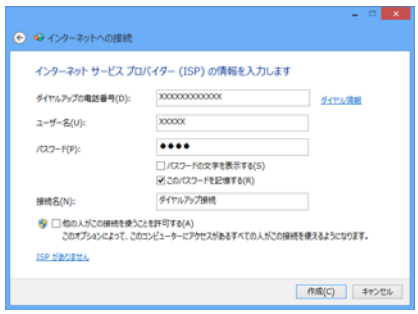
このアクションを使用したサンプルについては、本マニュアルの「実践編」「SQL を実行して DB のレコードを追加/削除する」項目を参照してください。

アクション仕様	
パラメータ	なし
ダイアログ	■ODBC 設定  ▽データソース 「アクション共通の設定」「ODBC 設定」を参照ください。 ▽SQL 文字列 SQL の文字列を記述します。 例）DELETE FROM TABLE01;

ダイアルアップアクション

ダイアルアップアクションはモデム等を用い、ダイアルアップ接続をおこないます。
コントロールパネルの「ネットワークと共有センター」で接続先を設定し、実際に接続を確認して下さい。





コントロールパネルで設定したエントリー（Xxxx）をダブルクリックし、実際に接続を確認して下さい。
接続 OK であれば、ダイアルアップアクションに同じユーザー名、パスワード、電話番号を設定してください。

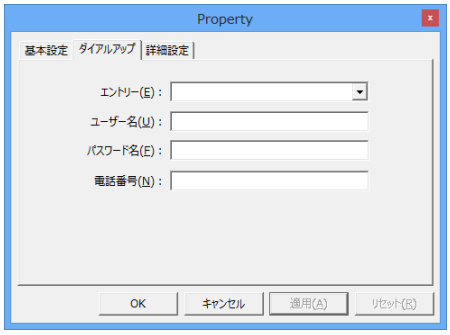
このアクションを使用したサンプルについては、本マニュアルの「実践編」「ダイアルアップ接続/切断を行う」項目を参照してください。

アクション仕様

パラメータ	パラメータ文字	動作
	DIAL（もしくは空文字）	ダイアルアップを行います。
	HANGUP	切断します。

ダイアログ

■ダイアルアップ

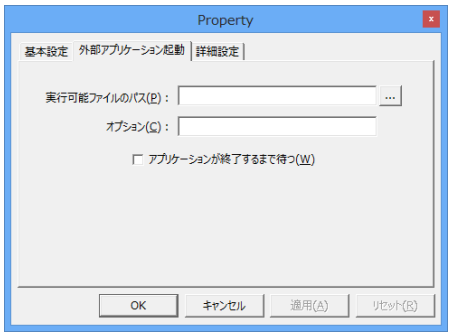


▽エントリー：「ネットワークとダイアルアップ接続」で設定した接続先を指定して下さい。
▽ユーザー名：接続を確認した設定と同じ設定をして下さい。
▽パスワード：接続を確認した設定と同じ設定をして下さい。
▽電話番号：接続を確認した設定と同じ設定をして下さい。

外部アプリケーション実行アクション

外部のアプリケーションを実行し、外部アプリケーションと連携することができます。

このアクションを使用したサンプルについては、本マニュアルの「実践編」「外部アプリケーションを実行」項目を参照してください。

アクション仕様	
パラメータ	なし
ダイアログ	■外部アプリケーション起動  ▽実行可能ファイルのパス 外部アプリケーションの実行可能ファイルのパスです。ファイルパスに空白が含まれる場合であっても、ダブルクォーテーション「"」でくくる必要はありません（オプションとは異なる点に注意してください)。 ▽オプション 起動時にオプションが必要な場合に設定します（下記ヒントを参照ください)。 ▽アプリケーションが終了するまで待つ このオプションにチェックを入れると、起動した外部アプリケーションが終了するまでアクションを終了せずに待機します。



ほとんどのアプリケーションでは、「オプション」に自動的に開くファイルを設定できます。ここで、オプションで指定するファイルパスにスペースが含まれる場合は、ダブルクォーテーション「"」でくくって下さい。
例）"c:¥Program Files¥App.exe"



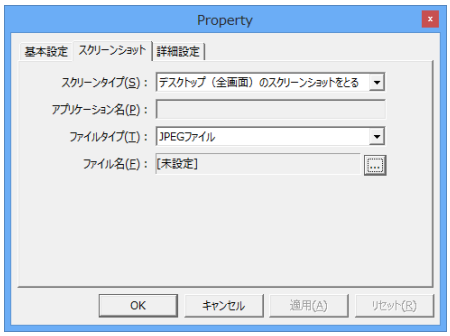
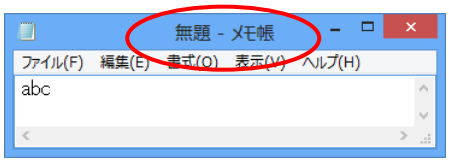
バッチ転送アクションと独自の外部アプリケーションを組み合わせて利用する事で、システム構成のバリエーションが広がります。
例えば、データの加工や複雑な集計処理などのバッチ的な処理を外部アプリケーションに行わせるといった機能実装も可能です。具体的には、タグの情報をテキストファイルやデータベースへ保存した後、外部のアプリケーションを起動します。起動された外部アプリケーションはテキストファイルやデータベースを介して先ほどのタグの情報を取得し、データの加工や集計などのバッチ的な処理を行います。ここで、集計した結果を再びタグへ転送したい場合には、「アプリケーションが終了するまで待つ」をチェックしておき、外部アプリケーションが終了した後に、再びバッチ転送アクションによって外部アプリケーションが加工したデータを取得し、タグに取り込みます。

スクリーンショットアクション

Windows アプリケーションの画面をキャプチャーし、JPEG もしくは BITMAP ファイルを作成します。

このアクションを使用したサンプルについては、本マニュアルの「実践編」「画面全体のスクリーンショットを取得」項目を参照してください。

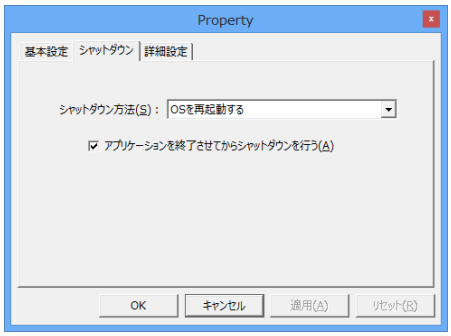
アクション仕様

パラメータ	なし
ダイアログ	■スクリーンショット  ▽スクリーンタイプ ・このアプリケーションのスクリーンショットをとる ・デスクトップ（全画面）のスクリーンショットをとる ・指定したアプリケーション名のスクリーンショットをとる ▽アプリケーション名 アプリケーションのウインドウの名前です。下記の例では「無題 – メモ帳」になります。  ▽ファイルタイプ JPEG か BITMAP かを選択します。 ▽ファイル名 「アクション共通の設定」「名前フォーマット」を参照ください。

シャットダウンアクション

アプリケーションや OS をシャットダウンします。

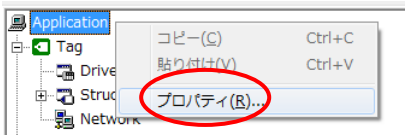
このアクションを使用したサンプルについては、本マニュアルの「実践編」「Server アプリケーションを終了させる」項目を参照してください。

アクション仕様	
パラメータ	なし
ダイアログ	■シャットダウン  ▽シャットダウン方法 ・OS をログオフする ・OS をログオフする（強制） ・OS を再起動する ・OS を再起動する（強制） ・OS をシャットダウンする ・OS をシャットダウンする（強制） ・OS をシャットダウン後、電源を切る ・OS をシャットダウン後、電源を切る（強制） ・アプリケーションを終了する ・アプリケーションを再起動する（連続稼働安全対策のチェックが必要）

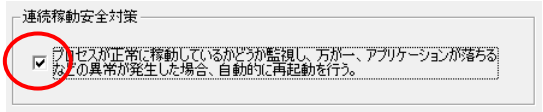


「アプリケーションを再起動する」を定刻イベントなどで実行することで、定期的に Server アプリケーションを再起動することが出来ます。この機能は、「連続稼働安全対策」の設定が必要です。

1. Application のプロパティを開く



2. 基本設定の「連続稼働安全対策」にチェックを入れる



時刻同期アクション

PC の時刻と PLC の時刻の同期を行います。

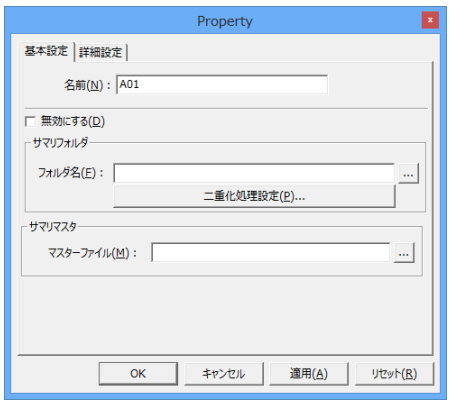
このアクションを使用したサンプルについては、本マニュアルの「実践編」「PC と PLC の時刻を同期させる」項目を参照してください。

アクション仕様

パラメータ	なし
ダイアログ	■時刻同期  ▽同期方法 ・パソコンの時刻を PLC（タグ）へ設定する ・PLC（タグ）の時刻をパソコンへ設定する（タグの値はキャッシュ値） ・PLC（タグ）の時刻をパソコンへ設定する（タグの値はリアル値） ▽年、月、日、時、分、秒 設定したい時刻の種類をチェックします。たとえば時間と分と秒のみ同期を行いたい場合は時、分、秒を選択します。 ▽タグパス 同期をとるタグ名です。例）U01.F01.T01

サマリアクション（CSV/ODBC）

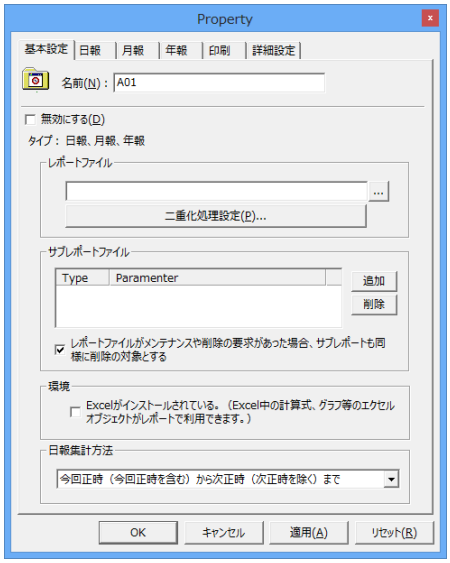
ログ、集計、日報月報年報作成を行います。詳しくはサマリマニュアルを参照下さい。

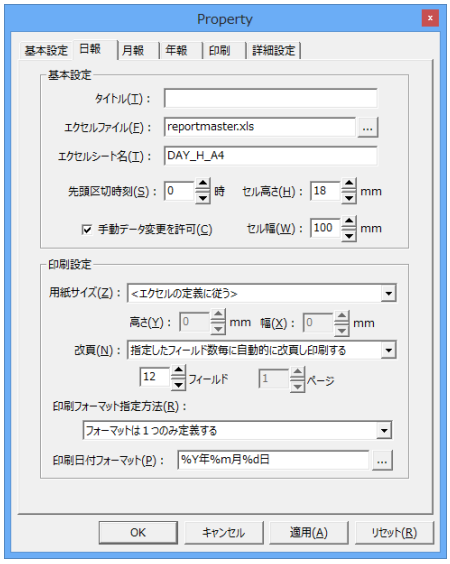
アクション仕様	
パラメータ	詳しくはサマリマニュアルを参照下さい。
ダイアログ	■基本設定  ▽サマリフォルダ サマリマニュアル「サマリアクションの設定方法について」を参照して下さい。 ▽マスターファイル サマリマニュアル「サマリアクションの設定方法について」を参照して下さい。

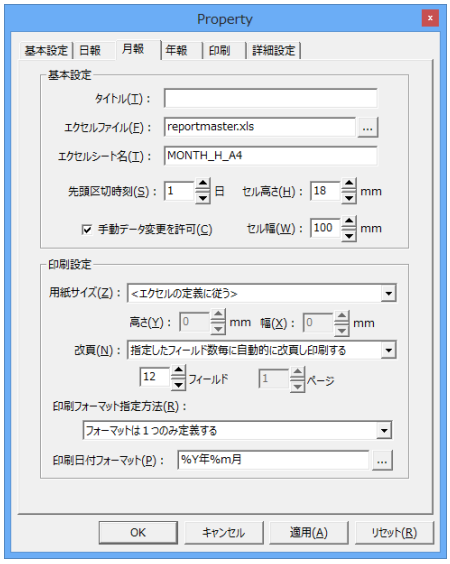
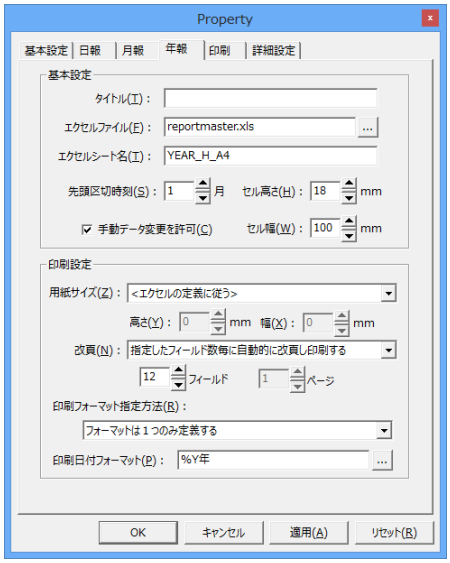
日報月報年報アクション

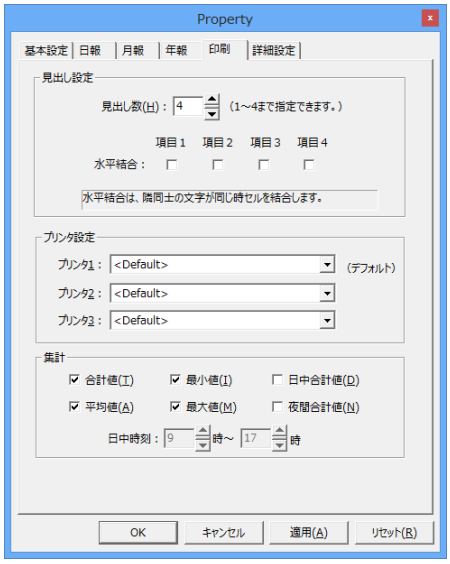
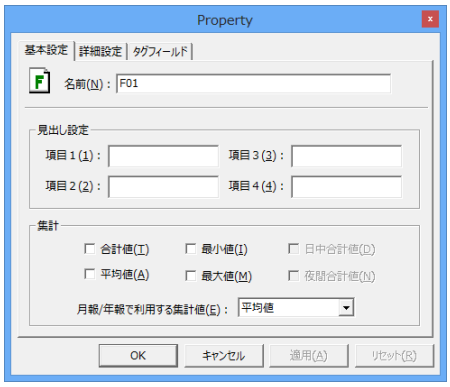
日報月報年報を作成するためのアクションです。

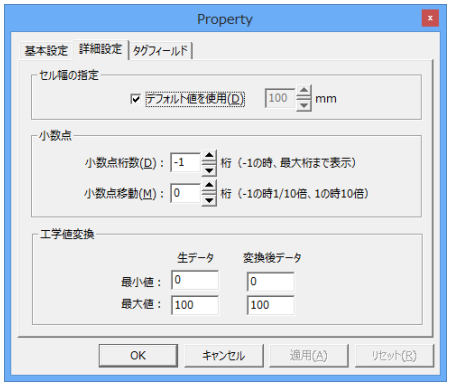
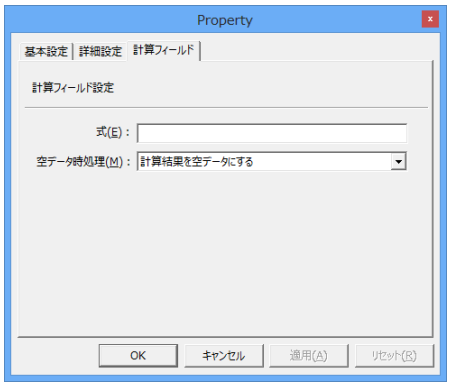
当アクションの利用に際しては、レポートのマニュアルも併せて参照してください。具体的な利用方法が解説されています。

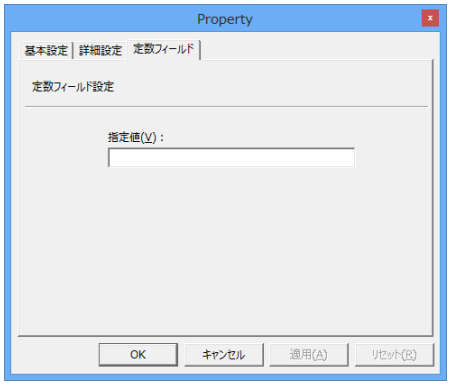
アクション仕様	
パラメータ	詳しくはレポートマニュアル「日報月報年報アクションの実行パラメータ」を参照してください。 ・CREATE(レポートタイプ, 開始時刻, 終了時刻, 集計が終わったものを除く) レポートの作成を行います（レポートファイルの集計処理を実行します)。 ・CREATESUB(レポートタイプ, 開始時刻, 終了時刻, サブレポートインデックス) サブレポートの作成を行います（レポートファイルの集計処理は行いません)。 ・PRINT(レポートタイプ, 開始時刻, 終了時刻, プリント番号) レポートの印刷を行います。 ・MAINTENANCE (レポートタイプ, レポート保持日数) レポートのメンテナンスを行います（レポートファイルの自動削除を行います)。
ダイアログ	■基本設定  ▽レポートファイル レポートマニュアル「日報月報年報で作成されるファイル」を参照してください。 ▽サブレポートファイル レポートマニュアル「サブレポートについて」を参照してください。 ▽Excel がインストールされている レポートマニュアル「グラフ、計算式などを使用したい場合」を参照してください。 ▽日報集計方法 レポートマニュアル「日報の集計方法と区切り時刻について」を参照してください。

アクション仕様	
ダイアログ	■日報  ▽タイトル 印刷時に使用されるタイトルです。 ▽エクセルファイル レイアウトマスターファイルとなるエクセルのファイル名を指定します。詳しくはレポートマニュアル「レイアウトマスターファイル」を参照下さい。 ▽エクセルシート名 日報のレイアウトが含まれるシート名です。詳しくはレポートマニュアル「レイアウトマスターファイル」を参照下さい。 ▽先頭区切時刻 レポートマニュアル「日報の集計方法と区切り時刻について」を参照してください。 ▽手動データ変更を許可 日報を画面から変更できるようにする場合はチェックしてください。 ▽セル高さ・セル幅 日報を画面で表示するときのセルの高さと幅を指定します。 （※FA-Panel のレポートクライアントグリッドコントロールの場合、この設定は無視されます。高さと幅はコントロールのプロパティで設定します） ▽用紙サイズ、高さ、幅 印刷する用紙のサイズ。「<エクセルの定義に従う>」を選択すると、レイアウトファイルのプリンタの設定に従い用紙が決定されます。 ▽改頁、印刷フォーマットの指定方法 レポートの改ページ設定です。レポートマニュアル「複数ページにわたるレポートのレイアウト」を参照してください。 ▽印刷日付フォーマット 印刷で使用される日付のフォーマット。デフォルトでは「xxxx 年 xx 月 xx 日」と印刷されます。

アクション仕様	
ダイアログ	■月報  ▽日報を参照下さい。 ■年報  ▽日報を参照下さい。

アクション仕様	
ダイアログ	■印刷  ▽見出し数 見出しとして表示したい行数。見出しに出力する内容はフィールドの設定で行います。 ▽水平結合 水平結合したい項目（見出し）をチェックします。隣同士の見出しの文字が同じ場合、セルが結合されます。 ▽プリンタ設定 印刷先のプリンタを選択します。何も指定しなければデフォルトプリンタに出力されます。 ▽集計 集計を行いたい項目をチェックします。ここでチェックされた項目は必ず印刷されます。尚、集計の設定はフィールド毎の個別の設定側でも行います。こちらの設定で集計にチェックを入れた状態で、フィールド側では集計を行わない設定とした場合、そのフィールドの集計欄（セル）は空白として印字されます。 ■基本設定（フィールド）  ▽見出し設定 見出し（見出しではカンマ「,」は使用できません）。 ▽集計 フィールド毎の集計項目を選択します。 ▽月報・年報で利用する集計値 月報、年報に展開するときに利用する集計値。

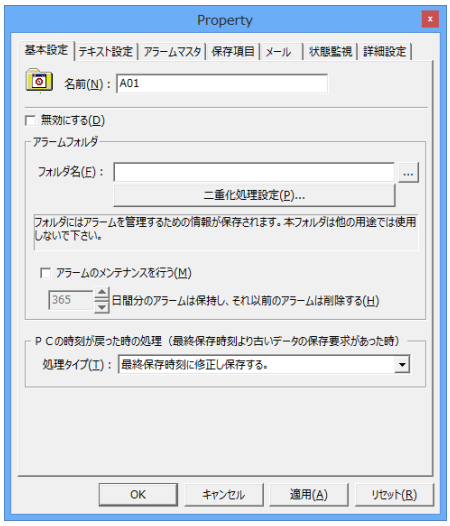
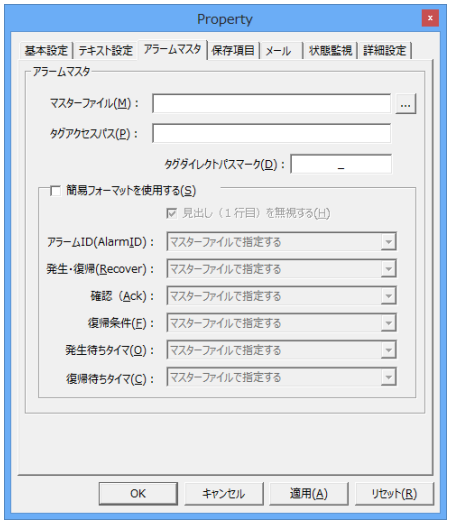
アクション仕様	
ダイアログ	■詳細設定（フィールド）  ▽セル幅の指定 日報を画面で表示するときのセルの幅をフィールド毎に設定したい場合には、「デフォルト値を使用」のチェックを外して幅を設定します。 ▽小数点 レポートマニュアル「印刷時の表示桁数について」を参照してください。 ▽工学値変換 タグの値を工学値変換したい場合設定してください。 ■計算フィールド  ▽式 フィールド同士を計算します。例）F01+F02　スクリプト Ver1 アクションで使用できるオペレータが使用できます。 ▽空データ時処理 計算元となるフィールドに値が存在しない場合の計算方法です。

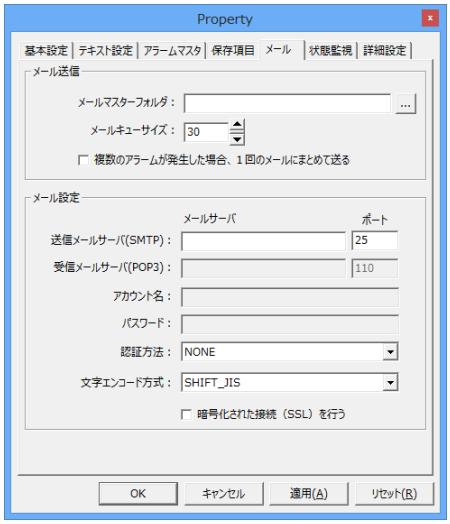
アクション仕様	
ダイアログ	■定数フィールド  ▽指定値 指定値で設定した値が格納されます。なにも指定しないと空文字（なにも表示されない）になります。

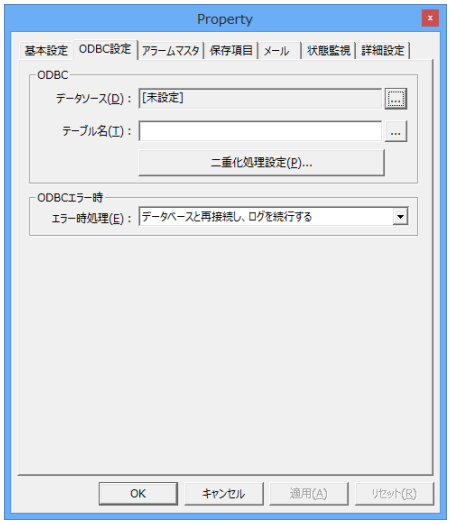
アラームサーバーアクション

FA-Panel 用のアラームサーバーです。詳しくは「アラームマニュアル」を参照下さい。

アクション仕様	
パラメータ	1）なにも指定しない場合 キューに格納されたアラームをチェックし、アラームが存在する場合にはアラームを発生させ、クライアントに通知します（※アラームの検出はアラームサーバーの実行とは別に行われ、検出されたアラームは一旦キューに格納された状態になります)。 2）「MAIL」を指定した場合 メールキューにたまったアラームメールを送信します。「複数のアラームが発生した場合、１回のメールにまとめる」がチェックされていると、アクション実行のタイミングでメールキューにたまっているアラームを１回のメールにまとめて送信します。 3）「RELOADMAILMASTER」を指定した場合 メールマスターファイルを再読み込みします。メールの送信先などをシステム運用中に変更や追加を行いたい場合には、まずメールマスターファイルを Panel Browser や VB などの外部アプリケーションで編集して保存します。続いて RELOADMAILMASTER パラメータを付加したイベント（タグイベントなど）でアラームサーバーアクションを実行すると、編集したメールマスターファイルが再読み込みされます。再読込み後、MAIL パラメータを指定したイベントでアラームサーバーアクションを実行すると、編集後の内容でメールが送信されます。 4）「RELOADALARMMASTER」を指定した場合 アラームマスターファイルを再読み込みします。アラーム条件などをシステム運用中に変更や追加をしたい場合は、まずアラームマスターファイルを編集して保存します。その後、RELOADALARMMASTER パラメータを付加したイベント（タグイベントなど）でアラームアクションを実行すると、アラームマスターファイルが再読み込みされ、編集後の内容でアラームの監視が行われます。 5）「ACKALL」を指定した場合 アラームの確認を制御盤上のボタン等の外部デバイスで行いたい場合､タグイベントで ACKALL パラメータを指定してアクションを実行します。

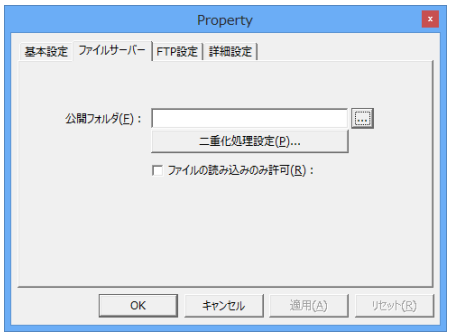
アクション仕様	
ダイアログ	■基本設定  ▽フォルダ名 アラームを管理するための情報を保存するために使用する専用のフォルダです。複数のアラームサーバーを設定するときは、各サーバで個別のフォルダを指定してください。
	▽アラームファイルのメンテナンスを行う 自動的に古いアラーム情報を削除します。 ▽ＰＣの時刻が戻った時の処理（最終保存時刻より古いデータ保存要求があった時） ＰＣの時刻合わせ等で時刻が戻された場合の処理です。詳しくはアラームマニュアル「ＰＣの時刻が戻った時の処理」を参照してください。 ■アラームマスタ  アラームマニュアル「アラームマスタのフォーマット」を参照下さい。

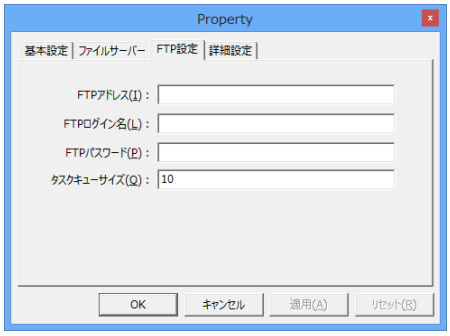
アクション仕様	
ダイアログ	■保存項目  アラームマニュアル「アラームマスタのフォーマット」を参照下さい。 ■メール アラームマニュアル「アラームメールを送信する」を参照下さい。 ■テキスト設定　（アラームサーバーアクションのみ） 

アクション仕様	
ダイアログ	▽フラッシュ設定 アラームサーバーアクションの場合、アラーム情報はテキストファイルへ書き込まれます。書き込まれるアラーム情報は、ここで指定した件数分まとめてハードディスクに書き込まれます。アラームの発生後、即座にファイルに書き込みたい場合には 1 を指定します。但し、1 を指定した場合には、ハードディスクへ毎回アクセスすることになるため、HDD の負荷が大きくなります。HDD の負荷を減らしたい場合はフラッシュの設定に 2 以上の数を設定してください。例えば 30 を設定すると、30 個のアラームが発生した後でまとめてハードディスクへ書き込みます。フラッシュの設定に 2 以上の設定をした場合、サーバーが正常終了した場合であればメモリ上に残った情報は確実に HDD へ保存されますが、停電などにより PC の電源が切れた時など、何らかの原因でシステムが異常終了してしまった場合には一部のアラーム情報が HDD へ保存されない可能性があります。フラッシュ設定を有効にしてかつアラーム情報を確実に保存させたい場合には、UPS 等を使用して停電時でもシステムが確実にシャットダウンされるようにして下さい。 ■ODBC 設定　（アラームサーバー（ODBC）アクションのみ）  アラームの情報をデータベースに保存したいときは「アラームサーバー（ODBC）」アクションを選択します。設定方法については「アクション共通の設定」「ODBC 設定」を参照ください。 アラーム情報を保存するデータベースには、下記のフィールドが設定されている必要があります。フィールド名は必ず一致させてください。 フィールド名 / フィールド型 ALARMDATE / 日付型 ALARMTYPE / テキスト型 ALARMID / 数値型 ALARMACK / 数値型 ALARMRECOVER / 数値型 ALARMCONDID / 数値型 尚、データベースの設定サンプルとして、Access データベースの設定例が以下に格納されています。 「¥¥sample¥alarm¥sample02¥alarm.mdb」

ファイルサーバーアクション

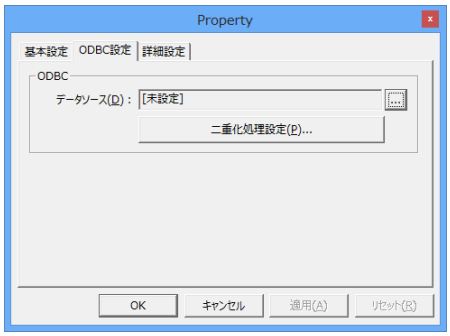
FA-Panel 用のファイルサーバーです。Panel Browser（ファイル転送コントロール）から Panel Server にファイルの送信や受信の指示をすることができます。詳しくはコントロールリファレンス「ファイル転送コントロール」を参照下さい。

アクション仕様	
パラメータ	なし
ダイアログ	■ファイルサーバー  ▽公開フォルダ 公開したいフォルダを設定します。 ▽二重化処理設定 次のメソッドの動作を二重化します。 **ReceiveFile/SendFile/CopyFile/CopyFileTime/RenameFile/DeleteFile/MakeFolder** **CopyFolder/RenameFolder/DeleteFolder** 例えば、CopyFile を行うと、メインとサブの両方のフォルダで実行されます。 もし、ファイルの書き込みに失敗した場合、復帰次第自動的にファイルが復帰処理されます。 例えば SendFile で送ったファイルが、片側のディスクしか書き込めなかった場合、故障したディスクが復帰した時、自動的に書き込めなかったディスクへファイルがコピーされます。 ▽ファイルの読み込みのみ許可 この設定にチェックを入れると、Panel Browser からフォルダの内容を変更できなくさせることができます。

アクション仕様	
ダイアログ	■FTP 設定  ▽FTP アドレス/FTP ログイン名/FTP パスワード ファイル転送コントロールを使用する場合、アクション側で FTP の設定を行いたい場合、設定を行います。その際はファイル転送コントロールでは、接続時「FTPSetServer("", "", "")」と引数を省略して呼び出します。 ▽タスクキューサイズ FTPReceiveFile/FTPSendFile メソッドで RequestID に 1 以上を設定し非同期通信を行う場合、同時に受け付ける FTPReceiveFile/ FTPSendFile 呼び出しの最大回数を指定します。

クエリーサーバーアクション

FA-Panel 用のクエリーサーバーです。Panel Browser（リモートクエリーデータベースコントロール）から Panel Server を通じてデータベースへクエリーを実行し、Panel Browser で結果を受け取ることができます。詳しくはオンラインコントロールリファレンス「リモートクエリーデータベースコントロール」を参照下さい。

アクション仕様	
パラメータ	なし
ダイアログ	■ODBC 設定  設定方法については「アクション共通の設定」「ODBC 設定」を参照ください。

3章　スクリプト Ver2 アクションについて

概要

スクリプト Ver2（以降 SC2 構文と呼ぶ）は高度な処理を目的とした本格的なスクリプト構文です。Panel Server にはスクリプト Ver2 アクションが搭載されており、定義したスクリプトを実行することができる機能が備わっています。この機能を用いることで、複雑な条件分岐や高度な計算を行うことが可能になります。

本章ではスクリプト Ver2 アクションを用いたサンプルについて説明します。

スクリプト Ver2 アクションの概要については、本マニュアルの「アクションリファレンス」「スクリプト Ver2」を参照してください。

スクリプト Ver2 の詳細な構文については、「スクリプトマニュアル」「スクリプト Ver2（SC2 構文）」項目を参照してください。

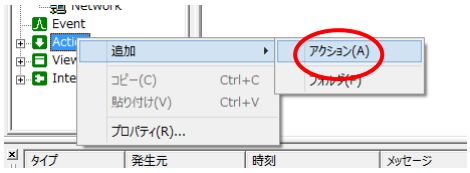
設定手順

タグ設定

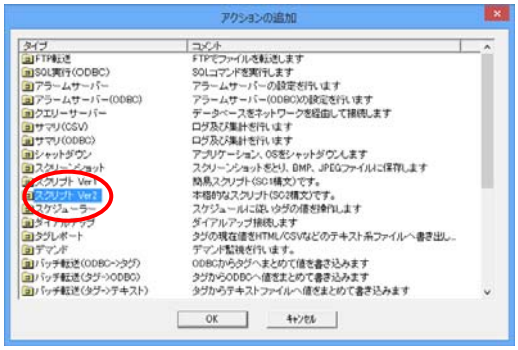
スクリプト Ver2 で使用するタグを追加してください。

アクション設定

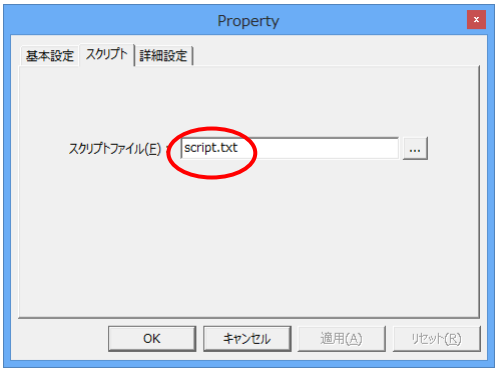
Action を右クリックして表意されるメニューから、「追加」-「アクション」を選択します。



スクリプト Ver2 を選択して、追加します。

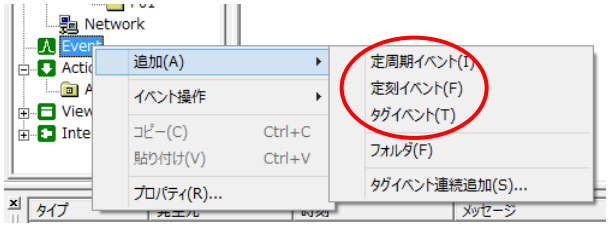


追加したアクションのプロパティを開き、スクリプトファイルを選択します。



イベント設定

追加したアクションをイベントに登録します。定周期/定刻/タグのイベントのうち、使用方法に該当するイベントを選択し、アクションを登録してください。
定周期でスクリプトを実行したい場合は、定周期イベントを選び、定刻でバッチ処理のようなことを行いたい場合は、定刻イベントを選んでください。また、タグの値をトリガにして実行したい場合は、タグイベントを選んでください。



ヒント

スクリプトの実体は、スクリプト用の外部ファイルを用意することになるため、アクションの設定自体は、スクリプトのファイルを選択するのみです。

3－1　基本的なスクリプトの実行方法について（Sample01）

概要

SC2 構文は定義されたイベントをトリガにして、記述されたスクリプトを実行させていきます。

本サンプルは、SC2 構文で使用できる基本的な各種イベントが定義されており、どのような状態でイベントが発生し、記述したスクリプトが実行されるかを確認することができます。

サンプルでは、
OnInitialize/OnTerminate/OnTimer/OnTagValueChanged/OnHeartBeat/OnExpressionValueChanged
イベントの実行を確認することができます。

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥script2¥sample01¥script2_sample01.txt
スクリプト本体：¥¥sample¥common¥action¥script2¥sample01¥script.txt



サンプルで使用している各イベントやメソッドの詳細については、オンラインコントロールリファレンスを参照してください。



OnHeartBeat イベントはイベントによりアクションが実行される毎に必ず呼び出されます。尚、定周期的な処理を実装する場合には、SetTimer メソッドによる OnTimer イベントの使用を推奨します。SetTimer には、複数のタイマーを同時に起動する、個々の周期を設定できるなどのメリットがあります。
例えば、定周期的な処理は OnTimer イベントを使用し、定刻イベントなどでスクリプトを実行したい場合は、OnHeartBeat イベントを使用するといった使い分けをしてください。



OnTimer イベントは、ScriptVer2 アクションがイベントにより呼ばれた時に、条件（OnTimer で設定した周期が経過している）を満たしていた場合に実行されます。
例えば、イベントの周期：10 秒　OnTimer の周期：11 秒と指定している場合、最初の実行から 11 秒後に OnTimer の処理がされるのではなく、下記のようにイベントで ScriptVer2 が再び呼ばれた段階で、処理が実行されます。

1：初回実行（0 秒）→OnTimer のカウント開始。
2：イベントで 1 度目の実行（10 秒経過）→周期は 11 秒なので、条件が不成立の為まだ処理はされない。
3：OnTimer の周期経過（11 秒経過）→ScriptVer2 は呼び出されていない為まだ実行されない。
4：イベントで 2 度目の実行（20 秒経過）→OnTimer の設定周期を超えている為、条件が成立し、処理が実行される。

イベント毎に OnTimer の処理を行いたい場合は、イベントの周期よりも短い周期を OnTimer の周期とする必要があります。

解説

スクリプトの定義ファイルに記述されているスクリプトについて説明します。ここでは、スクリプトを実行するためのトリガとなる各イベントについて主に説明します。

1）OnInitialize イベント

OnInitialize イベントはオンライン化した直後に実行されるイベントです。
初期化処理など、初期処理を行う際に使用します。

処理内容

① SvsDump メソッドは、OutPutLog にメッセージを出力するメソッドです。ここでは、OnInitialize イベントが実行されたことをログ出力しています。
② AddTag メソッドを用いて、使用するタグの登録を行います。
③ AddExpression メソッドで式の登録を行います。AddExpression メソッドで登録した式の値が変化すると、OnExpressionValueChanged イベントが発生されます。
④ SetTimer でタイマーのセットを行います。セットしたタイマーのタイムアップ時に OnTimer イベントが発生されます。ここでは、無制限で 5 秒周期のタイマーをセットします。

Script

```
event OnInitialize()
{
    ① ::SvsDump("OnInitialize:" & ::SvsGetActionName());
    ② ::AddTag(c("U01.F01.T01", "U01.F01.T02", "U01.F01.T03"));
    ③ var vExpressionName = c("EX01","EX02");
       var vExpression = c("(U01.F01.T04 && U01.F01.T05) ||
                            U01.F01.T06" ,"U01.F01.T04 || U01.F01.T05 || U01.F01.T06");
       ::AddExpression(vExpressionName, vExpression);
    ④ ::SetTimer(0, 5000, -1);
}
```

2）OnTerminate イベント

OnTerminate イベントはオフライン化すると実行されるイベントです。
終了時に何らかの処理を行いたい時に使用します。

処理内容

SvsDumpp メソッドで、OnTerminate イベントが発生したことをログ出力しています。

Script

```
event OnTerminate()
{
        ::SvsDump("OnTerminate:" & ::SvsGetActionName());
}
```

3）OnTimer イベント

OnTimer イベントは SetTimer でセットされたタイマーのタイムアップ時に OnTimer イベントが発生されます。
定周期的な処理を行いたい場合に使用します。

動作内容

SvsDumpp メソッドで、OnTimer イベントが発生したことをログ出力しています。

Script

```
event OnTimer(timerid,counter)
{
        ::SvsDump("OnTimer Id:" & ::CStr(timerid) & " counter:" & ::CStr(counter));
}
```

4）OnTagValueChanged イベント

OnTagValueChanged イベントは AddTag で登録されたタグの値が変化したときに発生します。
タグの値の変化があった際に、処理を行いたい場合に使用します。

OnInitialize イベントで、「U01.F01.T01」、「U01.F01.T02」、「U01.F01.T03」の 3 つのタグを登録しているので、いずれかのタグの値が変化することで、OnTagValueChanged イベントが発生します。

処理内容

① SvsDumpp メソッドで、OnTagValueChanged イベントが発生したことをログ出力しています。
② 値が変化したタグが「U01.F01.T01」だった場合、「U01.F01.T03」に 1 を書き込みます。また、値が変化したタグが「U01.F01.T02」だった場合、「U01.F01.T03」に 0 を書き込みます。

Script

```
event OnTagValueChanged(tagname,value)
{
    ① ::SvsDump("OnTagValueChanged(Tag:" & tagname & " Value:" & ::CStr(value) & ")");

    ② if(value){
            if(tagname == "U01.F01.T01"){
                ::WriteVal("U01.F01.T03", 1);
            }else if(tagname == "U01.F01.T02"){
                ::WriteVal("U01.F01.T03", 0);
            }
      }
}
```

5）OnHeartBeat イベント

OnHeartBeat イベントはスクリプト Ver2 アクションが呼び出されるたびに発生するイベントです。
定刻イベントなどと組み合わせて、一定の期間ごとに処理を行いたい場合に使用します。

サンプルでは 1 秒周期でアクションが実行されるため、 1 秒毎に OnHeartBeat イベントが発生されます。

処理内容
SvsDumpp メソッドで、OnHeartBeat イベントが発生したことをログ出力しています。

Script

```
event OnHeartBeat()
{
        SvsDump("OnHeartBeat:" + SvsGetActionName());
}
```

6）OnExpressionValueChanged イベント

OnExpressionValueChanged イベントは AddExpression を使用して登録した式の値に変化があったときに発生します。

OnInitialize イベントで、式の登録を行っているため、登録された式の値が変化するとイベントが発生します。

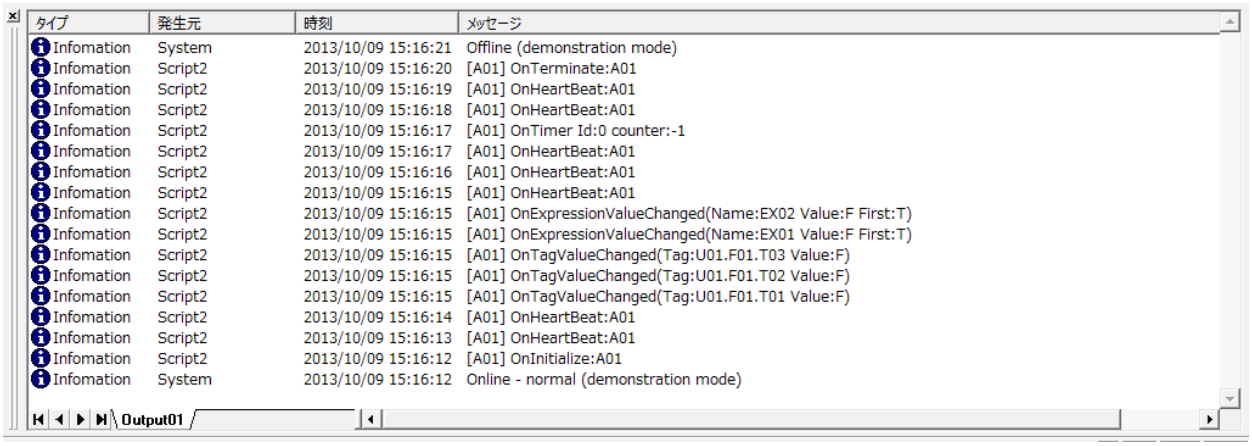
処理内容
SvsDumpp メソッドで、OnExpressionValueChanged がどのような内容で発生したのかをログ出力しています。

Script

```
event OnExpressionValueChanged(name,value,firstevent)
{
        ::SvsDump("OnExpressionValueChanged(Name:" & ::CStr(name) &
                " Value:" & ::CStr(value) & " First:" & ::CStr(firstevent) & ")");
}
```

動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、OutPutLog にスクリプトの動作結果がログとして出力されます。



タイプ	発生元	時刻	メッセージ
Infomation	System	2013/10/09 15:16:21	Offline (demonstration mode)
Infomation	Script2	2013/10/09 15:16:20	[A01] OnTerminate:A01
Infomation	Script2	2013/10/09 15:16:19	[A01] OnHeartBeat:A01
Infomation	Script2	2013/10/09 15:16:18	[A01] OnHeartBeat:A01
Infomation	Script2	2013/10/09 15:16:17	[A01] OnTimer Id:0 counter:-1
Infomation	Script2	2013/10/09 15:16:17	[A01] OnHeartBeat:A01
Infomation	Script2	2013/10/09 15:16:16	[A01] OnHeartBeat:A01
Infomation	Script2	2013/10/09 15:16:15	[A01] OnHeartBeat:A01
Infomation	Script2	2013/10/09 15:16:15	[A01] OnExpressionValueChanged(Name:EX02 Value:F First:T)
Infomation	Script2	2013/10/09 15:16:15	[A01] OnExpressionValueChanged(Name:EX01 Value:F First:T)
Infomation	Script2	2013/10/09 15:16:15	[A01] OnTagValueChanged(Tag:U01.F01.T03 Value:F)
Infomation	Script2	2013/10/09 15:16:15	[A01] OnTagValueChanged(Tag:U01.F01.T02 Value:F)
Infomation	Script2	2013/10/09 15:16:15	[A01] OnTagValueChanged(Tag:U01.F01.T01 Value:F)
Infomation	Script2	2013/10/09 15:16:14	[A01] OnHeartBeat:A01
Infomation	Script2	2013/10/09 15:16:13	[A01] OnHeartBeat:A01
Infomation	Script2	2013/10/09 15:16:12	[A01] OnInitialize:A01
Infomation	System	2013/10/09 15:16:12	Online - normal (demonstration mode)

Output01

3－2　スクリプト Ver2 の DB 連携について（Sample02）

概要

SC2 構文は DirectQueryDB コントロールを使用することで、ODBC 経由でデータベースに接続を行い、SQL を実行することができます。

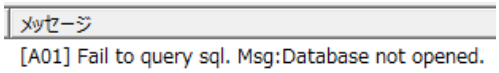
サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥script2¥sample02¥script2_sample02.txt
スクリプト本体：¥¥sample¥common¥action¥script2¥sample02¥script.txt
サンプルデータベース：¥¥sample¥common¥action¥script2¥dbsample.mdbxt



サンプルで使用している各イベントやメソッドの詳細については、オンラインコントロールリファレンスを参照してください。



本サンプルはデータベースにアクセスするため、Microsoft 社の Access を使用しています。そのため、Access がインストールされていない環境で動作させると、下記エラーメッセージが表示され、データベースのオープンに失敗します。



サンプルをそのまま動作させる場合は Access をインストールが必要です。もしくは、他の DB を導入し、ODBC の設定を行う必要があります。

ODBC の設定については、「Appendix： アクション共通の設定」「ODBC 設定」項目を参照してください。

解説

スクリプトの定義ファイルに記述されているスクリプトについて説明します。ここでは、データベースへのアクセス方法を説明します。

1）初期処理

初期処理は OnInitialize イベントで行います。

初期処理として New 演算子を使用して、DirectQueryDB コントロールオブジェクトの生成を行います。

処理内容

① New 演算子を使用し、DirectQueryDB コントロールのオブジェクトを生成します。
② SetTimer でタイマーのセットを行います。セットしたタイマーのタイムアップ時に OnTimer イベントが発生されます。ここでは、無制限で 5 秒周期のタイマーをセットします。
OnTimer イベント内で SQL を実行します。

Script

```
var objDB;

event OnInitialize()
{
        ::SvsDump("OnInitialize:" + ::SvsGetActionName());
    ① objDB = new DirectQueryDB;
    ② ::SetTimer(0, 5000, -1);
}
```

2）SQL 実行処理

SQL の実行は OnTimer イベント内で行います。

まず、DirectQueryDB コントロールが持っている Connect メソッドを使用して、データベースへ接続を行います。接続完了後、QuerySQL メソッドで SQL（SELECT によるテーブルデータの全件取得）を実行し、取得したデータを OutPutLog へ出力します。

処理内容

① データベースへ接続を行います。接続は、ユーザー関数「mConecctDB」で行います。mConecctDB では、接続先データベースの定義を行い、Connect メソッドで接続を行います。
② QuerySQL メソッドを実行し、指定の SQL「SELECT * FROM TABLE1」を実行します。
③ ②の SQL 実行結果を取得し、フィールド名とカラムデータを OutPutLog に出力を行います。

```
event OnTimer(timerid,counter)
{
        ::SvsDump("OnTimer Id:" + ::CStr(timerid) + " counter:" + ::CStr(counter));

①       if(!mConecctDB()){
                return;
        }
        var vFieldNum, vRecordNum, vFieldName, vFieldType, vData;
        if(!objDB.QuerySQL("SELECT * FROM TABLE1", vFieldNum,
②               vRecordNum, vFieldName, vFieldType, vData)){
                ::SvsDump("Fail to query sql. Msg:" + objDB.GetLastError());
                objDB.DisConnect();
                return;
        }

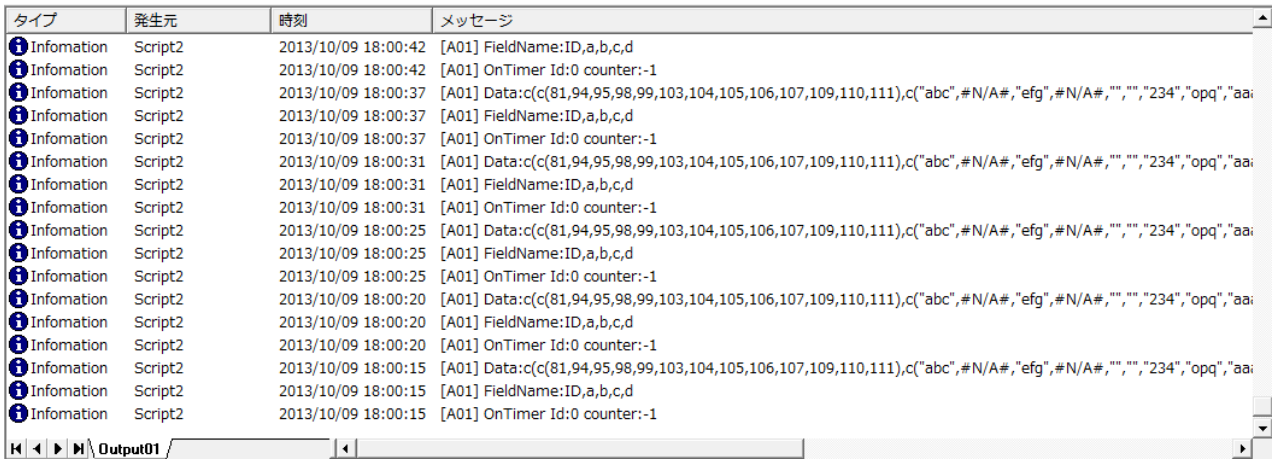
②      ::SvsDump("FieldName:" + ::CArrayToStr(vFieldName, ",", F));
        ::SvsDump("Data:" + ::CScript(vData));
}

function mConecctDB()
{
        var vDBInfo = "MS Access Database;" &
                    "DBQ=dbsample.mdb;" &
                    "DriverId=25;" &
                    "FIL=MS Access;" &
                    "MaxBufferSize=2048;" &
                    "PageTimeout=5;" &
                    "+DATASOURCE=MS Access Database";

        if(!objDB.Connect(vDBInfo, "", "", "")){
                ::SvsDump(objDB.GetLastError());
                return F;
        }
        return T;
}
```

動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、OutPutLog にスクリプトの動作結果がログとして出力されます。



タイプ	発生元	時刻	メッセージ
Infomation	Script2	2013/10/09 18:00:42	[A01] FieldName:ID,a,b,c,d
Infomation	Script2	2013/10/09 18:00:42	[A01] OnTimer Id:0 counter:-1
Infomation	Script2	2013/10/09 18:00:37	[A01] Data:c(c(81,94,95,98,99,103,104,105,106,107,109,110,111),c("abc",#N/A#,"efg",#N/A#,"","","234","opq","aa
Infomation	Script2	2013/10/09 18:00:37	[A01] FieldName:ID,a,b,c,d
Infomation	Script2	2013/10/09 18:00:37	[A01] OnTimer Id:0 counter:-1
Infomation	Script2	2013/10/09 18:00:31	[A01] Data:c(c(81,94,95,98,99,103,104,105,106,107,109,110,111),c("abc",#N/A#,"efg",#N/A#,"","","234","opq","aa
Infomation	Script2	2013/10/09 18:00:31	[A01] FieldName:ID,a,b,c,d
Infomation	Script2	2013/10/09 18:00:31	[A01] OnTimer Id:0 counter:-1
Infomation	Script2	2013/10/09 18:00:25	[A01] Data:c(c(81,94,95,98,99,103,104,105,106,107,109,110,111),c("abc",#N/A#,"efg",#N/A#,"","","234","opq","aa
Infomation	Script2	2013/10/09 18:00:25	[A01] FieldName:ID,a,b,c,d
Infomation	Script2	2013/10/09 18:00:25	[A01] OnTimer Id:0 counter:-1
Infomation	Script2	2013/10/09 18:00:20	[A01] Data:c(c(81,94,95,98,99,103,104,105,106,107,109,110,111),c("abc",#N/A#,"efg",#N/A#,"","","234","opq","aa
Infomation	Script2	2013/10/09 18:00:20	[A01] FieldName:ID,a,b,c,d
Infomation	Script2	2013/10/09 18:00:20	[A01] OnTimer Id:0 counter:-1
Infomation	Script2	2013/10/09 18:00:15	[A01] Data:c(c(81,94,95,98,99,103,104,105,106,107,109,110,111),c("abc",#N/A#,"efg",#N/A#,"","","234","opq","aa
Infomation	Script2	2013/10/09 18:00:15	[A01] FieldName:ID,a,b,c,d
Infomation	Script2	2013/10/09 18:00:15	[A01] OnTimer Id:0 counter:-1

Output01



サンプルでは SQL の実行結果を取得するため、QuerySQL メソッドを使用していますが、UPDATE や INSERT 文を発行する場合は、結果が必要ないため、ExecuteSQL メソッドを使用します。ExecuteSQL メソッドを使用する場合は、BeginTrans メソッドを使用して、トランザクション処理を行うことができます。

3－3　スクリプト Ver2 でヒストリカルデータを取得（Sample03）

概要

SC2 構文は HistoricalData コントロールを使用することで、ヒストリカルデータサーバから時系列データを取得することができます。

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥script2¥sample03¥script2_sample03.txt
スクリプト本体：¥¥sample¥common¥action¥script2¥sample03¥script.txt



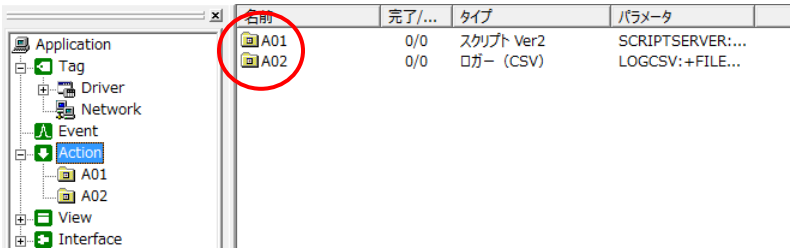
サンプルで使用している各イベントやメソッドの詳細については、オンラインコントロールリファレンスを参照してください。

解説

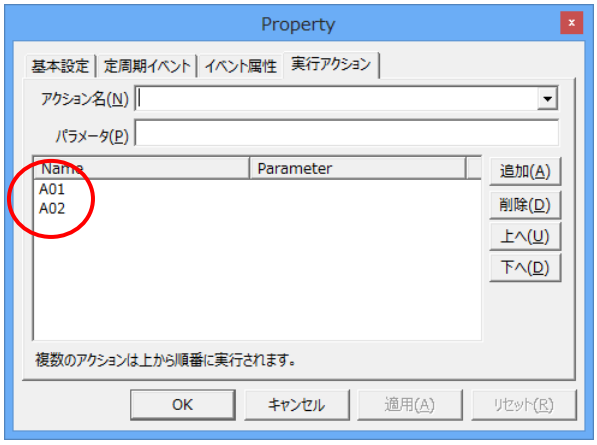
本サンプルではロガーアクションを使用しロギングを行います。そのロギングされたデータを、ロガーアクションが持っているヒストリカルデータサーバを利用して、SC2 構文から取得を行います。

A01：スクリプト Ver2 アクション
A02：ロガーアクション(CSV)

の 2 つのアクションを登録します。



定周期イベントを追加し、A01 と A02 を順に処理するように登録を行います。



次にスクリプトの定義ファイルに記述されているスクリプトについて説明します。ここでは、ヒストリカルデータからのデータ取得方法を説明します。

1）初期処理

初期処理は OnInitialize イベントで行います。

初期処理として New 演算子を使用して、HistoricalData コントロールオブジェクトの生成を行います。

処理内容

① New 演算子を使用し、HistoricalData コントロールのオブジェクトを生成します。
② HDSRequestType プロパティに「RAWDATA」を指定します。「RAWDATA」とすることで、各行のデータの時刻がすべて同じ生データを取得することができるようになります。
③ AddField メソッドで取得したいフィールドを指定します。ここではロガーアクションが持つヒストリカルデータサーバから時系列データを取得するため、ロガーアクションに定義されているフィールドを指定します。
④ SetTimer でタイマーのセットを行います。セットしたタイマーのタイムアップ時に OnTimer イベントが発生します。ここでは、無制限で 5 秒周期のタイマーをセットします。

Script

```
event OnInitialize()
{
        ::SvsDump("OnInitialize:" + ::SvsGetActionName());
①      objHistData = new HistoricalData;
②      objHistData.HDSRequestType = "RAWDATA";
③      objHistData.AddField(c("A02.U01_F01_SD0000","A02.U01_F01_SD0001",
                "A02.U01_F01_SD0002","A02.U01_F01_SD0003","A02.U01_F01_SD0004"));
④      this.SetTimer(0, 5000, -1);
}
```

2）ヒストリカルデータ取得処理

ヒストリカルデータの取得は OnTimer イベント内で行います。

まず、HistoricalData コントロールを使用し、時系列データを取得し、OutPutLog へ出力します。

処理内容

① ReadHistoricalData メソッドで時系列データの取得を行います。メソッドの引数で取得したいデータの範囲を指定します。ここでは「future」～「past」と指定し、全データの検索を行います。ただし、最大レコード数を 10 としているため、取得できるデータは 10 件となります。
また、データ取得に失敗した場合は、GetLastError メソッドを使用して、エラー内容をログ出力します。

② 取得したデータを、取得データ数、取得データの日付、取得データの順にログ出力します。

Script

```
event OnTimer(timerid,counter)
{
        ::SvsDump("OnTimer Id:" + ::CStr(timerid) + " counter:" + ::CStr(counter));
        var vCount, vDateTime, vData;

①      if(!objHistData.ReadHistoricalData("future", T, "past", T, 10, vCount, vDateTime, vData)){
                ::SvsDump("Fail to read historical data. Msg:" + objHistData.GetLastError());
                return;
        }

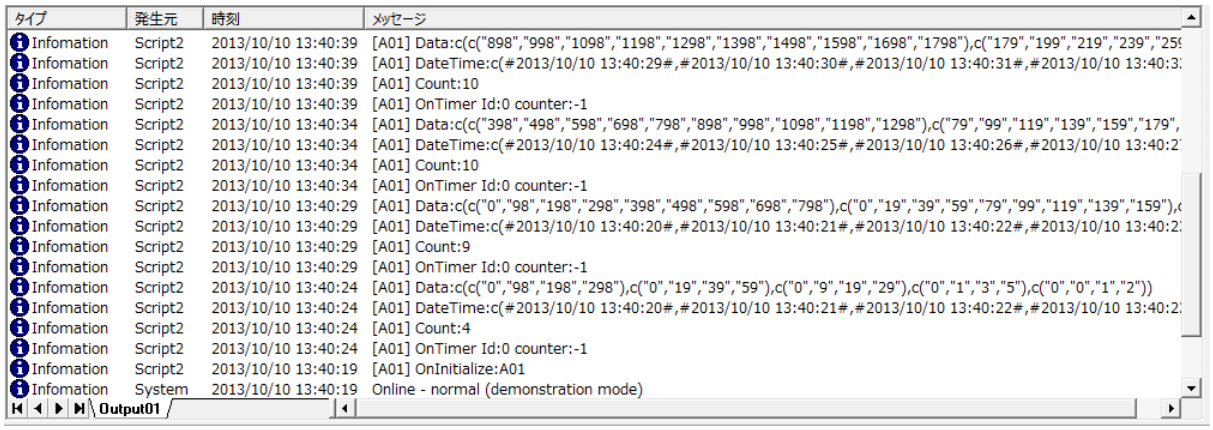
②      ::SvsDump("Count:" + ::CStr(vCount));
       ::SvsDump("DateTime:" + ::CScript(vDateTime));
       ::SvsDump("Data:" + ::CScript(vData));
}
```



「future」、「past」は基準時刻の指定です。「future」は未来を表し、「past」は過去を表します。詳細については、ユーザーズマニュアルの「時刻型の値の書式」項目を参照してください。

動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、OutPutLog にスクリプトの動作結果がログとして出力されます。



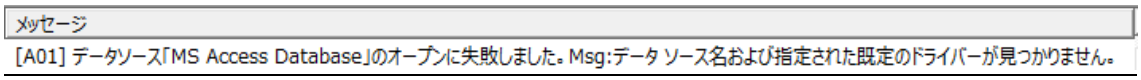
４章　実践編

概要

ここでは各アクションを使用した実践的な手法の説明を行います。



データベースへアクセスを行うサンプルについては、設定に Microsoft 社の Access を使用しています。そのため、Access がインストールされていない環境で動作させると、以下のようなエラーが表示され、正常に動作しません。



サンプルをそのまま動作させる場合は Access をインストールが必要です。もしくは、他の DB を導入し、ODBC の設定を行う必要があります。

ODBC の設定については、「Appendix： アクション共通の設定」の「ODBC 設定」項目を参照してください。

４－１　定周期で計算を行う

概要

ここでは、簡易計算を行うサンプルについて解説します。定周期で特定の計算を行い、計算結果を別のタグへ書き込みたい場合に活用することができます。

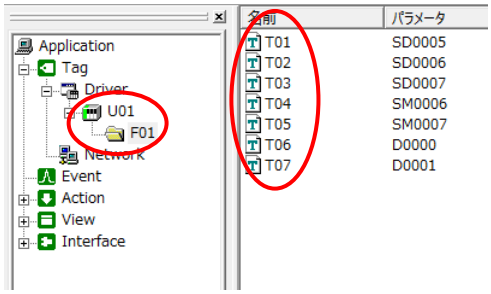
使用アクション：スクリプト Ver1 アクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥script1¥sample01¥script1_sample01.txt

設定手順

タグ設定

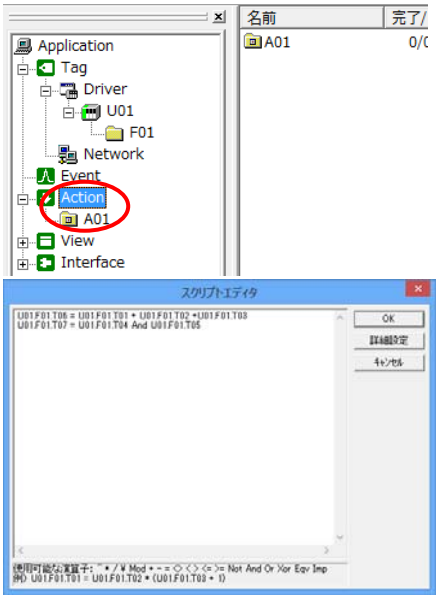
計算を行うためのタグと計算結果を転送するタグを設定します。



タグ名	アドレス	目的
U01.F01.T01～T03	SD0005～SD0007	1 つめの計算式で使用するタグ
U01.F01.T04～T05	SM0006～SM0007	2 つめの計算式で使用するタグ
U01.F01.T06～T07	D0000～D0001	計算結果格納用タグ

アクション設定

スクリプト Ver1 アクション「A01」を設定します。



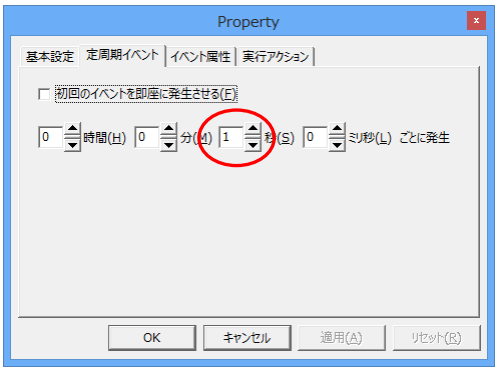
アクションのプロパティを開き、スクリプトタブからスクリプトの設定を行います。

スクリプトは以下の通り設定します。

```
U01.F01.T06 = U01.F01.T01 + U01.F01.T02 +U01.F01.T03
U01.F01.T07 = U01.F01.T04 And U01.F01.T05
```

イベント設定

アクションを実行するための定周期イベントを作成します。今回は計算を 1 秒ごとに行います。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T06～T07 の値が更新されることを確認してください。

ヒント

U01.F01.T06～T07 はどこからも参照されていないため、現在値が表示されません。そのため、該当タグを右クリックし、「タグ操作」-「モニタリング開始」を選択してください。

４－２　ＰＬＣ間のデータ共有（データ転送）

概要

ここでは、複数の PLC 間のデータをゲートウェイするサンプルについて解説します。例えば、PLC が複数台存在する状況下で、リアルタイムあるいはトリガが発生したタイミングで、PLC 間でデータを転送したい場合があります。そのような場面ではブリッジアクションを利用します。

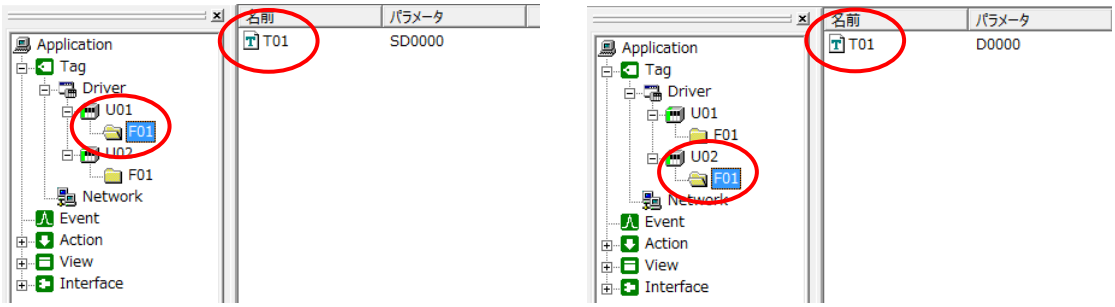
使用アクション：ブリッジアクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥bridge¥sample01¥bridge_sample01.txt

設定手順

タグ設定

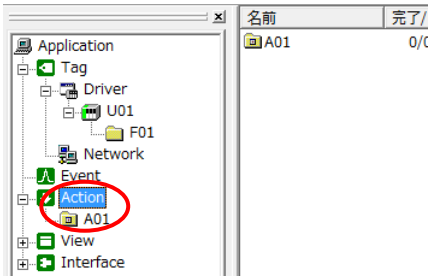
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	SD0000	転送元のタグ
U02.F01.T01	D0000	転送先のタグ

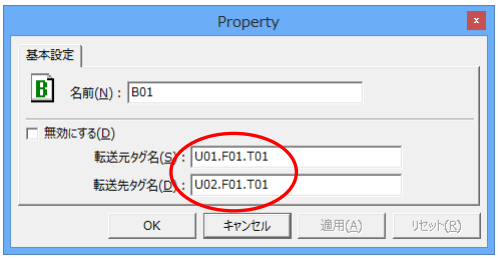
アクション設定

ブリッジアクション「A01」を設定します。



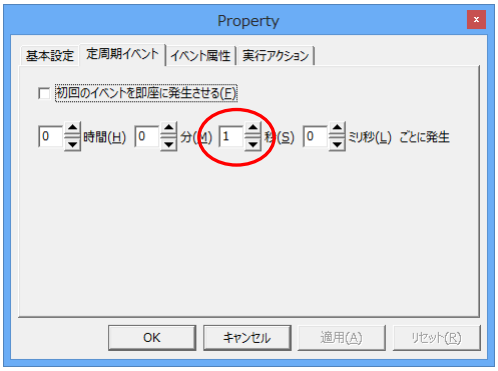
フィールドを追加して、転送設定を行います。

転送元タグ名：U01.F01.T01
転送先タグ名：U02.F01.T01



イベント設定

アクションを実行するための定周期イベントを作成します。今回は 1 秒ごとに値の転送を行います。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 の値が 1 秒周期で U02.F01.T01 へ転送されることを確認してください。

ブリッジアクションには配列タグを指定することもできます。配列タグを使用することでまとまった領域の値をまとめて転送することができます。
サンプルは、「¥¥sample¥common¥action¥bridge¥sample02」に格納されています。

このサンプルでは定周期イベントによるリアルタイムな転送を行いました。例えば、あるビットが ON した時のみ転送を行いたい場合などには、タグイベントをアクション実行のトリガとして使用します。

４－３　WEB 経由での簡易モニタリング

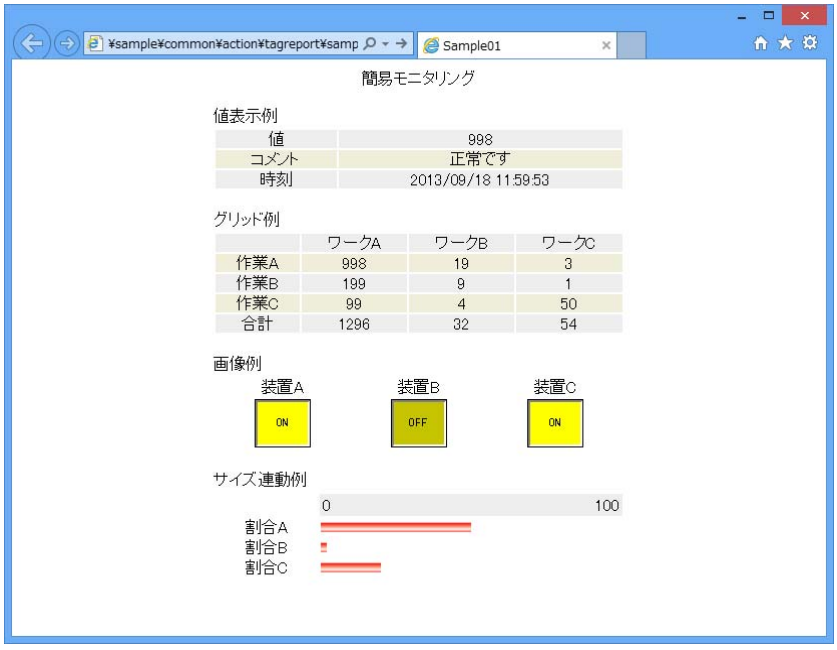
概要

ここでは、Web 経由で PLC の情報を簡易的にモニタリングするサンプルについて解説します。このようなケースでは、タグレポートアクションを利用して HTML ファイルをリアルタイムに出力するのが最も簡単な方法です。また、単なる値の表示のみならず、マーク文字列をうまく配置することによって JavaScript と連動させたり、値に応じて画像を入れ替えたりすることも可能です。

また、この機能を使用することで、出力された HTML ファイルを IIS などの HTTP サーバに設置し、遠隔監視の画面を作成することが可能になります。

使用アクション：タグレポートアクション

サンプルの設定ファイルは以下に格納されています。
設定ファイル：¥¥sample¥common¥action¥tagreport¥sample01¥tagreport_sample01.txt
埋め込み先ファイル：¥¥sample¥common¥action¥tagreport¥sample01¥CFG¥sample01.htm



本サンプルは、不特定多数のクライアントから同時に参照されるようなニーズを想定した処理方式の一例として紹介しています。FA-Panel のグラフィック画面として本サンプル相当の監視画面を構築することも可能ではありますが、Web サービスを介して不特定多数の PC（Web ブラウザ）から同時に監視画面を参照する必要があるような場面において、本サンプルの手法を組み合わせて利用することにより実現することができます。

表示される画面は HTML の記述に依存するため、OS に依存しない監視画面を作成することができます。例えば、各社タブレット機器などからも監視を行えるようになります。

尚、このサンプルではリアル値の表示を以下の方式ごとに実装しています。

■値表示例
リアル値を表示します。コメントや時刻などの情報も扱います。
■グリッド例
グリッドに現在値及び合計値を表示します。合計値はアプリケーション側で計算します。
■画像例
on.gif と off.gif の２つの画像を使用し、ビット情報を絵で表示します。
■サイズ連動例
redbar.gif の width プロパティと連動して、棒グラフのイメージで数値を表示します。

設定手順



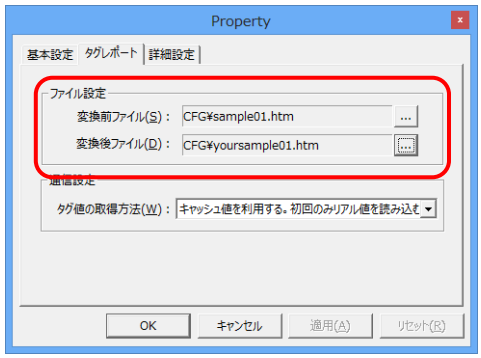
タグ設定

タグを設定します。

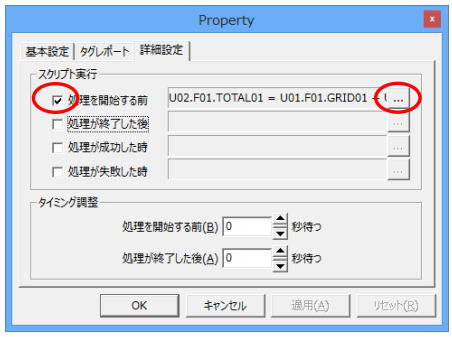
タグ名	マーク文字列	目的
U01.F01.VALUE01	[VALUE01]	値
U01.F01.VALUE02	[VALUE02]	コメント
U01.F01.GRID01	[GRID01]	グリッド例-作業 A ワーク A
U01.F01.GRID02	[GRID02]	グリッド例-作業 B ワーク A
U01.F01.GRID03	[GRID03]	グリッド例-作業 C ワーク A
U01.F01.GRID04	[GRID04]	グリッド例-作業 A ワーク B
U01.F01.GRID05	[GRID05]	グリッド例-作業 B ワーク B
U01.F01.GRID06	[GRID06]	グリッド例-作業 C ワーク B
U01.F01.GRID07	[GRID07]	グリッド例-作業 A ワーク C
U01.F01.GRID08	[GRID08]	グリッド例-作業 B ワーク C
U01.F01.GRID09	[GRID09]	グリッド例-作業 C ワーク C
U01.F01.PIC01	[PIC01]	画面例-装置 A
U01.F01.PIC02	[PIC02]	画面例-装置 B
U01.F01.PIC03	[PIC03]	画面例-装置 C
U01.F01.SIZE01	---	サイズ連動例-割合 A
U01.F01.SIZE02	---	サイズ連動例-割合 B
U01.F01.SIZE03	---	サイズ連動例-割合 C
U02.F01.VALUE03	[VALUE03]	時刻
U02.F01.TOTAL01	[TOTAL01]	グリッド例-ワーク A の合計
U02.F01.TOTAL02	[TOTAL02]	グリッド例-ワーク B の合計
U02.F01.TOTAL03	[TOTAL03]	グリッド例-ワーク C の合計
U02.F01.SIZE01	[SIZE01]	サイズ連動例-割合 A ピクセル換算値
U02.F01.SIZE02	[SIZE02]	サイズ連動例-割合 B ピクセル換算値
U02.F01.SIZE03	[SIZE03]	サイズ連動例-割合 C ピクセル換算値

アクション設定

タグレポートアクション「A01」を設定します。タグレポートアクションの変換前ファイル名及び変換後ファイル名を確認して下さい。

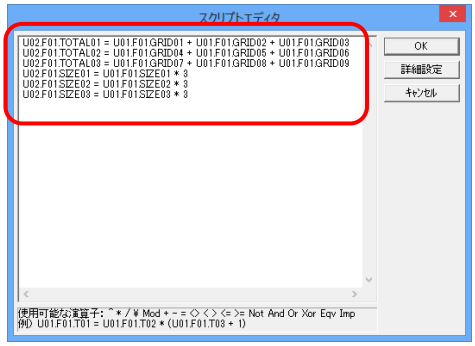


「詳細設定」の「処理を開始する前」のスクリプトでは、グリッドの合計値や棒グラフのピクセル換算値を求めます。

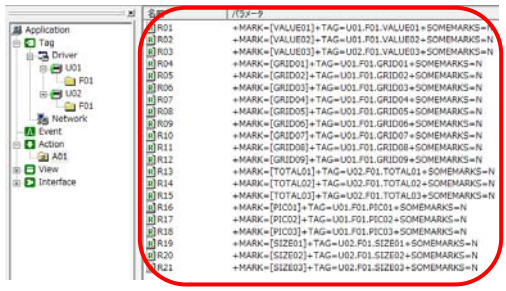


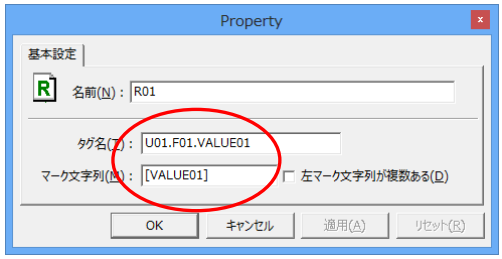
設定する計算式は以下の通りです。

```
U02.F01.TOTAL01 =
U01.F01.GRID01 + U01.F01.GRID02 + U01.F01.GRID03
U02.F01.TOTAL02 =
U01.F01.GRID04 + U01.F01.GRID05 + U01.F01.GRID06
U02.F01.TOTAL03 =
U01.F01.GRID07 + U01.F01.GRID08 + U01.F01.GRID09
U02.F01.SIZE01 = U01.F01.SIZE01 * 3
U02.F01.SIZE02 = U01.F01.SIZE02 * 3
U02.F01.SIZE03 = U01.F01.SIZE03 * 3
```



タグレポートアクション内に、タグ名とマーク文字列との対比を記述します。

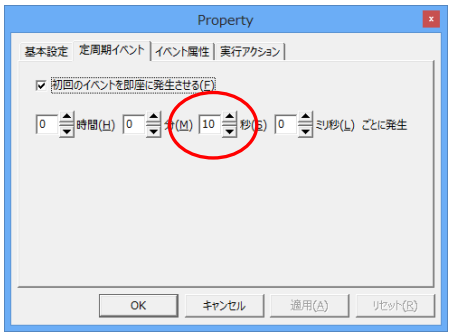




イベント設定

アクションを実行するための定周期イベントを作成します。今回はレポートファイルの作成を１０秒ごとに行います。実行アクションは「A01」を設定します。

※「初回のイベントを即座に発生させる」をチェックすることにより、オンライン直後１０秒待たずに、「A01」が実行されます。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、レポートアクションで指定した HTML ファイルが指定されたフォルダに作成されていることを確認し、ブラウザで表示してみてください。

解説

コメントの表示について

■画面表示

値表示例

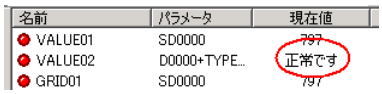
値	8898
コメント	正常です
時刻	2003/03/25 08:24:45

■HTML から抜粋

```
<TR>
<TD width="30%" align="center" bgcolor="#eeeedb">コメント</TD>
<TD align="center" width="70%" bgcolor="#eeeedb">[VALUE02]</TD>
</TR>
```

コメントを表示させるには、コメント用のタグを１つ作る必要があります。このサンプルでは「U01.F01.VALUE02」がコメント用のタグになります。コメント用タグはフィルタ機能（「テーブルファイルを参照して数値や文字列に変換する」）を用います。フィルタ機能の詳細は「タグマニュアル」を参照下さい。

「U01.F01.VALUE02」は下記のように、PLC の値に応じコメントが変化する文字列タグです。このコメント文字列が HTML 内の[VALUE02]へ埋め込まれます。



PLC の値	タグの値
0	正常です
1	ポンプに異常が発生しました
2	圧力に異常が発生しました
3	回転数に異常が発生しました

画像の入れ替えについて

■画面表示

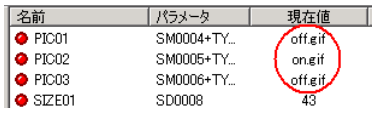
画像例



■HTML から抜粋

```
<TR>
<TD align="center"><IMG src="[PIC01]" width="56" height="48" border="0"></TD>
<TD align="center"><IMG src="[PIC02]" width="56" height="48" border="0"></TD>
<TD align="center"><IMG src="[PIC03]" width="56" height="48" border="0"></TD>
</TR>
```

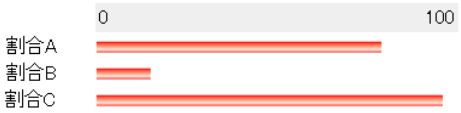
PLC の現在値による画像の入れ替えも、前のコメントの表示方法と同じように、フィルタ機能（「テーブルファイルを参照して数値や文字列に変換する」）を用います。PLC の値が ON の時「on.gif」、OFF の時「off.gif」になり、HTML に埋め込まれます。



PLC の値	タグの値
OFF(FALSE)	off.gif
ON(TRUE)	on.gif

棒グラフの表示について

■画面表示



■HTML から抜粋

```
<TR>
<TD width="100" align="center">割合 A</TD>
<TD width="300"><IMG src="redbar.gif" width="[SIZE01]" height="10" border="0"></TD>
</TR>
<TR>
<TD width="100" align="center">割合 B</TD>
<TD width="300"><IMG src="redbar.gif" width="[SIZE02]" height="10" border="0"></TD>
</TR>
<TR>
<TD width="100" align="center">割合 C</TD>
<TD width="300"><IMG src="redbar.gif" width="[SIZE03]" height="10" border="0"></TD>
</TR>
```

棒グラフの表示では、「詳細設定」の「処理を開始する前」のスクリプトで、HTML に値を埋め込む前に棒グラフの幅のピクセル換算値を計算しています。

「U01.F01.SIZE0X」は 0～100 の間で変化します。また、棒グラフの 100%時の幅は 300 ピクセルなので 3 倍することにより、棒グラフの幅を求めています。そして、計算結果を HTML の[SIZE01]～[SIZE03]へ埋めこんでいます。

```
U02.F01.SIZE01 = U01.F01.SIZE01 * 3
U02.F01.SIZE02 = U01.F01.SIZE02 * 3
U02.F01.SIZE03 = U01.F01.SIZE03 * 3
```



このように、マーク文字列はどのような位置にあってもかまいません。例えば、HTML 内に JavaScript が記述されている場合、関数の引数や変数の初期値の部分にマーク文字列を埋め込むなどの応用が可能です。

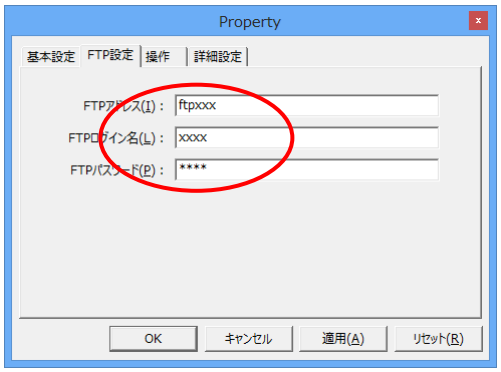


HTML ファイルに Javascript などで定期的に画面をリロードする処理を入れることで、画面を定期的に更新することができます。

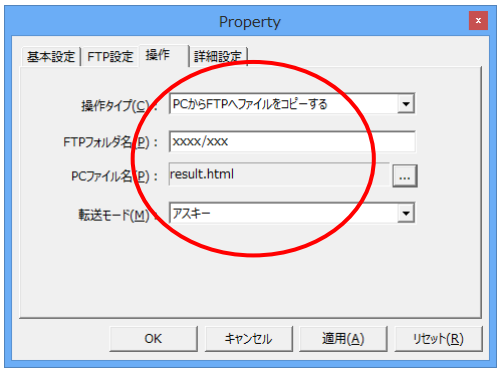
このサンプルの応用例

FTP サーバへ HTML ファイルを転送

作成された HTML ファイルをインターネットで公開するために、FTP 転送アクションで HTTP サーバにファイルを定期的に転送することが可能です。FTP 転送アクション「A02」を作成し、「E01」イベントで「A01」の後に「A02」を実行するようにします。



・FTP アドレス：FTP サーバのアドレス
・FTP ログイン名：ログイン名
・FTP パスワード：パスワード



・FTP フォルダ名：FTP のフォルダへのパス
・PC ファイル名：タグレポートアクションで作成したファイルへのパス

値の自動更新を行う

本サンプルは HTML ファイルを WEB ブラウザで表示を行っているだけなので、PLC の値が変化したとしても画面の値は更新されません。もし、値を更新したい場合は、WEB ブラウザのリロードボタンを押し、HTML ファイルの読み直しを行う必要があります。この値を自動的に更新したい場合があります。そのような場合、一定周期でリロードを行うような JavaScript を HTML ファイルへ組み込むことで実現出来ます。



サンプルは以下に格納されています。
¥¥sample¥common¥action¥tagreport¥sample02

このサンプルには JavaScript が組み込まれています。そのため、WEB ブラウザ側で JavaScript が有効である必要があります。また、WEB ブラウザによっては JavaScript の記述方法が異なる場合があり、サンプルが正常に動作しない場合もあるため、注意が必要です。

４－４　PLC の大量データを一括出力する（テキスト出力）

概要

ここでは、PLC の大量データをファイル出力するサンプルについて解説します。タグに設定されているデータを特定のタイミングでテキストファイルに一括出力します。PLC からの出力トリガを受け取って、タグに設定されたレシピ情報などを出力したい場合などに活用することができます。

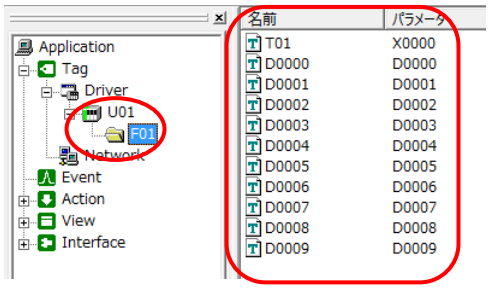
使用アクション：バッチ転送(タグ→テキスト)アクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥batch_tag_text¥sample01¥batch_tag_text_sample01.txt

設定手順

タグ設定

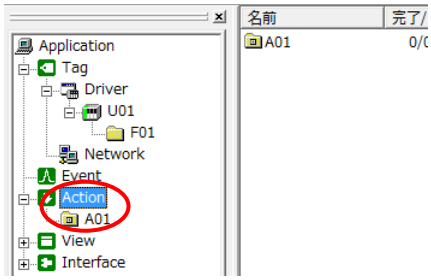
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ
U01.F01.D0000～D0009	D0000～D0009	情報出力を行うタグ

アクション設定

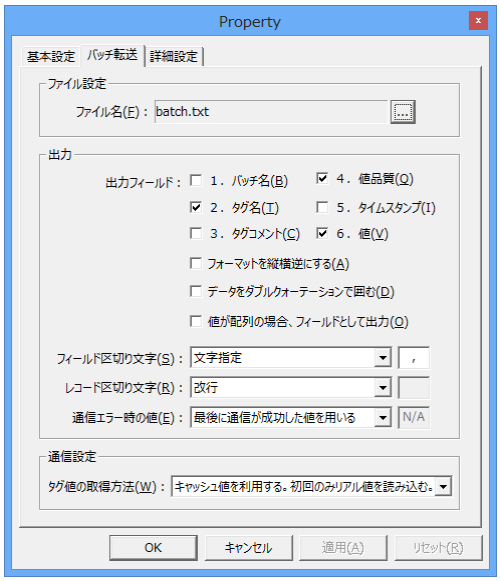
バッチ転送(タグ→テキスト)アクション「A01」を設定します。



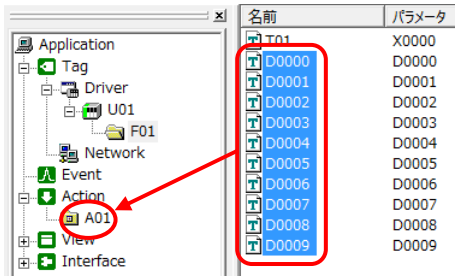
アクションのプロパティを開き、バッチ転送タブからどの情報を出力するかを設定します。

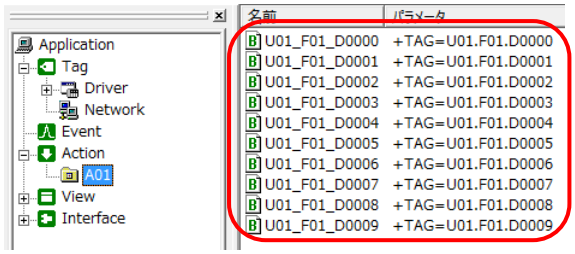
設定内容は以下の通りです。

ファイル名：batch.txt
出力：2.タグ名/4.値品質/6.値
フィールド区切り文字：文字指定「,」
レコード区切り文字：改行
通信エラー時の値：最後に通信が成功した値を用いる



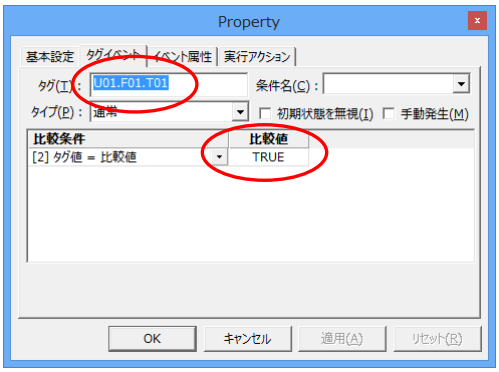
タグを A01 にドラッグアンドドロップします。このアクションはタグをドラッグアンドドロップするだけで簡単にタグを登録することができます。





イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、batch.txt が出力されることを確認してください。



４－５　PLC の大量データを一括出力する（DB 出力）

概要

上記では、PLC の大量データをファイルへ一括出力する方法をご紹介しましたが、ここでは、DB に対して出力を行います。

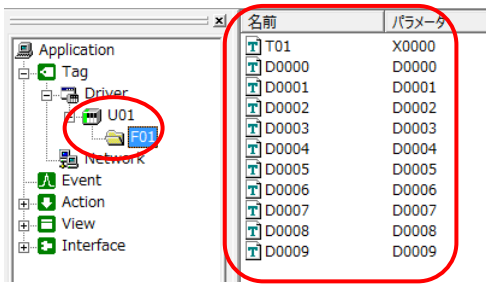
使用アクション：バッチ転送(タグ→ODBC)アクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥batch_tag_odbc¥sample01¥batch_tag_odbc_sample01.txt
サンプルデータベース：¥¥sample¥common¥action¥batch_tag_odbc¥sample01¥batchsample.mdb

設定手順

タグ設定

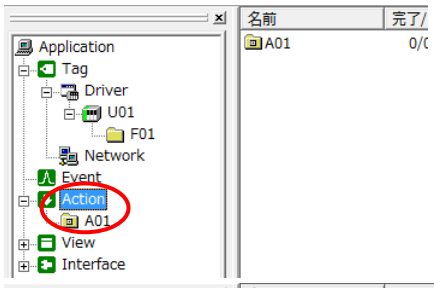
タグを設定します。



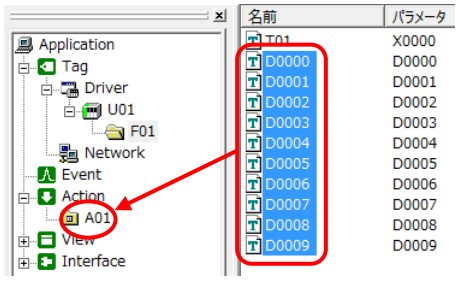
タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ
U01.F01.D0000～D0009	D0000～D0009	情報出力を行うタグ

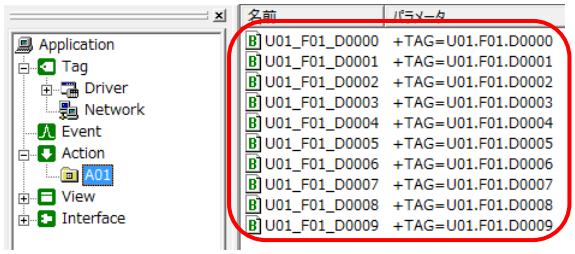
アクション設定

バッチ転送(タグ→テキスト)アクション「A01」を設定します。



タグを A01 にドラッグアンドドロップします。このアクションはタグをドラッグアンドドロップするだけで簡単にタグを登録することができます。

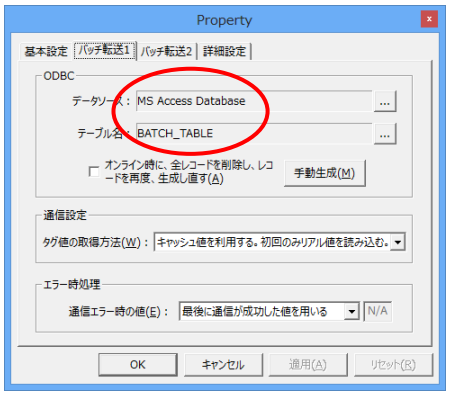




バッチ転送（タグ→ODBC）アクションのプロパティからデータベースを作成します。

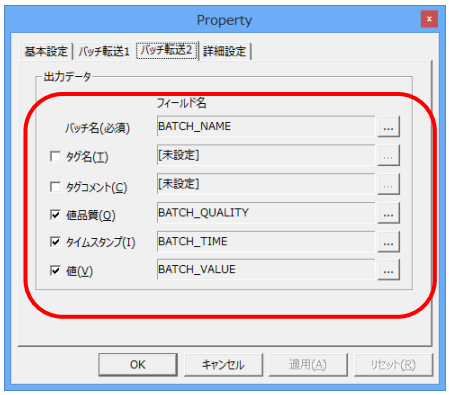
設定は以下の通りです。

バッチ名フィールド：BATCH_NAME
（「バッチ名」フィールドは必須）
値品質フィールド：BATCH_QUALITY
タイムスタンプフィールド：BATCH_TIME
値フィールド：BATCH_VALUE



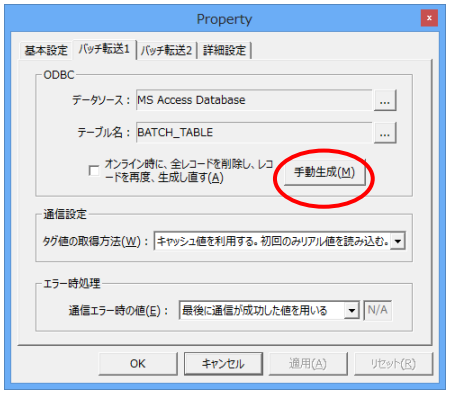
Access でのフィールド設定例：

	フィールド名	データ型
▶	BATCH_NAME	テキスト型
	BATCH_QUALITY	テキスト型
	BATCH_TIME	日付/時刻型
	BATCH_VALUE	数値型

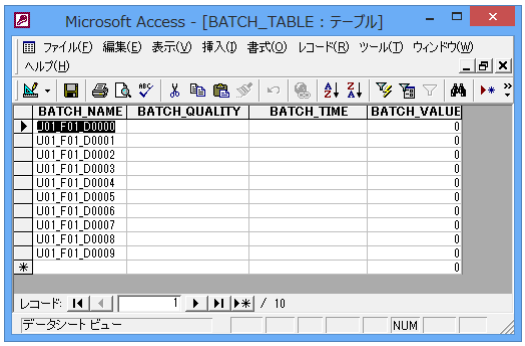


「手動作成」ボタンを押し、キーとなるバッチ名をデータベースへ書き込みます。

当アクションはバッチ名をキー項目としてデータベースに値を書き込むため、準備としてこの作業を行う必要があります。右の画面はバッチ名がデータベースに正しく書き込まれた状態です（Access の例）。

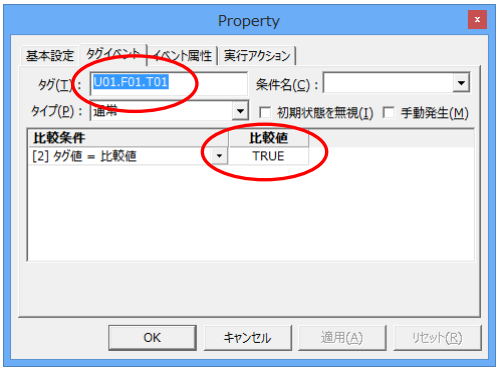


Access の例：



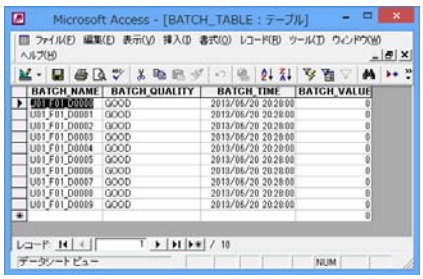
イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、batch.txt が出力されることを確認してください。アクションが実行されると、タグの値がリアルタイムにレコードへ書き出されます。

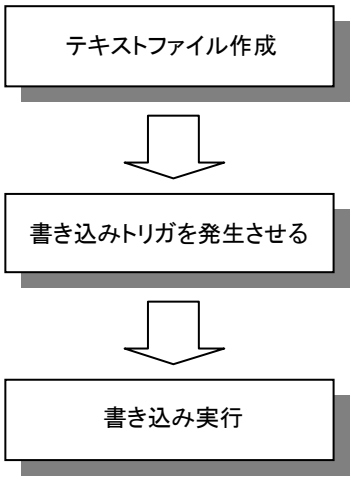


４－６　PLC へ大量データの一括書き込みを行う（テキスト経由）

概要

ここでは、ファイルに保存してあるデータ内容を PLC に対して書き込むサンプルについて解説します。

生産ラインの運転に必要となる運転パラメータやレシピ情報など、大量なデータの書き込みを PLC に対して行う必要がある場合があります。通常の SCADA ではタグを 1 点ずつ扱うのが一般的であるため、大量な点数の書き込みを行うと処理速度が遅くなってしまう場合があります。そのような場面では、バッチ転送アクションを用いてテキストファイルや ODBC 経由で PLC に値を転送することができます。



SCADA などのクライアントの機能によって、PLC に書込みたい値を任意のタイミングでテキストファイルに書き出します。常に固定の値を書き込む場合は、書き込む値が保存されたテキストファイルをあらかじめ用意しておきます。

ビットタグなどを利用し、PLC に書き込みを行いたいタイミングで、クライアント側から書き込み実行のトリガを発生させてください（タグイベント）。

上記のトリガによって、バッチ転送アクションはテキストファイルから値を取得後、PLC に対して一括書き込みを実行します。

テキストファイルのフォーマットとしては、区切り文字を入れて値を列挙します。ここで、区切り文字は自由に選択できます。例えば「,」カンマを区切り文字とした場合、フォーマットは下記の様になります。

1,2,3,4,5,6,7,8,9,10

バッチ転送アクションの定義に従って、ファイルから書き込み値を取得して PLC に転送します。

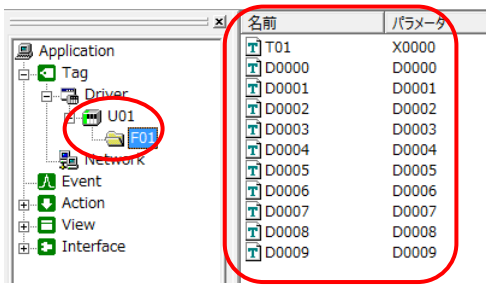
使用アクション：バッチ転送(テキスト→タグ)アクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥batch_text_tag¥sample01¥batch_text_tag_sample01.txt
読み込み用ファイル：¥¥sample¥common¥action¥batch_text_tag¥sample01¥batch.txt

設定手順

タグ設定

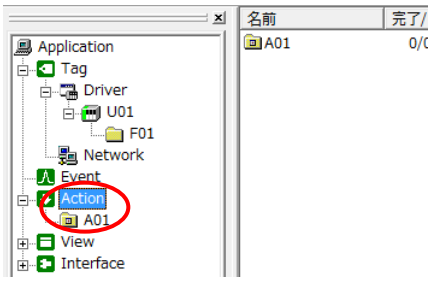
タグを設定します。



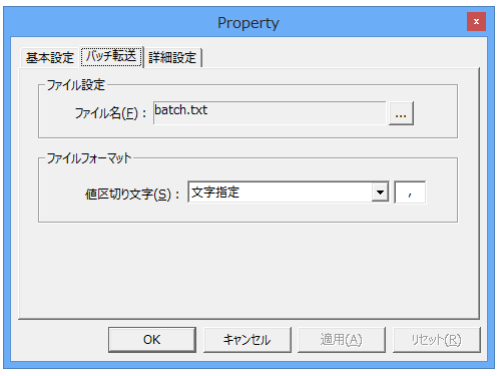
タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ
U01.F01.D0000～D0009	D0000～D0009	一括書き込みを行うタグ

アクション設定

バッチ転送(タグ→テキスト)アクション「A01」を設定します。



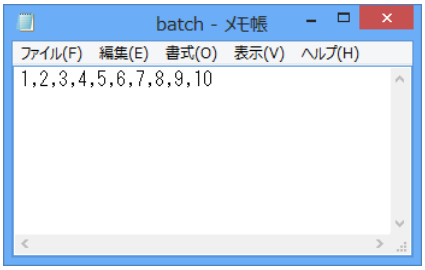
アクションのプロパティを開き、バッチ転送タブからどの情報を出力するかを設定します。



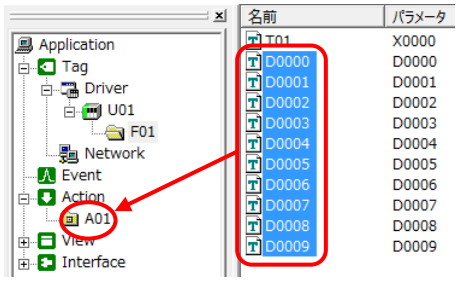
設定内容は以下の通りです。

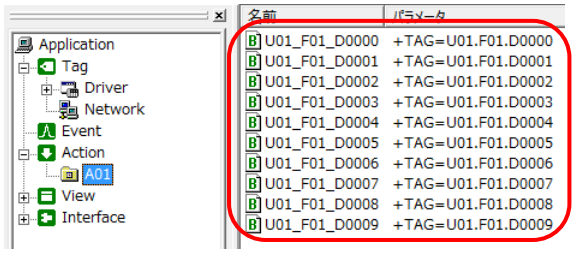
ファイル名：batch.txt
値区切り文字：文字指定「,」

batch.txt は 1～10 の値をカンマ区切りで設定します。



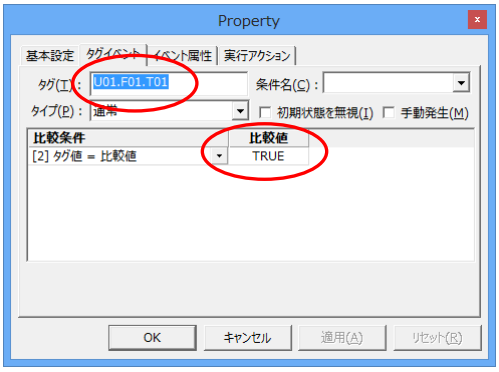
タグを A01 にドラッグアンドドロップします。このアクションはタグをドラッグアンドドロップするだけで簡単にタグを登録することができます。





イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、batch.txt に設定された値がタグに書き込まれることを確認してください。

ヒント

U01.F01.D0000～D0009 はどこからも参照されていないため、現在値が表示されません。そのため、該当タグを右クリックし、「タグ操作」-「モニタリング開始」を選択してください。

４－７　PLC へ大量データの一括書き込みを行う（DB 経由）

概要

上記では、PLC へ大量データをファイル経由でへ一括書き込みする方法をご紹介しましたが、ここでは、DB 経由で書き込みを行います。

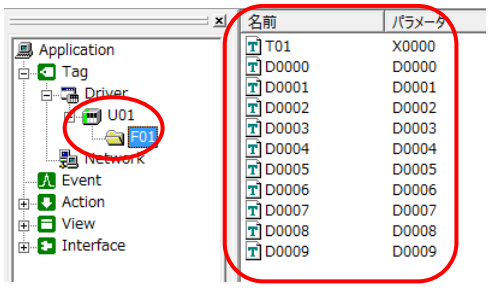
使用アクション：バッチ転送(ODBC→タグ)アクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥batch_odbc_tag ¥sample01¥batch_odbc_tag _sample01.txt
読み込み用ファイル：¥¥sample¥common¥action¥batch_odbc_tag¥sample01¥batchsample.mdb

設定手順

タグ設定

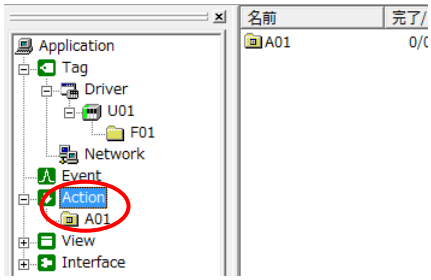
タグを設定します。



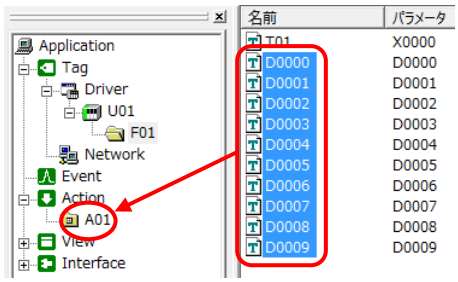
タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ
U01.F01.D0000～D0009	D0000～D0009	一括書き込みを行うタグ

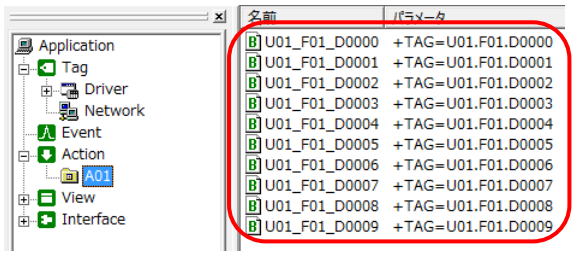
アクション設定

バッチ転送(ODBC→タグ)アクション「A01」を設定します。



タグを A01 にドラッグアンドドロップします。このアクションはタグをドラッグアンドドロップするだけで簡単にタグを登録することができます。

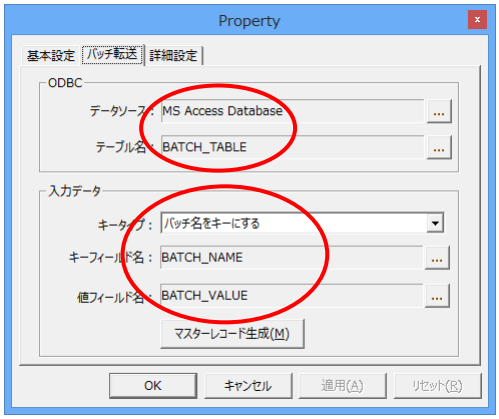




データベースを作成します。

アクションで定義しているフィールド名に合わせてデータベースのレイアウトを定義します。

キーフィールド名：BATCH_NAME
値フィールド名：BATCH_VALUE

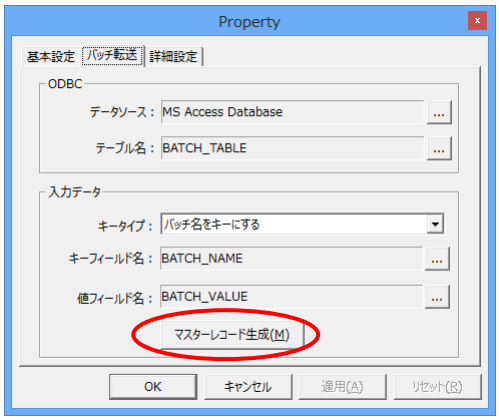


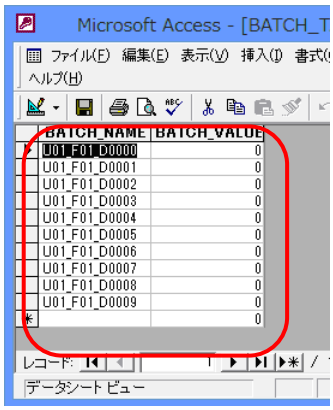
DB の設定：

フィールド名	データ型
BATCH_NAME	テキスト型
BATCH_VALUE	数値型

「マスターレコード生成」ボタンを押し、キーとなるバッチ名をデータベースへ書き込みます。

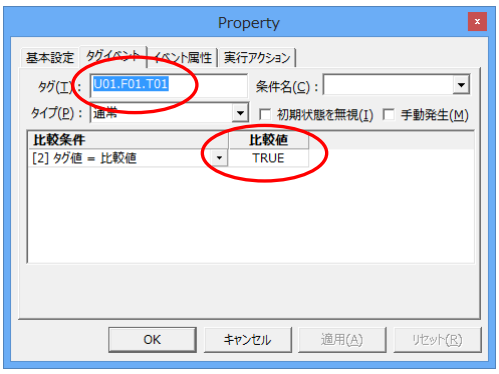
当アクションはバッチ名をキー項目としてデータベースから値を取得するため、準備としてこの作業を行う必要があります。右の画面はバッチ名がデータベースに正しく書き込まれた状態です（Access の例）。





イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、DB に設定された値がタグに書き込まれることを確認してください。

	BATCH_NAME	BATCH_VALUE
▶	U01_F01_D0000	1
	U01_F01_D0001	2
	U01_F01_D0002	3
	U01_F01_D0003	4
	U01_F01_D0004	5
	U01_F01_D0005	6
	U01_F01_D0006	7
	U01_F01_D0007	8
	U01_F01_D0008	9
	U01_F01_D0009	10
✳		0



名前	パラメータ	現在値
D0000	D0000	1
D0001	D0001	2
D0002	D0002	3
D0003	D0003	4
D0004	D0004	5
D0005	D0005	6
D0006	D0006	7
D0007	D0007	8
D0008	D0008	9
D0009	D0009	10



U01.F01.D0000～D0009 はどこからも参照されていないため、現在値が表示されません。そのため、該当タグを右クリックし、「タグ操作」-「モニタリング開始」を選択してください。

４－８　任意のタイミングでメール送信を行う

概要

ここでは PLC からのトリガをうけてメール送信を行うサンプルについて解説します。

使用アクション：メール送信アクション

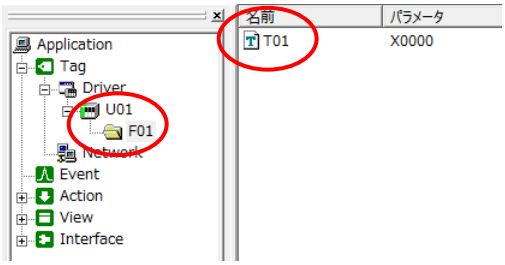
サンプルは下記に格納されています。

設定ファイル：¥¥sample¥common¥action¥mail¥sample01¥mail_sample01.txt
メールマスターファイル：¥¥sample¥common¥action¥mail¥sample01¥mailmaster.txt

設定手順

タグ設定

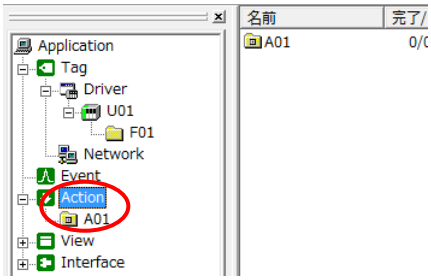
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ

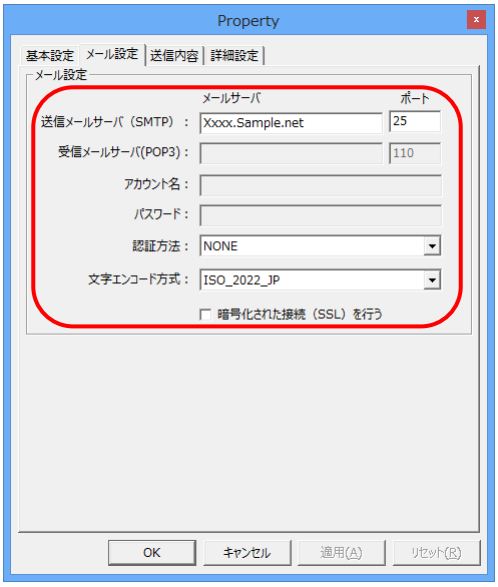
アクション設定

メール送信アクション「A01」を設定します。

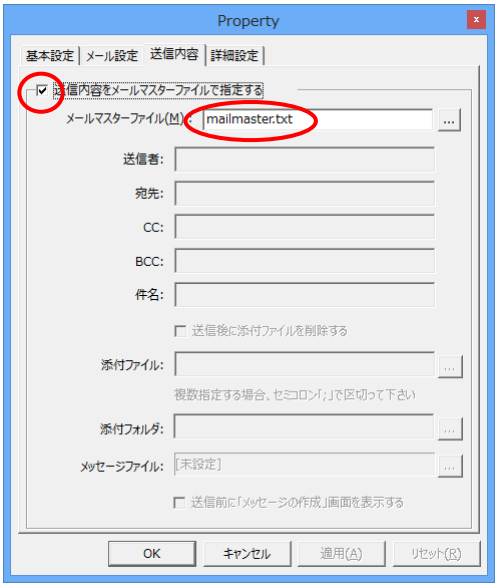


メール設定タブからメール送信時の設定を行います。

・メールサーバー(SMTP)：SMTP サーバのアドレス。
・送信者：送信者のメールアドレス。
・宛先：送信先のメールアドレス。
・件名：メールのタイトル。
・ メッセージファイル：メールのメッセージ部にあたるテキストファイル名のパスです。タグレポートアクションで作成したファイルへのパスを指定します。



送信内容タブから、送信内容を設定します。
ここでは、メールマスターを使用して送信内容の設定を行います。
使用するフォーマットは以下の通りです。




```
[MailSubject]
Title

[MailSender]
xxx@xxx.xx.xx

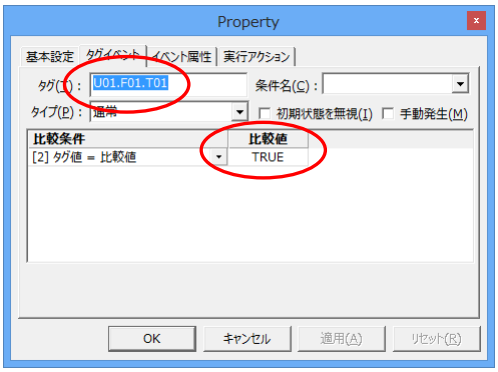
[MailSendToName]
xxx@xxx.xx.xx
(end)

[MailMessage]
メールの内容１行目。
メールの内容２行目。
メールの内容３行目。
(end)
```

フォーマットの詳細は「2 章 アクションリファレンス」の「メール送信アクション」の項目を参照してください。

イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、メールが送信されることを確認してください。

サンプルではタグイベントによるメール送信を行っていますが、定刻イベントや定周期イベントでメール送信アクションを実行することで、定期的な報告メール送信などを行うことができます。

４－９　アラームメール通知

概要

ここでは、設備上で発生したアラーム情報をメールで自動通知するサンプルについて解説します。この例では、PLC デバイスの M0000 が ON になった時に、各タグの値をメールで送信します。また、メールの文章の作成は「WEB 経由での簡易モニタリング」で紹介したタグレポートアクションを利用します。本サンプルでは各種情報が含まれたファイルを添付ファイルとして送付を行います。

使用アクション：メール送信アクション/タグレポートアクション

サンプルは以下に格納されています。
設定ファイル：¥¥sample¥common¥action¥mail¥sample02¥mail_sample02.txt
コンフィグフォルダ：¥¥sample¥common¥action¥mail¥sample02¥CFG

●メール文章作成の概要

メール文章のフォーマットは変換前ファイルで指定しておきます。このサンプルで使用しているメール文章のフォーマットは下記のとおりです。

変換前ファイル

```
[DATE]
エラーコード：[ERRCODE]
メッセージ：[ERRMSG]
装置コード：[DEVICE]
温度：[TEMP]
圧力：[PRES]
```

アラームが発生すると、変換前ファイルのフォーマットに従って下記のとおりにメール文章を作成し、メールの送信を行います。

変換後のメール文章

```
2003/01/01 00:00:00
エラーコード：0
メッセージ：正常です
装置コード：3918
温度：6959
圧力：1391
```

設定手順

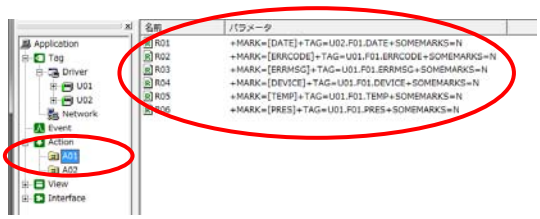
タグ設定

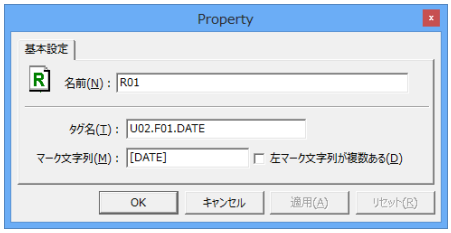
メール文章作成のためのタグを用意します。ここでは、以下のタグを登録してください。

タグ名	マーク文字列	目的
U01.F01.ERRCODE	[ERRCODE]	エラーコード
U01.F01.ERRMSG	[ERRMSG]	メッセージ
U01.F01.DEVICE	[DEVICE]	装置コード
U01.F01.TEMP	[TEMP]	温度
U01.F01.PRES	[PRES]	圧力
U01.F01.ALARM	---	警報タグ（M0000）
U01.F01. DATE	[DATE]	時刻文字タグ「%Y/%m/%d %H:%M:%S」

アクション設定

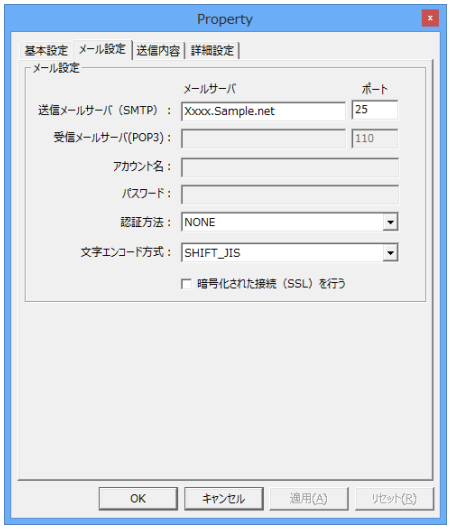
タグレポートアクション「A01」を設定します。タグレポートアクション内に、タグ名とマーク文字列との対比を記述します。





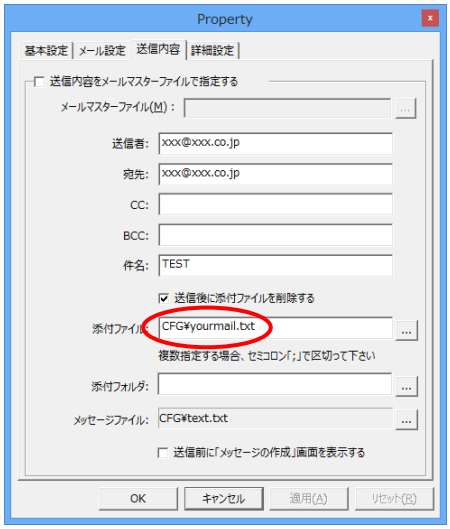
メール送信アクション「A02」を設定します。

・メールサーバー(SMTP)：SMTP サーバのアドレス。
・送信者：送信者のメールアドレス。
・宛先：送信先のメールアドレス。
・件名：メールのタイトル。
・メッセージファイル：メールのメッセージ部にあたるテキストファイル名のパスです。タグレポートアクションで作成したファイルへのパスを指定します。



送信内容タブで送信先の設定などを行います。ここでアラームの状態を知らせるための添付ファイルを設定します。

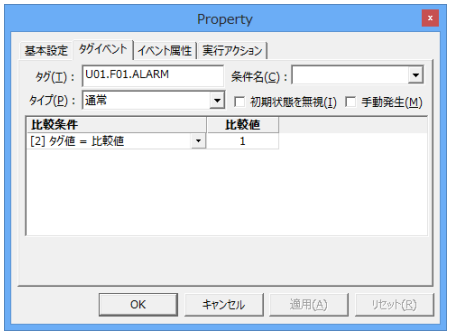
添付ファイルに「CFG¥yourmail.txt」を指定してください。





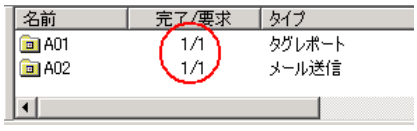
イベント設定

タグイベントを作成します。U01.F01.ALARM が TRUE になった時、「A01」「A02」アクションを実行します。



動作確認

オンライン後、U01.F01.ALARM を TRUE にして下さい。下記のようにアクションの実行回数がインクリメントされているのを確認後、メール送信先のメーラでメールを受信し、正しくメールが送信されたことを確認してみてください。

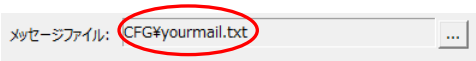




メール送信アクションは添付ファイルを送信することも可能です。例えば、異常発生時にトレンドグラフの JPEG 画像をスクリーンショットアクションで取得し、その画像を添付ファイルとして送信することができます。



本サンプルはメール送信の際に添付ファイルとして、データを送付していますが、メールの本文に情報を組み込むことができます。メール本文に組み込むためには、送信内容タブの設定で、メッセージファイルを指定します。この設定により、ファイルの内容が本文に組み込まれるようになります。



サンプルは、「¥¥sample¥common¥action¥mail¥sample03」に格納されています。

４－10　ファイルを別のフォルダにコピーする

概要

ここでは、指定ファイルを他のフォルダにコピーするサンプルについて解説します。共有フォルダにファイルをコピーして他のユーザーが参照できるようにするなど様々な活用方法があります。

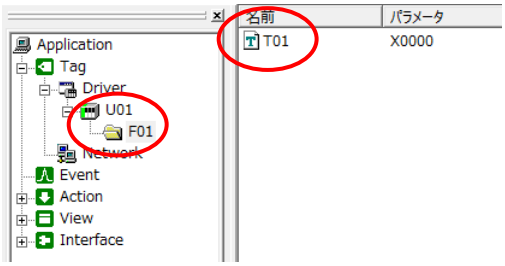
使用アクション：ファイル操作アクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥fileaction¥sample01¥ fileaction_sample01.txt

設定手順

タグ設定

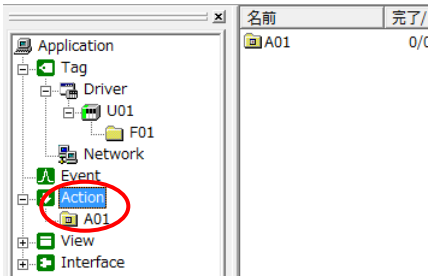
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ

アクション設定

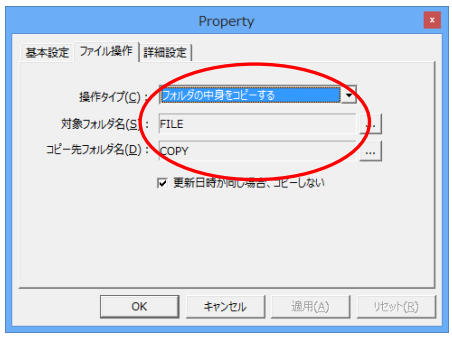
ファイル操作アクション「A01」を設定します。



アクションのプロパティを開き、ファイル操作タブから設定を行います。

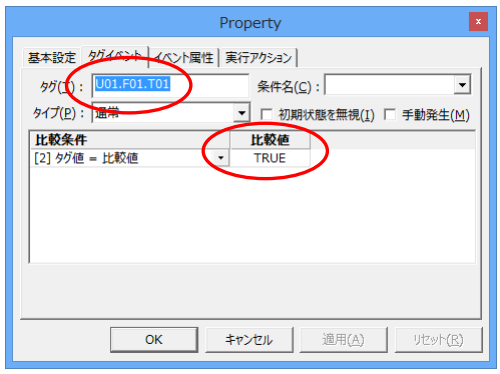
本サンプルでは、FILE フォルダの中身を COPY フォルダへ一括コピーさせます。

・操作タイプ：フォルダの中身をコピーする
・対象フォルダ名：FILE
・コピー先フォルダ名：COPY



イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、コピー先フォルダにファイルがコピーされているか確認してください。

４－11　ファイルバックアップ

概要

ここでは、指定フォルダを他のフォルダにバックアップするサンプルについて解説します。ファイル操作アクションを使用してもバックアップは行えますが、バックアップアクションは、バックアップ元のファイルが削除された場合、バックアップ先のファイルも削除することができるため、バックアップ元とバックアップ先フォルダの中身を常に同一とすることができます。

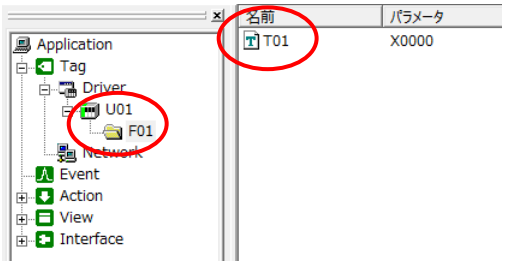
使用アクション：ファイルバックアップアクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥filebackup¥sample01¥filebackup_sample01

設定手順

タグ設定

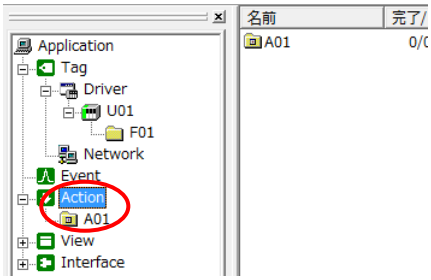
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ

アクション設定

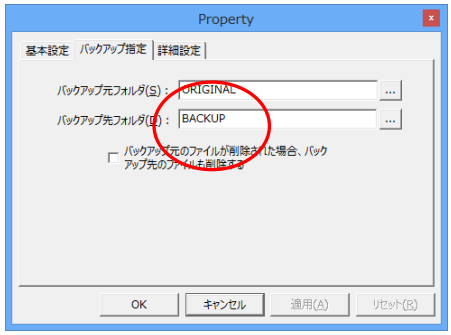
ファイルバックアップアクション「A01」を設定します。



アクションのプロパティを開き、バックアップ指定タブから設定を行います。

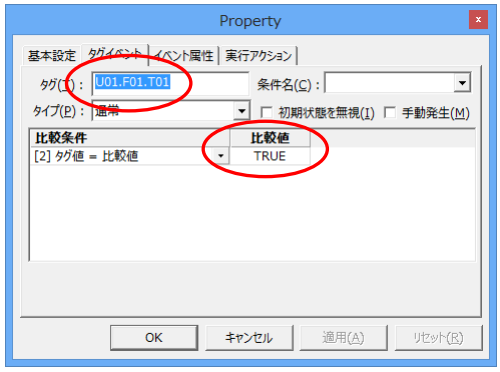
本サンプルでは、ORIGINAL フォルダの中身を BACKUP フォルダへ一括コピーさせます。

・対象フォルダ名：ORIGINAL
・コピー先フォルダ名：BACKUP



イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、バックアップ先フォルダにファイルがバックアップされているか確認してください。

４－12　ファイルを FTP 転送する

概要

ここでは、ファイルを FTP 転送するサンプルについて解説します。この機能を使用して、任意のファイルを FTP 経由で別のパソコンへ転送することができます。

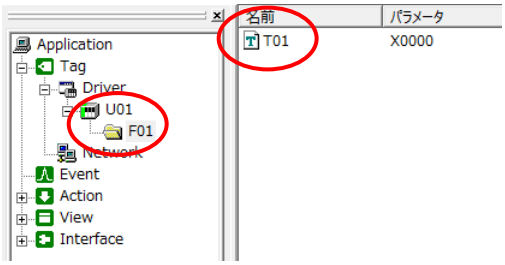
使用アクション：FTP 転送アクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥ftp¥sample01¥ftp_sample01.txt

設定手順

タグ設定

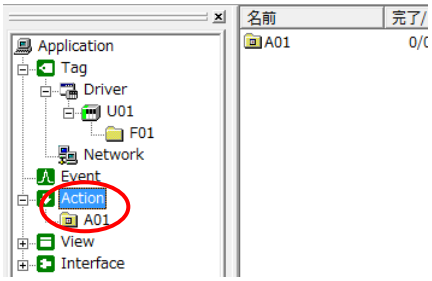
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ

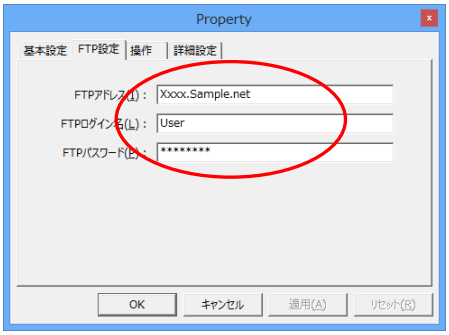
アクション設定

FTP 転送アクション「A01」を設定します。



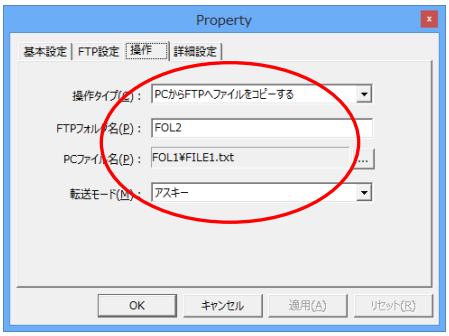
アクションのプロパティを開き、ファイル操作タブから設定を行います。

サンプルではサンプル値が設定されているので、環境に合わせて設定を行ってください。



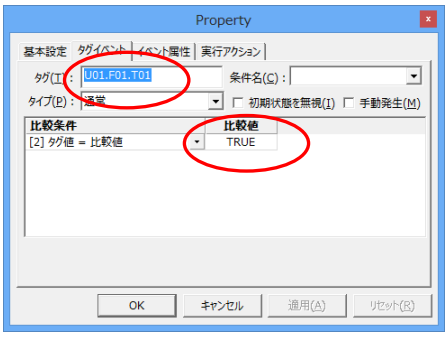
操作タブから転送設定を行います。

操作タイプ：PC から FTP ファイルをコピーする
FTP フォルダ名：FOL2
PC ファイル名：FOL1¥FILE1.txt
転送モード：アスキー



イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、FTP 転送先フォルダにファイルがコピーされているか確認してください。

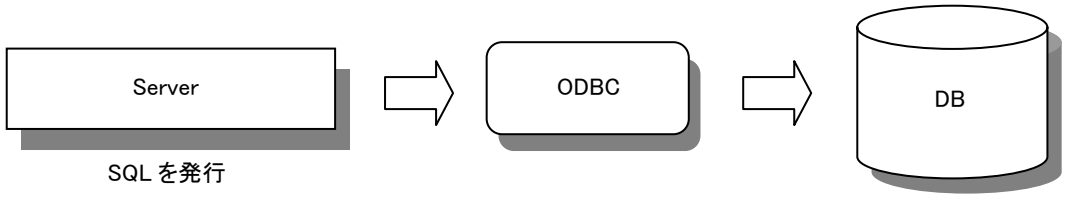
ヒント

本サンプルは PC のファイルを FTP 経由で転送するサンプルです。逆に FTP サーバに存在するファイルを PC へ転送することも可能です。
サンプルは、「¥¥sample¥common¥action¥ftp¥sample02¥ftp_sample02.txt」に格納されています。

４－13　SQL を実行して DB のレコードを追加/削除する

概要

ここでは、SQL を実行して DB の値を更新するサンプルについて解説します。DB には ODBC 経由でアクセスを行います。本サンプルでは insert コマンドによるデータ追加と delete コマンドによるテーブルデータの削除を行います。



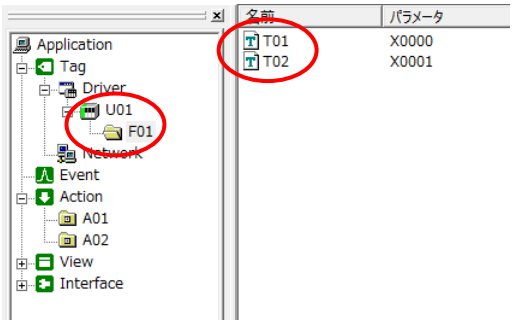
使用アクション：SQL 実行(ODBC)アクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥sql¥sample01¥sql_sample01.txt
サンプルデータベース：¥¥sample¥common¥action¥sql¥sample01¥dbsample.mdb

設定手順

タグ設定

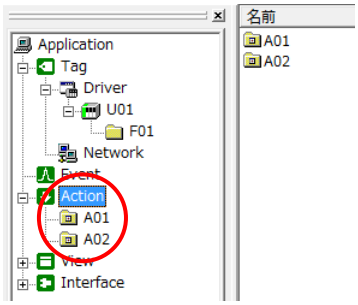
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	X0000	insert コマンドを実行するためのトリガとなるタグ
U01.F01.T02	X0001	delete コマンドを実行するためのトリガとなるタグ

アクション設定

SQL 実行(ODBC)アクション「A01」「A02」を設定します。



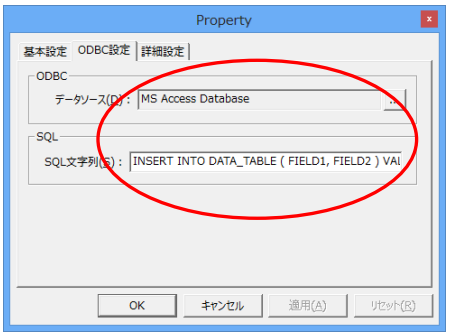
A01

アクションのプロパティを開き、ODBC 設定タブから ODBC と SQL の設定を行います。

SQL は以下の内容を設定します。

A01：

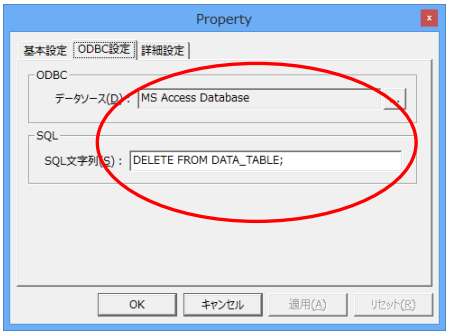
```
INSERT INTO DATA_TABLE ( FIELD1, FIELD2 ) VALUES (1, 'abc');
```



A02：

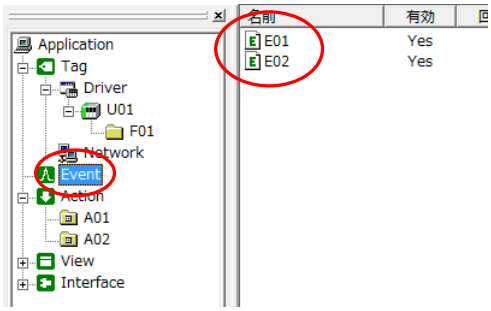
```
DELETE FROM DATA_TABLE;
```

A02



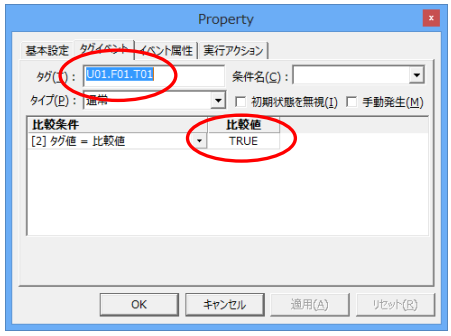
イベント設定

アクションを実行するためのタグイベントを作成します。今回は 2 つのアクションを使用するため、タグイベント「E01」「E02」の 2 つ用意します。



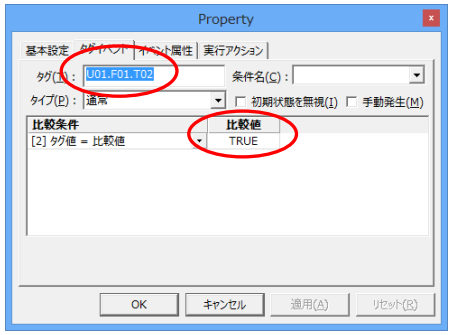
E01

E01 は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



E02

E02 は U01.F01.T02 が TRUE になったら、アクションを実行します。実行アクションは「A02」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、サンプルデータベースに新規レコードが追加されていることを確認してください。その後、U01.F01.T02 に TRUE を書き込み、サンプルデータベースのレコードが削除されていることを確認してください。

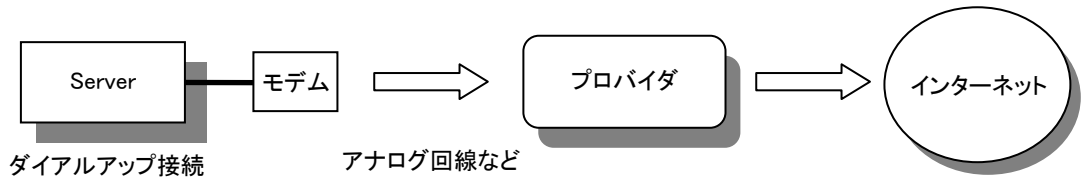
SQL 実行(ODBC)アクションで使用できる SQL は、アクションに設定されたもののみであるため、オンライン後に流動的に SQL の構文を変更することはできません。

状況によって、SQL の構文を変更したい場合は、スクリプト Ver2 を使用する方法があります。
詳細は「2 章 アクションリファレンス」の「スクリプト Ver2」項目を参照してください。

４－14　ダイアルアップ接続／切断を行う

概要

ここでは、ダイアルアップ接続を行い、インターネット接続をおこなうサンプルについて解説します。サンプルではトリガによって、接続/切断を行うことができます。



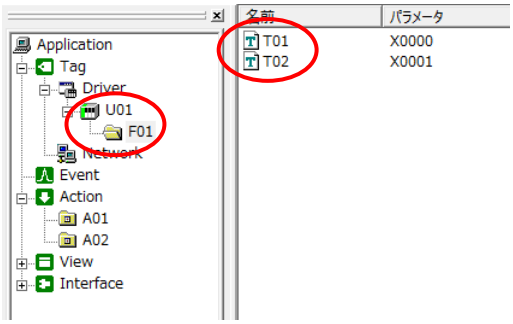
使用アクション：ダイアルアップアクション

サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥dialup¥sample01¥dialup_sample01.txt

設定手順

タグ設定

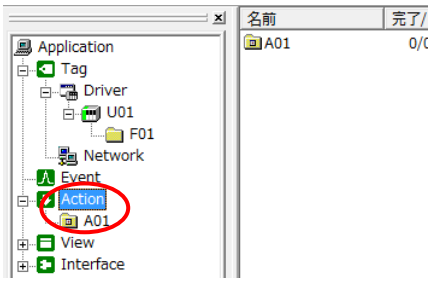
タグを設定します。



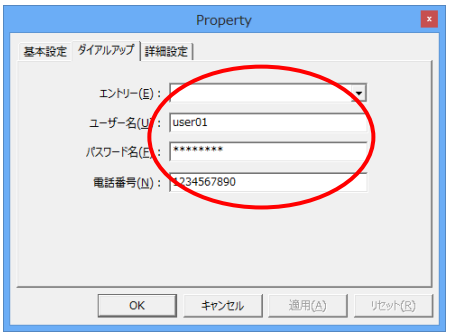
タグ名	アドレス	目的
U01.F01.T01	X0000	ダイアルアップ接続を実行するためのトリガとなるタグ
U01.F01.T02	X0001	ダイアルアップ切断を実行するためのトリガとなるタグ

アクション設定

ダイアルアップ接続アクション「A01」を設定します。

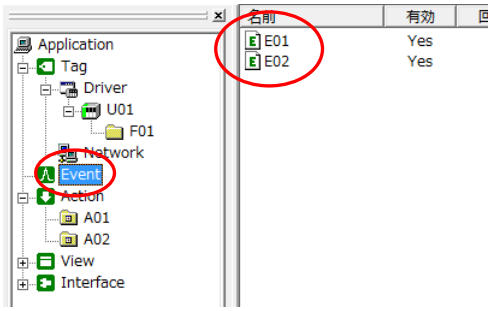


アクションのプロパティを開き、ダイアルアップタブからダイアルアップ接続時の設定を行います。



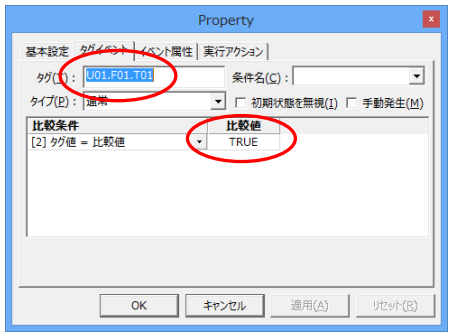
イベント設定

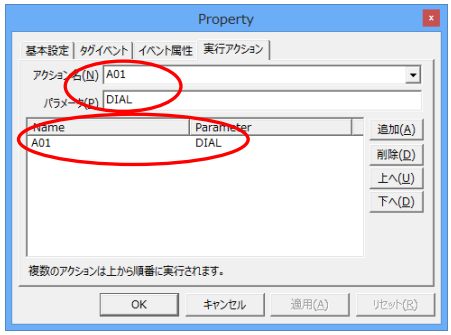
アクションを実行するためのタグイベントを作成します。今回は 2 つのアクションを使用するため、タグイベント「E01」「E02」の 2 つ用意します。



E01 は U01.F01.T01 が TRUE になったら、アクションを実行します。

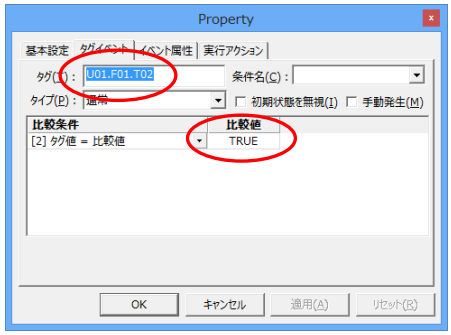
実行アクションは「A01」を設定し、パラメータに「DIAL」を指定します。

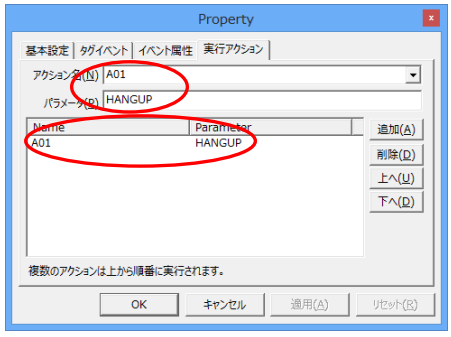




E02 は U01.F01.T02 が TRUE になったら、アクションを実行します。

実行アクションは「A01」を設定し、パラメータに「HANGUP」を指定します。





動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、ダイアルアップ接続されることを確認してください。その後、U01.F01.T02 に TRUE を書き込み、接続が切断されることを確認してください。

このサンプルの応用例

メール送信のためにダイアルアップ接続を利用することができます。ダイアルアップを行った後メール送信を行い、接続を切断します。メール送信アクション「A02」を作成し、「E01」イベントで「A01」の後に「A02」を実行するようにします。さらに、「A01」を「HANGUP」を指定します。



４－15　外部アプリケーションの実行

概要

ここでは外部アプリケーションを実行するサンプルについて解説します。自作した EXE などを PLC からのトリガを受けて実行したい場合など利用できます。サンプルでは WindowsOS に標準で付属するメモ帳から、自作したテキストファイルを開きます。

使用アクション：外部アプリケーション実行アクション

サンプルは下記に格納されています。

設定ファイル：¥¥sample¥common¥action¥application¥sample01¥application_sample01.txt
テキストファイル：¥¥sample¥common¥action¥application¥sample01¥file.txt

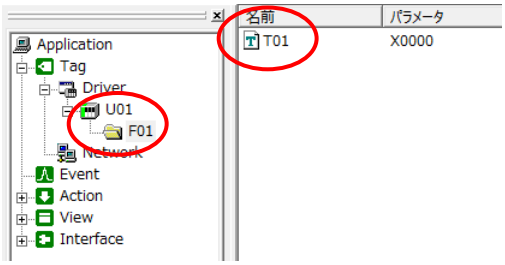


バッチファイル(.bat)を実行したい場合は、特殊な設定が必要であるため、本項の「このサンプルの応用例」を参照してください。

設定手順

タグ設定

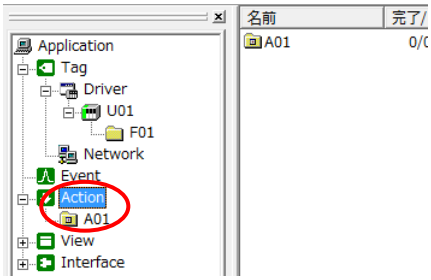
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ

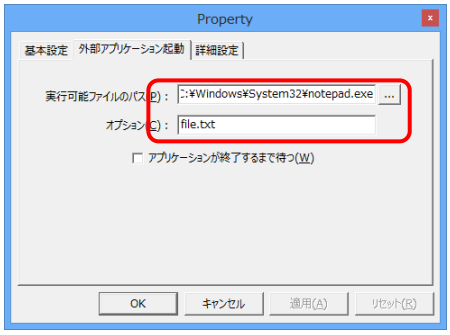
アクション設定

外部アプリケーション実行アクション「A01」を設定します。



外部アプリケーション起動タブから、起動するアプリケーションとオプションを指定します。

・実行可能ファイルパス：C:¥Windows¥System32¥notepad.exe
・オプション：file.txt



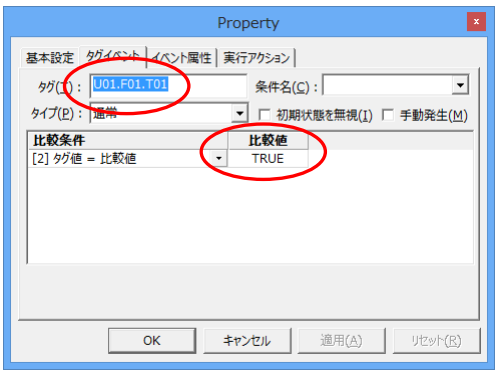
「C:¥Windows¥System32¥notepad.exe」はメモ帳のフルパスです。EXE の格納場所が異なる場合は、変更してください。

ヒント

外部アプリケーションの実行可能ファイルパスについては、フルパスで指定を行う必要があります。

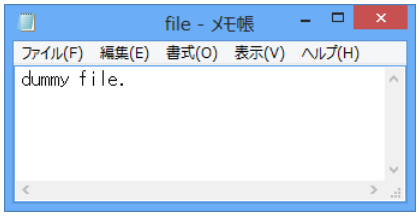
イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、「file.txt」が起動することを確認してください。



このサンプルの応用例

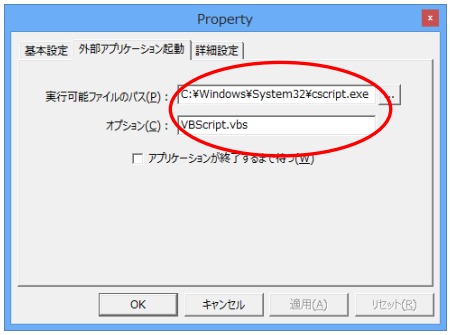
一括した処理を実行したい場合、バッチファイル(.bat)を作成して実行すれば便利ですが、外部アプリケーション実行アクションからはバッチファイルを実行することができません。
そのため、バッチファイルを実行したい場合、VBScript などを通して実行する方法があります。

VBScript を実行する場合、cscript.exe を利用して VBS ファイルを実行します。
cscript の本体は「C:¥Windows¥System32¥cscriptcscript.exe」になります。

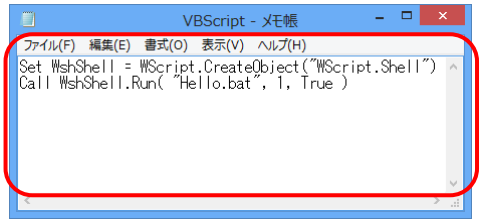
この応用サンプルは下記に格納されています。
設定ファイル：¥¥sample¥common¥action¥application¥sample02¥application_sample02.txt

実行可能ファイルのパスに、cscript のパスを設定し、オプションに実行したい VBS ファイルを設定します。

実行可能ファイルパス：C:¥Windows¥System32¥cscriptcscript.exe
オプション：VBScript.vbs



VBScript からバッチファイルを起動するスクリプトを記述します。
VBScript.vbs には下記のスクリプトが記述されており、「Hello.bat」を実行するようになっています。

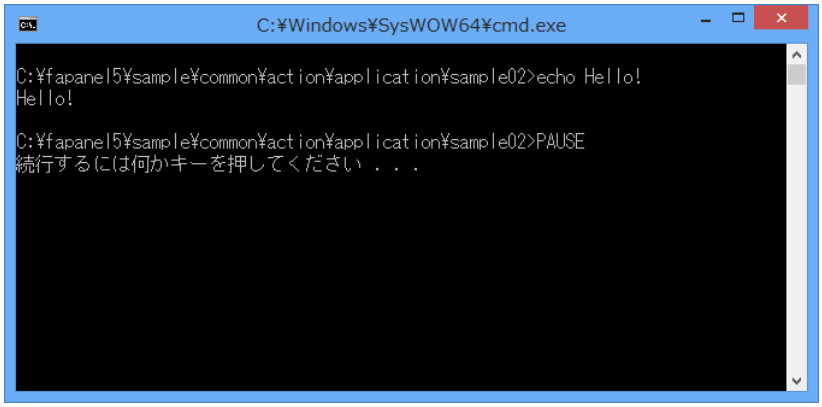


Script

```
Set WshShell = WScript.CreateObject("WScript.Shell")
Call WshShell.Run( "Hello.bat", 1, True )
```

サンプルでは、U01.F01.T01 をトリガにして、外部アプリケーション実行アクションを実行しています。

アプリケーションをオンライン(黄色矢印）にして、U01.F01.T01 に TRUE を書き込み、バッチファイルが起動することを確認してください。



４－16　画面全体のスクリーンショットを取得

概要

ここではデスクトップ全体のスクリーンショットを取得する実行するサンプルについて解説します。

使用アクション：スクリーンショットアクション

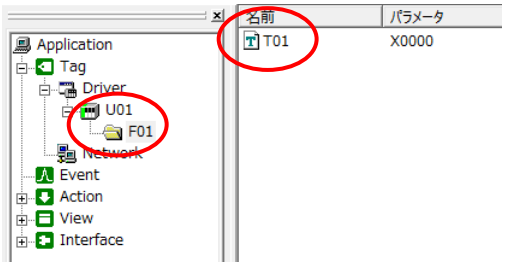
サンプルは下記に格納されています。

設定ファイル：¥¥sample¥common¥action¥screenshot¥sample01¥screenshot_sample01.txt

設定手順

タグ設定

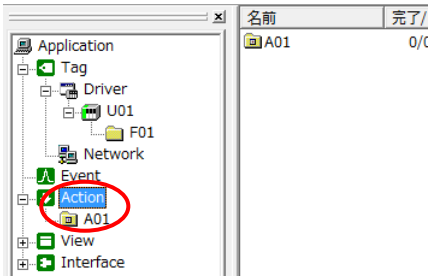
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ

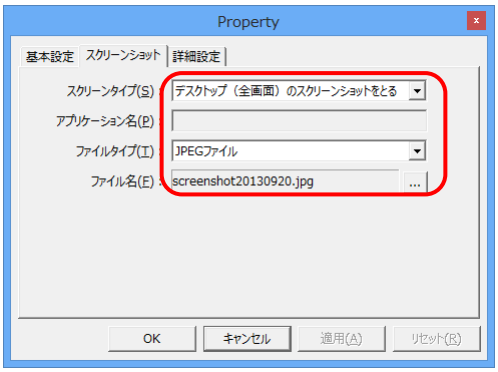
アクション設定

スクリーンショットアクション「A01」を設定します。



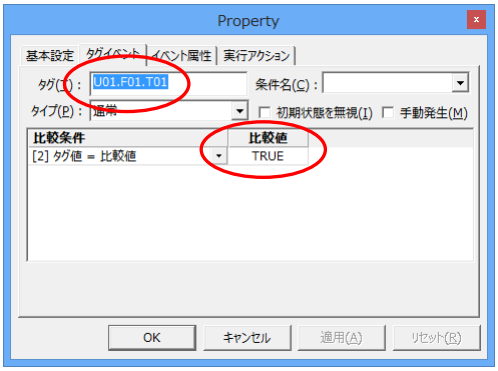
スクリーンショットタブから、スクリーンショットの設定を行います。

・スクリーンタイプ：
デスクトップ（全画面）のスクリーンショットをとる。
・ファイルタイプ：JPEG ファイル
・ファイル名：screenshot.jpg（日付を付加する）



イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、スクリーンショットの画像ファイルが生成されているか確認してください。

４－17　Server アプリケーションを終了させる

概要

ここでは Server アプリケーションを終了させるサンプルについて解説します。タグイベントを用いることで、PLC からのトリガなどから Server アプリケーションを終了させることができます。

使用アクション：シャットダウンアクション

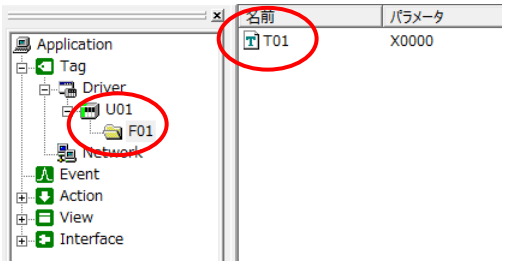
サンプルは下記に格納されています。

設定ファイル：¥¥sample¥common¥action¥shutdown¥sample01¥shutdown_sample01.txt

設定手順

タグ設定

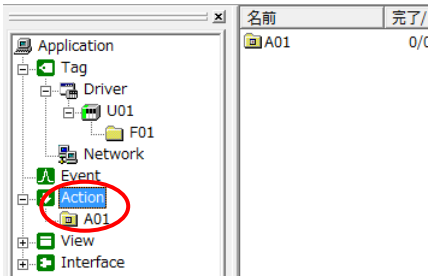
タグを設定します。



タグ名	アドレス	目的
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ

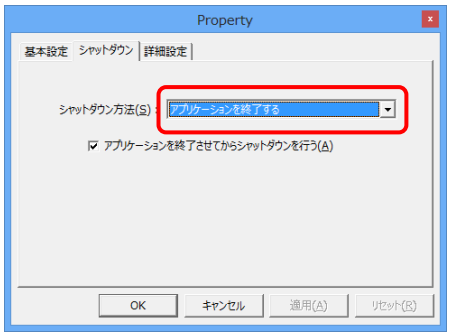
アクション設定

シャットダウンアクション「A01」を設定します。



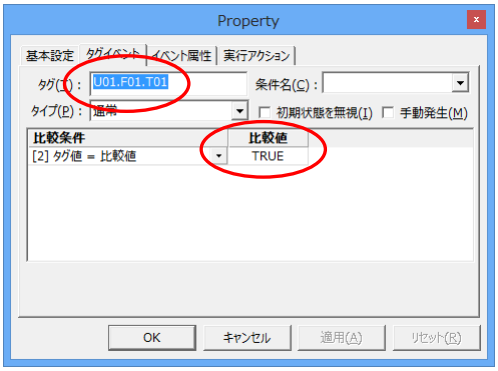
シャットダウンタブから、スシャットダウンの設定を行います。

・シャットダウン方法：アプリケーションを終了する。



イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、Server アプリケーションが正常に終了することを確認してください。

このサンプルの応用

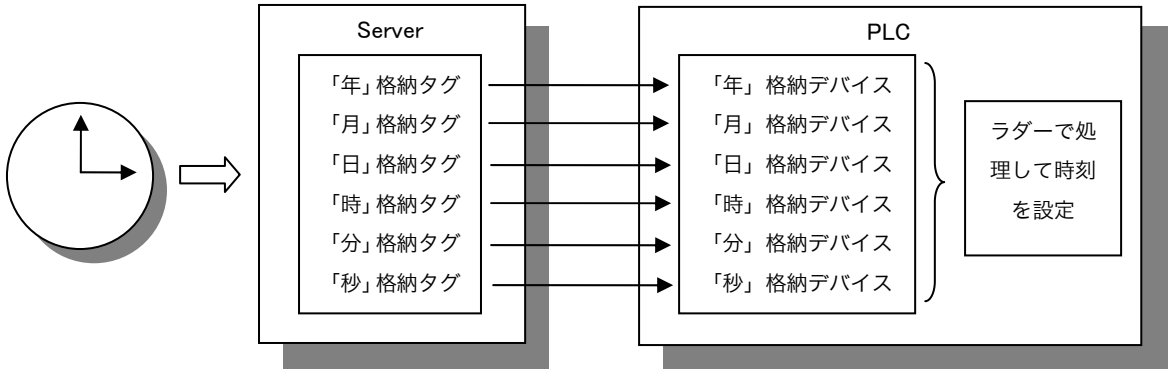
停電時 UPS が動作した場合、シャットダウンアクションを使用して Server アプリケーションを正常に終了することができます。例えば、IPlink を使用して独自アプリケーションを作成し、UPS が動作した場合、UPS 側で独自アプリケーションを起動させます。アプリケーションからアクションを実行するトリガタグを ON することで、アクションを実行することができます。ただし、この場合 UPS 側で動作時に特定のアプリケーションを実行できる機種である必要があります。

４－18　PC と PLC の時刻を同期させる

概要

ここでは PC と PLC の時刻を同期させるサンプルについて解説します。本サンプルでは PC の時刻を取得して、PLC に書き込みます。

時刻同期アクションは PLC（タグ）へ PC の時刻情報を書き込む機能です。年～秒までの情報を別々のタグに書き込み、PLC 側のラダーで、書き込まれた日付情報を PLC のフォーマットに合った時刻に処理して、時刻の同期を行います。



使用アクション：時刻同期アクション

サンプルは下記に格納されています。

設定ファイル：¥¥sample¥common¥action¥time¥sample01¥time_sample01.txt



時刻同期アクションは、直接 PLC へ時刻を設定できないため注意が必要です。PLC はメーカや機種によって日付の設定方法やフォーマットが異なります。そのため、時刻同期アクションはタグ経由で PLC へ日付情報を書き込むところまでを行い、フォーマット調整および日付の設定はラダーなどで行うようにすることで、あらゆる状態に対応できるようになっています。

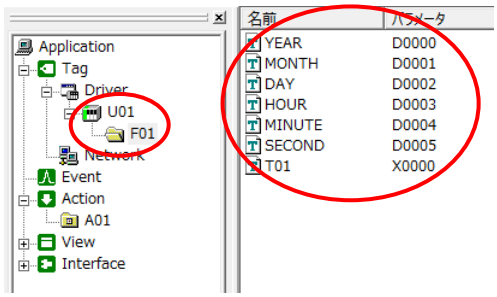


直接 PLC のデバイスへ時刻情報を書き込むのではなく、一旦、仮想デバイスタグなどに目的に応じた書式で日付情報を書き込んでおき、そのタグの値をスクリプト Ver2 アクションで更に加工してから PLC に書き込むという方法もあります。

設定手順

タグ設定

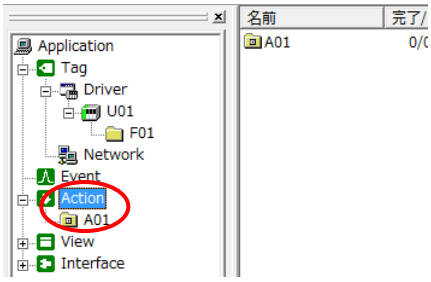
タグを設定します。



タグ名	アドレス	目的
YEAR	D0000	年を書き込むタグ
MONTH	D0001	月を書き込むタグ
DAY	D0002	日を書き込むタグ
HOUR	D0003	時を書き込むタグ
MINUTE	D0004	分を書き込むタグ
SECOND	D0005	秒をを書き込むタグ
U01.F01.T01	X0000	アクションを実行するためのトリガとなるタグ

アクション設定

時刻同期アクション「A01」を設定します。



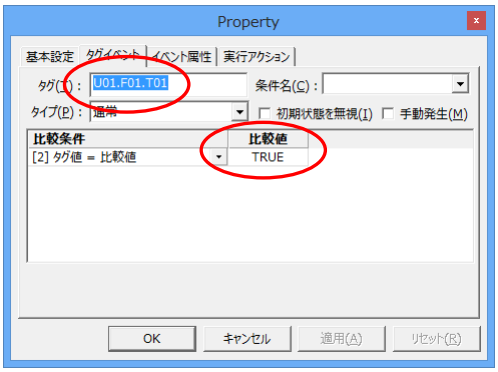
シャットダウンタブから、スクリーンショットの設定を行います。



・同期方法：パソコンの時刻を PLC（タグ）へ設定する
・年にチェック「U01.F01.YEAR」
・月にチェック「U01.F01.MONTH」
・日にチェック「U01.F01.DAY」
・時にチェック「U01.F01.HOUR」
・分にチェック「U01.F01.MINUTE」
・秒にチェック「U01.F01.SECOND」

イベント設定

アクションを実行するためのタグイベントを作成します。今回は U01.F01.T01 が TRUE になったら、アクションを実行します。実行アクションは「A01」を設定します。



動作確認

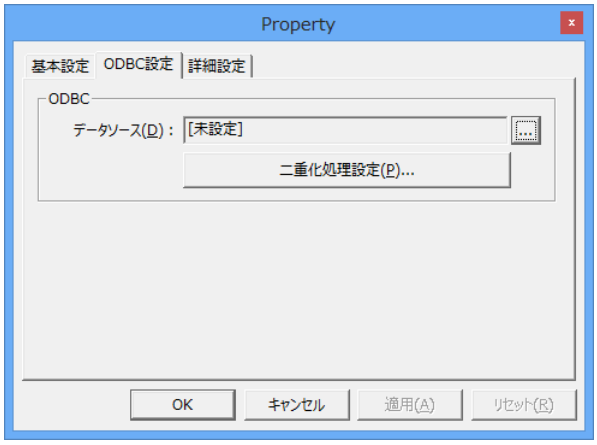
アプリケーションをオンラインにしてください（黄色矢印）。オンライン後、U01.F01.T01 に TRUE を書き込み、PC の時刻がタグに書き込まれることを確認してください。

このサンプルの応用例

本サンプルでは PC の時刻を PLC へ反映するものですが、同期方法で「PLC(タグ)の時刻をパソコンへ設定する」を選択することで、PLC の日付データを元に、PC の時刻を変更することができます。ただし、管理者権限など、時刻を変更できるユーザーでログインしておく必要があります。

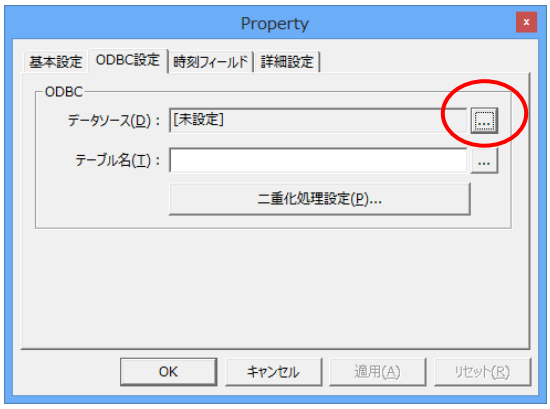
Appendix：　アクション共通の設定

ODBC 設定



データベースに接続するいくつかのアクションではODBCの設定が必要になります。ODBCの設定方法は共通です。ここではMicrosoft Access「sample.mdb」の「sampletable」というテーブルをODBC経由で使用できるように設定する例を紹介します。

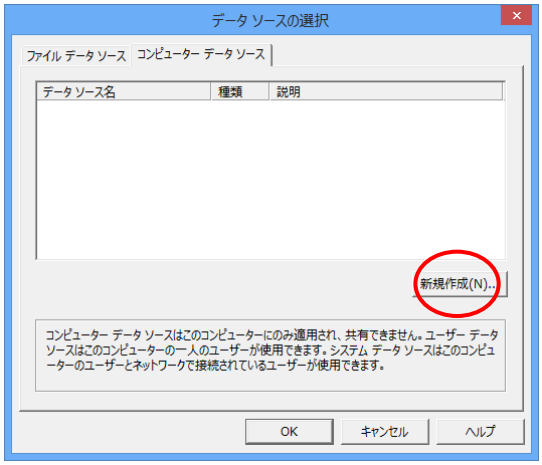
1



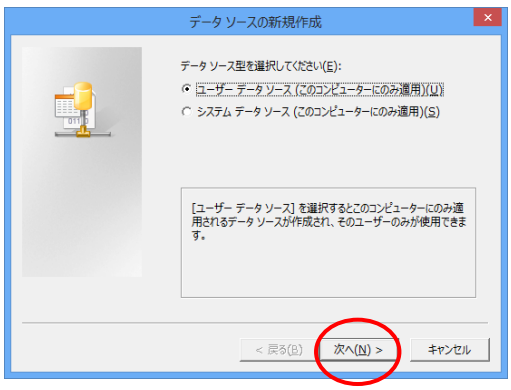
2



3



4



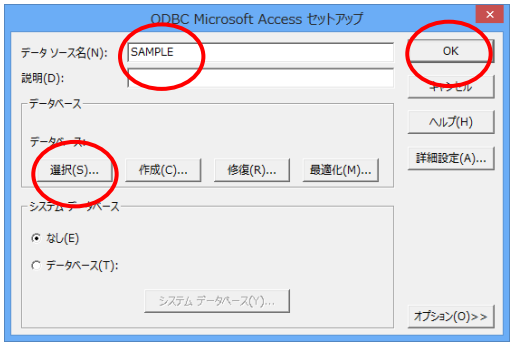
5



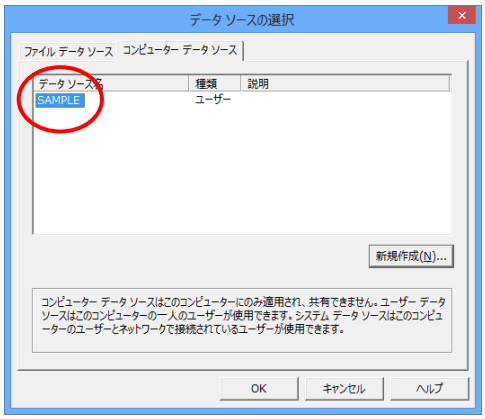
6



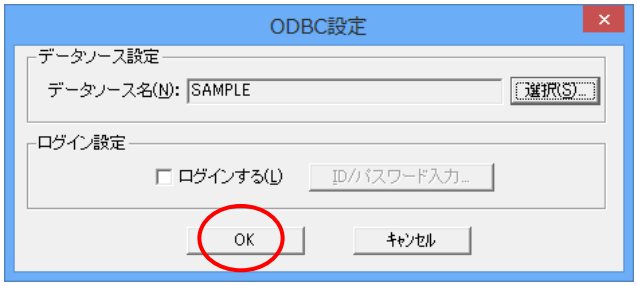
7



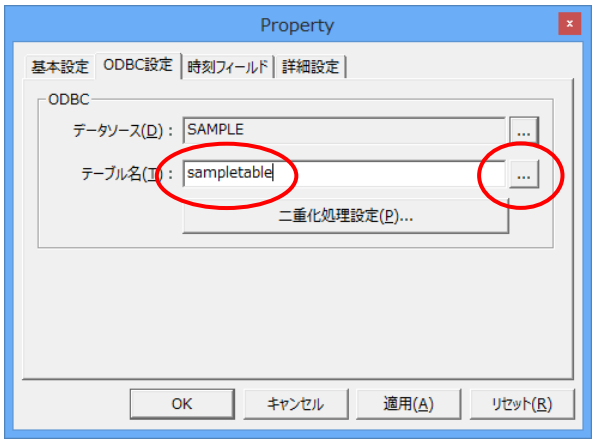
8



9



10



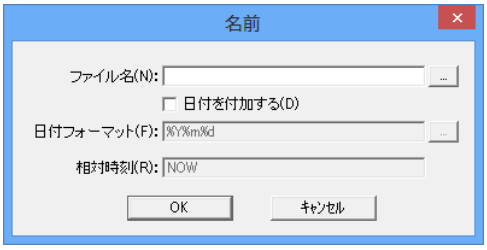
ヒント

ODBC ドライバは 64BitOS を使用している場合でも、32Bit 版を使用する必要があります。
64BitOS の場合、32Bit の ODBC ドライバは下記のものになります。
「C:¥Windows¥SysWOW64¥odbcad32.exe」

名前フォーマット

名前フォーマットとは、ファイル名やフォルダ名、データベースのフィールド名などに日付や時刻を付加して、動的な名称指定を可能とする便利な機能です。この機能により、開発時に設定する固定名称としてのみならず、例えば処理対象日付のフォルダ名を指定する、などの処理が可能になります。

■ファイル名として



■フォルダ名として



■名前として（データベースのフィールド名など）



まず、「ファイル名」「フォルダ名」「名前」の欄に名前を指定します。単純に固定の名前を指定したいのであれば、「日付を付加する」のチェックを外します。名前に日付を付加したい場合は「日付を付加する」をチェックし、「日付フォーマット」及び「相対時刻」を指定します。すると、名前の末尾に日付や時刻が自動的に付加されます。ファイル名やフォルダ名の場合は、拡張子（例えば.txt など）の前に日付や時刻が付加されます。「相対時刻」は現在を基準として、いつの時刻や日時なのかを「相対時刻フォーマット」で指定します。「相対時刻フォーマット」については次ページを参照下さい。

「日付フォーマット」は下記のように指定します。
例）

%Y%m%d	:	20030101
%Y%m%d%H%M%S	:	20030101000000
%Y/%m/%d	:	2003/01/01
%Y/%m/%d %H:%:M:%S	:	2003/01/01 00:00:00

フォーマット	解説
%Y	4 桁で表す西暦 (0100～9999)
%y	2 桁で表す西暦の下 2 桁 (00～99)
%#y	先頭に 0 がつかない西暦の下 2 桁 (0～99)
%m	2 桁で表す月 (01～12)
%d	2 桁で表す日 (01～31)
%p、%#p	午前/午後を表す、AM/PM
%I	2 桁で表す 12 時間表記の時間 (01～12)
%H	2 桁で表す 24 時間表記の時間 (00～23)
%M	2 桁で表す分 (00～59)
%S	2 桁で表す秒 (00～59)

相対時刻フォーマット

相対時刻フォーマットとは現在の時刻を起点として、1 日後や 1 時間後などのように相対的に指定する方法です。

例）現在が「2003/01/15 12:12:12」（水曜）の場合

now	現在。つまり「2003/01/14 12:12:12」。
now-1d	現在-1 日。つまり「2003/01/14 12:12:12」。
now+1d	現在+1 日。つまり「2003/01/16 12:12:12」。
now-1mo	現在-1 ヶ月。つまり「2002/12/15 12:12:12」。　※下記の注意を参照
month	今月。現在がつまり「2003/01/01 00:00:00」。
month-1d	先月の末日。つまり「2002/12/31 00:00:00」。
month-1d+9h	先月の末日の朝９時。つまり「2002/12/31 09:00:00」。
day-1d	昨日。つまり「2003/01/14 00:00:00」。
day+8h+30m	今日の８時３０分。つまり「2003/01/15 08:30:00」。
week+2d	今週の火曜日。つまり「2003/01/14 00:00:00」。

相対時刻のフォーマットは下記のようになります。

基準時刻[+時間間隔] [-時間間隔]...

上記フォーマットの基準時刻の部分には下記の基準時刻が指定できます。

基準時刻	解説	現在が「2003/06/18 12:12:12」（水曜）とすると
now	現在	2003/06/18 12:12:12
year	今年	2003/01/01 00:00:00
month	今月	2003/06/01 00:00:00
week	今週	2003/06/15 00:00:00
day	今日	2003/06/18 00:00:00
hour	今の時刻（時まで有効）	2003/06/18 12:00:00
minute	今の時刻（分まで有効）	2003/06/18 12:12:00

時間間隔の前には「+」か「-」及び数値を必ずつけます。「+」であれば未来へ、「-」であれば過去になります。例えば「+10m」であれば基準時刻から 10 分後を指します。

時間間隔	解説
y	年
d	日
mo	月
w	週
h	時
m	分
s	秒



現在が 31 日の場合、NOW に対して月の引き算を含めた相対時刻（例　NOW-1MO 等）を使用できません。理由は、31 日の 1 ヶ月前が 30 日までしかない月の場合、該当する日が存在しないためエラーが発生しますので注意してください。

「アクションマニュアル」Copyright © 2010 RoboticsWare, Inc. All rights reserved.
2010 年 6 月 1 日 発行　株式会社ロボティクスウェア
〒331-0811 埼玉県さいたま市北区吉野町 1-11-1　TEL:048-782-9861 FAX:048-782-9862

■本書は著作権上の保護を受けています。本書の一部または全部について、株式会社ロボティクスウェアの
文章での許諾を得ずに、いかなる方法においても無断で複写・複製することは禁じられています。
■記載されている会社名、製品名、ブランド名は、対応する法人または個人の（登録）商標です。
■このソフトウェアの仕様及び本書に記載されている事項は、将来予告無しに変更することがあります。