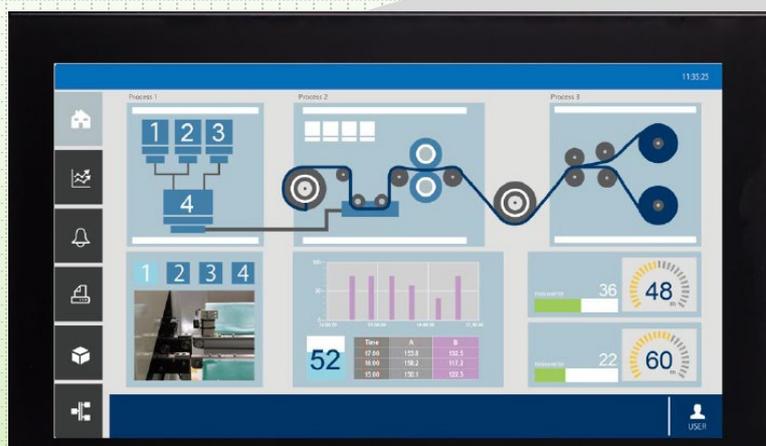


プログラマブル表示器 WH 虎の巻



Panasonic

Let's Visualize!

xAscender Studio



memo

目次

1 - WH 本体設定・仕様.....	3
ランタイムの削除.....	4
スプラッシュ画面の変更方法.....	5
システムログの取得方法.....	7
VNC.....	10
ブラウザでのシステムセッティング画面の開き方.....	11
本体メモリ使用状況.....	12
Web ページでサポートされていない部品/機能.....	13
各機能の上限数.....	15
内部変数タグを初期化する方法（ホームページ表示時初期化）.....	16
2 - xAscender studio 機能.....	17
タグのインポート.....	18
インポートしたタグの編集.....	20
文字列のエクスポート、インポート.....	22
PLC からのページ切り替え.....	24
Web ページの変更.....	25
ページカテゴリの追加.....	26
タグクロスリファレンス.....	27
オブジェクトビュー.....	28
ダイアログタイプの変更.....	29
ブザーの設定.....	30
更新パッケージ.....	32
数式の登録.....	34
ライブタグ.....	35
カラーパレット機能.....	36
インデックス付きタグセット.....	38
バックライトの設定.....	40
データ転送.....	41
デフォルトフォントの変更.....	44
トレンド.....	45
トレンドの出力(csv).....	47
トレンドデータのグラフ化.....	48
ユーザの切り替え.....	49
ユーザ権限の変更.....	50
ユーザ登録.....	52
レシピによる複数個の内部変数タグを初期化する方法 1（ホームページ表示時初期化）.....	53
レシピによる複数個の内部変数タグを初期化する方法 2（初期化画面で初期化）.....	55
WH の変数種類概要説明.....	57
変数タグ詳細説明.....	59
変数ウィジェットによるローカル変数定義.....	60
プロジェクト変数ウィジェットによるグローバル変数定義.....	62
システム変数の概要とデフォルト変数・システム変数.....	64
システム変数タグの保持メモリ変数.....	68
タグの一括読み込み（データ転送機能使用・非サイクリック通信）.....	70
タグの一括書き込み（データ転送機能使用・非サイクリック通信）.....	73
PDF 表示機能 1 PDF ファイルの WH への格納方法(FTP 転送).....	75
PDF 表示機能 2 PDF ビューの起動と PDF ファイルの閲覧.....	78

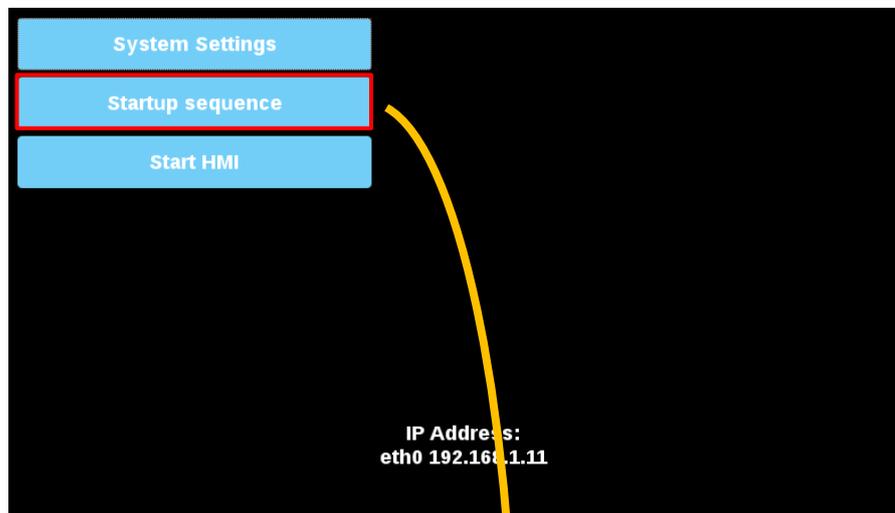
正規表現の使い方	80
タグ名の一括置換方法(正規表現で部分置換)	83
エクセルでのタグ名編集とタグのインポート	88
3 - 部品の設定	95
「ボタン」部品の ON/OFF による表示文字の切り替え	96
Web コントロール部品	98
プラグインのインストール方法	99
Combo Box	101
User Widgets の作成	103
IP カメラ	105
アイコン付きボタンのアイコン変更方法 1 (ファイル名を複製)	106
アイコン付きボタンのアイコン変更方法 2 (ファイル一覧より選択)	107
エクスプローラに SVG ファイルのサムネイルを表示させる方法	108
メディアウィジェット表示時に自動再生させない設定	109
メディアプレイヤーウィジェット表示時に自動再生させない設定	110
ラジオボタン作成方法 1 (PLC との通信タグを使用する場合)	111
ラジオボタン作成方法 2 (内部変数タグを使用する場合)	112
4 - 通信設定	113
Modbus 通信時の設定	114
FTP クライアント機能の使用法	115
FTP サーバの有効化	117
シミュレータでの PLC 連携	118
通信対象機器構成が変動するシステムでの通信局無効指定	120
5 - JavaScript 関連	122
WH での JavaScript の動作の仕組み	123
オブジェクト種類と JavaScript による操作	125
JavaScript による 1 ショットタイマー使用例	126
JavaScript による 1 ショットタイマー処理方法	127
JavaScript によるウィジェットプロパティ操作	128
JavaScript によるタイムインターバル使用例	130
JavaScript によるタイムインターバル処理方法	131
JavaScript によるラベル表示の折り返し表示方法	132
JavaScript によるタグ操作方法	133
JavaScript によるラベル付きボタンの表示内容変更方法	134
JavaScriptによる複数の内部変数タグ初期化方法 1 (ホームページ表示時初期化)	135
JavaScriptによる複数の内部変数タグ初期化方法 2 (初期画面で初期化実行)	136
タグの一括書き込み (JavaScript 機能使用・非サイクリック通信)	137
タグの一括読み込み (JavaScript 機能使用・非サイクリック通信)	139

1 - WH 本体設定・仕様

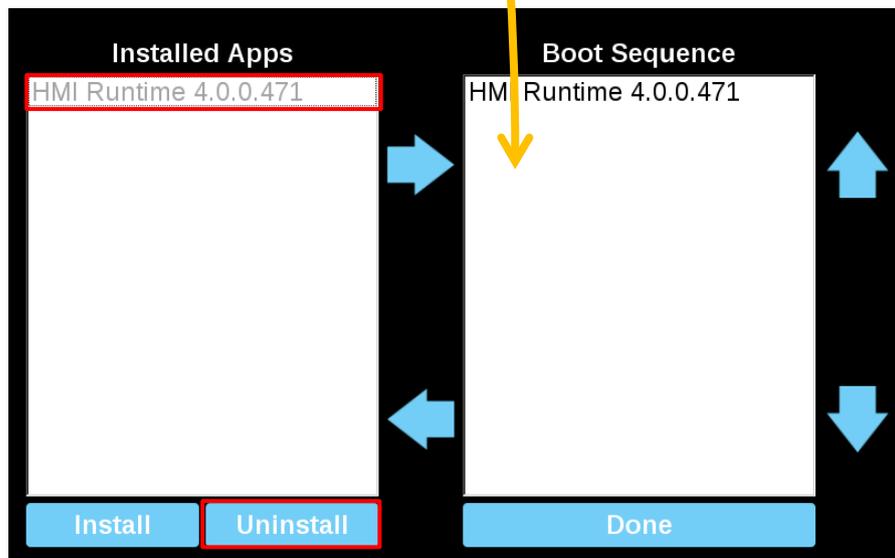
ランタイムの削除

WH 本体に入っているランタイムの削除方法を紹介します。

WH の電源投入し、直後に出てくる「xAscender」のsplash画面で本体画面を 3 回タッチします。
画面上部に「###」と表示されたら、そのまま数秒間待ち、下記画面が表示されます。



次に「Startup sequence」を選択します



インストールされているランタイムを選択し、「Uninstall」をタッチすることで、ランタイムを削除することができます

スプラッシュ画面の変更方法

WH 本体起動時のスプラッシュ画面を変更する方法を紹介します。

まず初めに変更したい画像のファイル名を「splash.bmp」に変更し、任意のフォルダ(ソースフォルダ)内に保存します

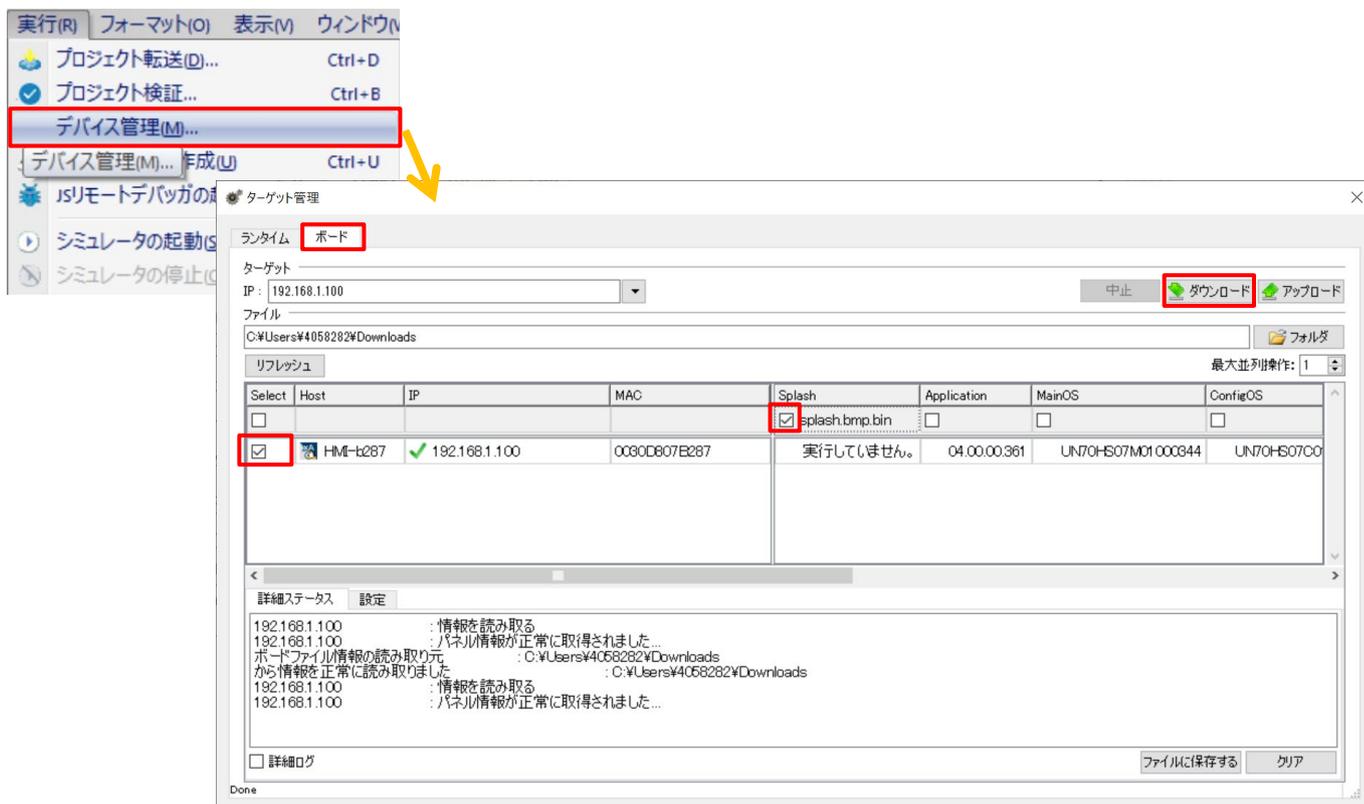
Splash 画面で使用される画像は、次の要件を満たしている必要があります。

ファイル名 : Splash.bmp
形式 : ビットマップ、RGB 565 形式
サイズ : 500KB 未満
ビットマップ幅 : 偶数(例:430×239)

メニューバーの「実行」-「デバイス管理」をクリックし、ターゲット管理ダイアログを表示します。

「ボード」タブを選択します。WH と接続した状態で「フォルダ」をクリックし、「splash.bmp」を保存したフォルダを選択します。フォルダを選択した後、「Select」に を入れると、WH の情報を読み取ります。

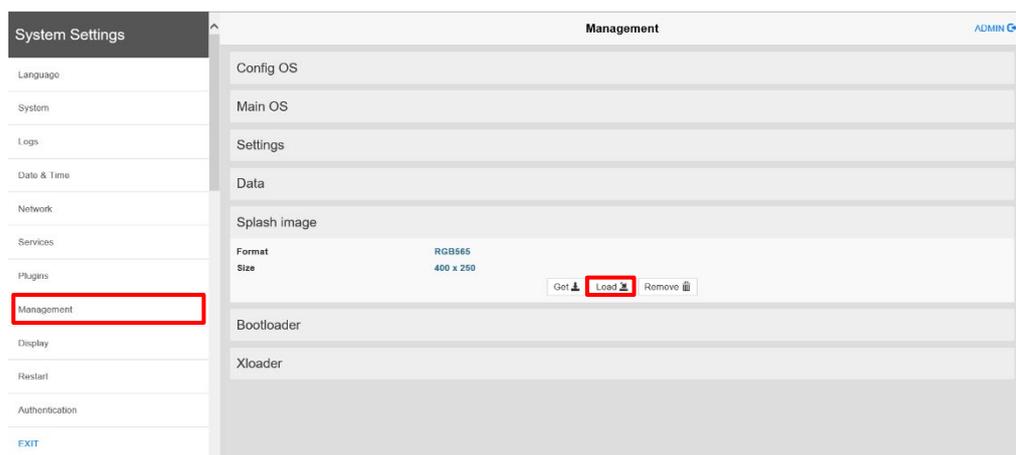
読み取った後、Splash に を入れて「ダウンロード」をクリックすると、起動画面が変更されます。



Web ブラウザで変更する方法

上記の操作を行うとダウンロード後は、ソースフォルダに「splash.bmp.bin」と「splash.bmp.bin.MD5」が生成されます。一度生成した上記ファイルは、別の WH でも使用することができます。

Web ブラウザで「https://(WH 本体の IP アドレス)/machine_config」を検索すると System Settings 画面が表示されます。「Management」-「Splash image」-「Load」をクリックし、「splash.bmp.bin」ファイルと「splash.bmp.bin.MD5」ファイルを選択することでスプラッシュ画面を変更することができます。



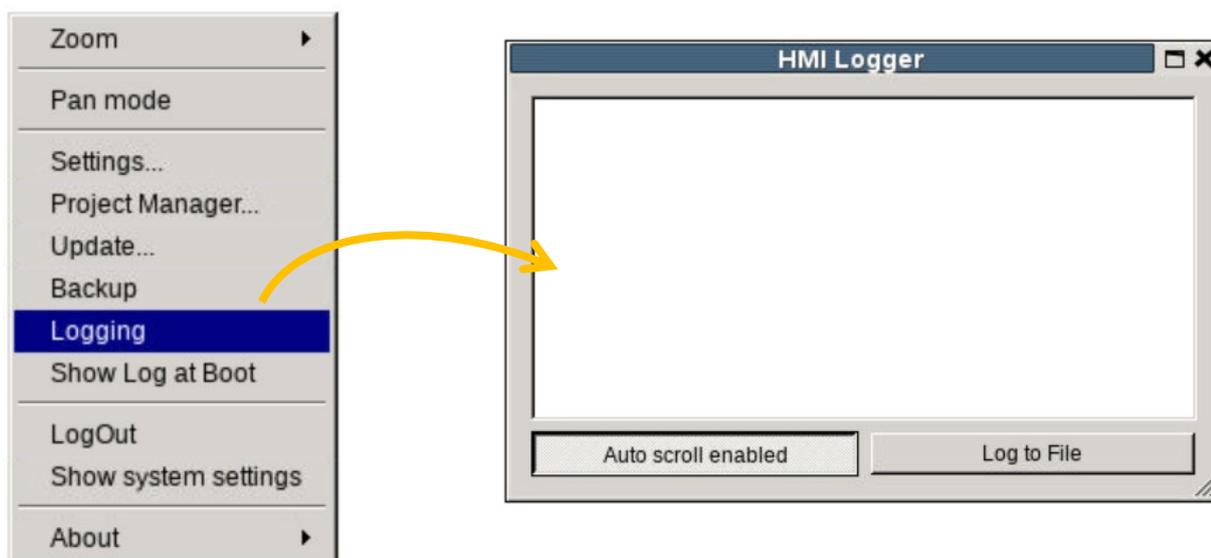
システムログの取得方法

WH 本体のシステムログを取得する方法を紹介します。

システムのログおよび本体でエラーが発生した際のログを確認することができます。
またログをファイルで保存することもできます。

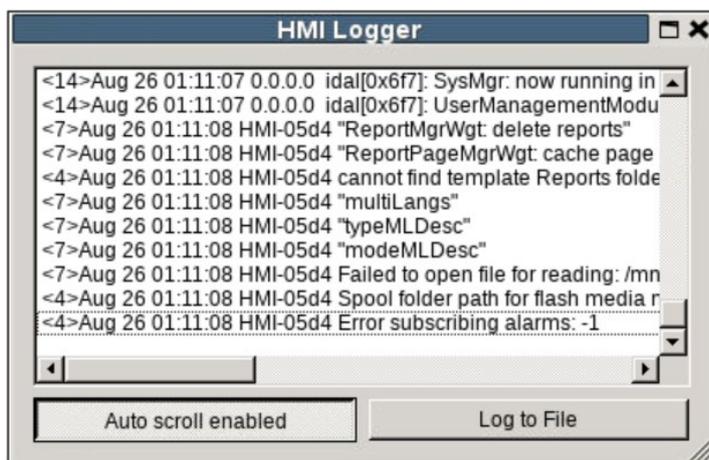
システムログの表示方法

本体画面を 2 秒以上タッチ→コンテキストメニュー内「Logging」を選択し、
「HMI Logger」を起動します。



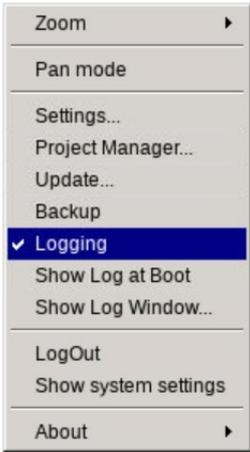
この状態でプロジェクトのダウンロードなど WH 本体に対して操作を行うとログが表示されます。
エラーが発生した際やシステムに何かしらの不具合が発生した際のログもここで表示されます。

例:プロジェクトダウンロード



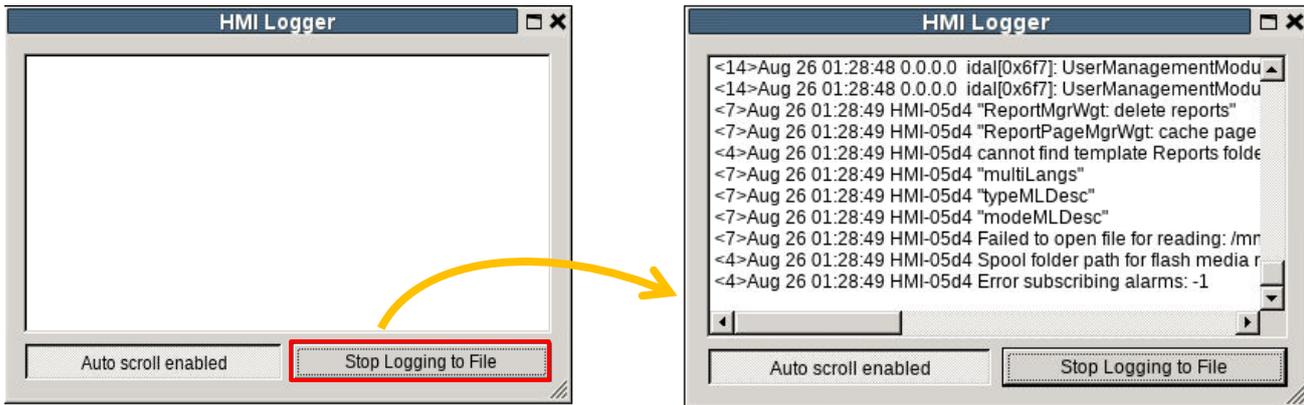
ロギングの停止

上記の操作でロギングを開始した場合、「HMI Logger」を閉じてもロギングは継続されます。ロギングを停止させるには、もう一度コンテキストメニューを開き、再度「Logging」を選択します。（Logging に☑が付いている場合、ロギングが実施されている状態になります）

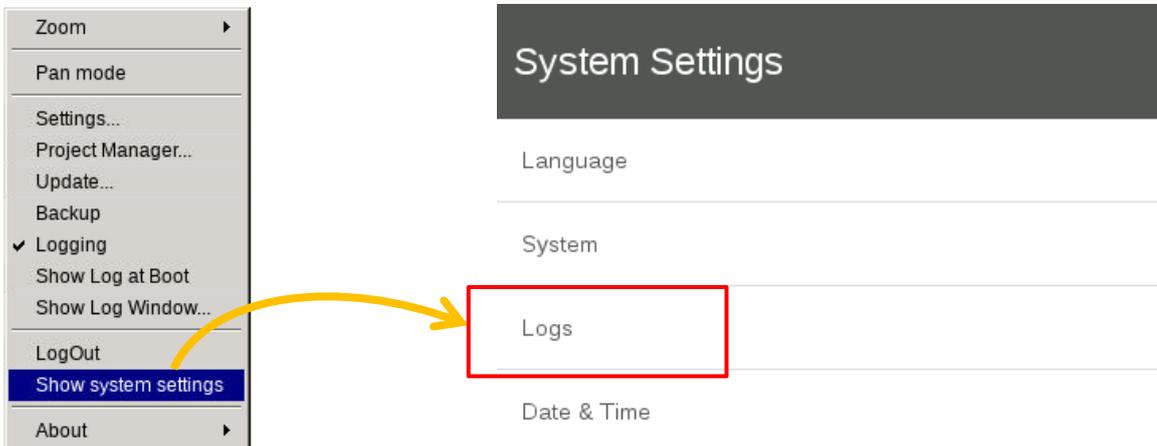


ファイルの保存

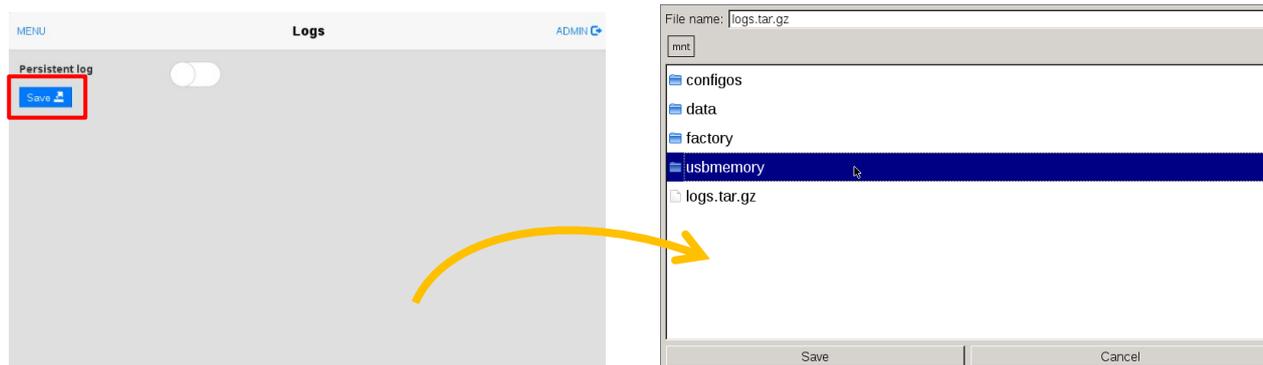
ロギングした内容をファイルに保存したい場合、「HMI Logger」内の「Log to File」をクリックします。（Log to File →→ Stop Logging to File に変化したことを確認）この状態で操作を行い「HMI Logger」上にログを表示させます。



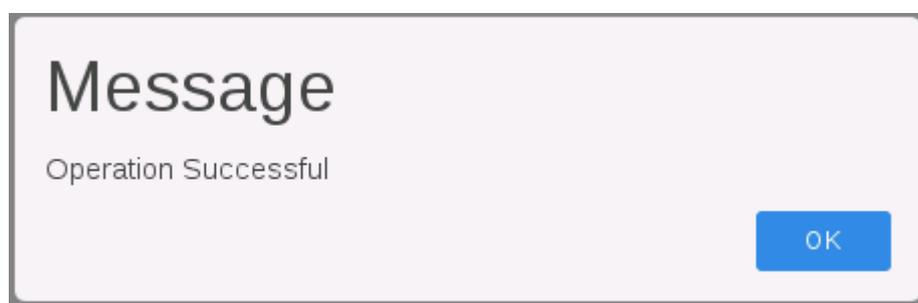
この状態でコンテキストメニュー内の「System Setting」→「Logs」を選択します。



「Logs」内の Save をタッチして、保存先を選択し、左下の「Save」を押します。



下記のメッセージが表示されたら保存は完了です。



VNC

VNC (Virtual Network Computing) 機能を使用することで、遠隔での WH の操作や Web 画面では表示できないコンテキストメニューなどを PC 上に表示させることができます。

VNC の有効化

WH 本体画面を 2 秒以上長押し→メニュー内「System Settings」を選択

「Service」を選択し、「VNC Service」を ON に変更します。

The screenshot displays the 'System Settings' web interface. On the left is a sidebar menu with categories: Language, System, Logs, Date & Time, Network, Services (highlighted with a red box), Management, and Display. A yellow arrow points from the 'Services' menu item to the 'Service Settings' page. The 'Service Settings' page has a header with 'MENU', 'Service Settings', and 'ADMIN' with a lock icon. Below the header is a list of services with their status and a chevron icon:

Service Name	Status
Fast Boot	Off
Firewall Service	Off
Router / NAT / Port forwarding	Off
Show loading bar during boot	On
SNMP Server	Off
VNC Service	On

The 'VNC Service' row is highlighted with a red box, and its status is 'On'.

PC と接続

PC 上で VNC クライアントソフトを用意し、WH と接続することで VNC 機能を使用することができます。

ブラウザでのシステムセッティング画面の開き方

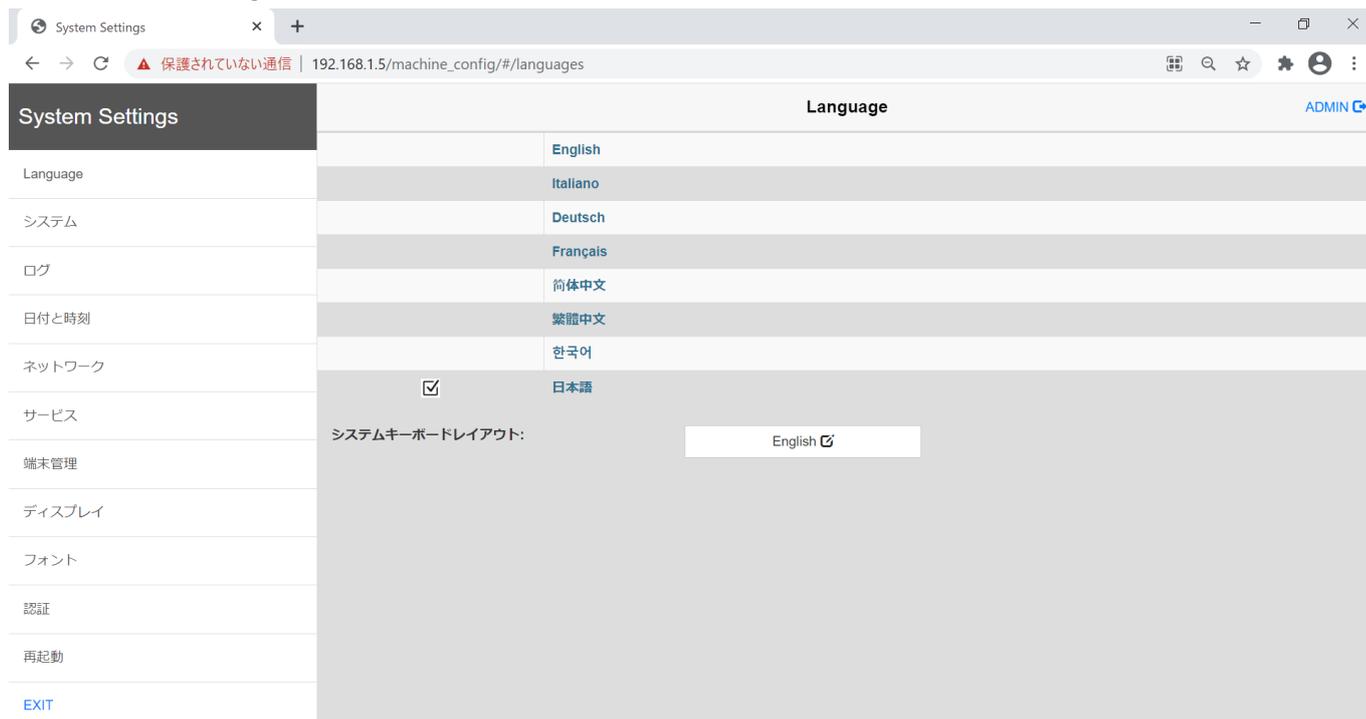
Google Chrome などのブラウザからシステムセッティング画面にアクセスする方法を説明します。

■PC と WH を同一のネットワークに接続

この記事では、PC と WH の IP アドレスをそれぞれ、192.168.1.6 と 192.168.1.5 と設定しています。

■ブラウザのアドレス欄に「https://192.168.1.5/machine_config/」と入力し、アクセスします。

下記キャプチャは Google Chrome でアクセスした場合の例です。



※初回アクセス時に ID/PASS を求められる場合があります。初期値は admin/admin で設定されています。

本体メモリ使用状況

本体メモリ使用状況の確認方法について説明します。

本体の画面を 2 秒間長押しするとダイアログが表示されます。下から二番目「Show system setting」をクリック、もしくは Web ブラウザで「https://MH 本体の IP アドレス/machine_config」を入力すると System Settings 画面が表示されます。

上記アドレスを検索すると、下図ログイン画面が表示されます。

(デフォルト)

ユーザ名 : admin

パスワード : admin

それぞれ入力し、ログインをクリックします。

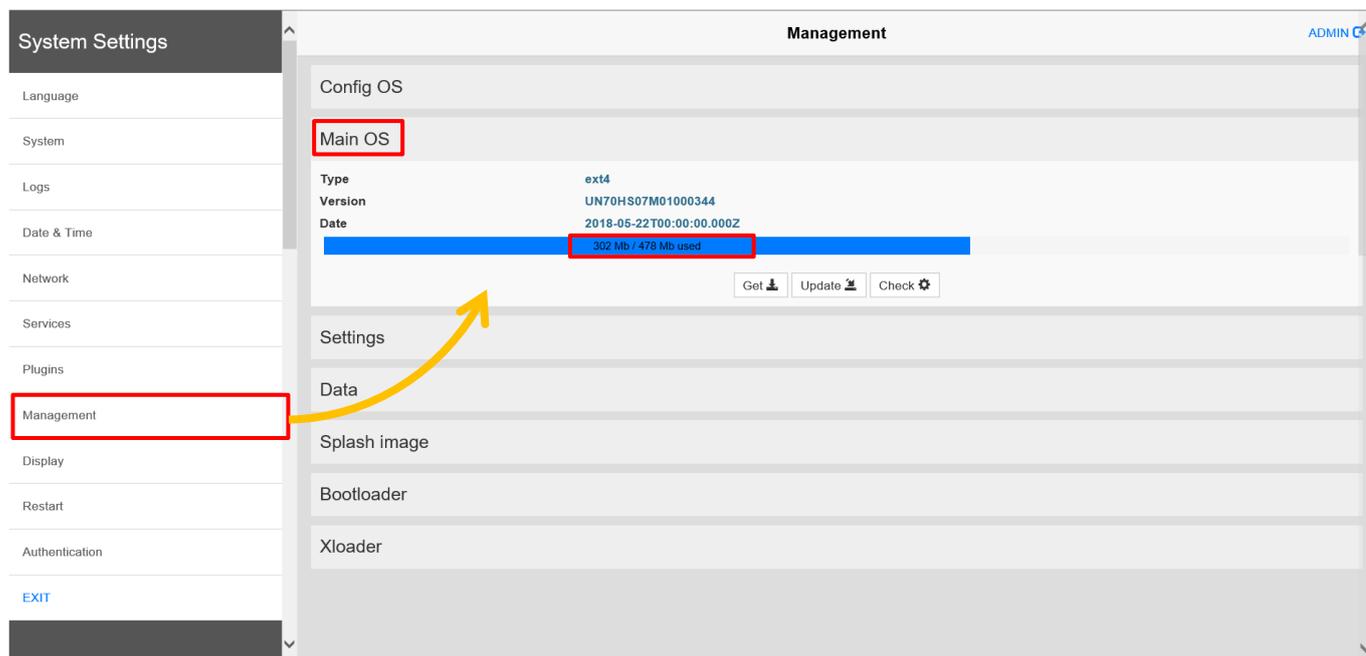


ログイン
https://192.168.1.100

ユーザ名

パスワード

「Management」-「Main OS」にて、使用しているメモリの状況を確認することができます。



The image shows the 'System Settings' application interface. On the left is a sidebar menu with 'Management' highlighted in red. The main content area is titled 'Management' and contains a 'Config OS' section. Within this section, 'Main OS' is highlighted in red. Below it, the OS details are listed: Type (ext4), Version (UN70HS07M01000344), and Date (2018-05-22T00:00:00.000Z). A blue progress bar indicates memory usage, with '302 Mb / 478 Mb used' highlighted in red. A yellow arrow points from the 'Management' menu item to the 'Main OS' section. At the bottom of the 'Config OS' section, there are buttons for 'Get', 'Update', and 'Check'.

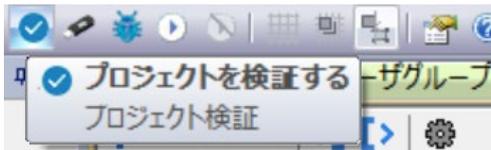
Web ページでサポートされていない部品/機能

Web 対応機能

一部の機能またはウィジェットのプロパティは、WH Web サーバではサポートされていません。
サポートされていないウィジェットが使用されている場合、ウィジェットは WH パネルのページにのみ表示されますが、Web ページではサポートされていないウィジェットは表示されません。
プロジェクト検証ツールを実行して、ページにサポートされていないウィジェットが含まれているかどうかを確認できます。

プロジェクト検証

メニューバー内アイコン



Web でサポートされていないウィジェット一覧

- Analog Clocks
- Analog Video
- BACnet
- Button gstArea
- Consumption meter
- Control List
- DateTime Combo (Date or Time can be set using other widgets)
- IP Widget (The IP can be configured via system settings with a browser)
- IPCamera (Supported for Chrome and Firefox. See "Web Browser" on page 464 for additional details)
- Media Player
- Multistate image multilayer (Multistate image widget can be used)
- Rotation menu widget
- RSS Feed
- RSS Scroll
- Scheduler
- Text Editor
- Web Browser
- Hyper Link

Web でサポートされていない機能のリスト

- Context menu
- Buzzer on touch
- Javascript debugger
- Wheel actions (Browser use wheel events to manage scroll bars)
- Combo box full-screen mode (Standard "context" mode is supported)
- Keypads
- ScreenSaver
- External Database (SQL4Automation)
- Display Rotation
- Electronic Signature

Web でサポートされていないアクション一覧

Widget	SlideWidget, BeginDataEntry, TriggerIPCamera, MoveIPCamera, RefreshEvent, ContextMenu, ReplaceMedia, OpenComboBox, CloseComboBox, SelectAlarmsOnSrc, ShiftTableDataSrcColumns, ResetTableDataSrcColumns, SetTableSortingColumn
Web Browser	All actions are not supported
Text Editor	All actions are not supported
MediaPlayer	All actions are not supported
Mail	All actions are not supported
FTP	All actions are not supported
Keypad	All actions are not supported
Page	LaunchApplication, LaunchBrowser, LaunchVNC, LaunchPDFViewer, LaunchUpdater, LaunchHMICloudEnabler, LockScreen, LoadProject, LastVisitedProject
Print	All actions are not supported
Tag	DataTransfer, ActivateGroup, DeactivateGroup, EnableNode, BACnetClearPriority, BACnetClearAllPriorities, BACnetSetPriority, ClearRetentiveMemory, ForceReadTag
Trend/Graph	ConsumptionMeterPageScroll
System	Restart, ResetProtoErrCount, SafelyRemoveMedia, ControlUserLED, SaveEventArchive, LogMessage, CopyCodesysProject
Database	Actions All actions are not supported
UserManagement	SwitchUser, ResetPassword, AddUser, DeleteUser, EditUsers, DeleteDynamicFiles, ExportUsers, ImportUsers
RemoteClient	All actions are not supported

Web でサポートされるシステム変数一覧

System Time
 X Screen resolution
 Y Screen resolution
 This Client Group-Name
 This Client User-Name
 Connection status
 This Client ID
 Available System Memory
 Current Language Id
 Current Language Name
 Current Language Code

Web でサポートされない Alarm 機能一覧

- ・トリガー条件に基づくアラームの色は Web ではサポートされていません
- ・実行時にアラームウィジェットを編集できない
- ・HW ベースのスマートフォン/タブレット(一般に組み込みデバイス)では、ユーザーはパフォーマンスの問題を予想できます
> 500 アラームより多いときに
- ・アラームトリガー条件ではページアクションはサポートされていません

Web でのその他の制限事項

ダイアログページはモーダルダイアログのみをサポートします。

一部の特定のウィジェットプロパティはまだサポートされていません。この場合、デフォルト値が使用されます。

プロジェクト検証を使用して、使用されているウィジェットにサポートされていないプロパティが含まれているかどうかを確認できます。

各機能の上限数

ここでは各機能や設定値の上限数を紹介します。

機能	上限数	
	WHS1 シリーズ、WHA1-05	WHA1 シリーズ(5inch を除く)
ページ数	1000	
基本ウィジェット数	2000	
タグ数	10000	
ダイアログページ数	50	200
同時に開けるダイアログ数	5	
レシピ数	32	
レシピのパラメータセット数	1000	
レシピの要素数	1000	
ユーザグループ数	50	
ユーザ数	500	
同時接続リモートクライアント数	4	
スケジューラ数	30	
アラーム数	2000	4000
データ転送数	1000	
テンプレートページ数	50	
ボタン部品ごとに登録できるアクション数	32	
トレンドバッファ数	30	
トレンドバッファごとのタグ数	200	300
トレンドバッファに予約されたメモリ	25Mb	50Mb
トレンドウィジェットごとの曲線数	5	10
散布図ウィジェットごとの曲線数	10	
トレンド表の印刷可能行の最大数	10000	
メッセージ部品のメッセージ数	1000	
言語数	24	
イベントバッファ数	4	
ページごとの JavaScript ファイルサイズ	64KB	
プロジェクトサイズ	240MB	
インデックス付きタグセット数	30	
プロトコル数	4	8
レポート数	32	64
レポートページ数	32	
ユーザフォルダサイズ(UpdatePackage.zip)	512MB	
同時 FTP セッション数	4	

内部変数タグを初期化する方法（ホームページ表示時初期化）

1 個の内部変数を初期化したい場合、

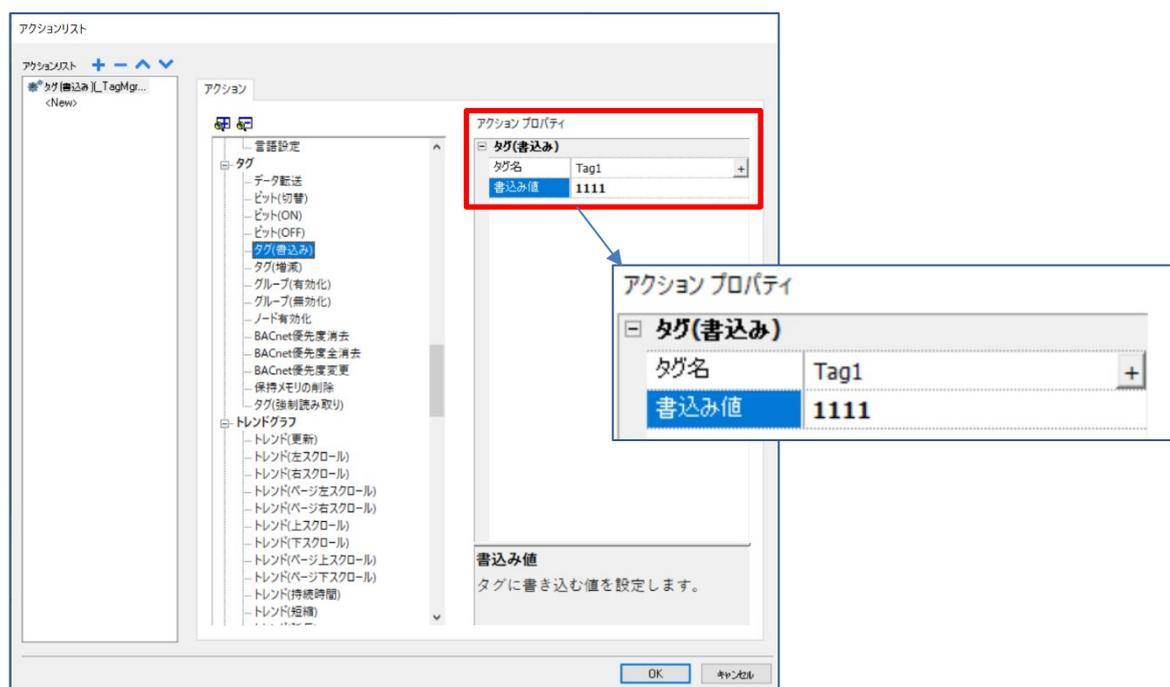
ページイベントのアクティブアクションに**タグ→タグ(書き込み)**を設定して初期化可能です。

ページイベントのアクティブアクションは、ページを表示した場合に起動されます。

複数の内部変数タグを初期化したい場合は、JavaScript による方法参照してください。

プロジェクト起動時だけでなく、ホームページでデータ変更後、別の画面を表示・操作してホームページに戻ってきた場合にも実行されて初期値に戻ってしまいますので、注意が必要です。

下図の例では、Tag1 に 1111 を書き込む設定です。



2 - xAscender studio 機能

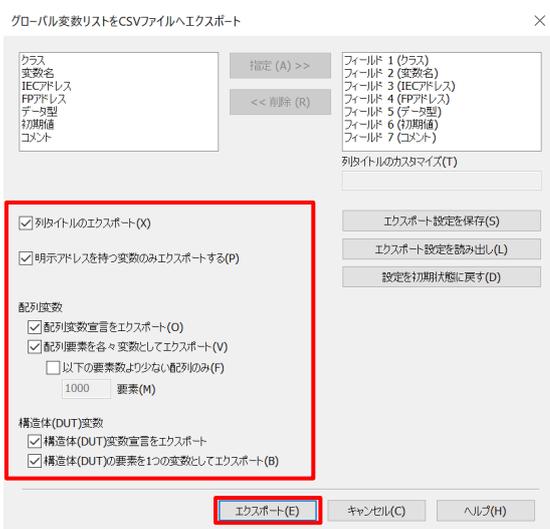
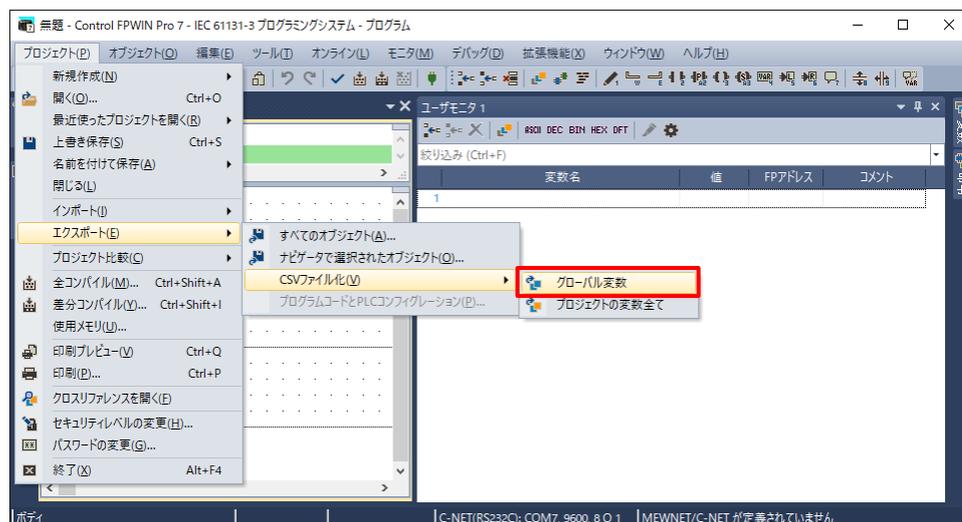
タグのインポート

Control FPWIN Pro7 で宣言したグローバル変数を xAscender Studio のタグにインポートできます。

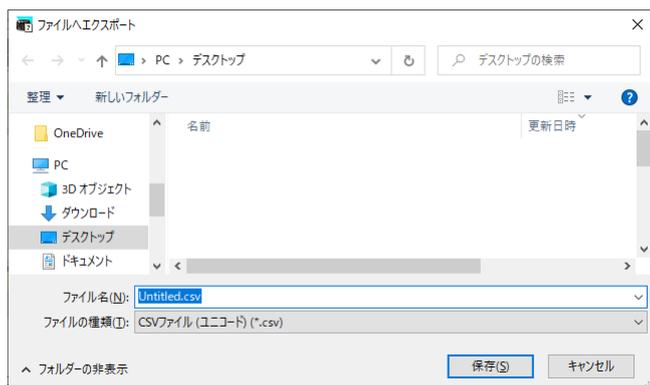
1.

Control FPWIN Pro7 で宣言したグローバル変数をエクスポートします。

メニューバーの「プロジェクト」-「エクスポート」-「CSV ファイル化」-「グローバル変数」をクリックします。



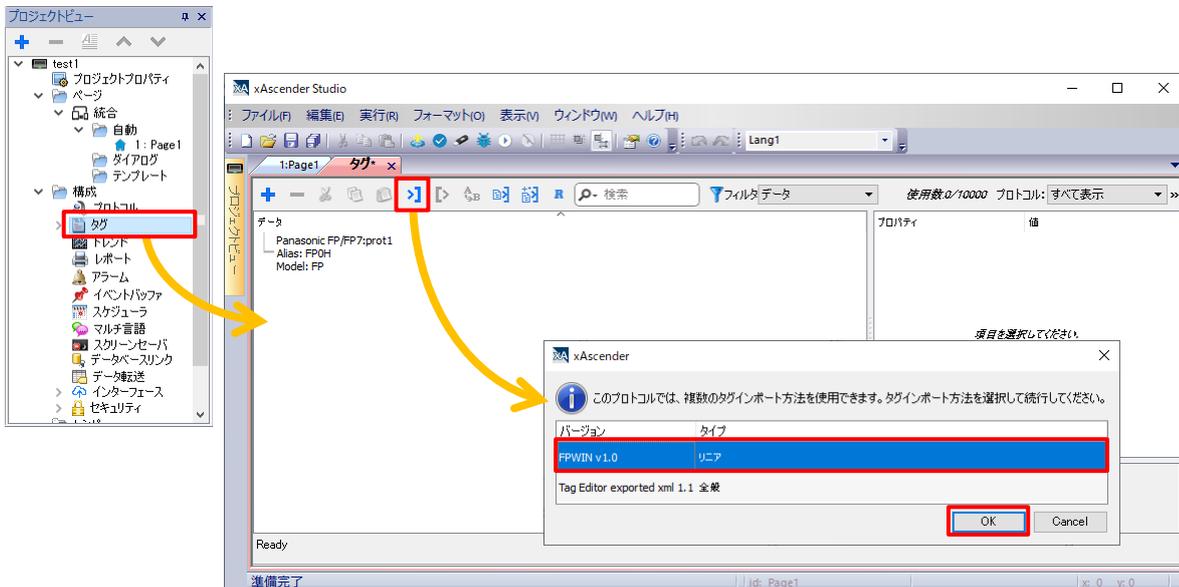
出力のオプションは「以下の要素数よりも少ない配列のみ」以外すべてにチェックを入れて、「エクスポート」をクリックします。



ファイル名を入力し、拡張子(.csv)で保存してください。

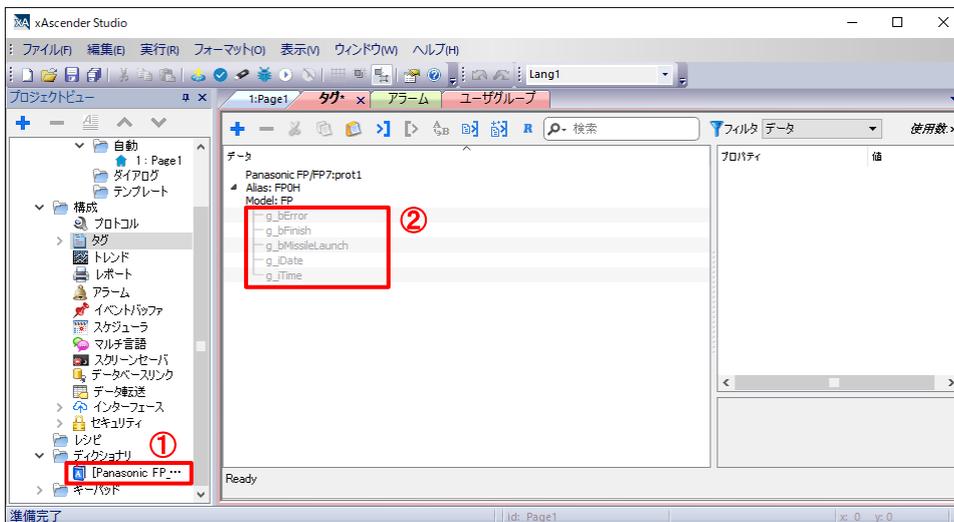
2.

Control FPWIN Pro7 のグローバル変数を xAscender Studio のタグ情報にインポートします。
プロジェクトビューの「構成」-「タグ」をダブルクリックします。



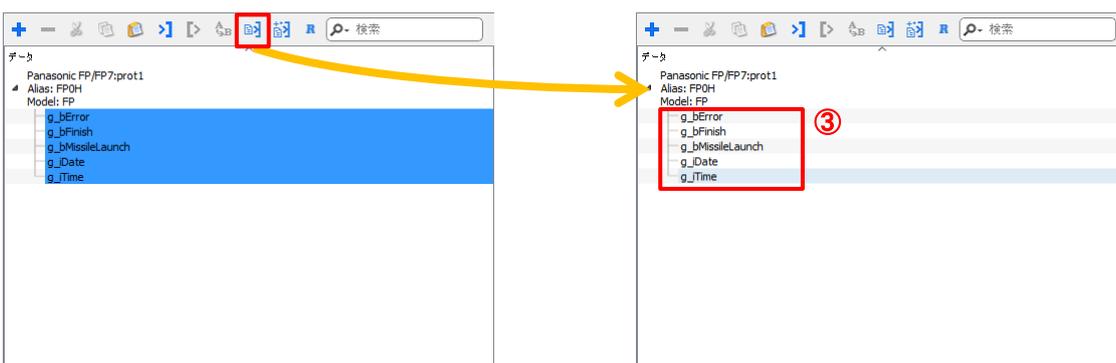
ディクショナリインポート ボタンをクリックし、バージョン“FPWIN v1.0”を選択し、「OK」をクリックします。

Control FPWIN Pro7 でエクスポートした CSV ファイルを読み込むと、ディクショナリが追加され、タグが表示されます。



- ① 追加されたディクショナリ
 - ② 追加されたタグ
- ※ この状態では
まだプロジェクトに追加
されていません。

タグを選択し、タグインポート ボタンをクリックし、プロジェクトに追加します。



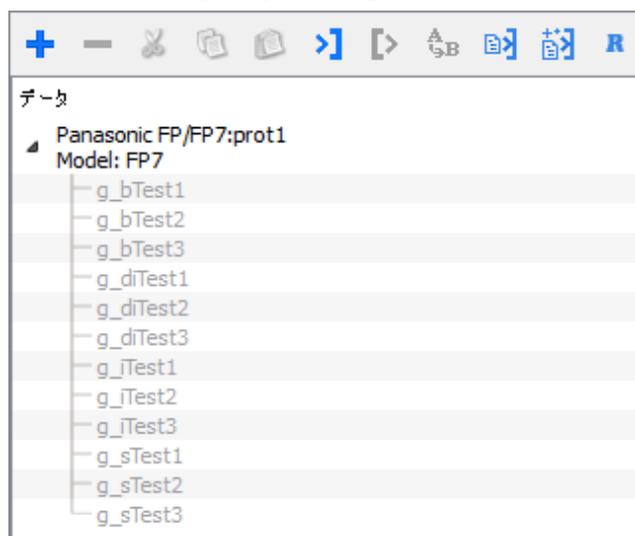
③ プロジェクトに追加されたタグ

インポートしたタグの編集

PLC のデバイスをタグにインポートした際に、使用/使用しない、表示/非表示、削除など編集をすることができます。
ここでは Panasonic 製 PLC を例に説明します。

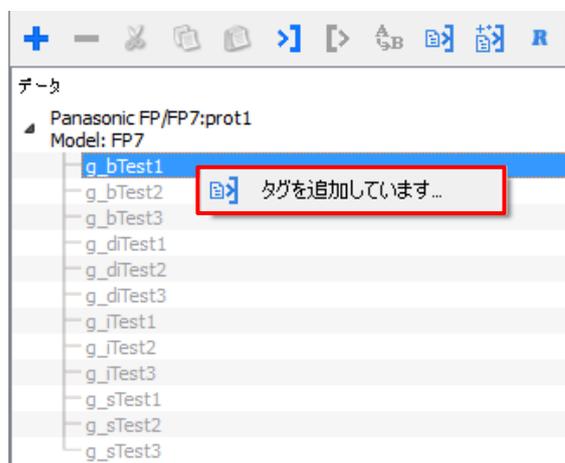
インポート後のタグ

下記のように、インポート直後ではタグ名(変数名)がグレーアウトしていて、プロジェクト内で使用することはできません。

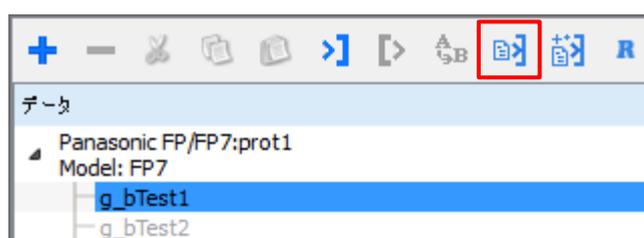


インポートしたタグを使用する

使用したいタグ名で右クリック→「タグを追加しています...」を選択する。



もしくは、使用したいタグ名を選択し、メニューバー内の「タグインポート」をクリックします。



グレーアウトしたタグの表示/非表示

インポートしたが、使用せずグレーアウトしているタグを非表示にすることができます。

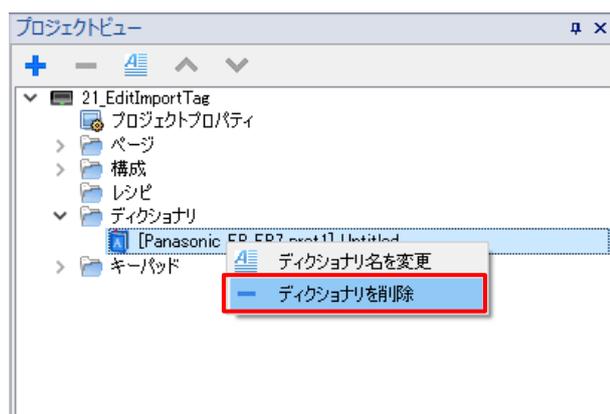
画面右上にある「すべてのタグを表示」の☑を外します



ディクショナリの削除

タグをインポートした際に、新しくディクショナリが作成されます。

このディクショナリを削除することでグレーアウトしているタグを一括で削除することができます。



文字列のエクスポート、インポート

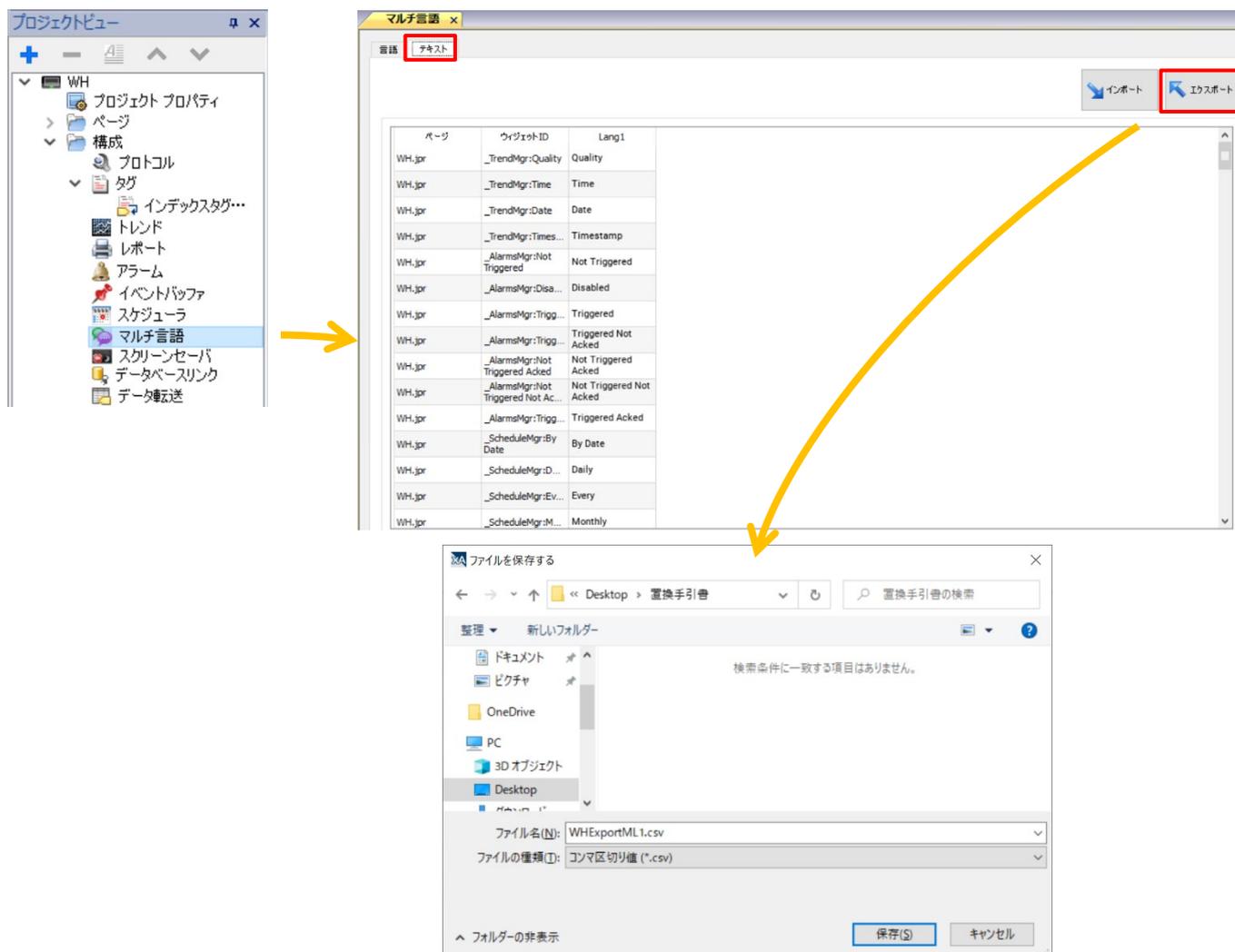
xAscender Studio での文字列のインポート方法を説明します。

WH で「マルチ言語」を使用する際に、他人に文字列データの共有、Excel での編集を行うために、エクスポート、インポート機能が用意されています。

エクスポート

プロジェクトビューの「構成」-「マルチ言語」をダブルクリックします。

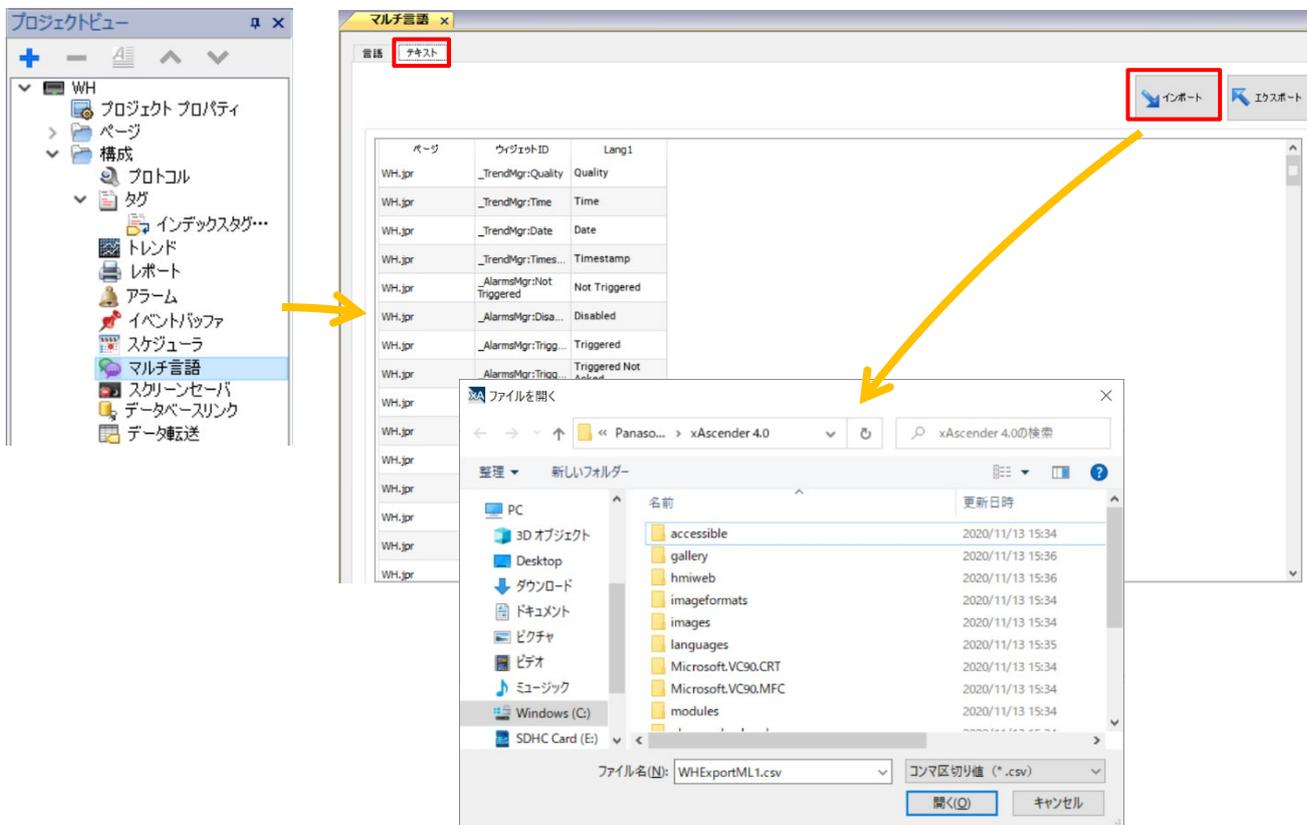
「テキスト」タブを選択し、右上の「エクスポート」をクリックし、csv ファイルとしてエクスポートします。



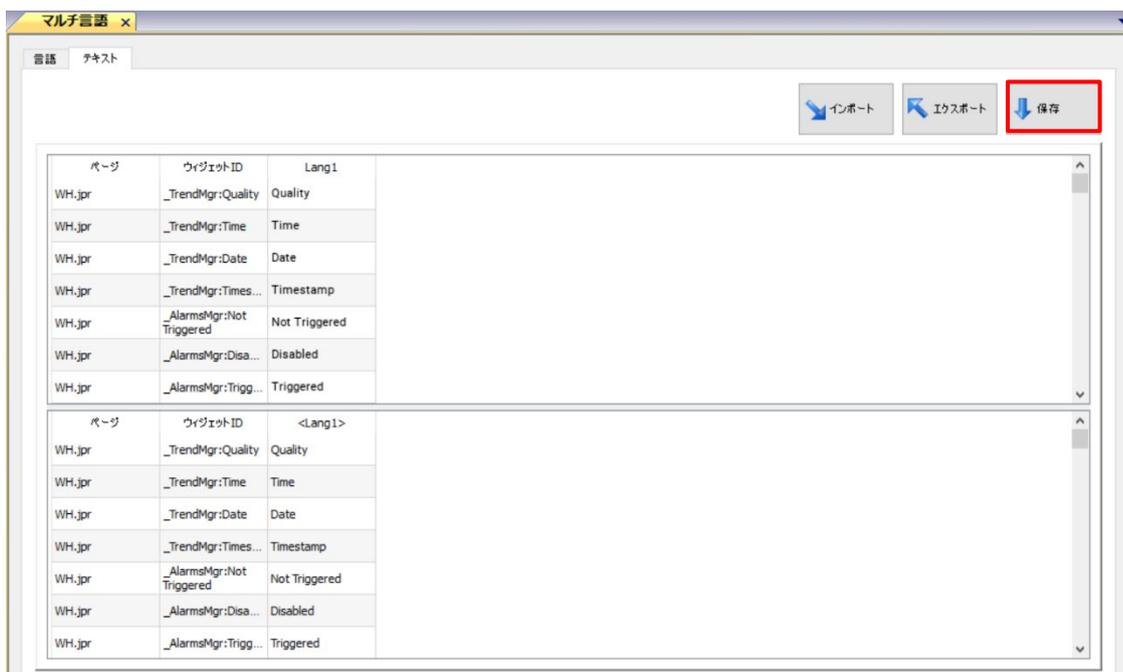
インポート

プロジェクトビューの「構成」-「マルチ言語」をダブルクリックします。

「テキスト」タブを選択し、右上の「インポート」をクリックし、任意の csv ファイルをインポートします。



「保存」をクリックすると、インポートが完了します。

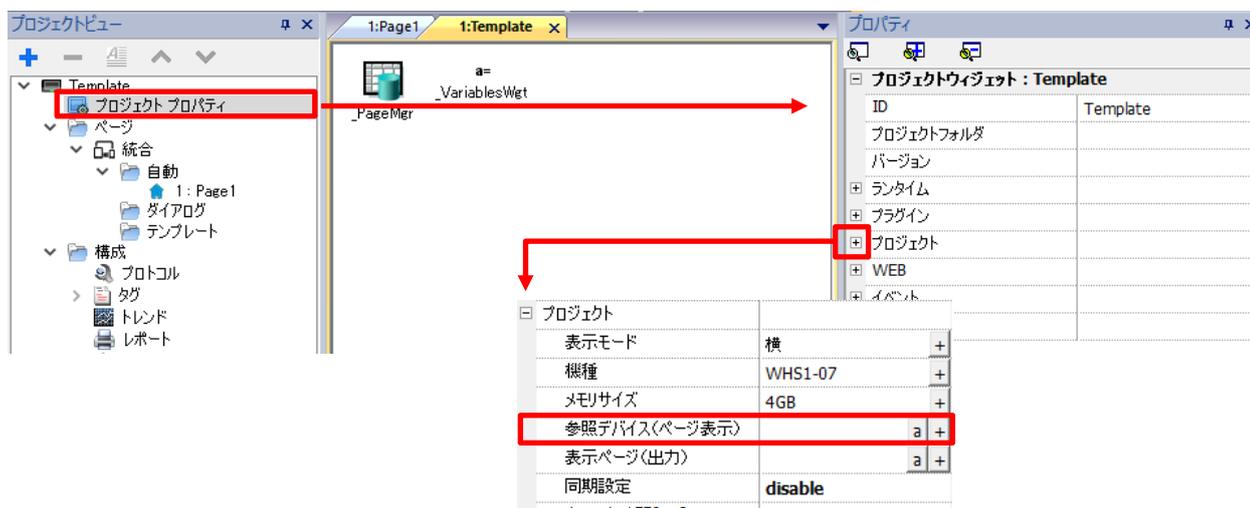


PLC からのページ切り替え

PLC からの指示で、WH 本体のページを切り替える方法を説明します。

プロジェクトツリーの「プロジェクトプロパティ」をダブルクリックしてプロパティを表示させます。

「プロジェクト」の **+** をクリックして展開する設定項目のうち、「参照デバイス(ページ表示)」に数値を書き込むことでページを切り替えることができます。



PLC から「参照デバイス(ページ情報)」の数値を書き込むには、データリンクにタグを割り付けます。

下図では例として、タグ「g_iPage」を割り付けています。

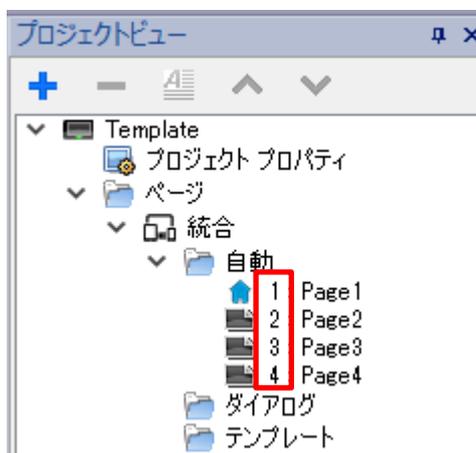
プロジェクト	
表示モード	横
機種	WHS1-07
メモリサイズ	4GB
参照デバイス(ページ表示)	
データリンク	g_iPage
アクセス種類	R
表示ページ(出力)	a
同期設定	disable
ホールド時間[ms]	2000

PLC から、タグ「g_iPage」の値を変更することで、WH 本体のページを切り替えることができます。

タグの値は、プロジェクトビューの「ページ」→「統合」→「自動」に表示されているページ数を使用します。

タグ「g_iPage」の値が 1 の場合: Page1 を表示。
タグ「g_iPage」の値が 2 の場合: Page2 を表示。
タグ「g_iPage」の値が 3 の場合: Page3 を表示。
タグ「g_iPage」の値が 4 の場合: Page4 を表示。
となります。

※タグ「g_iPage」の値が 0 場合、
ページは切り替わりません。



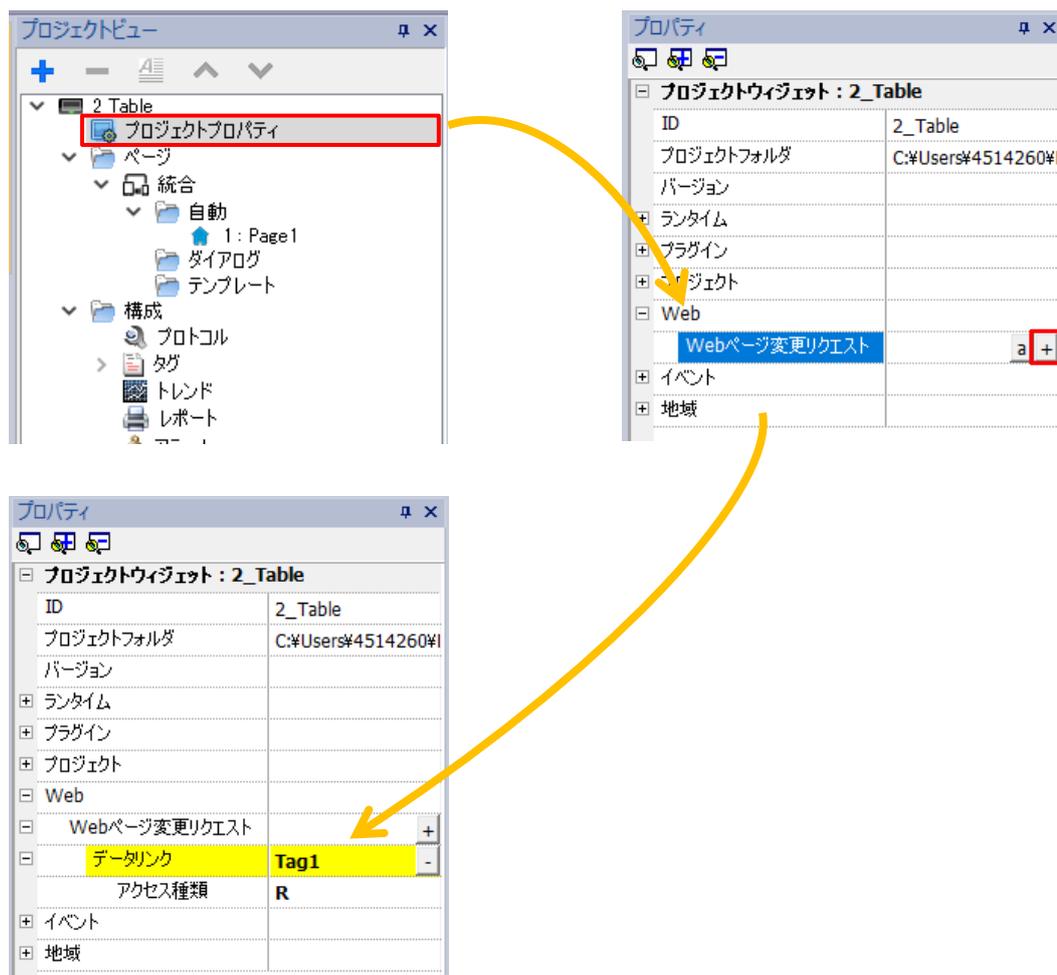
Web ページの変更

ここでは Web ブラウザで画面を表示した際のページ変更方法を紹介します。

Web ブラウザで WH の画面を表示する際、Web で表示するページと WH で表示するページは別々に管理されています。Web ページでも「次のページ」などのアクションを実行することにより、ページの変更はできますが、タグの値を参照してページの変更を行うこともできます。

「プロジェクトビュー内」→「プロジェクトプロパティ」→「Web」

Tag1(Short)を割り付けることで、Tag1 の値を参照して、Web 画面のページを変更することができます。

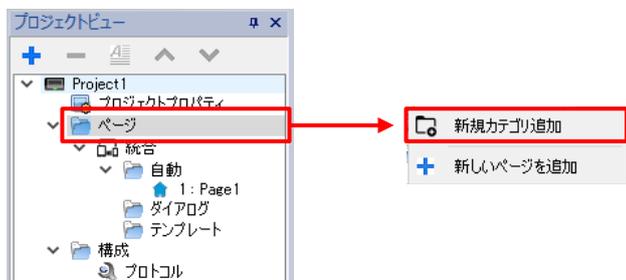


ページカテゴリの追加

WH シリーズは、表示させるデバイスによって異なるページを作成することができます。

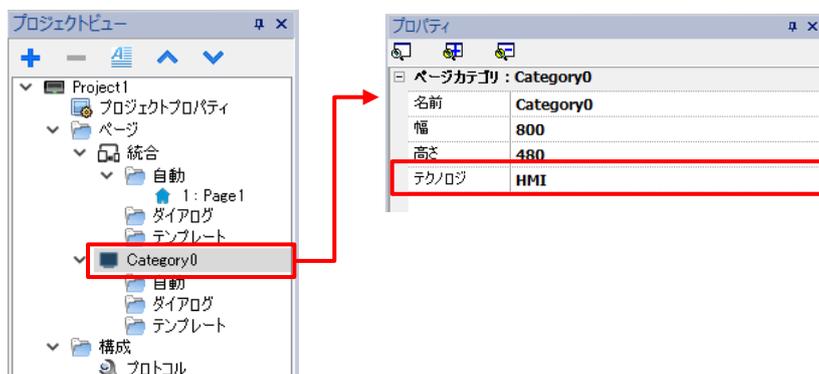
ここでは、すでに作成途中のプロジェクトに、異なる表示デバイスの設定(ページカテゴリ)を追加していく方法を説明します。

1. 「プロジェクトビュー」の「ページ」を右クリックし、「新規カテゴリ追加」を選択します。



2. 新しいページカテゴリが追加されました。

「Category 0」を選択している状態で、プロパティからカテゴリの設定を行うことができます。



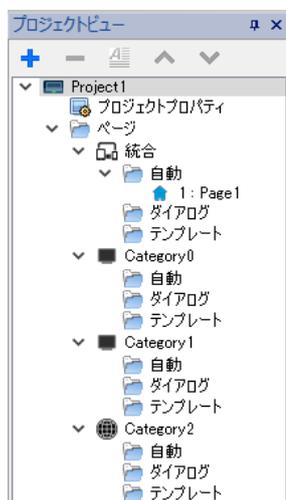
「テクノロジー」から接続するデバイスを選択します。

- ・HMI :WH 本体へ表示
- ・RemoteClient :xAsender Cilant へ表示
- ・Web :Web 画面(パソコン、タブレット、スマートフォンなど)へ表示

接続するデバイスは、単独で設定することも可能ですが、組み合わせでの設定も可能です。

例)「HMI と RemoteClient」「RemotClient と Web」

- ※ 下図の様に複数のページカテゴリを追加し、個別に設定していくことができます。

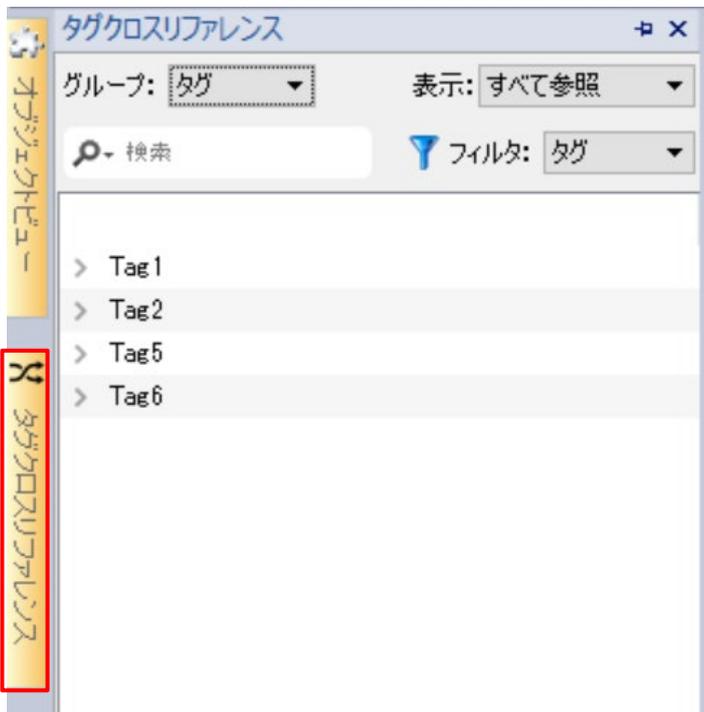


タグクロスリファレンス

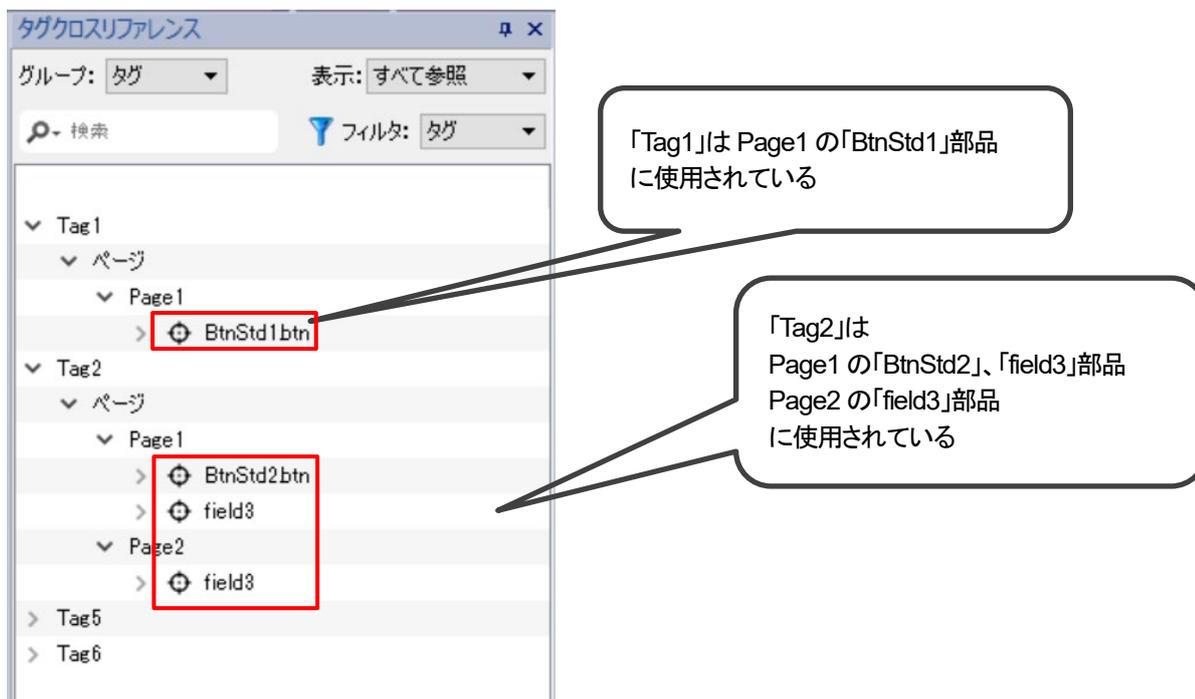
タグクロスリファレンスを使用することで、登録したタグがどのページ、どの部品に使用されているか確認することができます。

xAscender Studio の画面左側の「タグクロスリファレンス」にマウスのカーソルをあてます。

「タグクロスリファレンス」では現在使用されているタグが表示されています。



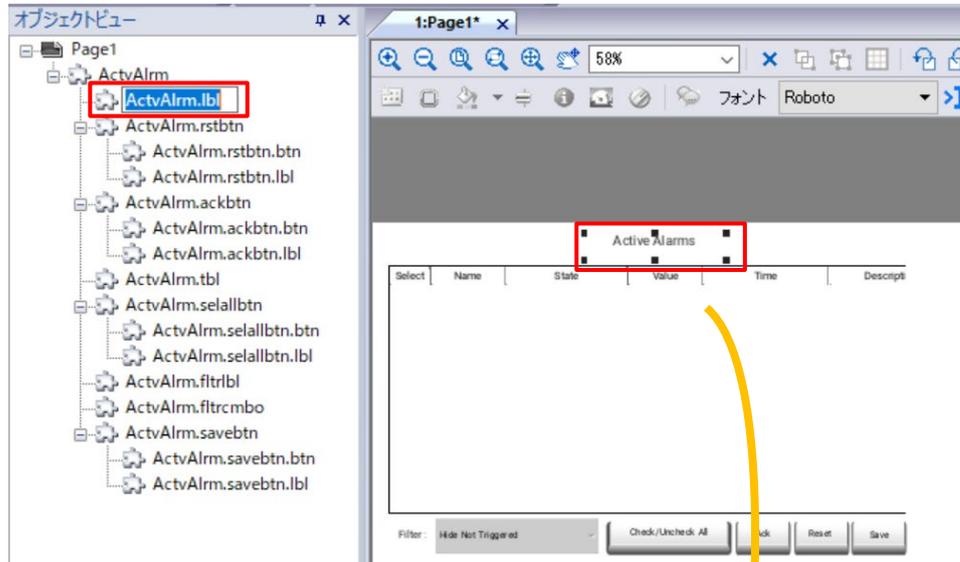
タグ名の左側の「>」を広げていくことで、タグがどこで使用されているか確認することができます



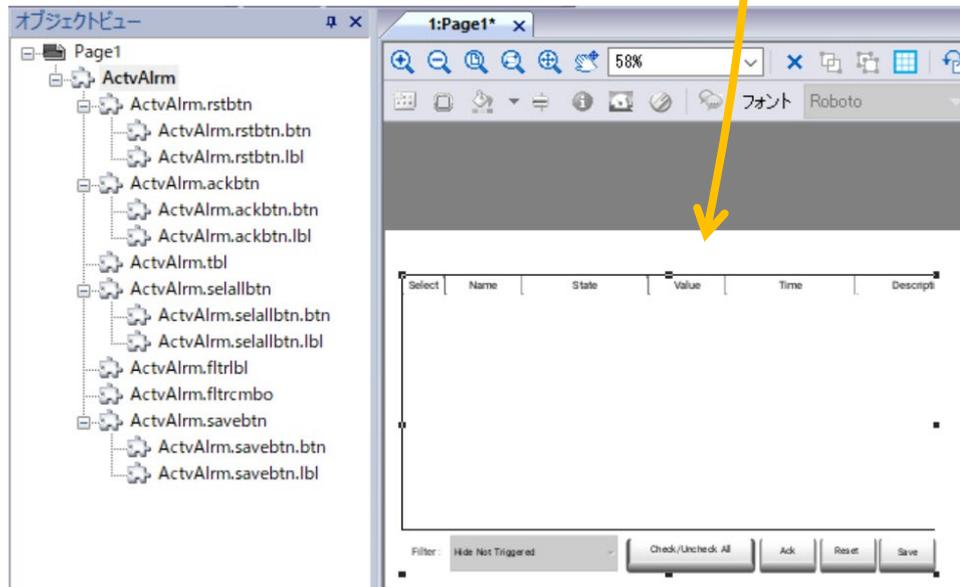
オブジェクトビュー

オブジェクトビューを使うと、複数の部品から成り立っているウィジェットの構成を確認するのに便利です。

下図のように、「Active Alarms」という部品はボタン部品、ラベル部品など様々な部品から構成されています。オブジェクトビューを使用することでその構成内容を確認することができます。
(例:「ActvAlrm.lbl」という部品はタイトルの文字表示に使用されています)



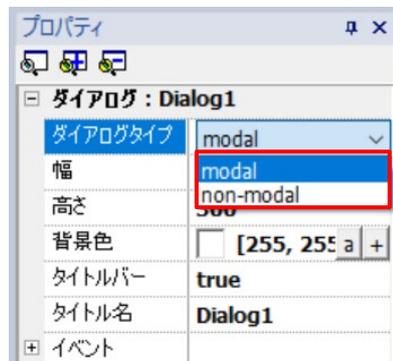
また、上記のように構成されている部品を選択した状態でキーボードの「Del」を押すことで、その部分のみ削除することもできます。



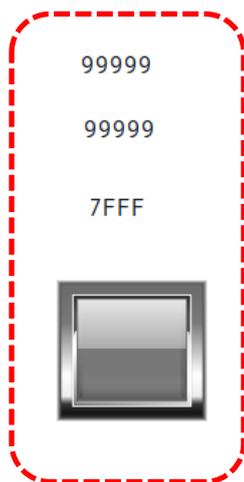
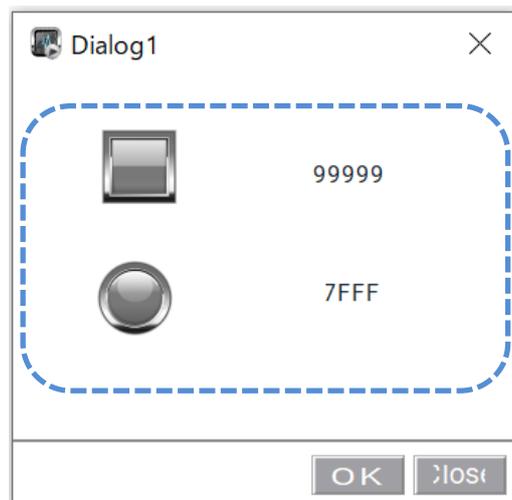
ダイアログタイプの変更

ダイアログのタイプは「modal」、「non-modal」の2種類があります。
これらを変更することでダイアログの動作を切り替えることができます。

ダイアログのプロパティ

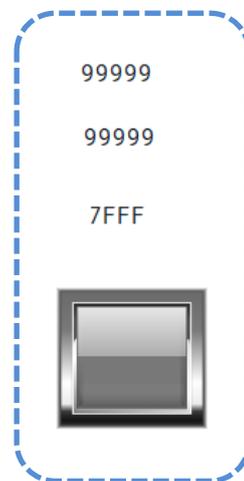
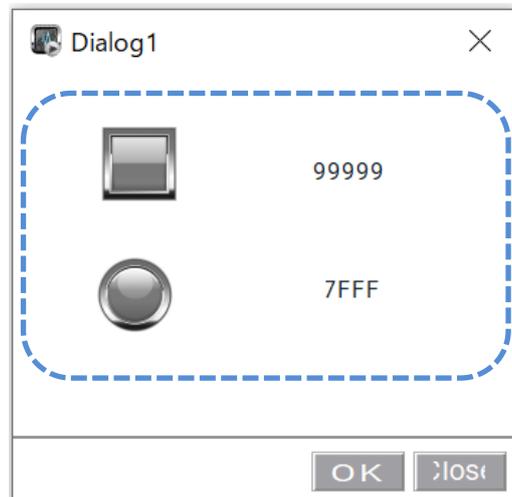


Modal を選択した場合



ダイアログを開いている状態で、
ダイアログページ以外の部品を
触ることができません。

Non-modal を選択した場合



ダイアログを開いている状態でも
ダイアログページ以外の部品を
触ることができます。

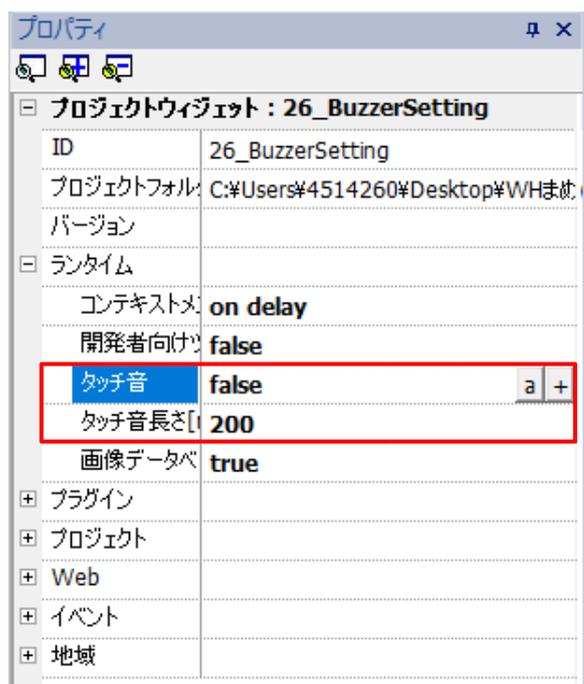
ブザーの設定

画面をタッチした際のブザーに関する設定を説明します。

ブザーに関する設定はプロジェクトプロパティ内での設定と、System Variable での詳細な設定が可能です。

プロジェクトプロパティ内での設定

「プロジェクトビュー」→「プロジェクトプロパティ」→「ランタイム」を開きます。

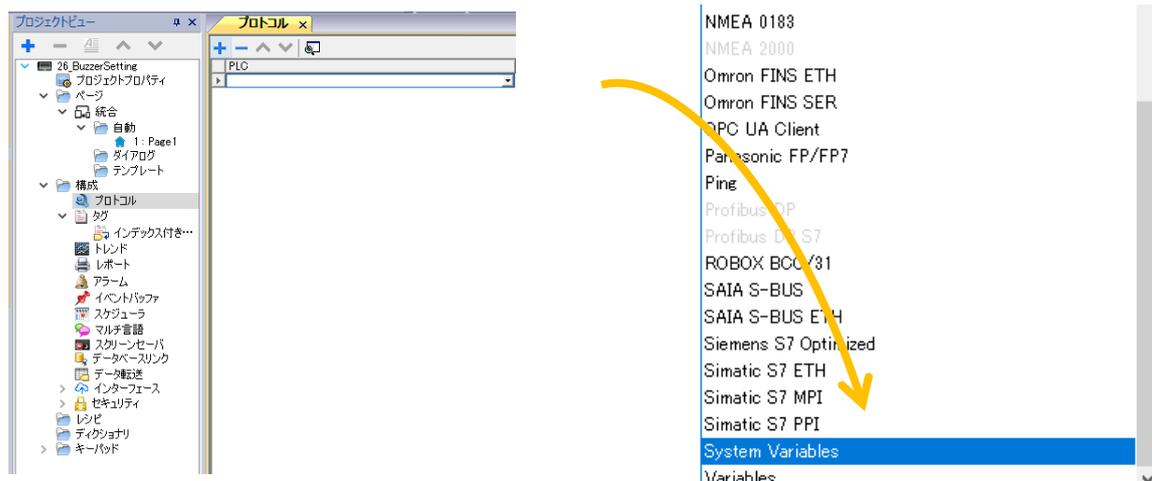


「タッチ」音の TRUE/FALSE を切り替えることで、タッチ音の ON/OFF を切り替えることができます。

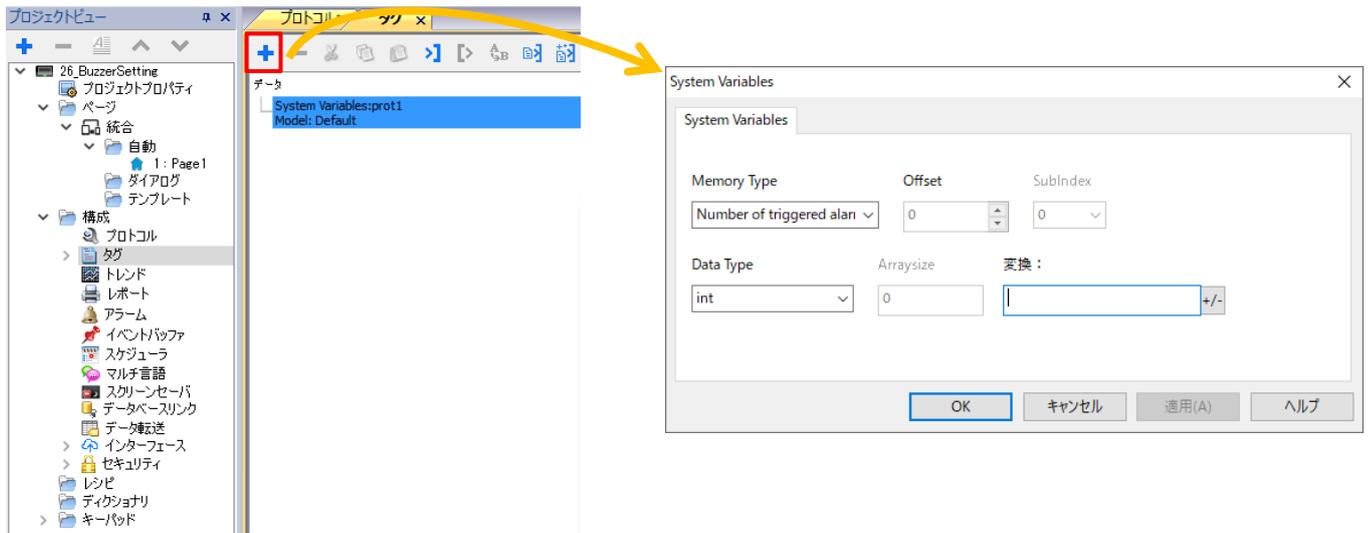
「タッチ音長さ」を調節することでタッチ音が ON した際の長さを変更できます。

xAscender Studio のプロトコル設定

プロトコルで「System Variable」を選択します。



「タグ」を開きタグの追加を選択します。



「Memory Type」を開くと、ブザーに関する変数がありますので、これらの変数の値を変更することで、ブザーの設定を変更することができます。

Buzzer Setup
 0=無効化
 1=有効化
 2=Buzzer Control またはプロジェクトプロパティ内の設定で制御されます。

Buzzer Control
 0=ブザーOFF
 1=ブザーON
 2=ブザー点滅

Buzzer Off Time(ms)
 ブザー点滅時のオフ持続時間(初期値 1000、範囲:100~5000)

Buzzer On Time(ms)
 ブザー点滅時のオン持続時間(初期値 1000、範囲:100~5000)

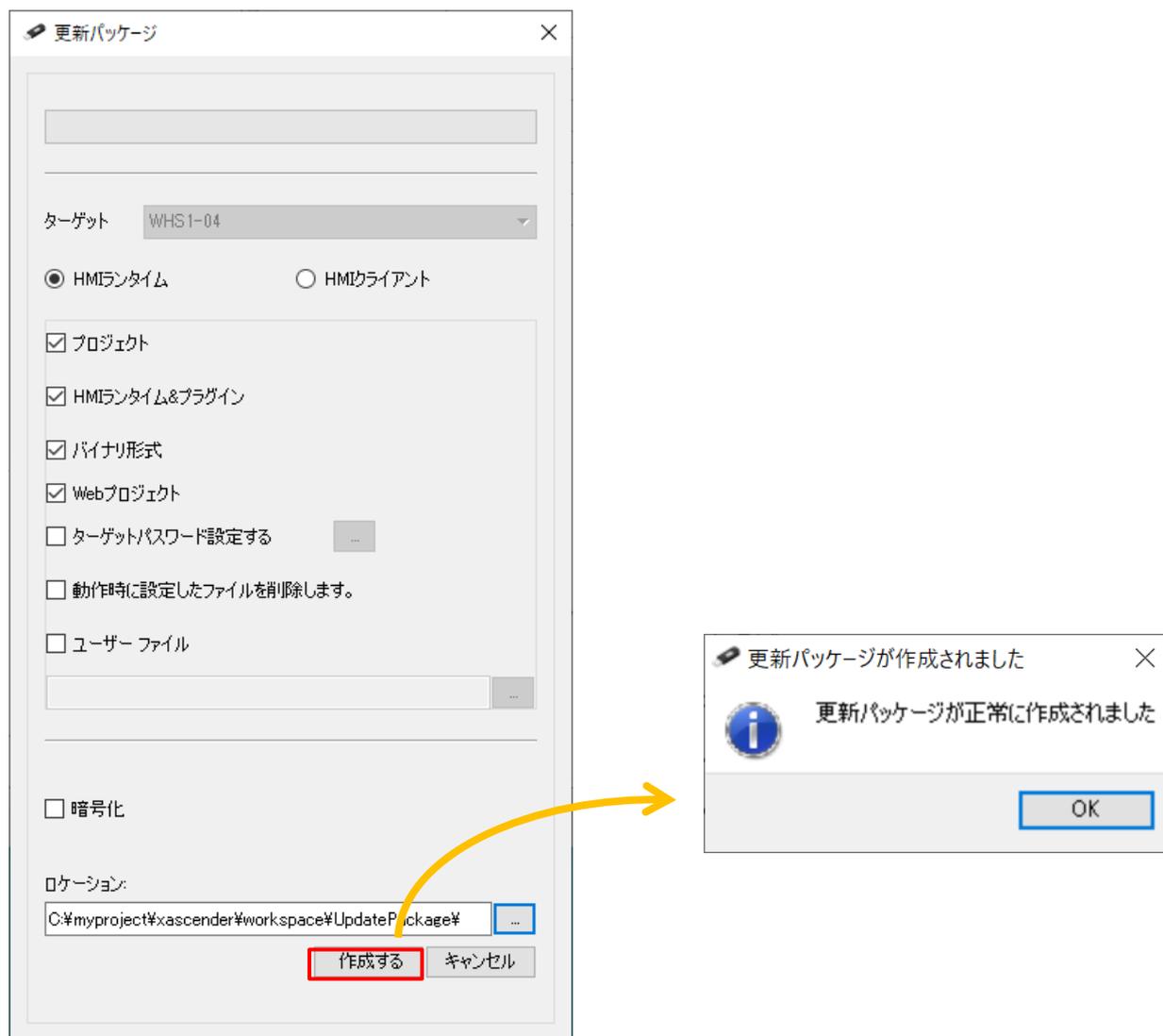
更新パッケージ

更新パッケージは USB メモリを使用して、WH 本体にプロジェクト、ランタイムのインストールを行う、更新ファイルを作成する機能です。

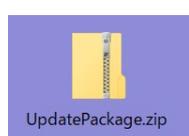
xAscender Studio では更新に使用する「UpdatePackage.zip」ファイルを作成します。

xAscender Studio の設定

「実行」→「更新パッケージ作成」を選択します。



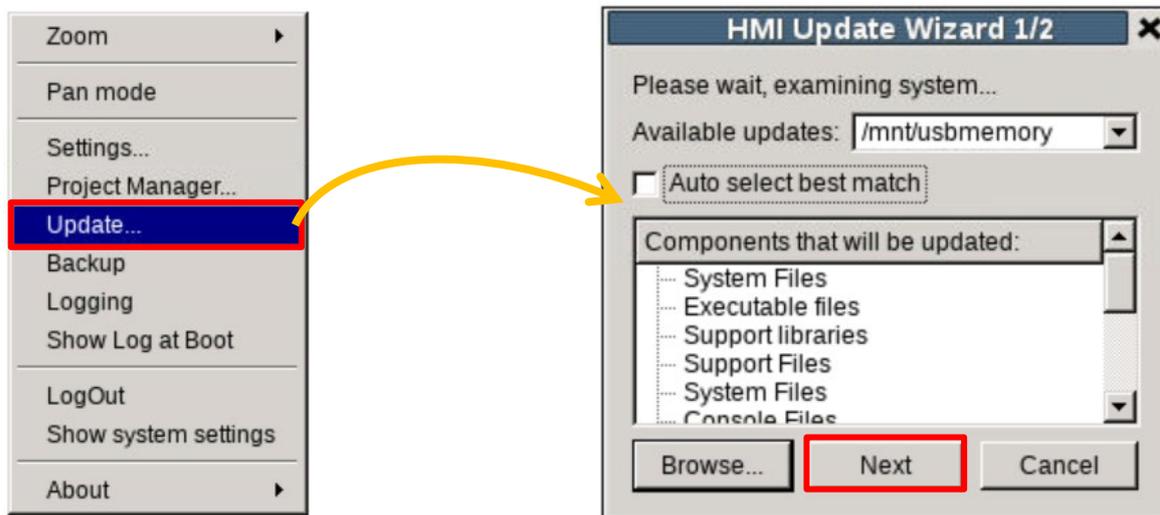
「作成する」をクリックすることで、「ロケーション」で指定したファイルパスに「UpdatePackage.zip」が作成されます。次に作成した「UpdatePackage.zip」を USB メモリに保存します。



WH 本体での操作

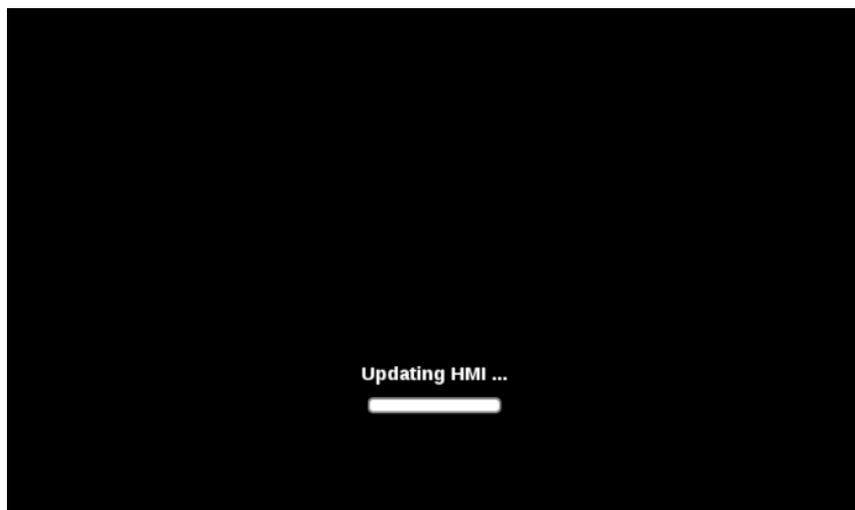
「UpdatePackage.zip」を保存した USB メモリを WH 本体の USB ポートに差し込みます。

その後、本体画面を 2 秒間以上長押しして、コンテキストメニューを表示させ、「Update」を選択します。



「HMI Update Wizard1/2」のダイアログが表示されたら、「Next」を選択します。

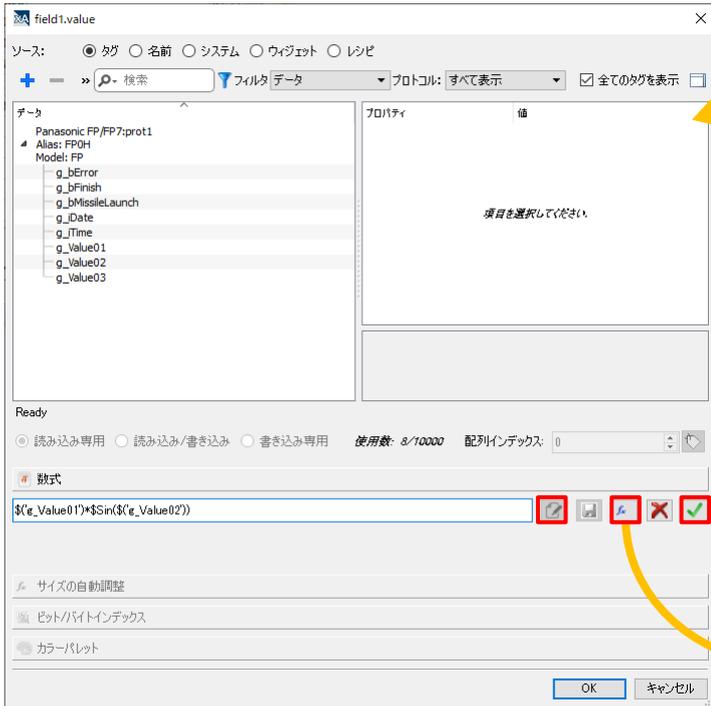
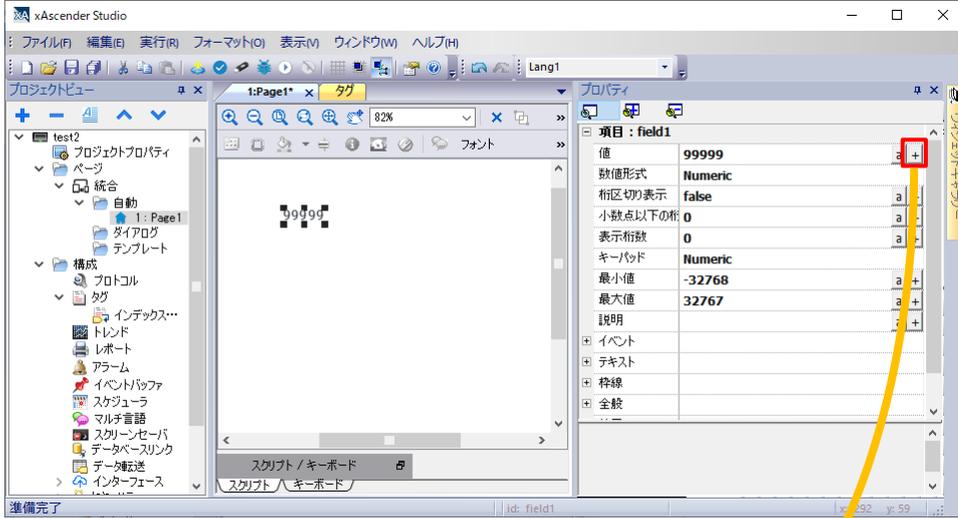
下記画面が表示され、再起動が終わりましたら、プロジェクト、ランタイムのインストール、更新は完了です。



数式の登録

ウィジェットの値などに数式を用いて設定することができます。

ランプ部品もしくはデータ部品を1つ配置し、「プロパティ」-「値」右の  をクリックします。



 ボタンをクリックすると数式が入力できる状態になります。

 ボタンをクリックすると関数入力用のダイアログが表示され、関数が入力できます。

入力が終わりましたら、 をクリックし、「OK」をクリックします。



ライブタグ

ライブタグ機能を使用することで、テキスト内の文字列を PLC で変更できます。

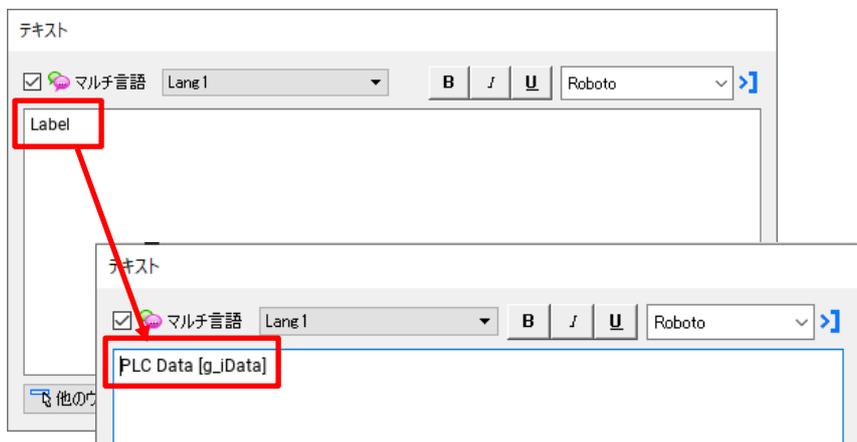
本テキストでは、ラベル部品のテキストの内容を変更する例で説明をします。

ラベル部品を追加して、「プロパティ」の「詳細なプロパティ」-「ライブタグ」を選択し、「ライブタグの有効化」を「true」に変更します。



追加したラベル部品をダブルクリックし、テキスト設定のウィンドウを開きます。

テキスト中に角括弧[]でタグ名を囲む形で入力を行うことで、テキストの表記が PLC とリンクします。



例えば、上図の様にテキスト「 PLC Data [g_iData] 」とテキストに登録した場合、WH 本体には下図の様に表示されます。

PLC Data 1234

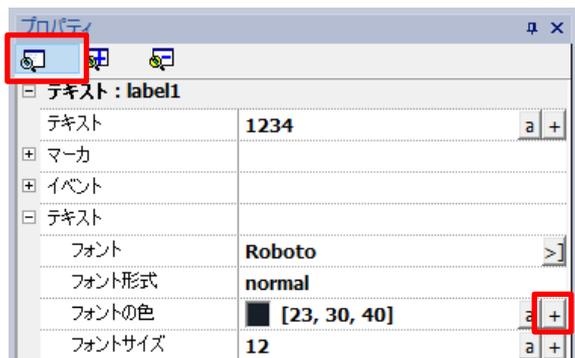
「1234」がタグ「g_iData」に格納された値となり、PLC でタグに割り当てたデータレジスタの値を変更することで表示されたテキストの値を変更することができます。

カラーパレット機能

カラーパレット機能を使用することで、PLC からの指示で部品の色を変更することができます。

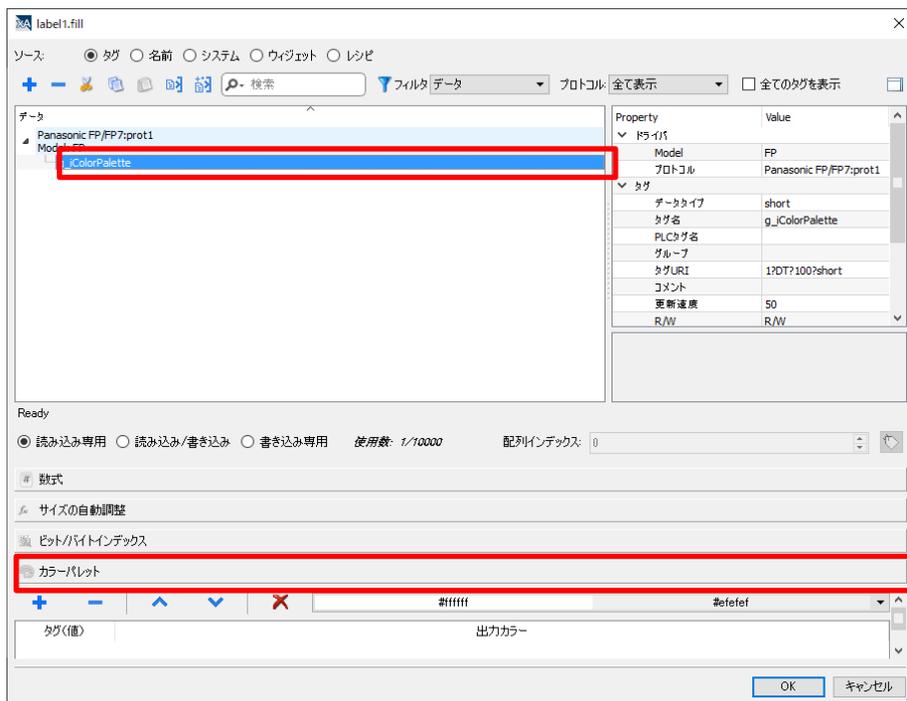
本テキストでは、ラベル部品のテキストの色を変更する例で説明をします。

ラベル部品を追加してテキストを入力し、ラベル部品を選択している状態で「詳細なプロパティ」-「テキスト」-「フォントの色」の **+** を選択してタグ設定画面を開きます。



任意のタグを選択します。ここでは「g_iColorPalette」を使用します。

タグを選択した状態で「カラーパレット」タブを選択すると、カラーパレットの登録部分が表示されます。

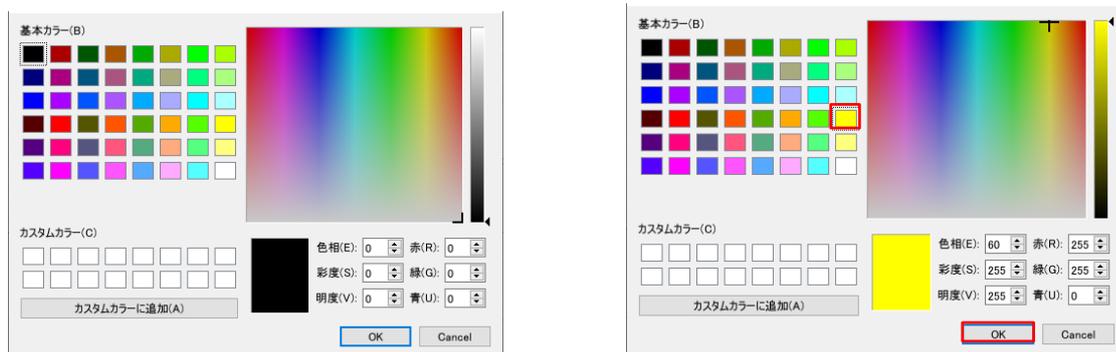


カラーパレットの登録を行います。

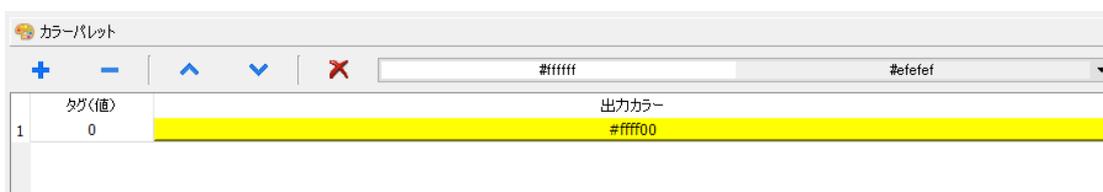
カラーパレットの **+** を選択します。



色の選択ウィンドウが立ち上がります。

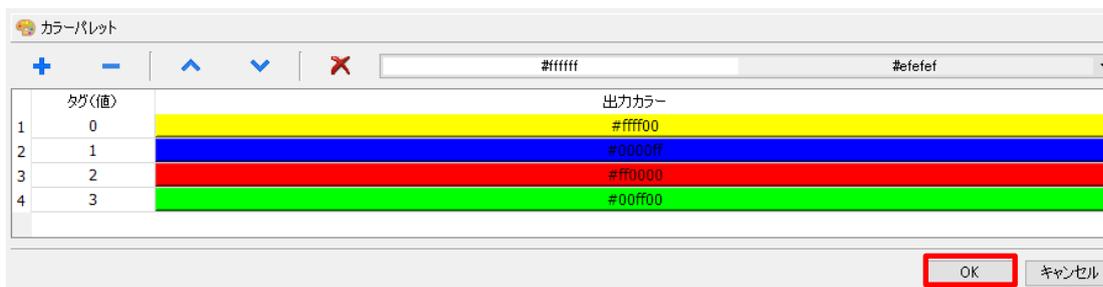


任意の色を選択し、「OK」を選択することでカラーパレットに登録された色が登録されます。



同様の操作を繰り返すことで、カラーパレットに色を追加していくことができます。

カラーパレット登録が完了したら、「OK」を選択することでラベル部品のプロパティ設定が完了します。



PLC から、タグ「g_iColorPalette」の値を変更することで下記のようにテキストの色を変更することができます。

タグ「g_iColorPalette」の値が 0 の場合： 黄色

タグ「g_iColorPalette」の値が 1 の場合： 青色

タグ「g_iColorPalette」の値が 2 の場合： 赤色

タグ「g_iColorPalette」の値が 3 の場合： 緑色

となります。

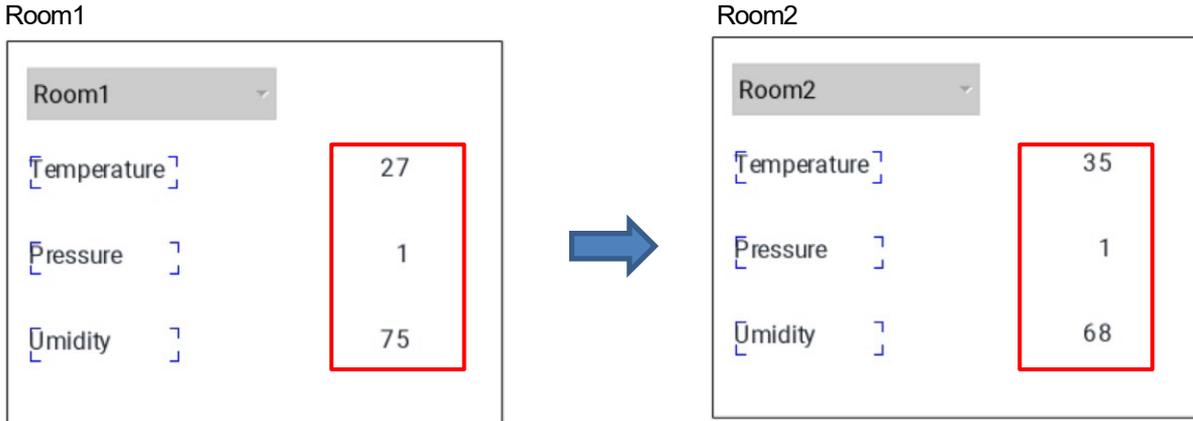
インデックス付きタグセット

インデックス付きタグセットの使用方法について説明します。

通常 1 つのデータ部品(「99999」)の値などには1つのタグしか割り付けることはできませんが、インデックス付きタグセットを使用することでデータ部品に割り付けるタグを変更することができます。

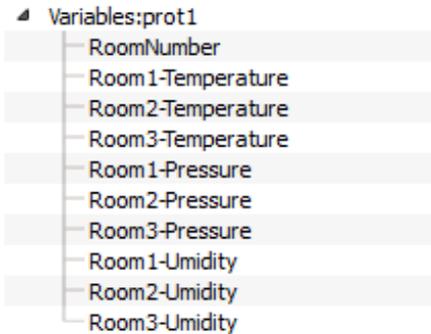
例) Combo Box と組み合わせて使用する場合

Room1→Room2 に変更した際に に割りついているタグが変わり、値が変化します。

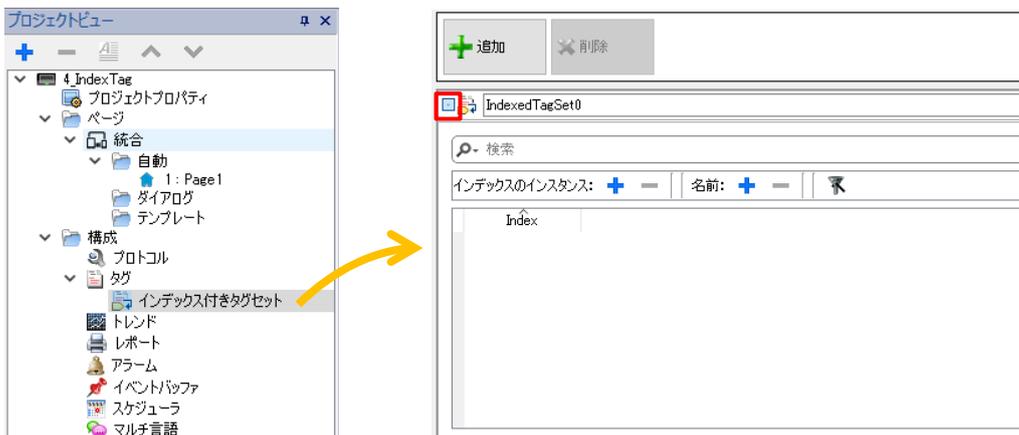


インデックス付きタグセットの設定

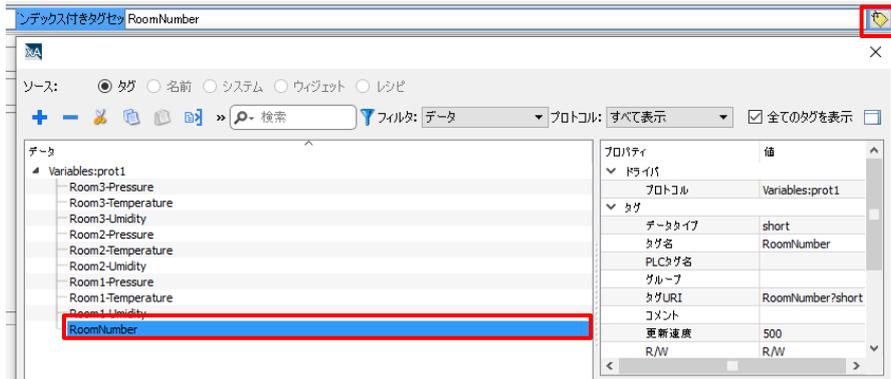
事前に下記のようにタグの登録を行います。



「プロジェクトビュー内」 → 「タグ」 → 「インデックス付きタグセット」をダブルクリック → 「追加」を選択し、 の+をクリックします。

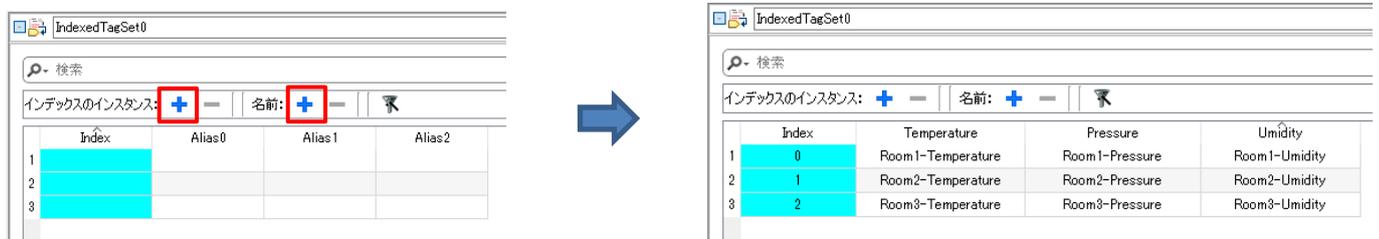


画面右上の「インデックス付きタグセット」に「RoomNumber」を割り付けます。
この「RoomNumber」の値を変更することで、タグの割り付けが変わります。



インデックスのインスタンスの「+」、名前前の「+」をクリックして、下記のように変更します。

マークをクリックしてタグの割り付けを行い、プロジェクトの保存をします。



ここまで完了したら、画面上のデータ部品に割り付けを行います。
インデックス付きタグセットに登録されたものは、ソース:名前
に格納されます。



このように登録を行った場合、データ部品にはそれぞれ下記のようにタグが割りつきます。

Index:0 → Room1-Temperature、Room1-Pressure、Room1-Umidity

Index:1 → Room2-Temperature、Room2-Pressure、Room2-Umidity

Index:2 → Room3-Temperature、Room3-Pressure、Room3-Umidity

バックライトの設定

バックライトの設定方法について説明します。

本体の画面を 2 秒間長押しするとダイアログが表示されます。下から二番目「Show system setting」をクリック、もしくは Web ブラウザで「https://WH 本体の IP アドレス/machine_config」を検索すると System Settings 画面が表示されます。

上記アドレスを検索すると、下図ログイン画面が表示されます。

(デフォルト)

ユーザ名 : admin

パスワード : admin

それぞれ入力し、ログインをクリックします。

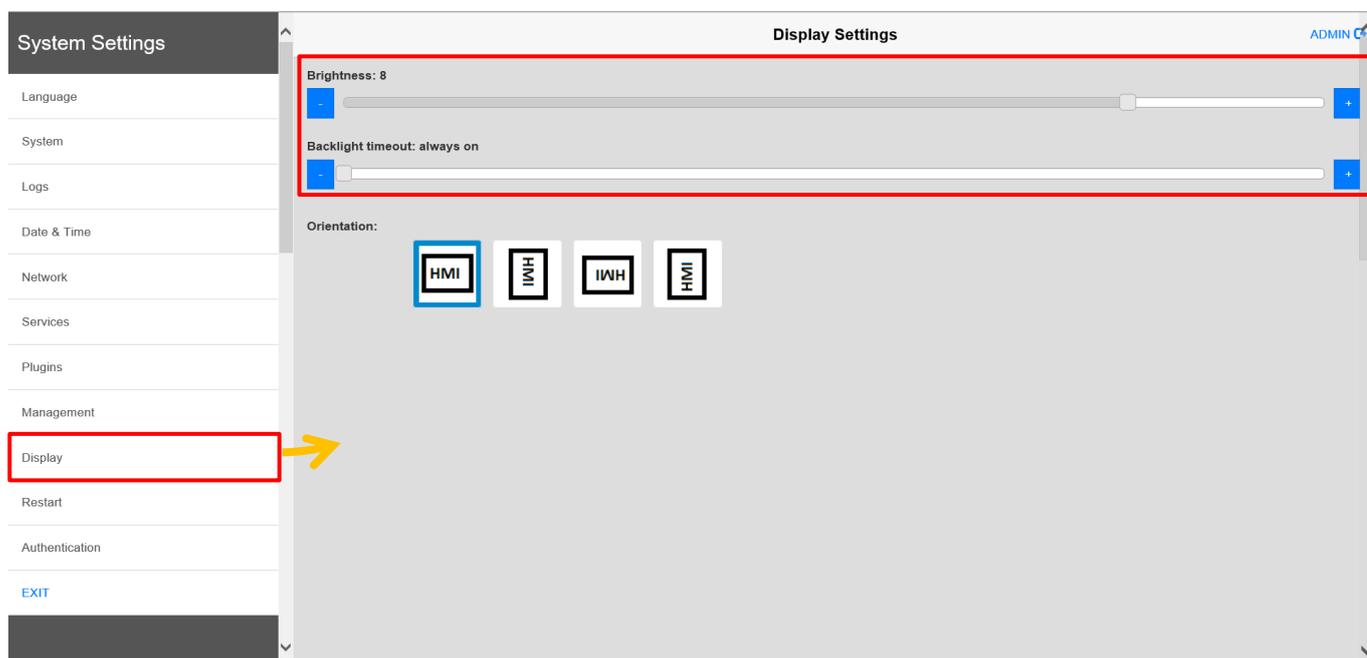


ログイン
https://192.168.1.100

ユーザ名

パスワード

「Display」-「Brightness」、「Backlight timeout」でバックライトに関する設定を変更することができます。



System Settings

Language

System

Logs

Date & Time

Network

Services

Plugins

Management

Display

Restart

Authentication

EXIT

Display Settings ADMIN

Brightness: 8

Backlight timeout: always on

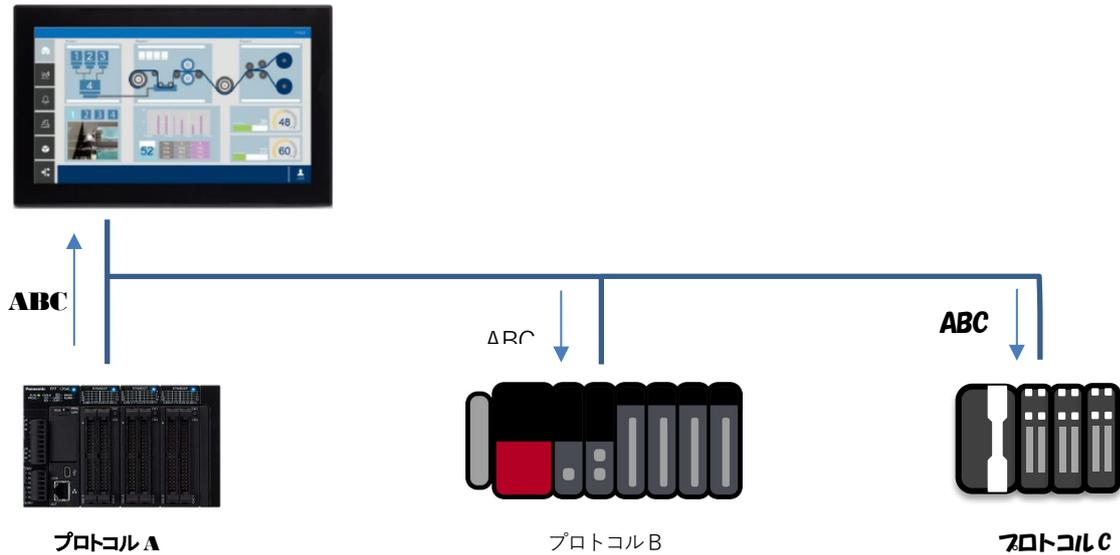
Orientation:

HMI HMI IWH IWH

データ転送

データ転送機能は異なるプロトコルを持つデバイス間で、設定のみでデータを転送することができます。

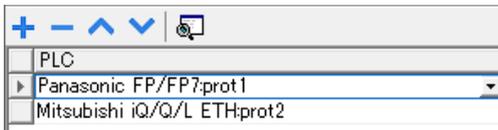
WH はデバイス間でゲートウェイとして動作することができます。



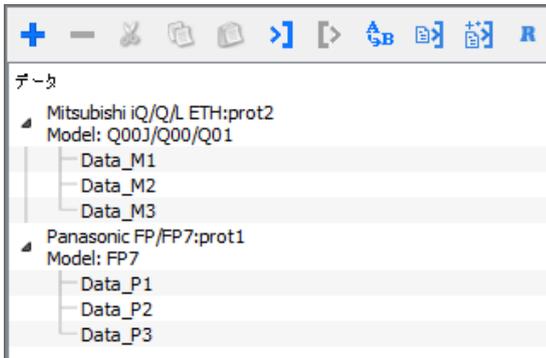
データ転送の設定

例)

プロトコル設定

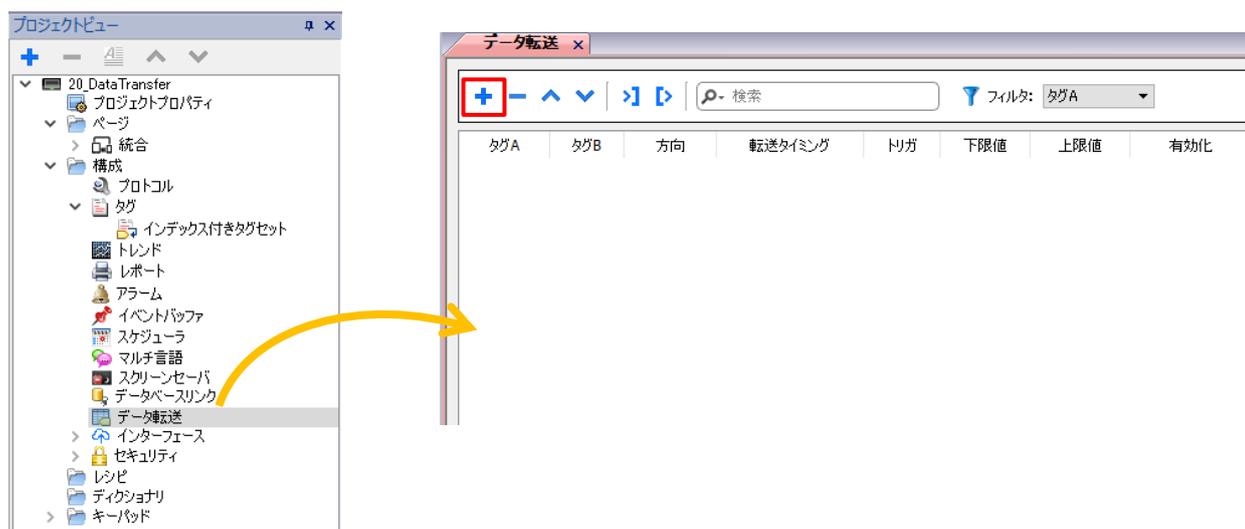


タグ設定

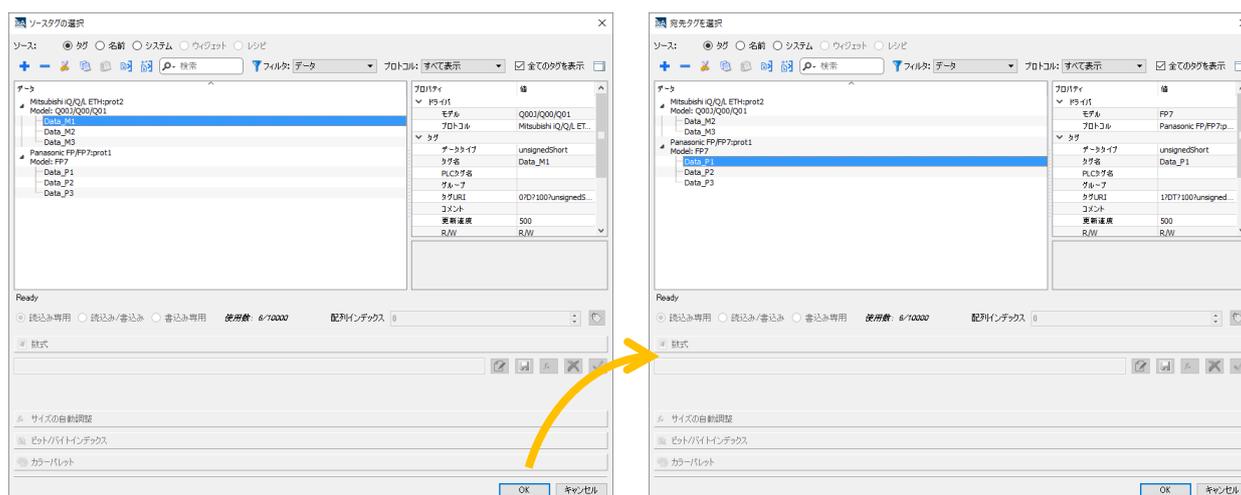


データ転送を使用するには、
上記のように二つのプロトコルが使用されていて、かつタグ登録されている必要があります。

「プロジェクトビュー内」 → 「構成」 → 「データ転送」をダブルクリックし、開きます。



データ転送の編集画面で「+」を押し、
ソースタグ(転送元)を選択し「OK」をクリック、
次に宛先タグ(転送先)を選択し、「OK」をクリックします。



データ転送

検索

フィルタ: タグA

	タグA	タグB	方向	転送タイミング	トリガ	下限値	上限値	有効化
1	Data_M1	Data_P1	A->B	タグ値更新時		0	0	<input checked="" type="checkbox"/>

上記のように登録され、
「Data_M1」の値が、タグ値が更新された際に Data_P1 に転送されます。

トリガ、上限値、下限値

「トリガ」にはタグを登録することができ、そのタグの値の変化が「上限値」、「下限値」を超える場合データ転送が実行されます。

※下限と上限の両方が「0」に設定されている場合、トリガタグの値が変更されるたびにデータ転送が実行されます。

例)

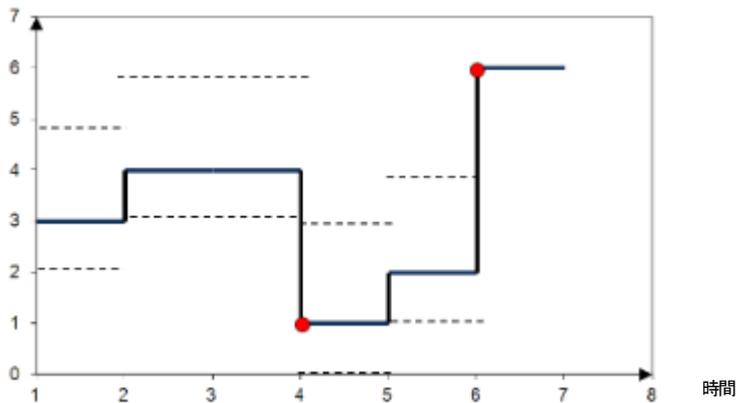
トリガ: Trigger (Unsignedshort)

上限値: 1.9 下限値: -0.9

タグA	タグB	方向	転送タイミング	トリガ	下限値	上限値	有効化
1 Data_M1	Data_P1	A->B	トリガ時	Trigger	-0.9	1.9	<input checked="" type="checkbox"/>

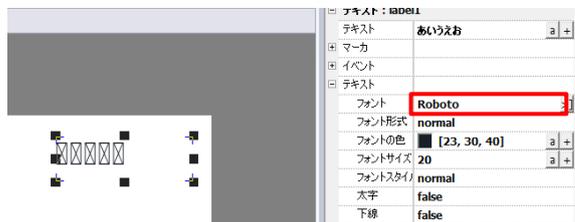
データ転送が実行されるタイミング: ●

Trigger の値



デフォルトフォントの変更

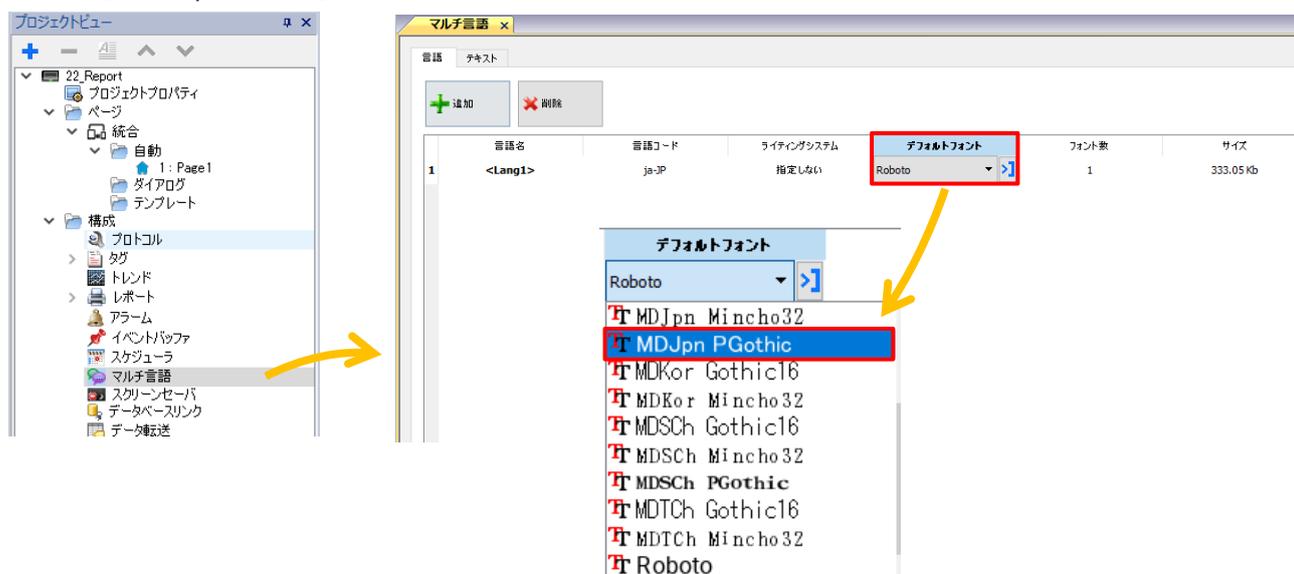
Label 部品などで日本語を表示する場合、デフォルトフォントが「Roboto」になっているため、日本語を入力しても、正しく表示することができません。



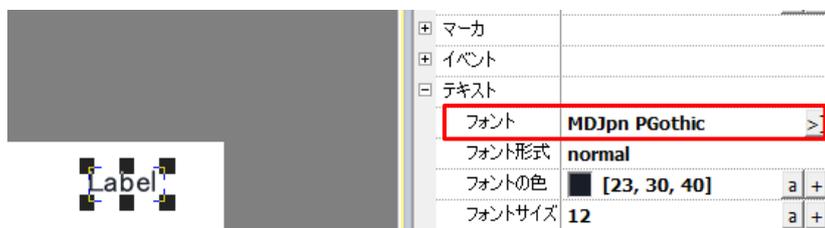
都度フォントを変更すれば日本語表示が可能ですが、プロジェクトのデフォルトフォントを変更することで、スムーズに画面作成を行うことができます。

デフォルトフォントの変更

「プロジェクトビュー内」 → 「マルチ言語」 → 「デフォルトフォント」
「Roboto」を「MDJpn PGothic」などの日本語フォントに変更する



この操作を画面作成前に行うことで、「Label」部品などを使用する際にフォントの変更が不要になります。



トレンド

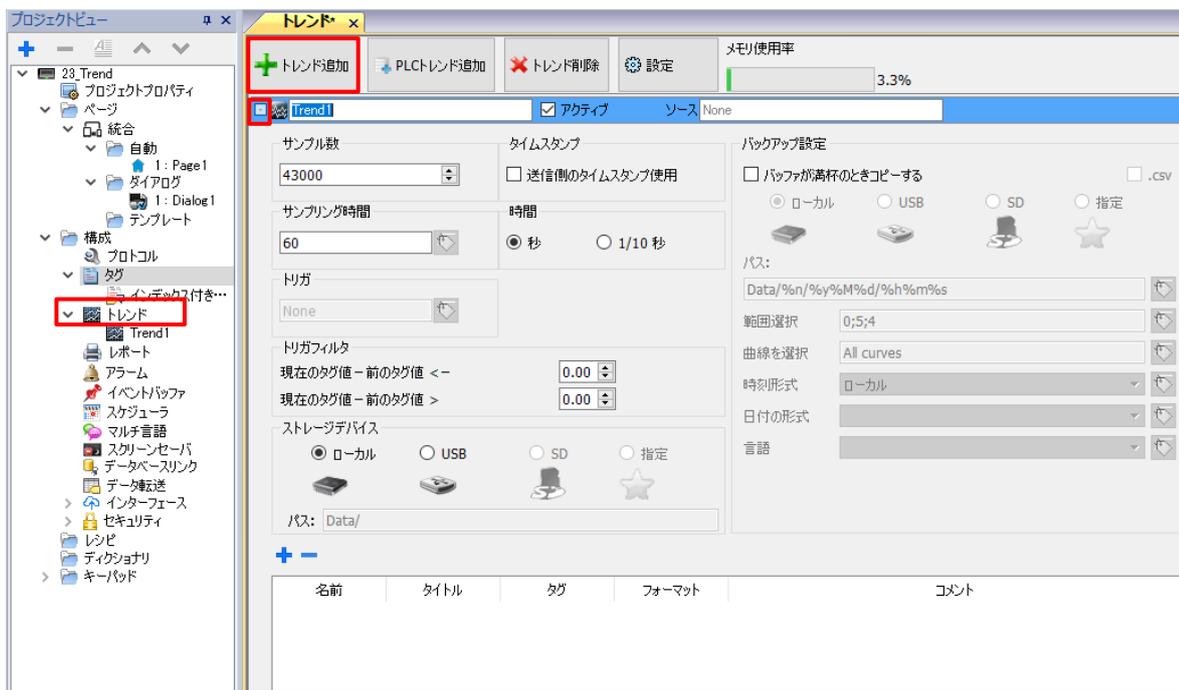
トレンド機能は、特定のサンプリング条件に従い、指定されたタグの値をサンプリングする機能です。

この機能で保存したデータは、トレンドウィジェットやグラフに使用することができます。

トレンドの設定

「プロジェクトビュー」→「構成」→「トレンド」をダブルクリックします。

1. 「トレンド追加」をクリックし、新しいバッファを追加します。
2. 新しく追加されたバッファの左側にある + をクリックし、すべての構成パラメータを表示します。



トレンド名:使用するトレンドの名前(任意)

アクティブ:☑を入れることで、システム起動時に実行が開始されます

ソース:トレンド機能でサンプリングされるタグの一覧

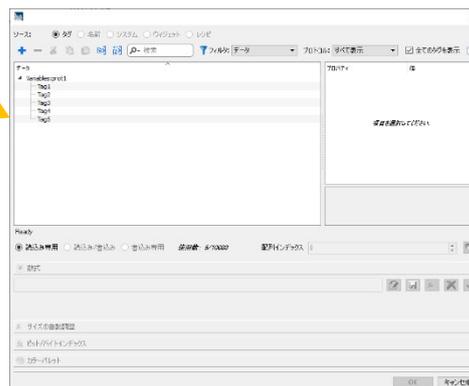
サンプル数	タイムスタンプ
43000	<input type="checkbox"/> 送信側のタイムスタンプ使用
サンプリング時間	時間
60	<input checked="" type="radio"/> 秒 <input type="radio"/> 1/10 秒

サンプル数:最大サンプリング数(実際に使用できるサンプル数はメモリサイズと各サンプルのサイズによって異なります)

タイムスタンプ:☑を入れると、外部機器から提供されたタイムスタンプを使用します(プロトコル:OPC UA クライアント、BACnet)

サンプリング時間:サンプリングの間隔(時間間隔でサンプリングを行う場合はこの設定を行います)

時間:サンプリング時間の単位



トリガ:登録したタグの値の変化でサンプリングする際に使用します(サンプリング時間は無効になります)

トリガフィルタ:

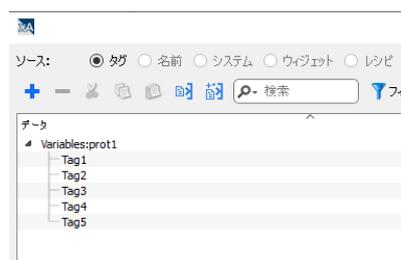
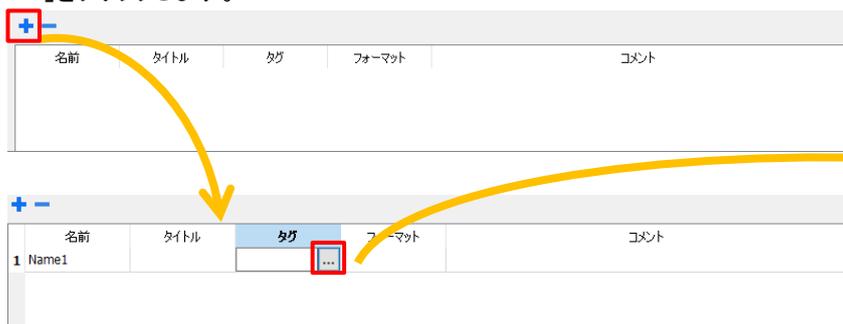
トリガタグ値の変更が指定された制限を超える場合、新しいサンプルが取得されて保存されます。それ以外の場合、サンプリングは行われません

ストレージデバイス:トレンドバッファの保存先を指定します(アドバンスモデルは SD 選択可)

タグの登録

サンプリングしたいタグの登録方法

「+」をクリックします。



「Name1」が追加されます、「タグ」の「…」をクリックし、サンプリングしたいタグを登録します。

※トレンド機能では、タグ文字列は8バイトまでサポートされています。

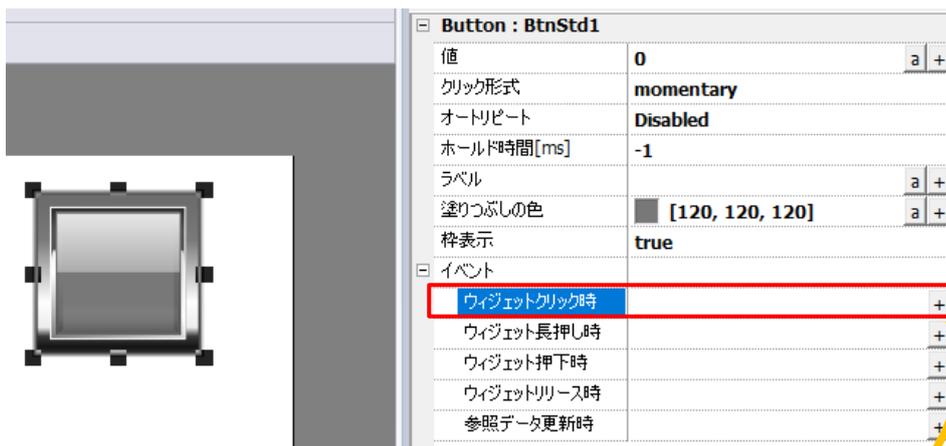
タグサイズが8バイトより大きい場合、最初の8バイトのみがトレンドに保存されます。Unicode 文字はサポートされていません。

トレンドの出力(csv)

サンプリングしたトレンドデータを CSV 形式で出力する方法を説明します。

ウィジェット、アラーム、スケジューラなどのアクションで設定することができます。
(ここではボタン部品のイベント→アクションを用いて説明します)

ボタン部品を一つ作成し、
「イベント」→「ウィジェットクリック時」の右側の「+」をクリックします。



アクションリストが開き、システム→トレンドダンプをクリックします。



トレンド名

トレンド名
保存先 Trend1

CSV 形式で出力したいトレンドを
選択します

保存先

保存先
ファイル形式
タイムスタンプ #Flash#QTHM#workspace#Dump
時刻形式 #USBMemory
ファイル名 #Storage Card

WH 本体、USB メモリ、
SD カード(アドバンスタイプのみ)

上記の設定を WH 本体にダウンロードします。
WH 本体でボタンを押すと、指定したトレンドデータが保存先に CSV 形式で出力されます。

トレンドデータのグラフ化

サンプリングしたトレンドデータをグラフ上に表示する方法を説明します。

画面上に「History Trend」部品を一つ追加します。

ここでの操作は「トレンド」機能の設定がされている前提となります。

プロパティ→グラフ線1→トレンドグラフ線1の「+」をクリックし「トレンドの選択」を選択します。

The screenshot shows the 'History Trend' widget on the left and its configuration panel on the right. The widget displays a line graph with a blue curve. The configuration panel shows the following settings:

- Trend : HistoricTrend
- グラフ線の数: 1
- ページ持続時間: 5 min
- 終了時間: 0
- ページ移動量Y: 100
- 開始地点: Right
- 目盛り
- テキスト
- グリッド
- カーソル
- X軸
- Y軸
- グラフ線1
 - トレンドグラフ線1
 - Y軸の最小値: 0
 - データリンク: y0:HistoricTr (アクセス種別: R)
 - Y軸の最大値: 100
 - データリンク: y1:HistoricTrend.wnd (アクセス種別: R)
 - 色: [0, 0, 255]
 - 枠線の幅: 2

グラフ上に表示させたいトレンドのタグをダブルクリックします。

The screenshot shows the 'HistoricTrend.buffer1.Value' configuration window. The 'ソース' (Source) is set to 'タグ' (Tag). The '名前' (Name) list shows 'HistoricTrend.trend' expanded to 'Trend1', with 'Name1' selected. A callout box titled 'トレンドの登録内容' (Trend Registration Content) shows a table with columns '名前' (Name), 'タイトル' (Title), and 'タグ' (Tag). The table contains two entries:

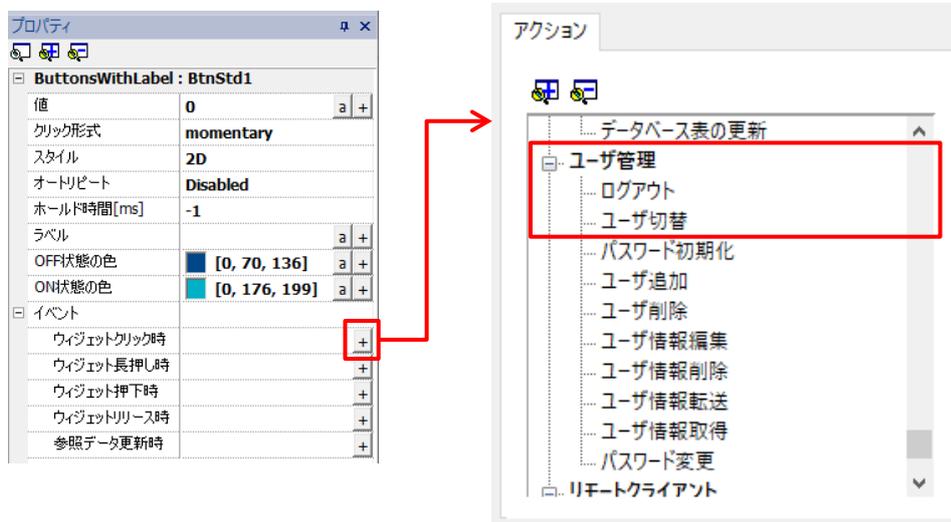
	名前	タイトル	タグ
1	Name1		Tag1
2	Name2		Tag2

上記の設定を行うことで、Trend1 に登録されている Name1(Tag1)のトレンドデータをグラフ上に表示させることができます。

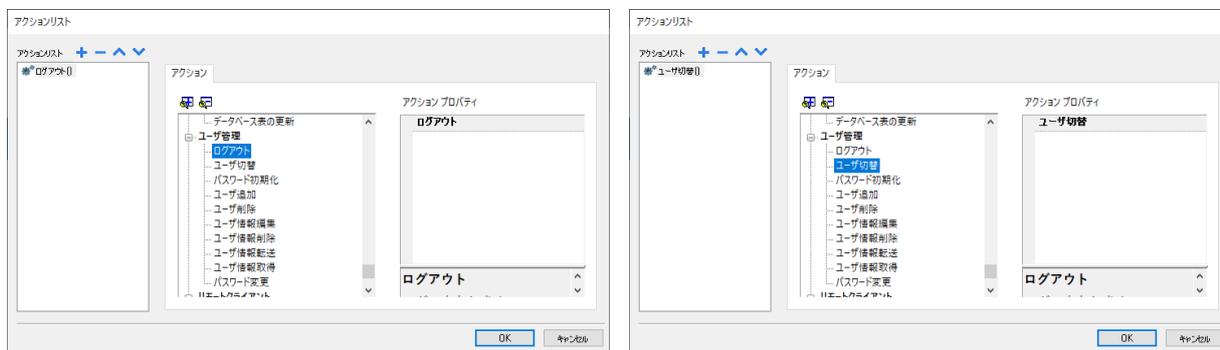
ユーザの切り替え

登録済みのユーザ切り替えは、ボタンなどのウィジェットにアクションを設定して実行することで切り替えることが可能です。

ページ内に配置したボタン等のウィジェットの「プロパティ」-「イベント」からアクションの実行条件を選択します。ここでは「ウィジェットクリック時」で設定します。「ウィジェットクリック時」欄にある **+** をクリックしてください。その後、「アクションリスト」が開きますので、「ユーザ管理」ツリー内の「ログアウト」もしくは「ユーザ切替」を選択します。



「ログアウト」「ユーザ切替」のアクションプロパティは下図の様になります。いずれも設定はなく、「OK」のクリックでアクションの設定は完了します。



※「ログアウト」と「ユーザ切替」の違い

「ログアウト」実行時は、ログアウト処理を行うため、パスワード等の入力がなければログインができませんが、「ユーザ切替」実行時は、ログアウト処理が行われていないため、「Back」から元のユーザ画面に戻ることができます。

「ログアウト」実行時

User name:	<input type="text"/>
Password:	<input type="password"/>
	<input type="checkbox"/> Show password
	<input type="button" value="Sign In"/>

「ユーザ切替」実行時

User name:	<input type="text"/>
Password:	<input type="password"/>
	<input type="checkbox"/> Show password
	<input type="button" value="Back"/> <input type="button" value="Sign In"/>

ユーザ権限の変更

「ユーザグループ」ごとに「権限」として、さまざまなウィジェットや操作へのアクセス制限を設定することができます。「権限」は「ユーザ」ごとでの設定はできませんので注意してください。

権限の設定を行うには下記の手順を実施します。

「プロジェクトツリー」から「構成」-「セキュリティ」の順でツリーを展開して「ユーザグループ」をダブルクリックし、「ユーザグループ」タブを開きます。

名前	認証	ホームページ	最後に表示したページを指す	コメント	権限設定
admin	true		<input checked="" type="checkbox"/>	administrator group	adminAuth guestAuth unauthorizedAuth
guest	true		<input checked="" type="checkbox"/>		
unauthorized	false		<input checked="" type="checkbox"/>		

権限を変更したいユーザグループの「権限設定欄」をクリックし、表示される をクリックします。

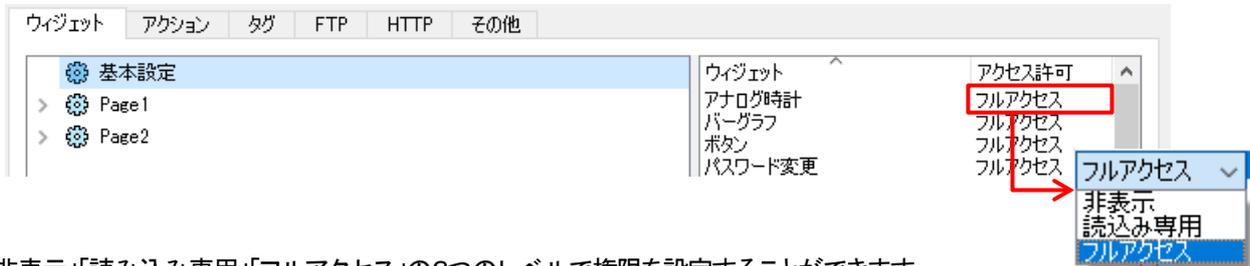
設定ダイアログが表示されます。

「ウィジェット」「アクション」「タグ」「FTP」「Http」「その他」の各レベルで権限設定を行うことができます。

特に、「ウィジェット」や「アクション」は、プロジェクト内に存在しているページや部品に対して行う権限設定となります。

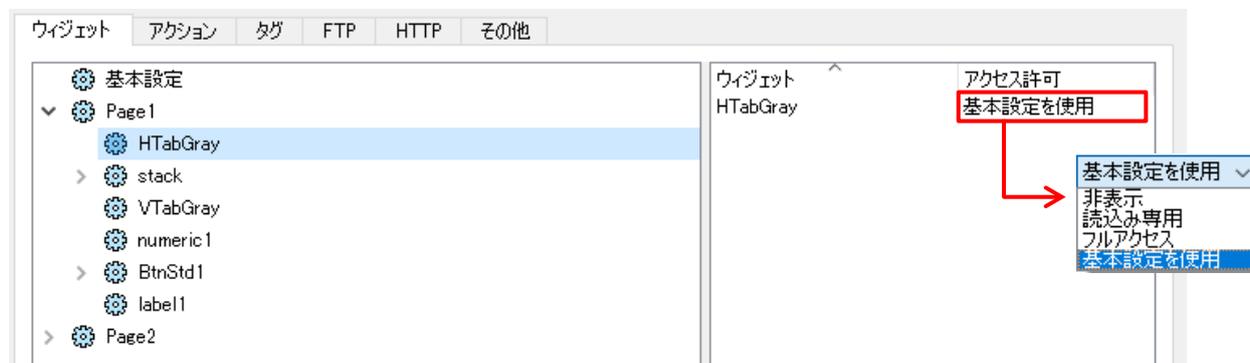
ウィジェット	アクセス許可
基本設定	フルアクセス
Page1	フルアクセス
Page2	フルアクセス
IPカメラ	フルアクセス
RSS	フルアクセス
RSSフィード	フルアクセス
RSSフィードグループ	フルアクセス
XMLファイル	フルアクセス
XY曲線	フルアクセス
アナログ時計	フルアクセス
イベントバッファ	フルアクセス
イベントバッファのグループ	フルアクセス
インジケータ	フルアクセス
ウェブブラウザ	フルアクセス
エディットボックス	フルアクセス
キーボードラベル	フルアクセス
キーボード形状	フルアクセス
キーボード編集	フルアクセス
キーボタン	フルアクセス
グリッド	フルアクセス
グループ	フルアクセス
コントロールリスト	フルアクセス
コンボボックス	フルアクセス
スケジュール	フルアクセス
ダイアログ	フルアクセス
トレンドウィンドウ	フルアクセス
トレンドバッファ	フルアクセス
トレンド曲線	フルアクセス
ハイパーリンク	フルアクセス
バーグラフ	フルアクセス
パスワード変更	フルアクセス

「ウィジェット」ではウィジェットへの書き込みや読み出し(表示/非表示)を設定できます。
「基本設定」はプロジェクト全体に対して反映させることができる設定です。
「admin」ではすべてのウィジェットに対して「フルアクセス」が設定されています。



「非表示」「読み込み専用」「フルアクセス」の3つのレベルで権限を設定することができます。

各ページに配置されているウィジェットごとで設定を行うこともできます。
各ページ名のツリーを展開すると、各ページ内で使用されているウィジェットが確認できます。
ウィジェット名で管理されているので、ウィジェット一つ一つに対して権限の設定を行うことができます。
デフォルトは「基本設定を使用」となっており、「基本設定」の内容が反映されています。



「アクション」に対する設定も、基本的な考え方は「ウィジェット」と同様です。



「アクション」に関しては、アクションを「許可」「許可しない」の2つのレベルでの設定になります。

ユーザ登録

新しくユーザを登録するには下記の手順を実施します。

「プロジェクトツリー」から「構成」-「セキュリティ」の順でツリーを展開して「ユーザ」をダブルクリックし、「ユーザ」タブを開きます。

・デフォルトユーザ「admin」が登録されていますので、ユーザ追加のために **+** をクリックします。

・「名前」(ユーザ名)は「User1」が自動で追加されます。
「グループ」や「パスワード」の内容は「admin」のものが反映されますので、必要に応じて変更を行ってください。「名前」も同様に任意に変更が可能です。

+ を再度クリックしていくことで、ユーザをさらに追加していくことができます
「User1」を追加したときと同様に、必要に応じて各設定の変更を行ってください。

※デフォルトユーザは、ユーザ名やパスワードを入力しなくても、電源立ち上げ後自動ログインします。
デフォルトユーザは、1ユーザのみ登録が可能です。

レシピによる複数個の内部変数タグを初期化する方法 1 (ホームページ表示時初期化)

複数の内部変数を初期化したい場合に、**レシピに初期化データを登録してHMIに保存しておく**
ページイベントのアクティブアクションでレシピを変数にダウンロードすることにより初期化可能です。

ページイベントのアクティブアクションは、ページを表示した場合に起動されます。

プロジェクト起動時だけでなく、ホームページでデータ変更後、別の画面を表示・操作してホームページに戻ってきた場合にも実行されて初期値に戻ってしまいますので、注意が必要です。

注)内部変数タグのみでなく、**PLC との通信タグに対しても有効**です。

レシピを使用したくない場合は、JavaScript による方法を参照してください。

(具体例)ここでは、**内部変数タグ Tag1, Tag2, Tag3 に対する初期値を設定**します。

1. 下図のように**レシピ機能に recipe0 を登録して、内部変数タグ Tag1, tag2, tag3 を割り当て初期値を設定**します。

設定完了後、ローカルデバイス(プロジェクト)に保存しておきます。(右図)

2. **ページイベントのアクティブアクションにタグ→タグ(書き込み)**を設定します。

3. 起動時の内部変数タグの内容を表示確認するための画面を作成しておきます。(左下図)
各データ部品には、対応する内部変数タグをリンクしておきます。(右下図)

Home page	
Tag1の内容	99999
Tag2の内容	99999
Tag3の内容	99999

項目 : numeric1	
値	99999
データリンク	Tag1
アクセス種類	R
数値形式	Numeric

4. WHに転送して動作を確認します。下図のように1で設定したTag1, Tag2, tag3の初期値が表示されました。

Home page	
Tag1の内容	111
Tag2の内容	1111
Tag3の内容	11111

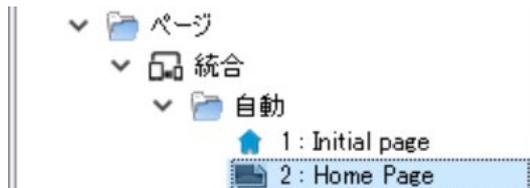
レシピによる複数個の内部変数タグを初期化する方法 2（初期化画面で初期化）

前頁のレシピによる複数個の内部変数タグ初期化方法 1 では、ホームページの表示アクション時にデータを初期化していたのでホームページ内で、データ内容を変更後に別の画面に遷移し、再度ホームページに戻ってきた場合にも初期化が実行されるので変更したデータ内容が保持されませんでした。

ここでは、そのような場合にも変更されたデータが保持される方法を説明いたします。

つまり**初期化専用のページ**を用意して、初期化完了後にホームページを自動表示する方法です。

下図のようにページを構成してください。



Initial page の内容は、前ページと同じです。ここでは、自動でホームページに移行するのでこの画面はほとんど目視できません。

初期化 page	
Tag1の内容	99999
Tag2の内容	99999
Tag3の内容	99999

項目 : numeric1	
値	99999
データリンク	Tag1
アクセス種類	R
数値形式	Numeric

通常ページ(ホームページ)も初期値設定を確認するための同様の画面にします。

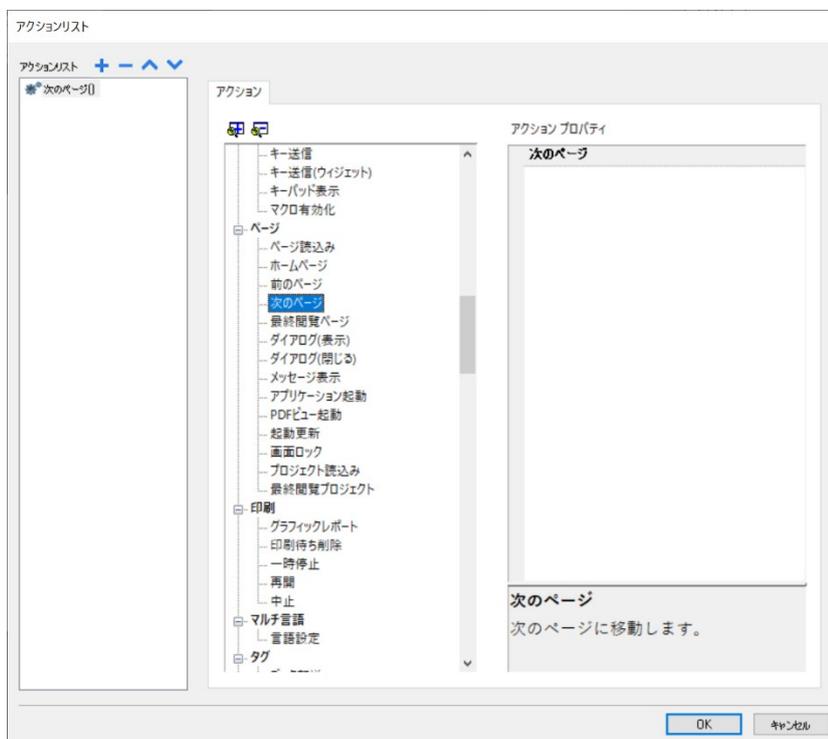
通常ページ	
Tag1の内容	99999
Tag2の内容	99999
Tag3の内容	99999

最初に Initial page が表示され、レシピ機能による Tag の初期化設定が完了後に通常ページ(ホームページ)を表示させるために Initial page に追加でデータ部品を配置し、**値に Tag3 をリンク**します。

項目 : numeric4	
値	99999
データリンク	Tag3
アクセス種類	R

さらにイベント→参照データ更新時にアクション→ページ→次のページを設定します。

イベント	
参照データ更新時	1アクション
アクション[0]	NextPage()



プロジェクトをWHにダウンロードして、動作を確認します。
 下図のように初期化ページで内部変数タグの初期化完了後、通常ページが表示されます。

Home page	
Tag1の内容	111
Tag2の内容	1111
Tag3の内容	11111

WH の変数種類概要説明

WH で変数(内部メモリ)を定義するには、以下の 2 種類の方法があります。

変数とは名前とデータ型を指定して、WH 内のウィジェットやアクション、JavaScript などから読み出し書き込みできるものです。

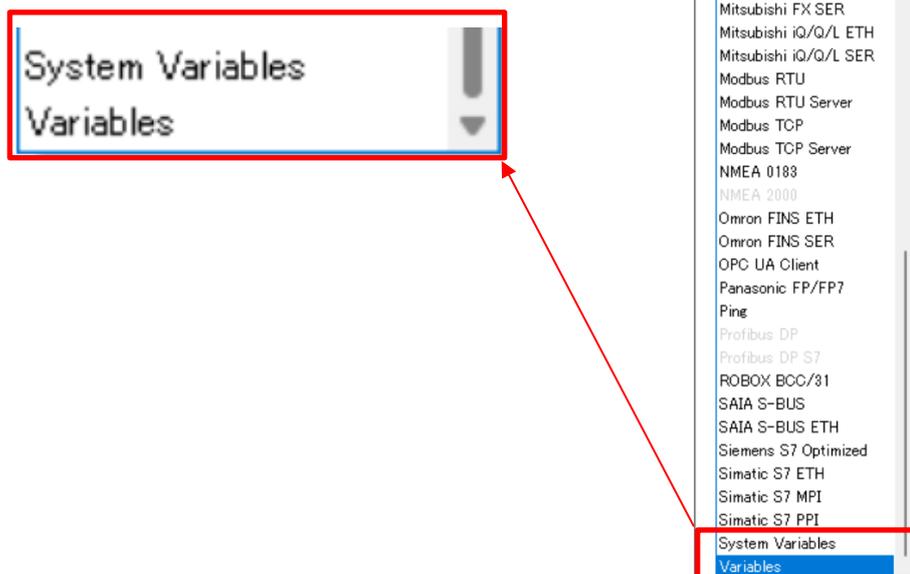
- ① 変数タグ
- ② 変数ウィジェット

上記①の**変数タグ**には、以下の 2 種類があります。

1. 変数タグ (Variables プロトコル内に定義)
2. システム変数タグ (System Variables プロトコル内に定義)

通常のタグは、通信プロトコル内に名前・データ型・PLC 内のアドレス等を指定して定義し、通信対象 PLC とのデータ読み出し・書き込みに使用しますが、変数タグは内部プロトコル Variables または System Variables 内に定義し外部との通信には使用されません。

Variables と System Variables の設定(下図)
プロトコル選択部の一番下に表示されます。



上記②の**変数ウィジェット**で定義できる変数には、以下の2種類があります。

1. **ローカル変数:** 変数ウィジェット(ページ内配置)による定義
2. **グローバル変数:** プロジェクトの変数ウィジェットによる定義

ローカル変数は、変数ウィジェットを使用して定義します。

具体的には、ページのウィジェットギャラリー→[その他]タブ→[データソース]→[変数]をページに貼り付けて使用します。

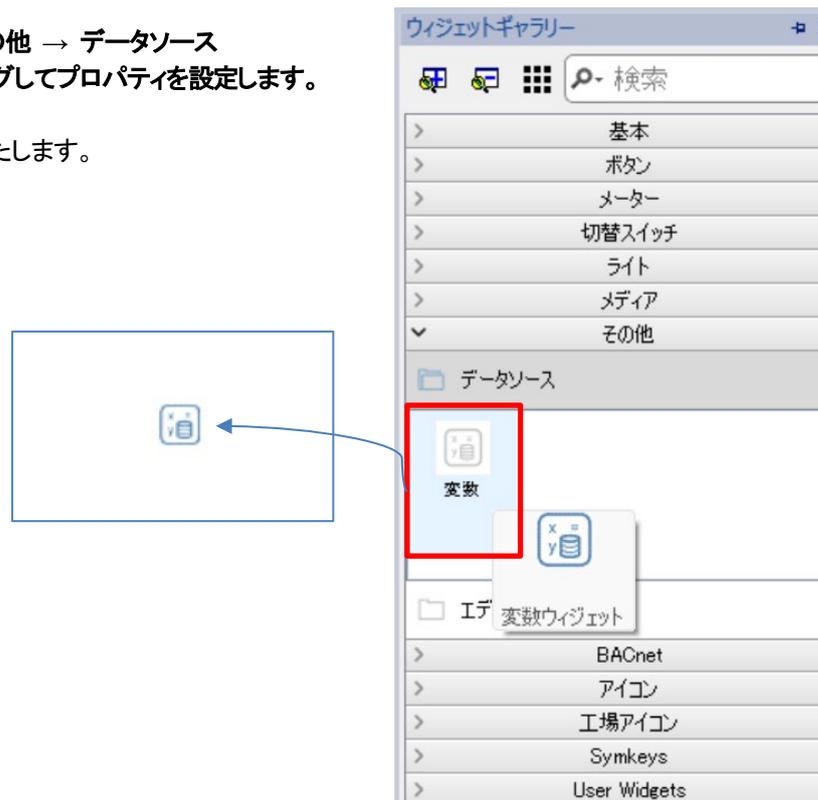
ローカル変数(変数ウィジェットで定義した変数)の使用可能範囲は配置されたページ内です。

グローバル変数は、**プロジェクトプロパティ内の変数ウィジェット**で定義します。

グローバル変数の使用可能範囲はプロジェクト全体です。

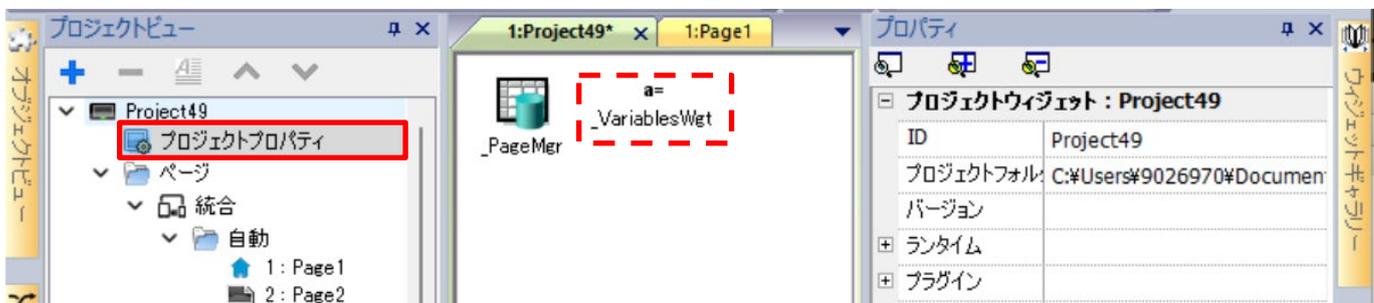
■変数ウィジェットを選択

ウィジェットギャラリー → その他 → データソース
→ 変数 をページ内にドラッグしてプロパティを設定します。
(右図の赤枠)
この先の詳細は、別途説明いたします。



■プロジェクトの変数ウィジェットを選択

プロジェクトプロパティをダブルクリックすると、中央にプロジェクトプロパティの編集画面が表示されます。
画面中央の _VariableWgt(下図赤点線枠)をクリックして、プロジェクトの変数ウィジェットを設定します。



変数欄①をクリックすると右端に+記号②が表示され
+記号をクリックすると[変数の選択]メニュー③が表示され
それをクリックして設定画面を表示します。
この先の詳細は、別途説明いたします。



変数タグ詳細説明

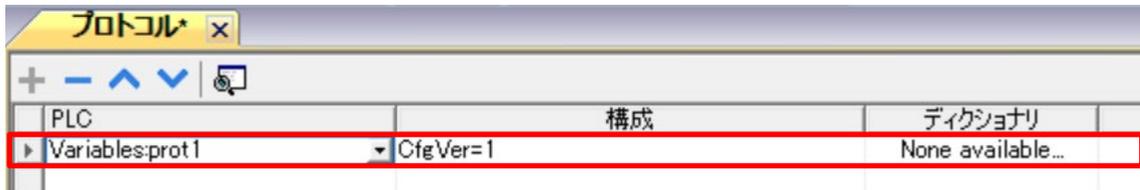
本項では、変数タグについて説明します。システム変数タグについては、別項で説明します。

【設定方法】

通常のタグと同様の方法で以下の手順で登録します。

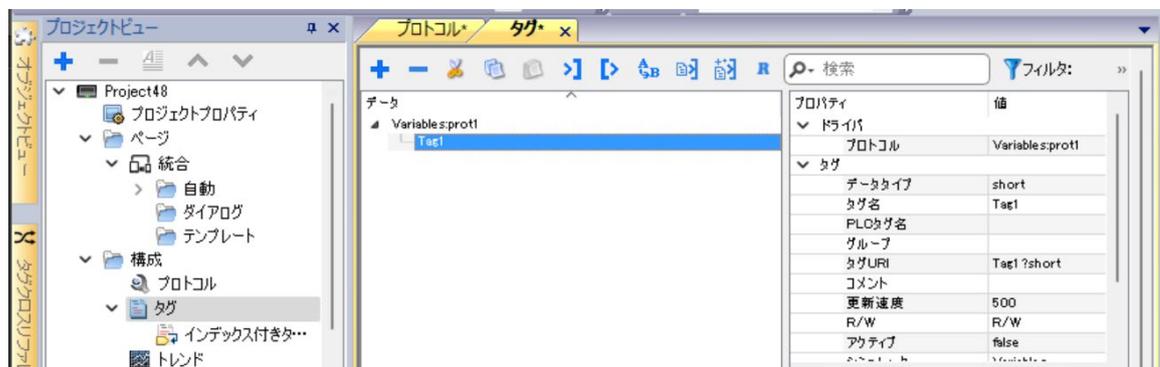
- ① プロトコル → Variables を登録 ② タグ設定 Variables 配下にタグを登録

①



注)プロトコルに Variables を登録すると後ろに portN (N は 1~)が付加されます。複数登録時の識別用です。

②



【変数タグの特徴】

Variables と System Variables プロトコルは、通信実体が無いので、**最大プロトコル数制限の対象外**です。

変数タグは、通常のタグと同様にデータタイプの指定が必要です。

変数タグ (Variables) の場合、**タグプロパティの更新速度は無効**です。

タグなのでグローバル扱いで、どのページのどのウィジェットや JavaScript からでもリンク(アクセス)可能です

変数タグは、**初期値 0 から始まります** (初期値を設定したい場合は別途説明の各種方法があります)

【データリンク方法】

通常のタグと同様に、各ウィジェットのプロパティ → ページエディタで変数タグを選択します。

【JavaScript でのアクセス方法】

通常のタグと同様に**プロジェクトオブジェクトのメソッド getTag または setTag** を使用してアクセスします。

具体例: Tag1 の内容を取得する場合。変数 tag1value に読み出しています。

```
var state = new State();  
var tag1value = project.getTag ( "Tag1", state );
```

具体例: Tag2 の内容を設定する場合。変数 settingValue の値 100 を Tag2 に設定しています。

```
var settingValue= 100;  
project.setTag("Tag2", settingValue);
```

【JavaScript によるアクセスのメリットデメリット】

JavaScript からは、1つのイベントで複数の変数タグへアクセスできます。

変数タグの初期値は設定できません。

変数ウィジェットによるローカル変数定義

本項では、変数ウィジェットによるローカル・グローバル変数定義について説明します。

【設定方法】

- 1:ローカル変数 変数ウィジェットをページにドラッグして、定義します。変数種類概要説明で説明済み
- 2:グローバル変数 プロジェクトプロパティの _VariableWgt

【ローカル変数の設定】

ページに配置した変数ウィジェット①をクリックして、
変数ウィジェットのプロパティを表示します。(右下図)



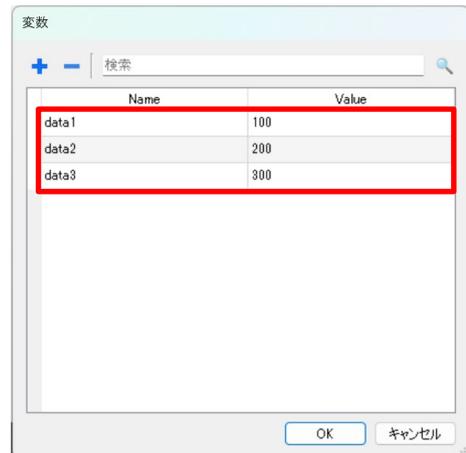
変数欄の<empty>②をクリックすると右端に+記号③が表示され
+記号をクリックすると[変数の選択]メニュー④が表示され
それをクリックして設定画面を表示します。



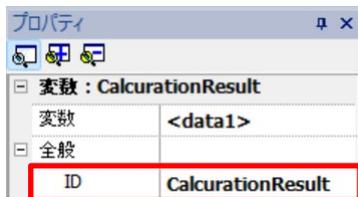
変数の設定画面 下図赤枠の+記号をクリックして変数を追加します。



初期値 100 の変数 data1
初期値 200 の変数 data2
初期値 300 の変数 data3
を追加します。



ここでは、変数ウィジェット ID を **CalcrurationResult** (下図赤枠)に変更し、変数名は、**data1, data2, data3** を定義しています。



変数ウィジェットの名前は、Variable1～の連番で自動命名されますが、そのままと同じような名前が多数でわかりにくくなります。
できるだけ意味の分かりやすい名前を付けるのがお勧めです。

具体例: SystemCounter、InitialValueA、data1、data2、data3 …等々

同様にウィジェットの ID も Variables のままではなく、意味の分かる名前に変更してください。

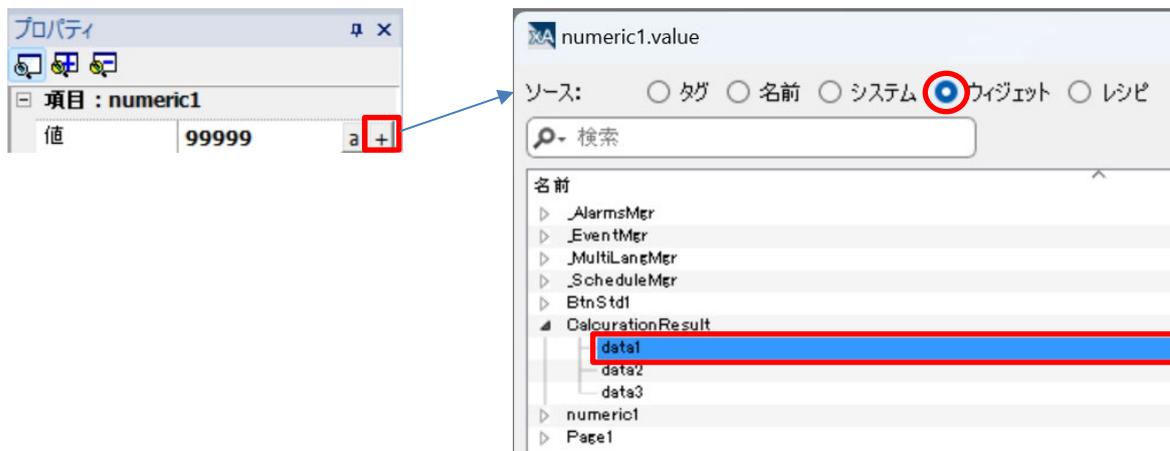
具体例: StatisticalProcessingResult

【ローカル変数の特徴】

1. 変数なので**データタイプの指定は不要**です。
(参考)タグの場合、PLC との通信サイズを確定する必要があるのでデータ型を指定します。
2. 変数ウィジェットによる変数定義はローカル変数です。ページ内でのみリンク(アクセス)可能です。
ページ内のウィジェットプロパティまたは JavaScript からデータリンクまたはアクセス可能です。
3. それぞれ**初期値を設定することが可能**です。
4. 配列変数は定義できません。
5. **アクションリストには、変数に関するアクションはありません。**

【データリンク方法】

各ウィジェットのプロパティ→データリンクエディタでソースにウィジェットを選択(右下図赤丸)→
変数ウィジェット(CalcurationResult)を展開→変数を選択(右下図赤枠・ここでは data1)



【JavaScript でのアクセス方法】

変数ウィジェットオブジェクトのプロパティアクセスメソッドを使用します。

- ・ローカル変数の場合、**変数ウィジェットの getProperty または setProperty メソッド**を使用します。
- ・グローバル変数の場合、**プロジェクトの変数ウィジェットの getProperty または setProperty メソッド**を使用します。

以下の例は、**変数ウィジェット名(ID) = CalcurationResult** で、**変数が data1** の場合の例です。

ローカル変数の読み出し時の具体例

```
// ページオブジェクトの getWdget メソッドで、変数ウィジェット"CalcurationResult"のオブジェクトを取得します。  
var varWgt = page.getWdget("CalcurationResult");  
// 変数ウィジェットの getProperty メソッドで、変数名"data1"の値を読み出して Var1Value にセットします。  
var Var1Value = varWgt.getProperty("data1");
```

ローカル変数の書き込み時の具体例

```
// ページオブジェクトの getWdget メソッドで、変数ウィジェット CalcurationResult のオブジェクトを取得します。  
var varWgt = page.getWdget("CalcurationResult");  
// 変数ウィジェットの setProperty メソッドで、変数"data1"に変数 Var1Value を書き込みます。  
var result = varWgt.setProperty("data1", Var1Value );
```

【メリットデメリット】

変数の初期値が設定できる

JavaScript としてのアクセスはタグに比べて手順が多い

プロジェクト変数ウィジェットによるグローバル変数定義

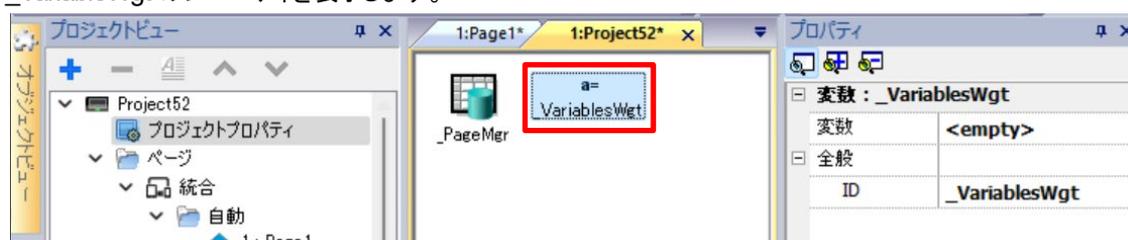
本項では、変数ウィジェットによるローカル・グローバル変数定義について説明します。

【設定方法】

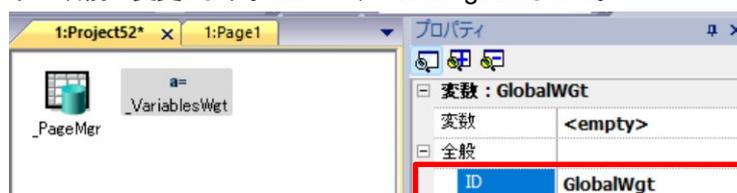
- 1:ローカル変数 前項で説明済み
- 2:グローバル変数 プロジェクトウィジェットの _VariableWgt で定義します。

【グローバル変数の設定】

プロジェクトツリーの一番上から2番目に表示されているプロジェクト・プロパティをクリックすると中央の編集領域に _PageMgr と _VariableWgt(下図赤枠)が表示されますので、_VariableWgt をクリックして右側に _VariableWgt のプロパティを表示します。



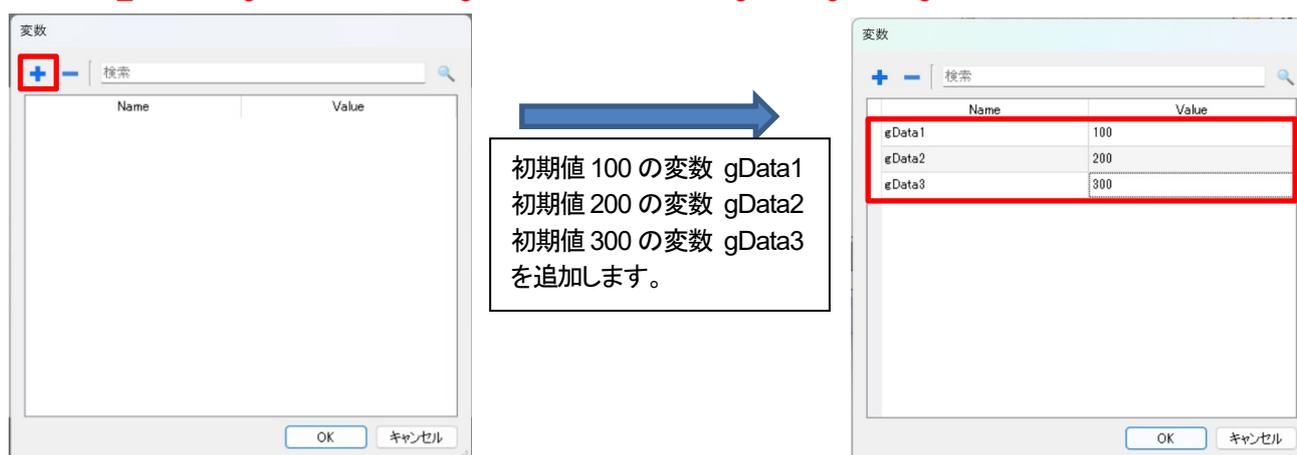
最初に _VariableWgt の ID をわかりやすい名前に変更します。ここでは、GlobalWgt としました。



変数欄の <empty> をクリックすると右端に + 記号が表示され、+ 記号をクリックすると [変数の選択] メニューが表示されそれをクリックして設定画面を表示します。

変数の設定画面 下図赤枠の + 記号(左下図赤枠)をクリックして変数を追加します。

ここでは、_VariableWgt の ID を GlobalWgt に変更し、変数名は、gData1, gData2, gData3 を定義しています。



【グローバル変数の特徴】

1. 変数なのでデータタイプの指定は不要です。
(参考)タグの場合、PLC との通信サイズを確定する必要があるのでデータ型を指定します。
2. プロジェクトプロパティの _VariableWgt による変数定義はグローバル変数です。プロジェクト内のどのページからでもアクセス可能です。但し、JavaScript のみ可能です。ウィジェットプロパティなどとデータリンク設定はできません。
3. それぞれ初期値を設定することが可能です。
4. 配列変数は定義できません。
5. アクションリストには、変数に関するアクションはありません。

【データリンク方法】

各ページのウィジェットのプロパティにデータリンク設定する方法はありません。

JavaScript でのみアクセスできます。

【JavaScript でのアクセス方法】

変数ウィジェットオブジェクトのプロパティアクセスメソッドを使用します。

・グローバル変数の場合、プロジェクトの変数ウィジェットの `getProperty` または `setProperty` メソッドを使用します。

以下の例は、変数ウィジェット名(ID) が `GlobalWgt` で、変数が `gDdata1` の場合の例です。

グローバル変数の読み出し時の具体例

// プロジェクトオブジェクトの `getWdget` メソッドで、変数ウィジェット `GlobalWgt` のオブジェクトを取得します。

```
var varWgt = project.getWidget("GlobalWgt");
```

// 変数ウィジェットの `getProperty` メソッドで、変数名 `gDdata1` の値を読み出します。

```
var Var1Value = varWgt.getProperty("gDdata1");
```

グローバル変数の書き込み時の具体例

// プロジェクトオブジェクトの `getWdget` メソッドで、変数ウィジェット `GlobalWgt` のオブジェクトを取得します。

```
var varWgt = project.getWidget("GlobalWgt");
```

// 変数ウィジェットの `setProperty` メソッドで、変数 `gDdata1` に変数 `Var1Value` を書き込みます。

```
var result = varWgt.setProperty("gDdata1", Var1Value);
```

システム変数の概要とデフォルト変数・システム変数

システム変数は、WH 本体動作に関する情報を含む特殊な変数です。

システム変数をタグとして使用する場合は、System Variables プロトコルを登録し、その配下にタグを登録する必要があります。

下表のデフォルト変数は、**タグを使用しなくても変数を直接ウィジェットプロパティにデータリンク**することができます。

System Variables プロトコルの種類

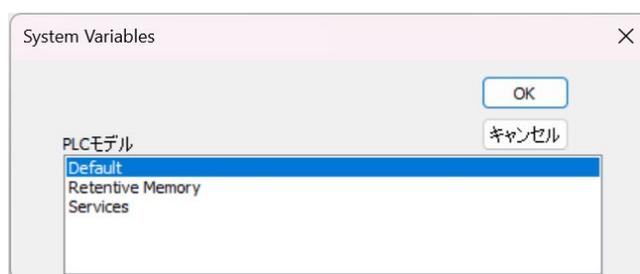
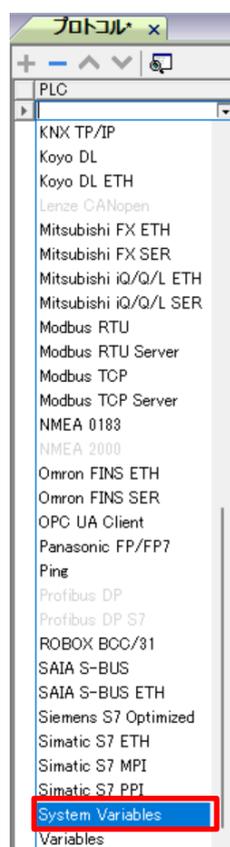
System Variables は、登録時に以下の 3 種類から選択します。

システム変数の種類		使用方法	変数名	タグ名	
1	デフォルト変数	Default	タグ変数または変数	既定	任意
2	保持メモリ変数	Retentive Memory	タグ変数	既定	任意
3	サービス変数	Services	タグ変数	既定	任意

システム変数タグの登録方法

System Variables プロトコルを選択(左下図の赤枠・リストの下から 2 番目)すると、右下図が表示されるので、

Default, Retentive Memory, Services の中から必要なものを選択して登録します。



下図は、Default, Retentive Memory, Services を順番に選択して登録した結果です。

PLC	構成	ディクショナリ
System Variables:prot1	CfgVer=1 model=Default	None available...
System Variables:prot2	CfgVer=1 model=RETENTIVE_MEM	None available...
System Variables:prot3	CfgVer=1 model=Services	None available...

デフォルト変数は、

WH システム内部の状態参照や設定をおこなう変数で名前が決まっている変数です。

タグとして使用する場合は、System Variables → Default プロトコルを登録し、そのプロトコル内にタグを追加します。

メモリアイプ(システム変数名)・データタイプ・オフセットを選択して登録します。

デフォルト変数の2種類の使用方法について

方法1: 変数として使用: **ウィジェットプロパティにデータリンク**して使用する (こちらの利用が一般的)

方法2: タグ変数として使用: System Variables → Default プロトコルとして登録し、その中で選択する方法

タグ変数を使用する必要性には、以下の理由などが考えられます。

1. システム変数とデバイスのタグの間でデータを転送する必要がある場合
2. システム変数の更新レートを選択する必要がある場合

デフォルト変数の種類

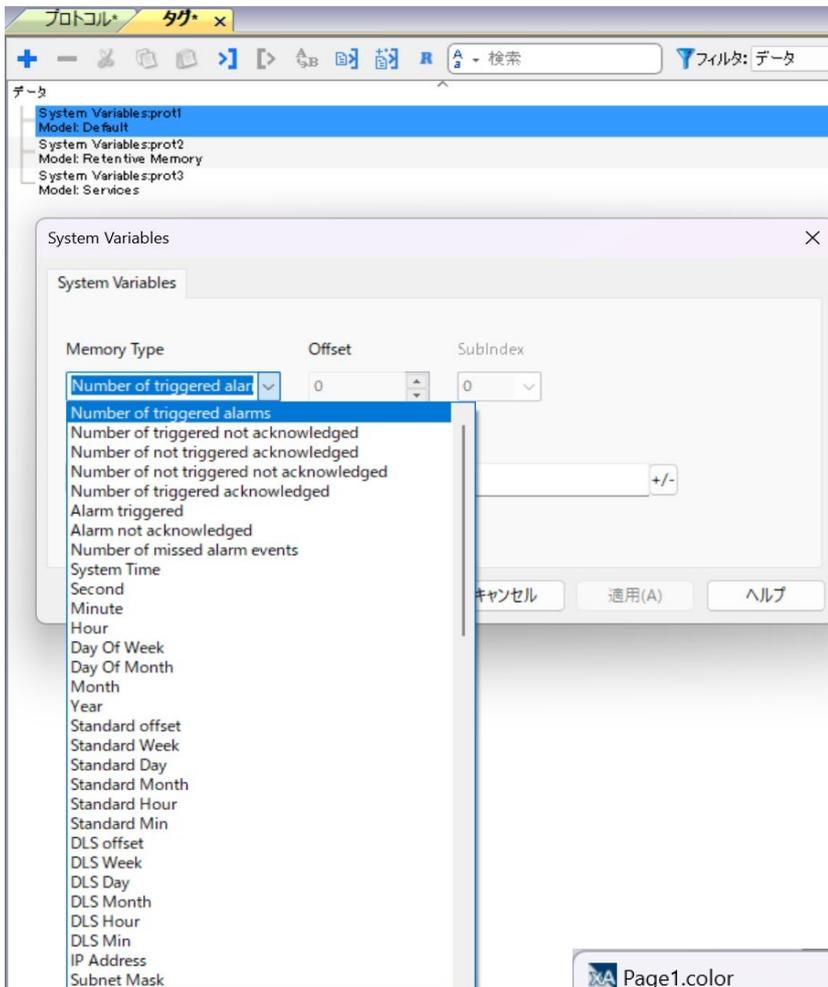
デフォルト変数のカテゴリ	
アラーム変数	Alarms
ブザー変数	Buzzer
通信変数	Communication
サマータイム変数	Daylight Saving Time
デバイス変数	Device
ダンプ情報変数	Dump Information
FTP クライアント変数	FTP
キーパッド変数	Keypad
言語変数	Language
ネットワーク変数	Network
印刷変数	Printing
リモートクライアント変数	Remote Client
画面変数	Screen
SD カード変数	SD Card
サーバ変数	Server
時間変数	Time
USBドライブ変数	USB Drive
ユーザ管理変数	User Management
バージョン変数	Version

タッチスクリーン変数

注)この変数は、ソース→ウィジェットに入っています

■ デフォルト変数タグの登録方法(下図)

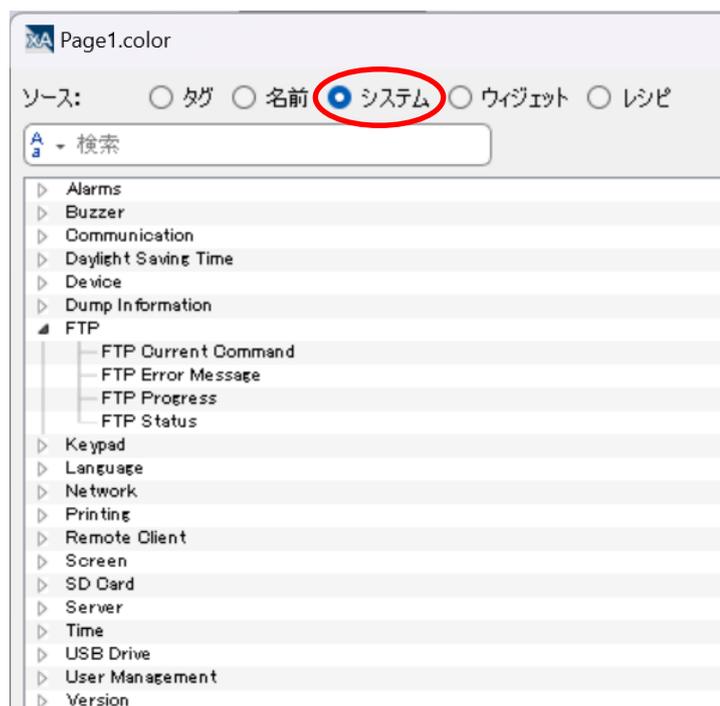
タグの編集画面で、+をクリックするとタグの設定画面が表示されます。Memory Type でシステム変数を選択します。タグ名は、自動的に Tag1～順番に命名されます。TagN のままでは、何の変数なのか識別しにくいので、後で変更可能です。



■ ウィジェットプロパティにデータリンクする方法

デフォルト変数は、タグとして割り付けしないでウィジェットのプロパティにそのままデータリンクして使用することが可能です。

右図で、ソースにシステムを選択すると(赤丸)各カテゴリのシステム変数が表示されるので展開してシステム変数を選択できます。

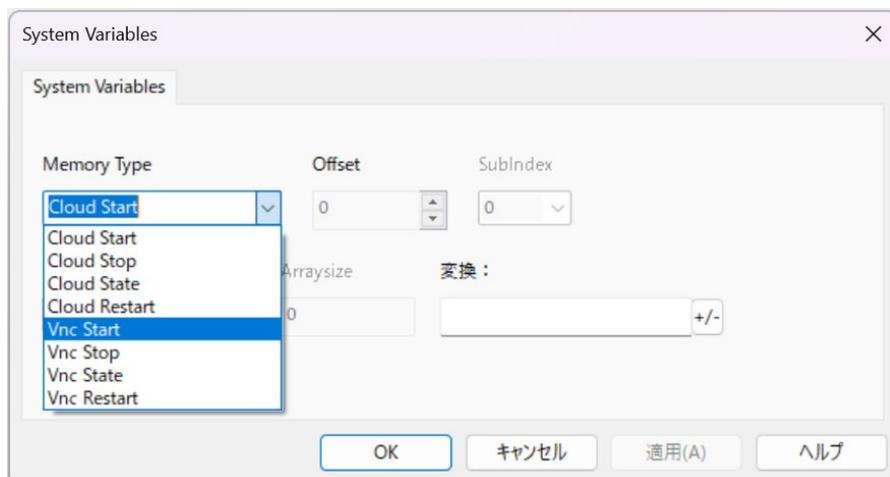


■ サービス変数は、

System Variables → Services プロトコルを登録してタグを追加して使用します。

Cloud や VNC サービスの制御用に割り当てられており、名前や用途が決まっている変数です。

下図は、サービス変数の登録画面です。



システム変数タグの保持メモリ変数

■保持メモリ変数

保持メモリ変数タグとして定義できます。**保持メモリ**というのは**電源断後も値を保持する**という意味です。

System Variables → **Retentive Memory** プロトコルを登録して任意のタグ名を登録して使用します。

通常のタグの様に名前と型付けを行い、WH 内部の不揮発メモリ (FRAM) にタグを割り付けて使用します。

※**FRAM**※ SRAM と同レベルの速度で書き込み可能な不揮発メモリ

WH のタイプによって FRAM デバイスを搭載していない機種では、フラッシュメモリに割り当てられるので注意が必要です。フラッシュメモリへの書き込みは手順と時間がかかるので、頻繁な書き換えが発生する場合などに電源断と重なるとフラッシュメモリの対象領域周辺データが破壊される場合があります。

下表の赤字の機種は、FRAM を搭載していません。

機種	FRAM
WHS1-04	No
WHS1-07	No
WHS1-10	No
WHA1-07	Yes
WHA1-10	Yes
WHA1-15	Yes
WHA1-21	Yes
WHA1-05	Yes

保持メモリは、16kB の連続した不揮発メモリを**バイトオフセット**を指定してアクセス可能です。

bool 型の場合は、オフセットと**サブインデックス** (バイト内のビットオフセット) を指定します。

※**バイトオフセット** 領域の先頭を 0 バイト目としたときのバイト単位の相対番地

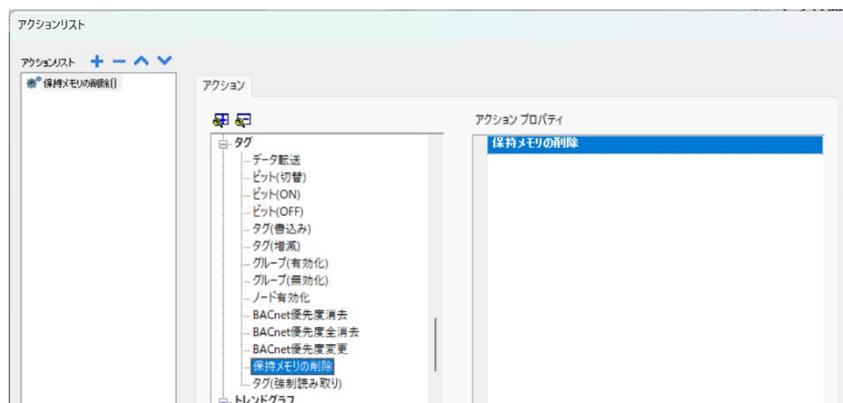
※**サブインデックス** バイト内のビット位置で、LSB が 0、MSB が 7 です。Bool 型のみ指定します。

占有する領域サイズは、指定したデータタイプによって異なります。

■保持メモリの消去タイミング

保持メモリは、以下の二つのタイミングで消去されます。

1. ウィジェットのイベントプロパティに**タグアクション**→**保持メモリの削除**を指定し、消去するときに**イベント**を実行します。

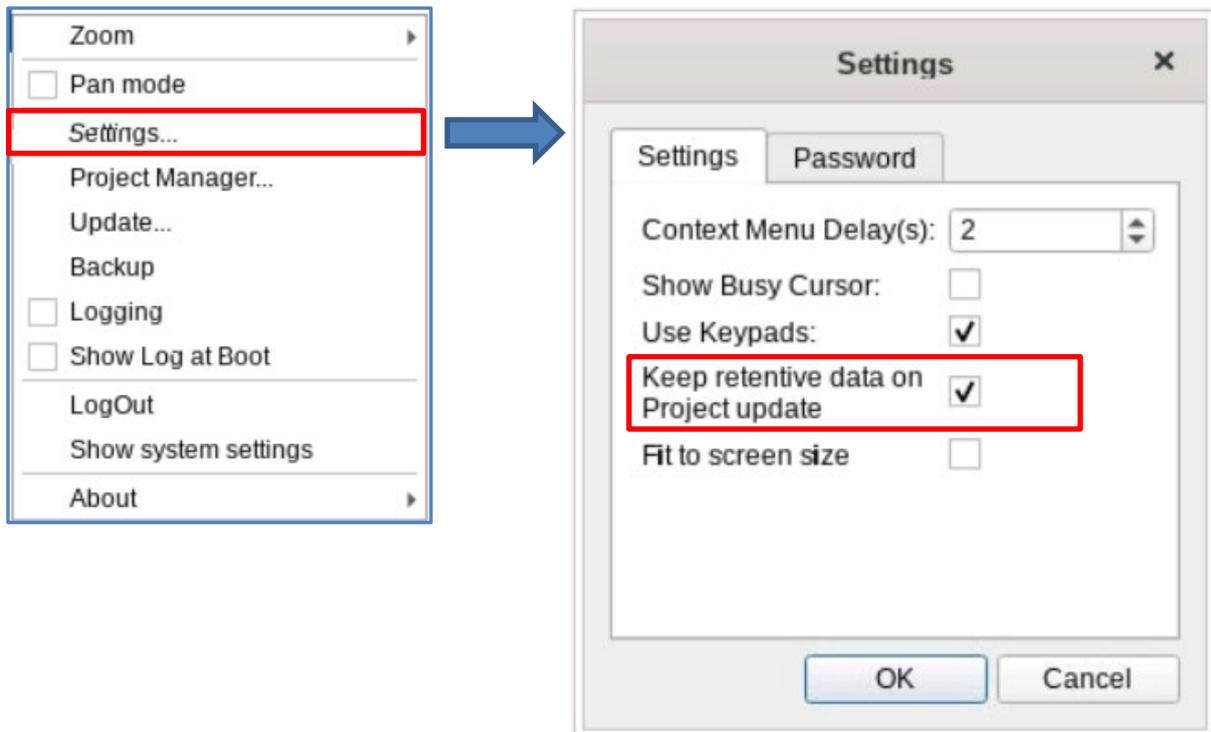


2. **プロジェクトをダウンロード**します。

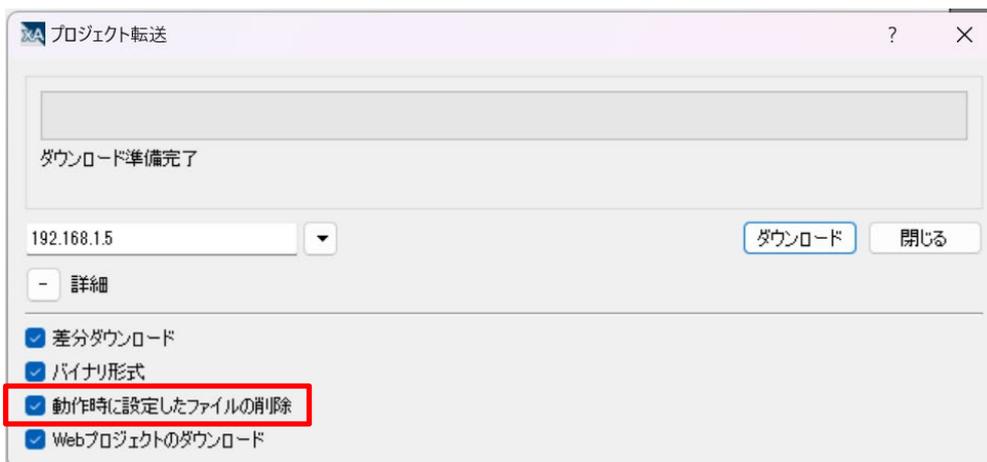
プロジェクトダウンロード時は、デフォルトで保持メモリを消去する設定になっています。

■プロジェクトダウンロード時に保持メモリを消去しない設定

WH 画面長押しまたは、ウィジェットアクションのコンテキストメニューで左下図を表示し、Settings を選択し、Settings 画面を表示(右下図)し、Keep retentive data on Project update をチェックしてください。



但し、プロジェクトダウンロード画面で、[動作時に設定したファイルの削除]にチェックを入れた場合(下図赤枠)は、WH 側での保持メモリを消去しない設定は無効となり、保持メモリは消去されます。



タグの一括読み込み(データ転送機能使用・非サイクリック通信)

PLC からタグのデータを周期的に読み出す処理が表示器の基本モニタ動作ですが、通常必要としないデータがある場合にも周期的に読み出す動作を行っている通信データ量が増大して他の重要な通信動作に影響する場合があります。

特にそのデータ量が多い場合は、なおさらです。

また、タグの更新周期を遅く設定する方法では、更新周期が遅すぎて必要時に間に合わない場合があります。

そのような場合に**任意のタイミングでタグの一括読み出し動作を行う**にはどうすれば良いでしょうか？

以下の方法があります。

- ① 一括読み出しを行う**タグの更新周期を「手動」**に設定します。この設定を行えば、**サイクリック通信を行いません**。
- ② 一括読み出し要求トリガ(トリガデータまたはイベント)を使用して、データ読み出しを行います。
この読み出し方法には、以下の2つの方法があります。
 1. データ転送機能を使用 **データ転送機能を使用**し、転送要求フラグによってデータ読み出しを実行します。
 2. JavaScript を使用 イベント処理で、ウィジェットアクションの **JavaScript プログラム** でデータ読み出しを実行します。

ここでは、**上記 ① と ②-1 の方法**について具体例を用いて説明します。

具体例の仕様説明

【PLC 側の設定と動作】

- ① DT0 から DT4 の 5 個のグローバル変数(UINT)を定義します。変数名は、g_uiTestDTn (n=0~4)とします。
- ② DT0 を 10 秒周期でカウントアップします。DT0 カウントアップ時に DTn=DT(n-1)+1(n=1~4)の動作を行います。
つまり、DT0=1 の時、DT1~DT4 は、それぞれ 2,3,4,5 となります。
動作確認がしやすいように 10 秒周期としています。必要に応じて変更してください。

PLC 側の通信設定は、通信可能な設定を行ってください。

ここでは、IP アドレス = 192.168.1.6 、システムコネクション 1 のポート番号 = 32769 とします。

グローバル変数定義

	クラス	変数名	FPア...	IECアドレス	データ型	初期値
1	VAR_GLOBAL	g_uiTestDT0	DT0	%MW5.0	UINT	0
2	VAR_GLOBAL	g_uiTestDT1	DT1	%MW5.1	UINT	0
3	VAR_GLOBAL	g_uiTestDT2	DT2	%MW5.2	UINT	0
4	VAR_GLOBAL	g_uiTestDT3	DT3	%MW5.3	UINT	0
5	VAR_GLOBAL	g_uiTestDT4	DT4	%MW5.4	UINT	0

プログラム

```
if (DF(sys_bPulse1s)) then
    uiCounter:=uiCounter+1;
    if (uiCounter = 10) then
        uiCounter:=0;
        uiSetdata := uisetdata+1;
    end_if;
    g_uiTestDT0:= uiSetdata;
    g_uiTestDT1:= g_uiTestDT0 + 1;
    g_uiTestDT2:= g_uiTestDT1 + 1;
    g_uiTestDT3:= g_uiTestDT2 + 1;
    g_uiTestDT4:= g_uiTestDT3 + 1;
end_if;
```

【WH 側の設定】

① プロトコル設定

1. 使用する PLC に合ったプロトコルと通信設定を行います。
2. Variables プロトコルを設定します。

PLC	構成	ディクショナリ	オフラインデバイスチェック	ラインデバイスチェック間隔
Panasonic FP/FP7:prot1	CfgVer=1 defNodeId=1 ForceRe None available..	None available..	<input checked="" type="checkbox"/>	30
Variables:prot2	CfgVer=1	None available..	<input type="checkbox"/>	Not applicable

② タグ設定

1. 上記①- 1 プロトコルに unsignedshort 型の 5 個のタグを登録します。読み出しデータの定義用です。
タグ名は、g_uiTestDTn (n=0~4) で、PLC アドレスは DT0~DT4、更新周期を「手動」に設定してください。
2. 上記①- 2 Variables プロトコルに unsignedshort 型の 5 個の変数タグを登録します。読み出しデータの格納用です。
タグ名は、Tag1, Tag2, Tag3, Tag4, Tag5 とします。
3. さらに Variables プロトコルに boolean 型のタグを定義します。タグ名は Tag6 です。
データ転送の一括読み出し用のトリガビットです。

データ	タグURI
Panasonic FP/FP7:prot1 Model: FP7	
g_uiRdTag	1?DT?1 0?unsignedShort-1 0
g_uiWtTag	1?DT?1 00?unsignedShort-1 0
g_uiTestDT0	1?DT?0?unsignedShort
g_uiTestDT1	1?DT?1?unsignedShort
g_uiTestDT2	1?DT?2?unsignedShort
g_uiTestDT3	1?DT?3?unsignedShort
g_uiTestDT4	1?DT?4?unsignedShort
WTtag0	1?DT?1 0?unsignedShort
WTtag1	1?DT?1 1?unsignedShort
WTtag2	1?DT?1 2?unsignedShort
WTtag3	1?DT?1 3?unsignedShort
WTtag4	1?DT?1 4?unsignedShort
Variables:prot2	
Tag1	Tag1?unsignedShort
Tag2	Tag2?unsignedShort
Tag3	Tag3?unsignedShort
Tag4	Tag4?unsignedShort
Tag5	Tag5?unsignedShort
Tag6	Tag6?boolean
Tag7	Tag7?boolean

プロパティ	値
▼ ドライバ	
モデル	FP7
プロトコル	Panasonic FP/FP7:prot1
▼ タグ	
データタイプ	unsignedShort
タグ名	g_uiTestDT0
PLCタグ名	
グループ	
タグURI	1?DT?0?unsignedShort
コメント	
更新速度	手動
R/W	R/W
アクティブ	false
シミュレータ	Variables
スケール	None
最小値	0
最大値	65535

③ ページ設定

1. Tag1, Tag2, Tag3, Tag4, Tag5 の値を表示するためのデータウィジェットを 5 つ定義します。
それぞれのデータウィジェットの値プロパティには、Tag1~Tag5 を読み出し専用 (R) でデータリンクします。
2. データを一括読み出しするためのトリガボタンを配置します。
このボタンの値プロパティにデータ転送起動トリガ用の boolean 型のタグ (Tag6) を定義します。

DT0 → Tag1	968
DT1 → Tag2	969
DT2 → Tag3	970
DT3 → Tag4	971
DT4 → Tag5	972

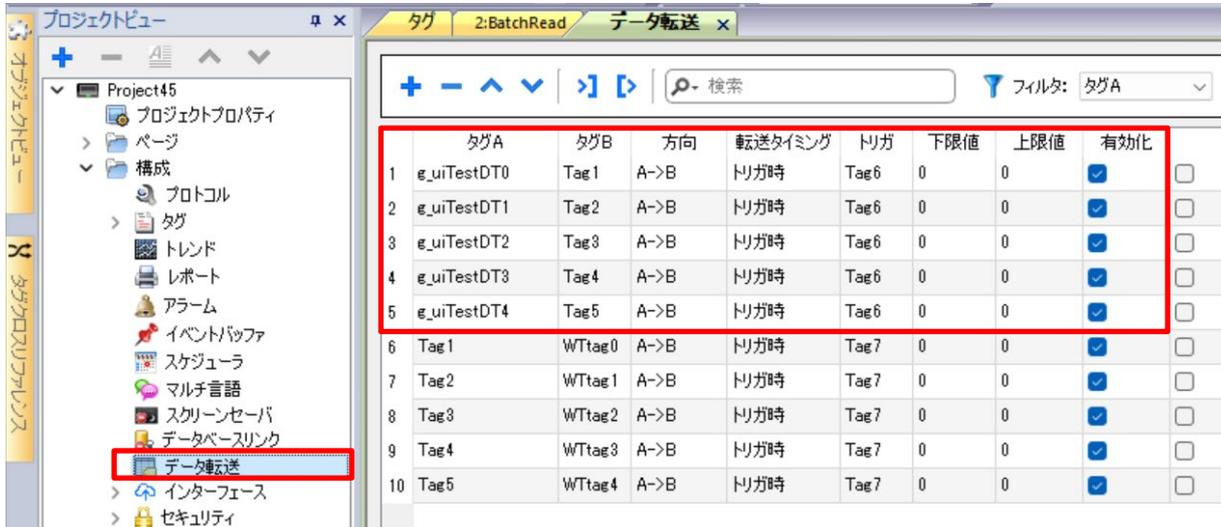
タグ値一括読み出し

プロパティ	
項目	numeric3
値	99999
データリンク	Tag1
アクセス種	R
数値形式	Numeric

プロパティ	
ButtonsWithLabel : BtnStd1	
値	0
データリンク	Tag6 R/W
アクセス種	R/W
クリック形式	momentary
スタイル	2D
オートリピート	Disabled
ホールド時間[m]	-1
ラベル	タグ値一括読み出し

④ データ転送設定

データ転送エディタを開いて、転送元(タグ A)に g_uiTestDTn(n=0~4)、転送先(タグ B)に Tag1~Tag5 を設定します。
 転送タイミングは「トリガ時」、トリガには「Tag6」、上限値・下限値には 0 (デフォルト値)を指定します。
 トリガのデータが上限値・下限値の範囲外になった時にデータ転送トリガ(要求)発生とみなされます。



【具体例の動作確認】

PLC のプロジェクトを転送し、xAscender でシミュレータを起動して、タグの通信をリアルに設定して、動作確認を行います。
 下図のようにシミュレーションで使用のチェックをはずして、[OK]します。



FPWIN-Pro での PLC 側のデータモニタの値と xAscender シミュレータ上の読み出し値を見比べて一括読み出しができていないことを確認します。(通常はデータ更新されず、一括読み出しボタンをクリックしたときのみ読み出しが実行されること)
 タグ値一括読み出しボタンをクリックして、下図のようにデータが転送され表示されることを確認してください。



【動作の仕組み】

タグ値一括読み出しボタンをクリックすると Tag6 が ON します。Tag6 は、データ転送機能の転送トリガとして登録しているので、データが PLC から読み出されて、Tag1, 2, 3, 4, 5 にセットされ、データウィジェットに表示されます。

【データ転送機能を使用して PLC から一括データ読み出しする際の注意点】

1. 転送元(タグ A)のタグ値は更新されません。本来読み出しデータが格納される転送元タグには読み出しデータ値は入りません。転送元タグは、PLC のデータ領域定義(データ読み出し元)の参照にのみ使用されているようです。
2. タグに配列変数を定義しても、データ転送エディタに登録する際は、各要素を 1 つずつ登録する必要があります。
3. 転送元(タグ A)・転送先(タグ B)のタグ指定に変数ウィジェットで定義した変数は指定できません。
 グローバル変数ウィジェットで定義した変数は指定できます。

タグの一括書き込み(データ転送機能使用・非サイクリック通信)

タグの一括読み出し(データ転送機能・非サイクリック通信)の項では、データ転送機能を使用して PLC のデータレジスタ DT0~DT4 から Tag1, 2, 3, 4, 5 への一括読み出しを実施しました。

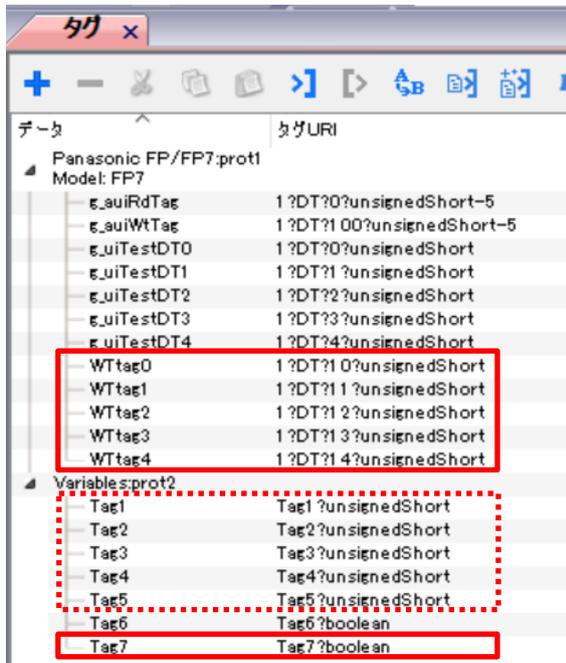
注) DT0~DT4 は、タグ g_uiTestDTn(n=0~4)で指定

ここでは、上記の一括読み出しにより、Tag1, 2, 3, 4, 5 へ読み出したデータをそのまま PLC のデータレジスタ DT10~にデータ転送機能を使用して一括書き込みする方法を説明します。

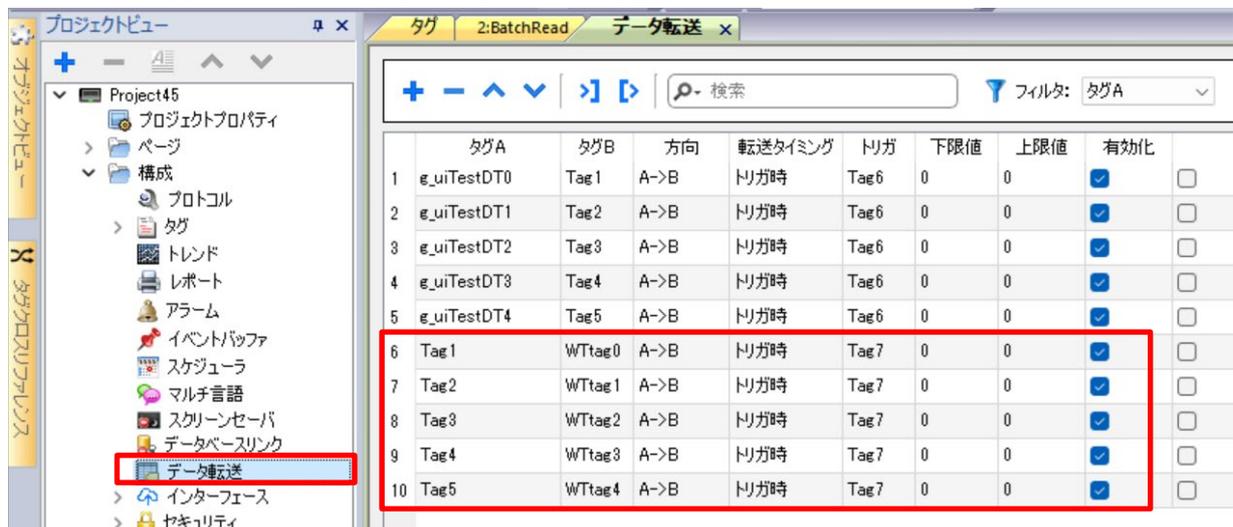
「タグの一括読み出し(データ転送機能・非サイクリック通信)」で使用したプロジェクトに以下の内容を追加します。

1. PLC への書き込み用の 5 つのタグを定義します。
タグ名は WTagN (N= 0,1,2,3,4)、PLC アドレスは DT10~DT14、unsignedshort 型、RW、更新周期は「手動」
2. Variables プロトコルに boolean 型のデータ転送の要求を行うためのタグ Tag7 を定義します。
3. Tag1, 2, 3, 4, 5 から上記の書き込み用タグ(WTagN (N= 0,1,2,3,4))へのデータ転送テーブルを追記します。
4. 一括書き込み用のボタンを配置し、値プロパティをデータ転送要求用タグ Tag7 にデータリンクします。

追加するタグ(下図赤枠)



PLC へのデータ書き込み用のデータ転送テーブル定義(下図赤枠)



一括読み出しと一括書き込みのテスト画面

上段は、「タグの一括読み込み(データ転送機能使用・非サイクリック通信)」のサンプル画面と同じです。

タグ値一括読み出しボタンの値プロパティには、Tag6 がデータリンクしてあります。

【動作】タグ値一括読み出しボタンをクリック → データ転送要求 Tag6 を ON → PLC から TagN へデータ読み出し

下段は、ここで追加したデータ転送テーブルを使用して、Tag1～Tag4 のデータを PLC の DT10～DT14 に書き込むための画面です。タグ値一括書き込みボタンの値プロパティには、Tag7 がデータリンクしてあります。

【動作】タグ値一括書き込みボタンをクリック → データ転送要求 Tag7 を ON → Tag1～Tag4 のデータを PLC に転送

DT0 → Tag1	0	タグ値一括読み出し
DT1 → Tag2	0	
DT2 → Tag3	0	
DT3 → Tag4	0	
DT4 → Tag5	0	
Tag1 → DT10		タグ値一括書き込み
Tag2 → DT11		
Tag3 → DT12		
Tag4 → DT13		
Tag5 → DT14		

シミュレータを起動して動作を確認します。

1. 一括読み出しボタンをクリックして、Tag1,2,3,4,5 に PLC のデータを読み出します。
2. 一括書き込みボタンをクリックして、書き込み用のデータ転送の起動要求フラグ Tag7 をセットし PLC にデータ書き込みを行います。FPWIN-Pro7 など、DT10～をモニタし、これらの動作が実行できていることを確認します。

DT0 → Tag1	213	タグ値一括読み出し ①
DT1 → Tag2	214	
DT2 → Tag3	215	
DT3 → Tag4	216	
DT4 → Tag5	217	
Tag1 → DT10		タグ値一括書き込み ②
Tag2 → DT11		
Tag3 → DT12		
Tag4 → DT13		
Tag5 → DT14		

変数名	値	FPアドレス
1 g_auiWtTag	③	
2 [0]	213	DT10
3 [1]	214	DT11
4 [2]	215	DT12
5 [3]	216	DT13
6 [4]	217	DT14

【データ転送機能を使用して PLC に一括データ書き込みする際の注意点】

- 1: **転送先(タグ B)のタグ値は更新されません。** 本来書き込みデータが格納される転送先タグには書き込みデータ値は入りません。転送先タグは、PLC のデータ領域定義(データ書き込み先)の参照にのみ使用されているようです。
- 2: **タグに配列変数を定義しても、データ転送エディタに登録する際は、各要素を1つずつ登録する必要があります。**
- 3: **転送元(タグ A)・転送先(タグ B)のタグ指定に変数ウィジェットで定義した変数は指定できません。**
グローバル変数ウィジェットで定義した変数は指定できます。

PDF 表示機能 1 PDF ファイルの WH への格納方法(FTP 転送)

ここでは、PDF 表示機能の動作確認するための準備として、表示させる PDF ファイルを WH に格納する方法を説明いたします。具体的には、WH の FTP サーバを利用します。PC 側では、ファイルエクスプローラを使用します。

(WH に外付けする USB メモリに pdf ファイルを格納する場合は、PC から USB メモリに複製して WH に装着するだけです。)

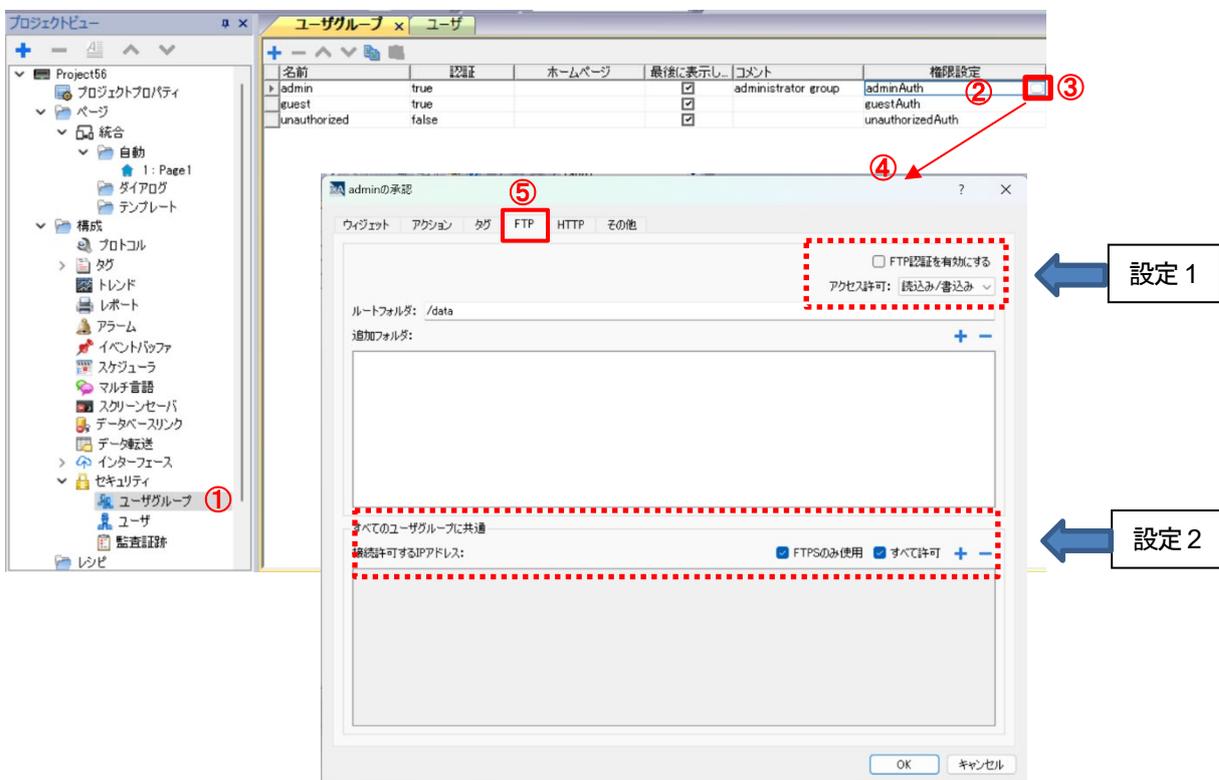
ここでは、デフォルト設定を優先しながら、できるだけ簡単な設定を説明いたします。以下の手順で進めてください。

1. FTP サーバのセキュリティ設定
2. プロジェクトのダウンロード
3. ファイルエクスプローラで、WH の FTP サーバへログイン
4. ファイルエクスプローラで、WH に PDF ファイルをコピー

【FTP サーバのセキュリティ設定】

最初に FTP サーバのログイン認証設定を行います。下記の手順で admin グループの FTP サーバ認証の画面を開きます。プロジェクトビュー内「セキュリティ」→ ① ユーザグループ → ② 権限設定 [adminAuth] をクリック → ③ 右端のボタン(下図赤枠) →

④ admin の承認画面 → ⑤ FTP タブを選択



次に、FTP サーバの認証設定を行います。

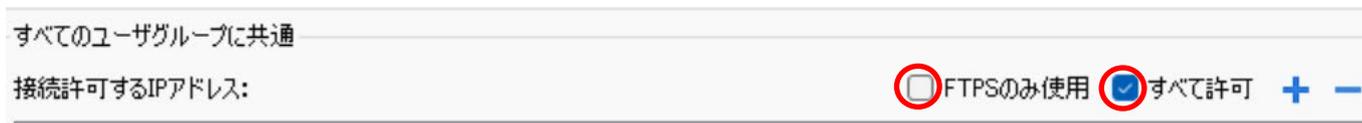
【admin グループに対する設定】 上図設定 1 の部分

[FTP 認証を有効にする] をチェックして、FTP 認証を有効にします。(右図赤丸)



【すべてのユーザグループに対する設定】 上図設定 2 の部分

デフォルトでは、FTPS のみ使用とすべて許可の両方にチェックが入っていますが、ここでは簡単化の為に FTPS のみ使用の **チェックを外し**、すべて許可はそのままとします。(下図赤丸)

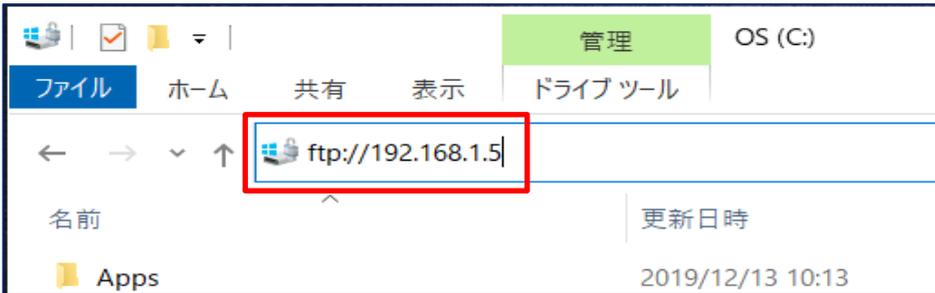


最後に[OK]をクリックして admin の承認画面を閉じ、プロジェクトを WH に転送します。

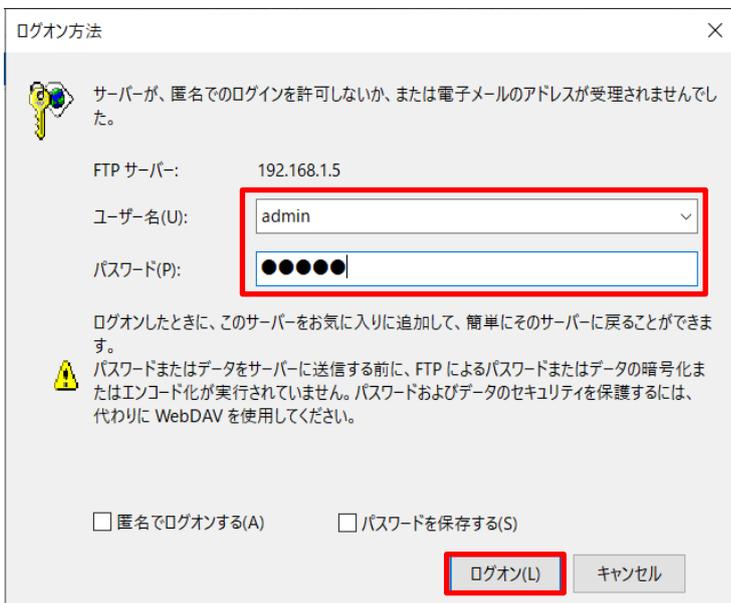
【ファイルエクスプローラで、WHにPDFファイルを転送】

ここでは、ファイルエクスプローラでWHのFTPサーバへログインして、PDFファイルをWHに転送します。

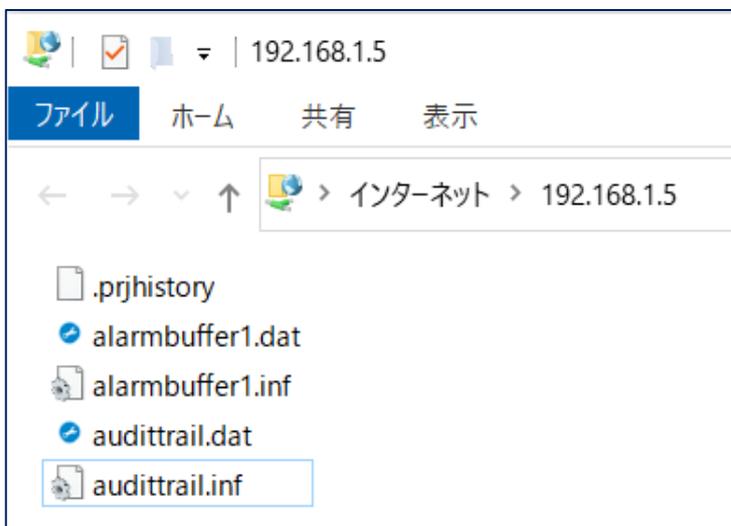
1. エクスプローラのアドレスバー(下図赤枠)に <ftp://192.168.1.5> と入力して ENTER します。



2. ログイン画面が表示されるので、ユーザ名とパスワード(初期値共に admin)を入力し、[ログオン]ボタンをクリックします。



3. WHのFTPサーバに接続出来たら、ルートフォルダの内容一覧が別画面で表示されます。

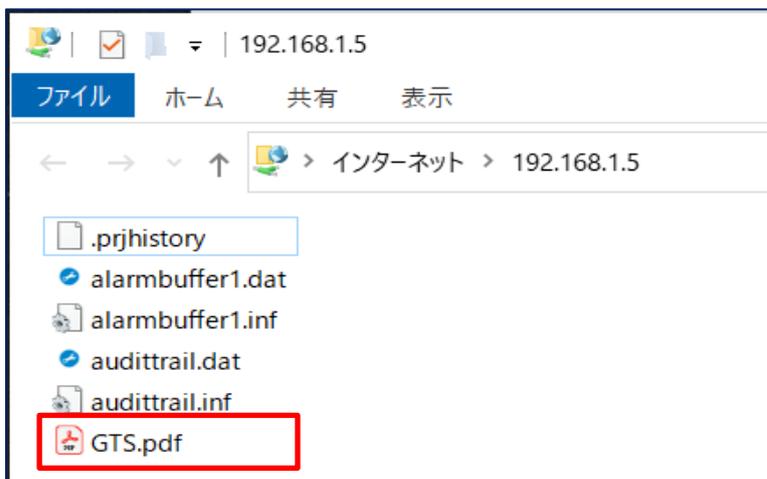


4. WH で表示させたいファイルが格納されているフォルダを開いて(下図)

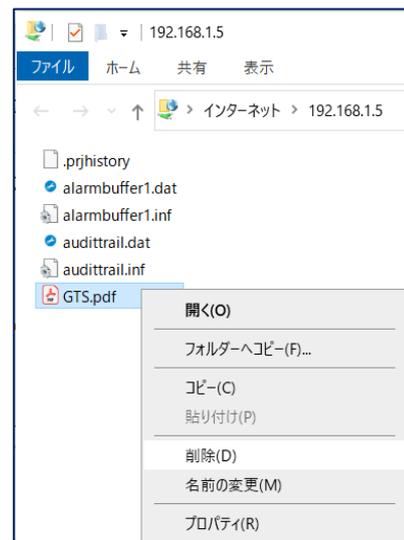
ファイルを WH の FTP サーバのルートディレクトリにコピーします。ここでは、**GTS.pdf** ファイルです。コピーは、通常のファイルと同様に**ドラッグ&ドロップ**または、**コピー&ペースト**で実行できます。



GTS.pdf ファイルが WH の FTP サーバのルートフォルダに複写できました。



複写したファイルは、通常のファイル操作と同様の操作で、FTP サーバ上から削除可能です。右クリック→削除



その他の情報

ブラウザから ftp サーバにアクセスできる機能は、Google Chrome、Edge ともにありません。

Firefox にはありますが、会社 PC では使用できないかもしれません。

Windows のコマンドプロンプトから利用できる FTP クライアントは、active 接続のみで passive 接続に対応していません。

ログインはできますが、ファイル一覧表示やファイルのアップロードはできません。

PDF 表示機能 2 PDF ビューの起動と PDF ファイルの閲覧

PDF 表示機能は、**ページアクションの PDF ビュー起動**を利用します。
表示用のボタンを配置して、ボタンウィジェットのクリックイベントに上記のアクションを登録します。

ページに下図のようなボタンを配置してください。



このボタンのクリックイベントに PDF ビュー起動のアクションを登録します。(下図)



各項目は、以下のように指定します。

アプリケーション名には、**起動する PDF ビューワの名前**、保存先には、**そのアプリの格納先**、引数には**そのアプリが表示するファイルの場所とファイル名**を指定します。

PDF ファイルが WH 本体内蔵メモリに格納されている場合

アプリケーション名	PDF.exe
保存先	%Flash%QTHMI%PDF
引数	/mnt/data/hmi/qthmi/deploy/data/GTS.pdf

PDF ファイルが WH に外付けする USB メモリに格納されている場合

アプリケーション名	PDF.exe
保存先	%Flash%QTHMI%PDF
引数	/mnt/usbmemory/GTS.pdf

注)上表では、引数(PDF ファイルの場所と名前)のみが異なります

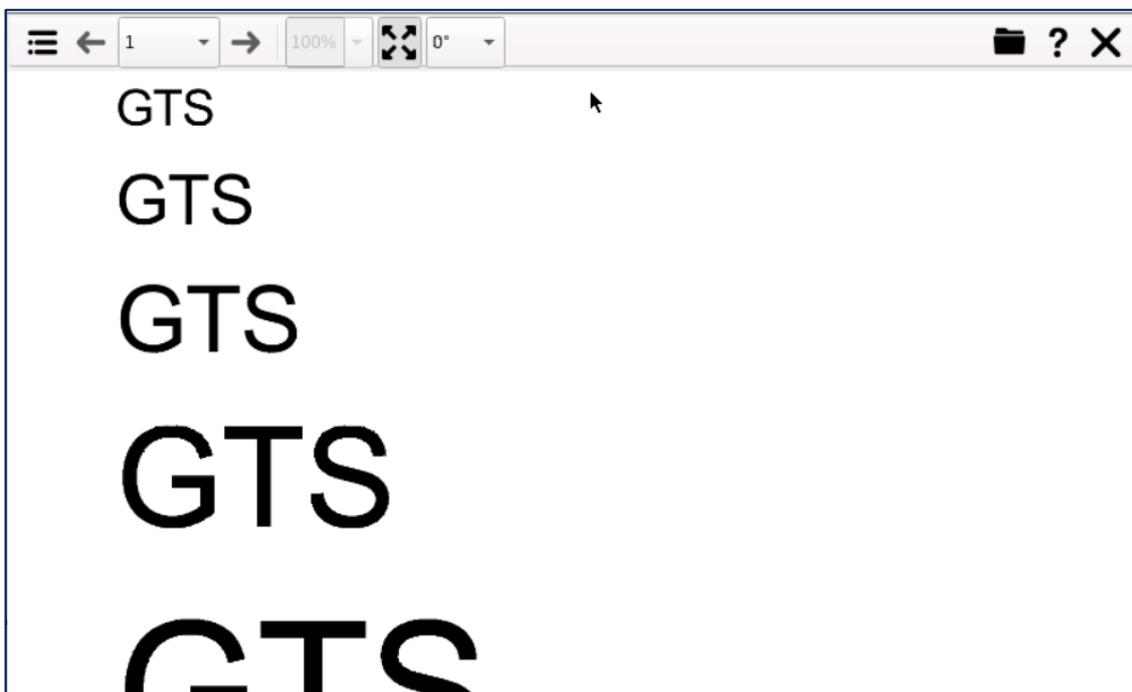
単一インスタンスには、true または false を指定します。

単一インスタンスが true とは、1 本の pdf ファイルのみ表示許可されるということです。

プロジェクトを WH にダウンロードして、表示ボタンをクリックし、GTS.pdf が表示されることを確認してください。



画面を上下にスワイプすることにより、PDF ファイルの全体を見ることができます。



正規表現の使い方

ここでは WH での正規表現の使用方法を説明します。

エディタ等の検索ツールなどで文字列の検索時に文字列のパターンを指定出来たら、一度に複数の文字を検索出来て便利です。

例えば、下記のような文字列が含まれるファイルがある場合に検索したい文字列が abc0,abc1,abc2 のみだった場合

それらの条件が 1 つの検索文字列として指定出来れば、一回の検索で希望する文字列が検索できます。

その検索文字列のパターンを指定する方法が文字列の正規表現です。

文字列のパターンマッチングを行うためのパターンを指定する表現方法の事を正規表現と呼びます。

abc0

abc1

abc2

abc5

abc6

本項では、xAscender 中の検索や置換機能で使用可能と推定できる正規表現について一般的な説明を行います。

注) xAscender で使用可能な正規表現の範囲について明確な説明がないので、使用不可のものが含まれている可能性があります。

正規表現の文字列の構成

1. 検索対象に含まれる文字そのもの
2. 文字列の **パターンを表現するための特殊文字(メタ文字とも呼ばれる)**

特殊文字(メタ文字)で各種のパターンを表現します。

特殊文字(メタ文字)の種類

.	^	\$	[]	*	+	?		()	¥
---	---	----	----	---	---	---	--	----	---

検索用メタ文字	意味
.	ドット 任意の 1 文字を意味します。
^	^に続く文字が行頭
\$	\$の直前の文字が行末
[]	[]内に含まれるいずれかの 1 文字
[^]	[^]内に含まれない 1 文字
*	直前のパターンの 0 回以上繰り返し(最長一致)
+	直前のパターンの 1 回以上繰り返し(最長一致)
?	直前のパターンの 0~1 回繰り返し(最長一致)
	(exp1 exp2) という指定で exp1 または exp2 という意味
()	()内の文字パターンのグループ化、または 置換文字列に引用する部分の指定(¥1~¥9 での部分参照)
¥	直後のメタ文字をエスケープする。メタ文字を文字として指定する。

次ページに続く

¥n	改行文字 ※CR+LF(制御コード 0x0d+0x0a)とLF(制御コード 0x0a) ※エディタや設定によっては CR(制御コード 0x0d)もマッチ
¥t	タブ文字(制御コード 0x09)
¥s	空白文字(半角スペース、¥t、¥n、¥r、¥f)すべて
¥S	空白文字(半角スペース、¥t、¥n、¥r、¥f)以外すべて
¥x##	16進数 ※#は0~9、a~f、A~Fのいずれか
¥b	語の区切り位置
¥B	語の区切り位置以外
¥d	すべての半角数字 例 1)¥d → 半角数字 例 2)¥d+ → 数字列 例 3)¥d{4} → 4桁の数字 例 4)¥d{4,} → 4桁以上の数字 例 5)¥d{1,4,} → 4桁までの数字(1~4桁の数字) 例 6)¥d{4,10} → 4~10桁の数字(最長一致) 例 7)¥d{4,10}? → 4~10桁の数字(最短一致) 例 8)¥d*[135791]¥b → 奇数
¥D	半角数字以外すべて
¥w	すべての半角英数字とアンダースコア
¥W	半角英数字とアンダースコア以外すべて
¥l	半角英小文字すべて
¥L	半角英小文字以外すべて(英大文字、数字、全角文字などすべて)
¥u	半角英大文字すべて
¥U	半角英大文字以外すべて(英小文字、数字、全角文字などすべて)
¥a	アラーム(制御コード 0x07)
¥c#	対応するコントロール文字 ※#はA-Zの半角英字のみ指定可
¥e	エスケープ(制御コード 0x1b)
¥f	改ページ(制御コード 0x0c)
¥r	リターン、CR(制御コード 0x0d) ※エディタや設定によっては CR+LFのCR部分にはマッチしない
¥v	垂直タブ(制御コード 0x0b)
¥Q~¥E	~部分に含まれるメタ文字をメタ文字として解釈しない (「¥」でエスケープしなくてよい)
(?)	以降の英字の大文字・小文字を同一視する
(?-i)	以降の英字の大文字・小文字を区別する
(?:~)	パターンのグループ
*?	直前のパターンの0回以上繰り返し(最短一致)
+?	直前のパターンの1回以上繰り返し(最短一致)
??	直前のパターンの0~1回繰り返し(最短一致)
{min,max}	直前のパターンのmin回からmax回繰り返し(最長一致)
{min,}	直前のパターンのmin回以上繰り返し(最長一致)
{num}	直前のパターンのnum回繰り返し
{min,max}?	直前のパターンのmin回からmax回繰り返し(最短一致)
{min,}?	直前のパターンのmin回以上繰り返し(最短一致)
{num}?	直前のパターンのnum回繰り返し(最短一致)

置換用メタ文字	意味
¥N	検索文字列で指定した N 番目の () で囲まれたパターンを引用する。 N は 1~9。指定例 ¥1¥2¥3¥4
¥n	改行文字(CR+LF)
¥t	タブ文字(ハードタブ)
¥s	半角スペース
¥xXX	コード XX(16 進数 2 桁)の文字
¥0	見つけた文字列全体(0 は半角数字のゼロ)
¥	直後のメタ文字をエスケープする。メタ文字を文字として指定する。

タグ名の一括置換方法(正規表現で部分置換)

タグ名の手動登録時、登録順に自動的に tagN (N=1,2,3,4...) というタグ名が付けられます。

このまま使用すると意味が分かりにくいので、後で名前を変更しようとすると手間がかかります。

特に、インデックス付きタグセットや JavaScript で複数のタグを使用する場合、タグ名は前もって昇順の番号やアルファベット等を付加して設計します。このような場合にタグ名を置換機能で置換すると便利です。

さらにその際、検索文字列や置換文字列を正規表現で指定すると文字列のパターンを一定の規則で置換できるので置換処理の回数を減らし、かつ間違いを減らすことが可能です。

ここでは、以下の2つの場合について置換の具体例を説明いたします。

- ① 「インデックス付きタグセットの登録方法」の項で説明したタグ名の置換による登録方法
上項では、下図のようなタグの登録状態で説明しています。

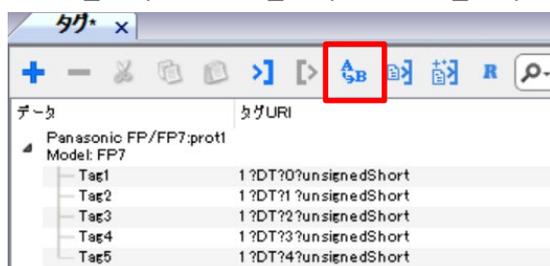
名前	PLC 上のアドレス	データ型
Room1_TempA	DT0	unsigned short
Room2_TempA	DT1	unsigned short
Room3_TempA	DT2	unsigned short
Room4_TempA	DT3	unsigned short
Room5_TempA	DT4	unsigned short
Room1_TempB	DT5	unsigned short
Room2_TempB	DT6	unsigned short
Room3_TempB	DT7	unsigned short
Room4_TempB	DT8	unsigned short
Room5_TempB	DT9	unsigned short
Room1_TempC	DT10	unsigned short
Room2_TempC	DT11	unsigned short
Room3_TempC	DT12	unsigned short
Room4_TempC	DT13	unsigned short
Room5_TempC	DT14	unsigned short

- ② ①の登録では、Room1_TempA,B,C の PLC 上のアドレス配置が、DT0, DT5, DT10 となっていました、データ収集の手順を考慮すると、以下のように DT0, DT1, DT2 の配置になっている場合もあります。

名前	PLC 上のアドレス	データ型
Room1_TempA	DT0	unsigned short
Room1_TempB	DT1	unsigned short
Room1_TempC	DT2	unsigned short
Room2_TempA	DT3	unsigned short
Room2_TempB	DT4	unsigned short
Room2_TempC	DT5	unsigned short
Room3_TempA	DT6	unsigned short
Room3_TempB	DT7	unsigned short
Room3_TempC	DT8	unsigned short
Room4_TempA	DT9	unsigned short
Room4_TempB	DT10	unsigned short
Room4_TempC	DT11	unsigned short
Room5_TempA	DT12	unsigned short
Room5_TempB	DT13	unsigned short
Room5_TempC	DT14	unsigned short

①の場合の設定手順説明

下図のように Tag1, Tag2, Tag3, Tag4, Tag5 を登録します。これらを Room1_TempA, Room2_TempA, Room3_TempA, Room3_TempA, Room4_TempA, Room5_TempA に変更します。



(下図)

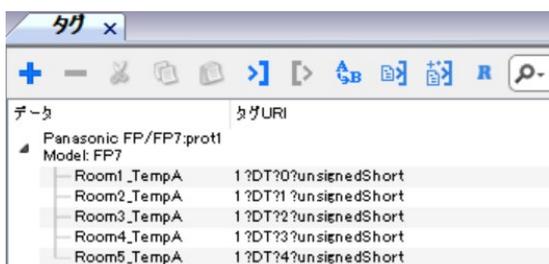


検索する値:に Tag(¥d) 、置換する値に Room¥1_TempA と入力し、[大文字と小文字を区別する]と[正規表現を使用する]にチェックし、最後に[すべて置換]ボタンをクリックします。

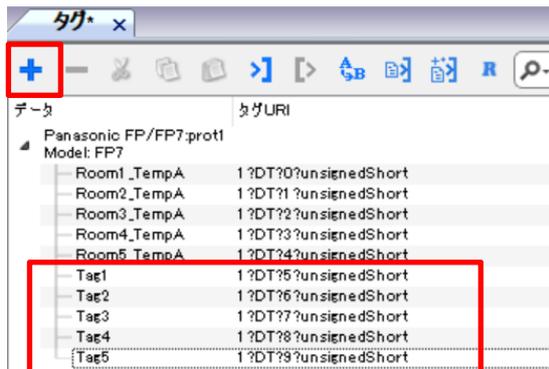
- 検索文字列に指定した Tag の部分は、そのままの値が使用されます。
- (¥d)の¥d は、0～9のいずれかが 1 文字の事です。()部は、置換文字列で指定するためのグループの意味です。
- 置換文字列に指定した Room の部分は、そのままの値が使用されます。
- ¥1 は、検索文字列で指定した()内の文字の事です。検索文字列に出現するグループ()の順番により¥1～¥9 まで指定できます。
- _TempA は、そのままの値が使用されます。
- つまり、TagN (N は 0～9)のタグ名を検索して、RoomN_TempA という名前に置換するという動作になります。



TagN (N は 0～9)のタグ名が、RoomN_TempA に置換されました。(下図)



Room5_TempA をクリックした後にタグの追加ボタン+をクリックして Tag1~Tag5 を続きに追加します。

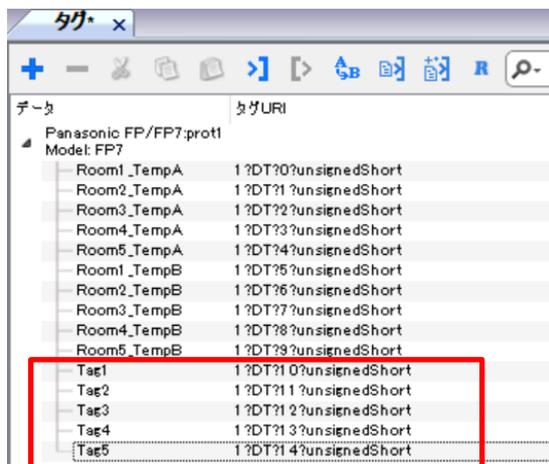


最初と同様にして、TagN (Nは0~9)のタグ名を RoomN_TempB に置換します。

検索する値:に Tag(#{d})

置換する値に Room#{1}_TempB

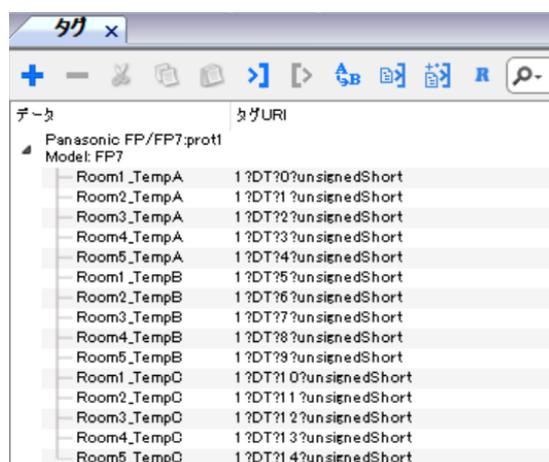
置換が完了したら、Room5_TempB をクリックした後にタグの追加ボタン+をクリックして Tag1~Tag5 を続きに追加します。



最初と同様にして、TagN (Nは0~9)のタグ名を RoomN_TempC に置換します。

検索する値:に Tag(#{d})

置換する値に Room#{1}_TempC

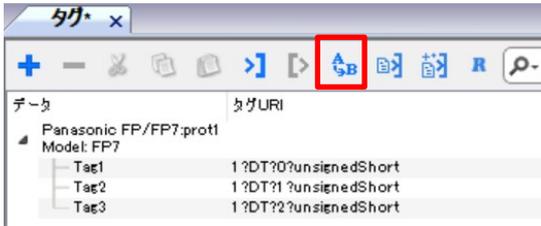


以上で、①の場合のタグ名の正規表現を使用した置換による設定ができました。

②の場合の設定手順説明

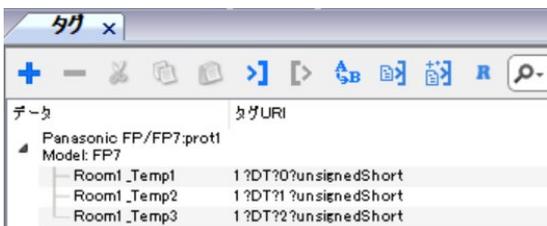
①の場合に比べて、設定手順が多くなります。

下図のように Tag1, Tag2, Tag を登録します。これらを Room1_TempA, Room1_TempB, Room1_TempC に変更します。



次に上部の置換ボタン(上図赤枠)をクリックして、置換画面を起動します。

検索する値に Tag(¥d) 、置換する値に Room1_Temp¥1 と入力し、[大文字と小文字を区別する]と[正規表現を使用する]にチェックし、最後に[すべて置換]ボタンをクリックします。



Room1_Temp3 をクリック後、+ボタンを 3 回クリックして Tag1, 2, 3 を追加します。次に置換画面を開いて

検索する値に Tag(¥d) 、置換する値に Room2_Temp¥1 と入力し、[大文字と小文字を区別する]と[正規表現を使用する]にチェックし、すべて置換します。

Room2_Temp3 をクリック後、+ボタンを 3 回クリックして Tag1, 2, 3 を追加します。次に置換画面を開いて

検索する値に Tag(¥d) 、置換する値に Room3_Temp¥1 と入力し、[大文字と小文字を区別する]と[正規表現を使用する]にチェックし、すべて置換します。

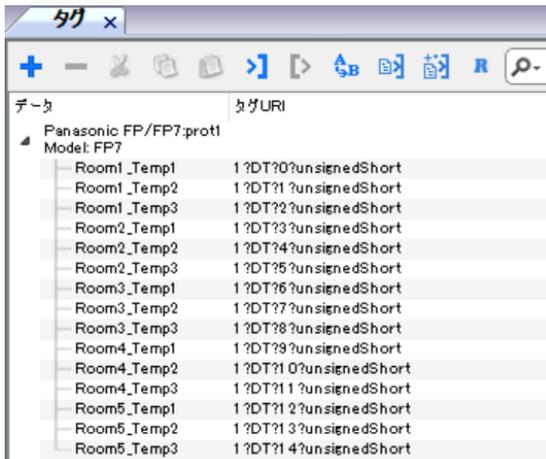
Room3_Temp3 をクリック後、+ボタンを 3 回クリックして Tag1, 2, 3 を追加します。次に置換画面を開いて

検索する値に Tag(¥d) 、置換する値に Room4_Temp¥1 と入力し、[大文字と小文字を区別する]と[正規表現を使用する]にチェックし、すべて置換します。

Room2_Temp4 をクリック後、+ボタンを 3 回クリックして Tag1, 2, 3 を追加します。次に置換画面を開いて

検索する値に Tag(¥d) 、置換する値に Room5_Temp¥1 と入力し、[大文字と小文字を区別する]と[正規表現を使用する]にチェックし、すべて置換します。

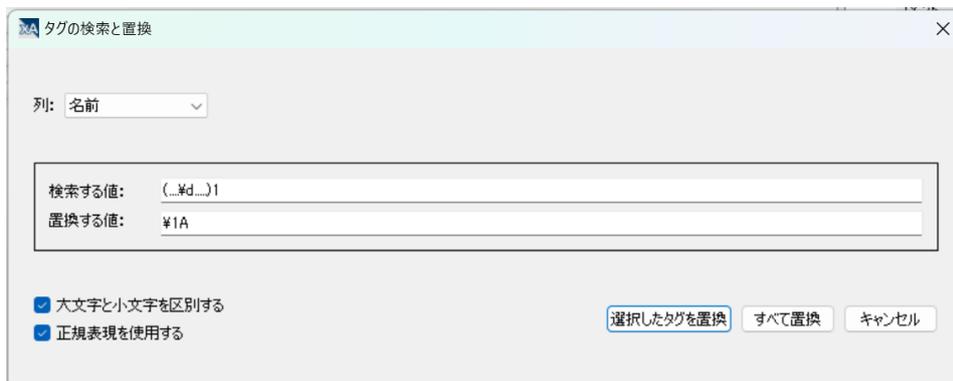
下図のように各部屋の温度 A,B,C のタグが、PLC 上の連続するメモリ領域に割り当たりました。



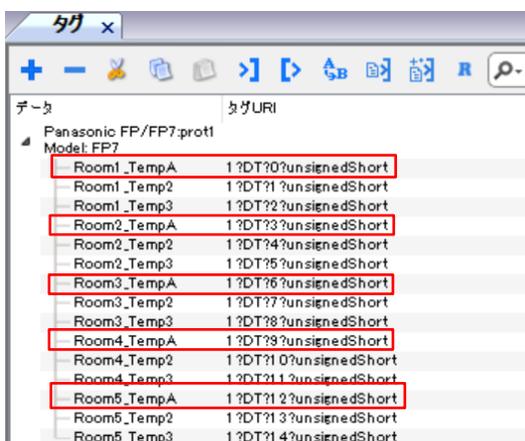
次に、各タグ名末尾 TempN を次の様に変更します。Temp1→TempA, Temp2→TempB, Temp3→TempC

置換画面を開いて、

検索文字列に(...%d...)1 置換文字列に¥1A を設定してすべて置換します。



RoomN_Temp1 が RoomN_TempA に置換されました。

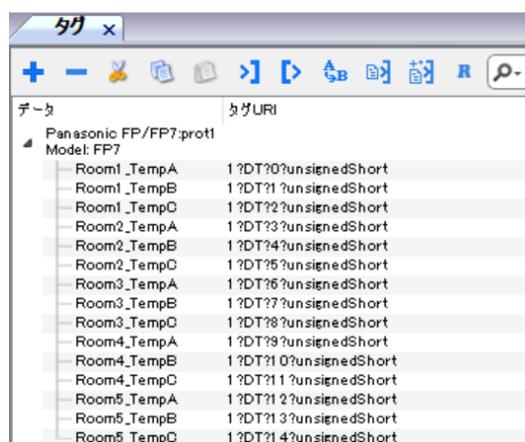


同様に置換画面を開いて、

検索文字列に(...%d...)2、置換文字列に¥1B を設定してすべて置換します。

さらに置換画面を開いて、

検索文字列に(...%d...)3、置換文字列に¥1C を設定してすべて置換します。



以上で、②の場合のタグ名を置換機能を使用して設定できました。

①の場合(置換回数3回)に比べて、かなり手数(置換回数8回)がかかりました。

②のような場合には、エクセルでタグの辞書を作成して、それをインポートする方法が修正などにも対応しやすくお勧めです。次項では、エクセルでのタグ設定を説明します。

エクセルでのタグ名編集とタグのインポート

ここでは、エクセルを利用してタグ名を編集し、xAscender プロジェクトにインポートする方法を説明します。

概略の手順は、FPWIN-Pro7 でグローバル変数を定義して csv ファイルにエクスポートし、その csv ファイルをエクセルに読み込んで複数のタグを編集・保存し、xAscender にインポートすることにより、簡単にタグ名を設定する方法です。

この方法によるとエクセルの編集機能や保存機能を利用できるので、xAscender のタグ編集機能よりも簡単に編集・保存できます。

ここでの説明は、以下の場合などを想定しています。

1. 制御・モニタシステムにおいて PLC と WH 表示器間の通信インターフェース仕様を決めて並行して開発する場合。
2. 既存の GR7 の PLC プロジェクトをベースにして、WH 表示器を追加する場合
3. PLC を介さずに直接機器と通信する場合

既存の Pro7 の PLC プロジェクトをベースに WH 表示器を追加する場合は想定していません。

その場合は、Pro7 プロジェクトのグローバル変数を xAscender プロジェクトにインポートするだけで良いからです。

エクセルでのタグ編集手順

① 前準備

タグ編集用の元ファイル(エクセルブック)がある場合(以下の説明 5 で作成するファイル)、そのファイルを開いて、

②に進んでください。

1. FPWIN-Pro7 でグローバル変数を定義して、csv ファイルにエクスポートします。Pro7export.csv

注)エクスポートしたファイルは、csv 属性のファイルで、項目の区切り文字はセミコロンです。

2. エクスポートファイルをテキストエディタで開いて、全て選択し、クリップボードにコピーします。
3. 新規にエクセルを開いて、A1 セルにクリップボードの内容を貼り付けます。
4. 列幅をそろえます。
5. エクセルブックとして保存します。→PRO7export.xlsx

以降は、このファイルをタグ編集用元ファイルとして読み込むことができます。

② タグの編集・保存、xAscender プロジェクトのタグ編集画面でのインポート

1. タグデータの編集を行います。
2. コンマ区切りの csv ファイルとして保存します。→TagdataWH.csv
エクセルブックとしても保存しておくことタグ編集の中断再開時に便利です。
3. csv ファイルをテキストエディタで開いて、コンマをセミコロンに全て変更し、保存します。
4. xAscender プロジェクトのタグ編集画面でインポートします。

次ページ以降で、上記①②の手順の詳細を説明します。

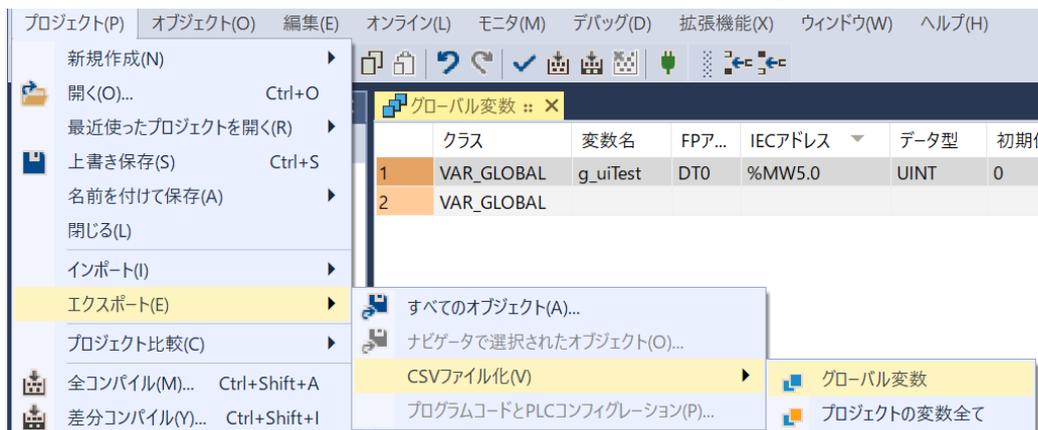
①前手順の詳細説明

1. FPWIN Pro7 で、下図のようにグローバル変数を定義します。

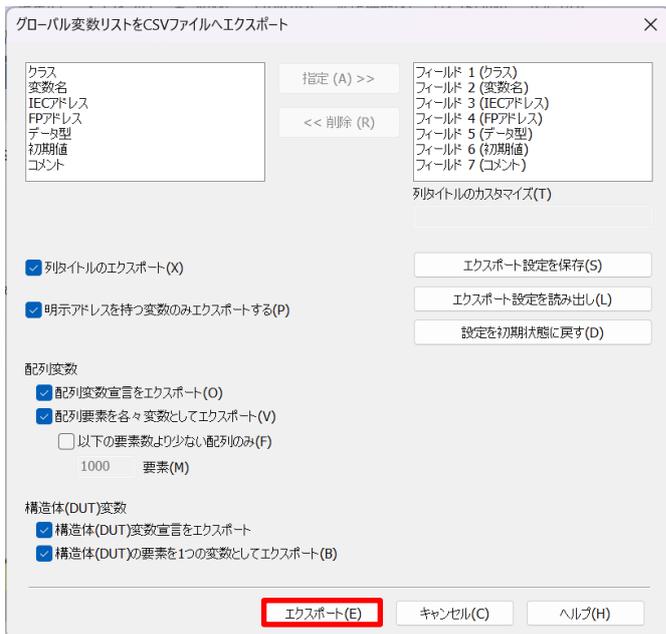
	クラス	変数名	FPア...	IECアドレス	データ型	初期値	外..	コメント
1	VAR_GLOBAL	g_uiTest	DT0	%MW5.0	UINT	0		

2. グローバル変数をエクスポートします。

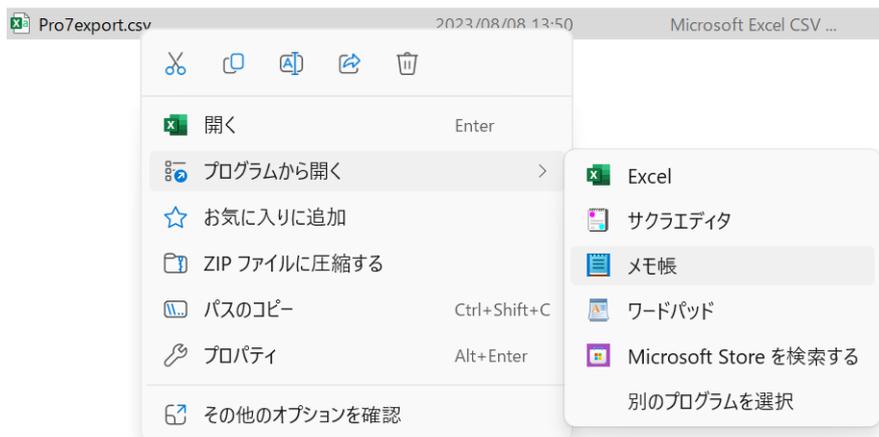
[プロジェクト]メニュー→[エクスポート]→[CSV ファイル]→[グローバル変数]



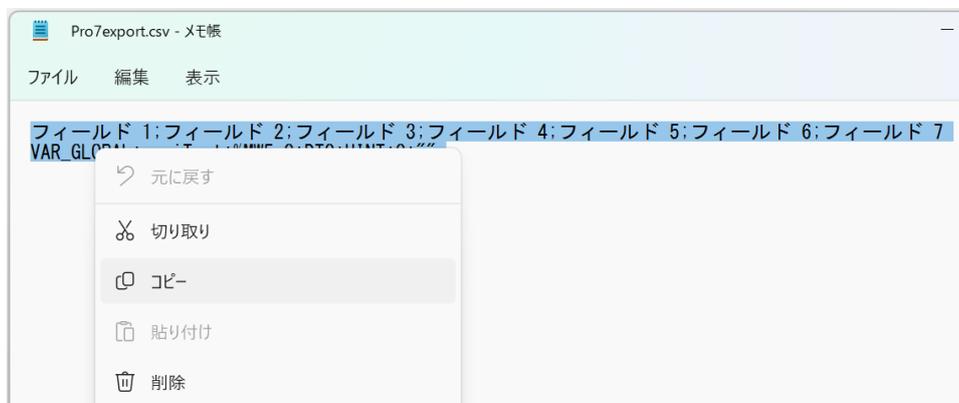
下図画面で、エクスポートを実行してください。ファイル名はここでは、Pro7export.csv としています。



3.2 で作成したグローバル変数のエクスポートファイル Pro7export.csv をメモ帳で開いて、

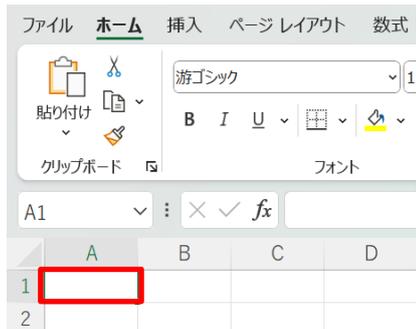


テキストをすべて選択してクリップボードにコピーします。

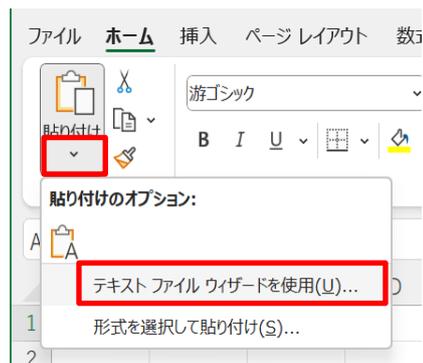


3. 新規にエクセルを開いて、A1 セルにクリップボードの内容を貼り付けます。

エクセルの A1 セルをクリックして、クリップボードから貼り付けます。



貼り付けボタンの下部▼をクリックし、オプションのテキストファイルウィザードを使用して貼り付けます。



[コンマやタブなどの区切り文字によってフィールドごとに区切られたデータ]を選択し、次にボタンをクリックします。



区切り文字にセミコロンを選択し、データのプレビューで、各フィールドが列ごとに分離されていることを確認して完了します。



データが貼り付けられました。

	A	B	C	D	E	F	G	H
1	フィールド 1	フィールド 2	フィールド 3	フィールド 4	フィールド 5	フィールド 6	フィールド 7	
2	VAR_GLO	g_uiTest	%MW5.0	DT0	UINT	0		

列の幅を拡張して確認します。

	A	B	C	D	E	F	G
1	フィールド 1	フィールド 2	フィールド 3	フィールド 4	フィールド 5	フィールド 6	フィールド 7
2	VAR_GLOBAL	g_uiTest	%MW5.0	DT0	UINT	0	

5. エクセルブックとして保存します。→PRO7export.xlsx

以降は、このファイルをタグ編集用元ファイルとして読み込むことができます。

②タグの編集・保存、xAscender プロジェクトのタグ編集画面でのインポート

1. タグデータの編集を行います。

ここでは部屋番号 1~5 の 3 つの温度データタグを下表のように設定します。

名前	PLC 上のアドレス	データ型
Room1_TempA	DT0	unsigned short
Room1_TempB	DT1	unsigned short
Room1_TempC	DT2	unsigned short
Room2_TempA	DT3	unsigned short
Room2_TempB	DT4	unsigned short
Room2_TempC	DT5	unsigned short
Room3_TempA	DT6	unsigned short
Room3_TempB	DT7	unsigned short
Room3_TempC	DT8	unsigned short
Room4_TempA	DT9	unsigned short
Room4_TempB	DT10	unsigned short
Room4_TempC	DT11	unsigned short
Room5_TempA	DT12	unsigned short
Room5_TempB	DT13	unsigned short
Room5_TempC	DT14	unsigned short

フィールド 2 の行 2, 3, 4 行にそれぞれ、Room1_TempA, Room1_TempB, Room1_TempC と入力し、3 つのセルを選択し、カーソルを一番下のセルの右下のフィルハンドルに合わせ、カーソルが+に変わったら下方向ヘドラッグします。

	A	B	C
1	フィールド 1	フィールド 2	フィールド 3
2	VAR_GLOBAL	Room1_TempA	%MW5.0
3		Room1_TempB	
4		Room1_TempC	

(注) 自動的にカウントアップされない場合は、複製後に表示されるオートフィルオプションボタンをクリックして、連続データを選択してください。

RoomN の N が自動的にカウントアップされて、各セルに複製されます。(注)

フィールド 4 も同様にして DT0 を選択して、カーソルをセル右下のフィルハンドルに合わせて下ヘドラッグします。

DTN の N が自動的にカウントアップされて、各セルに複製されます。(注)

その他のフィールドは同じ値を複製します。

	A	B	C	D	E	F	G
1	フィールド 1	フィールド 2	フィールド 3	フィールド 4	フィールド 5	フィールド 6	フィールド 7
2	VAR_GLOBAL	Room1_TempA	%MW5.0	DT0	UINT		0
3		Room1_TempB					
4		Room1_TempC					
5		Room2_TempA					
6		Room2_TempB					
7		Room2_TempC					
8		Room3_TempA					
9		Room3_TempB					
10		Room3_TempC					
11		Room4_TempA					
12		Room4_TempB					
13		Room4_TempC					
14		Room5_TempA					
15		Room5_TempB					
16		Room5_TempC					

xAscender プロジェクトにインポートする際に採用されるデータは、B 列のフィールド 2、D 列のフィールド 4、E 列のフィールド 5 のデータのみです。A 列フィールド 1、C 列フィールド 3、F 列フィールド 5 については、軽いチェックのみが行われているようです。

例えば、フィールド 3 を %QX0.2 で埋めても (Y2 のアドレス) インポート可能です。

よって、フィールド 3 の IEC アドレス部のデータは、先頭の設定値を複製するのみで OK です。

2. **コンマ区切りの csv ファイルとして保存**します。→TagdataWH.csv

各セルの設定が完了後、データを xAscender でインポート可能な csv ファイルとして保存します。

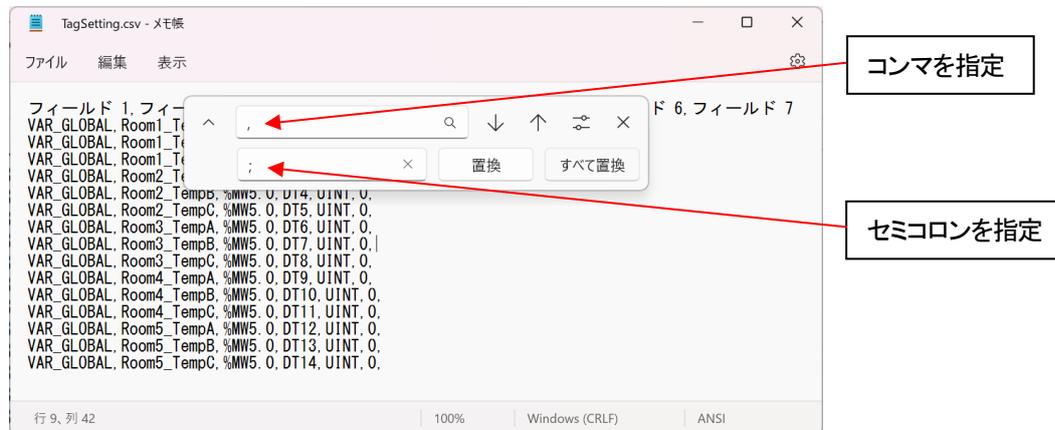
	A	B	C	D	E	F	G
1	フィールド 1	フィールド 2	フィールド 3	フィールド 4	フィールド 5	フィールド 6	フィールド 7
2	VAR_GLOBAL	Room1_TempA	%MW5.0	DT0	UINT		0
3	VAR_GLOBAL	Room1_TempB	%MW5.0	DT1	UINT		0
4	VAR_GLOBAL	Room1_TempC	%MW5.0	DT2	UINT		0
5	VAR_GLOBAL	Room2_TempA	%MW5.0	DT3	UINT		0
6	VAR_GLOBAL	Room2_TempB	%MW5.0	DT4	UINT		0
7	VAR_GLOBAL	Room2_TempC	%MW5.0	DT5	UINT		0
8	VAR_GLOBAL	Room3_TempA	%MW5.0	DT6	UINT		0
9	VAR_GLOBAL	Room3_TempB	%MW5.0	DT7	UINT		0
10	VAR_GLOBAL	Room3_TempC	%MW5.0	DT8	UINT		0
11	VAR_GLOBAL	Room4_TempA	%MW5.0	DT9	UINT		0
12	VAR_GLOBAL	Room4_TempB	%MW5.0	DT10	UINT		0
13	VAR_GLOBAL	Room4_TempC	%MW5.0	DT11	UINT		0
14	VAR_GLOBAL	Room5_TempA	%MW5.0	DT12	UINT		0
15	VAR_GLOBAL	Room5_TempB	%MW5.0	DT13	UINT		0
16	VAR_GLOBAL	Room5_TempC	%MW5.0	DT14	UINT		0

続けて、名前を変えてエクセルブック形式でも保存しておいてください。

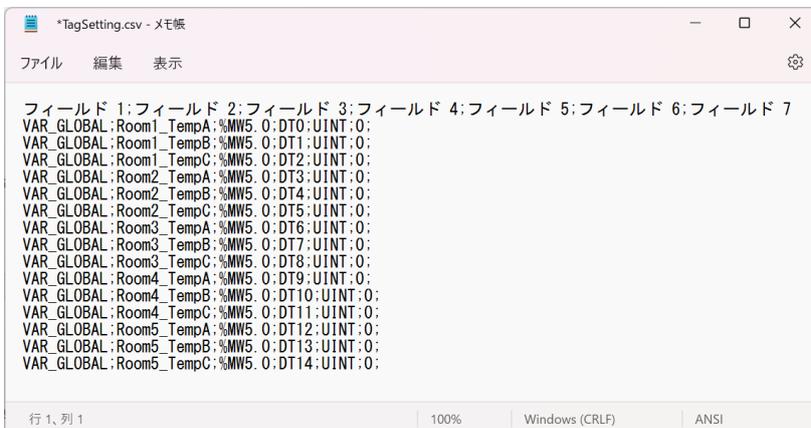
このデータを修正する際には、エクセルブック形式のファイルを読み出す方が簡単です。

3. csv ファイルをテキストエディタで開いて、**コンマをセミコロンに全て変更**し、保存します。(下図はメモ帳の例)

エクセルで保存できる csv ファイルは、コンマ区切りです。Pro7 でエクスポートした csv ファイルは、セミコロン区切りなので、保存した csv ファイルをメモ帳で開いて、コンマをセミコロンに全て置換します。

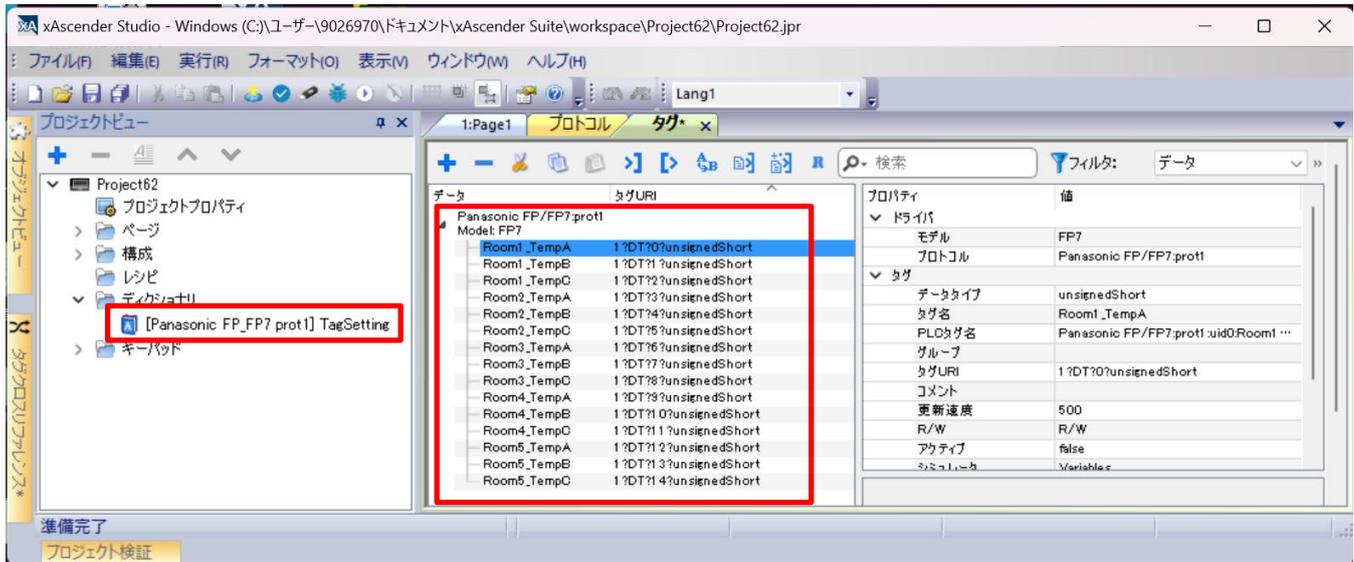


コンマがすべてセミコロンに修正されました。



ファイル→保存でファイルを上書き保存します。

4. xAscender プロジェクトのタグエディタでインポートできることを確認してください。



3 - 部品の設定

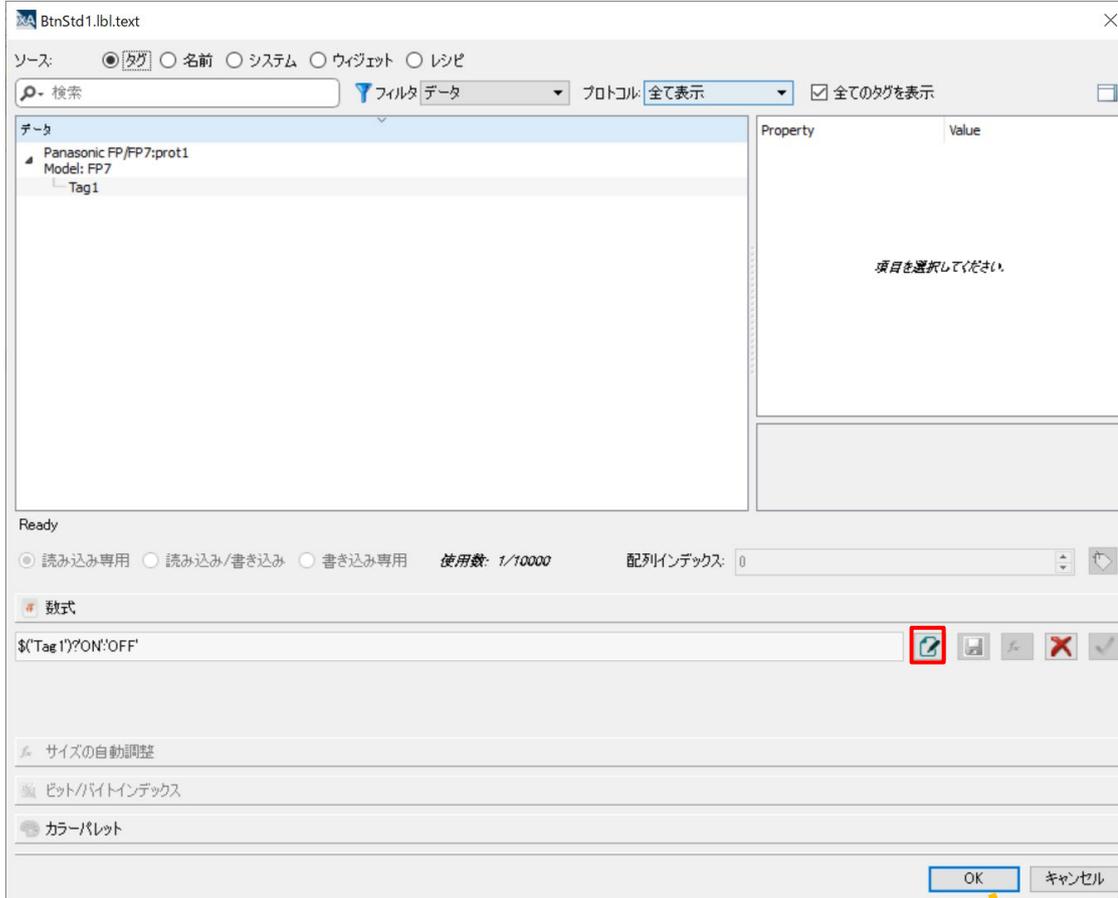
「ボタン」部品の ON/OFF による表示文字の切り替え

「ボタン」部品の ON/OFF による表示文字の切り替え方法について説明します。

スイッチボタンを一つ配置し、「プロパティ」-「値」でタグを登録します。

ラベル右の  をクリックし、下記ダイアログを表示します。

※今回はスイッチボタンなので、Tag1 のデータタイプは Boolean で登録しています。



赤枠の「数式を挿入」をクリックし、数式部分に「\$('Tag1')?'ON':'OFF」と入力すると、ON の時「ON」と表示され、OFF の時は「OFF」と表示されます。

数式部分の拡大

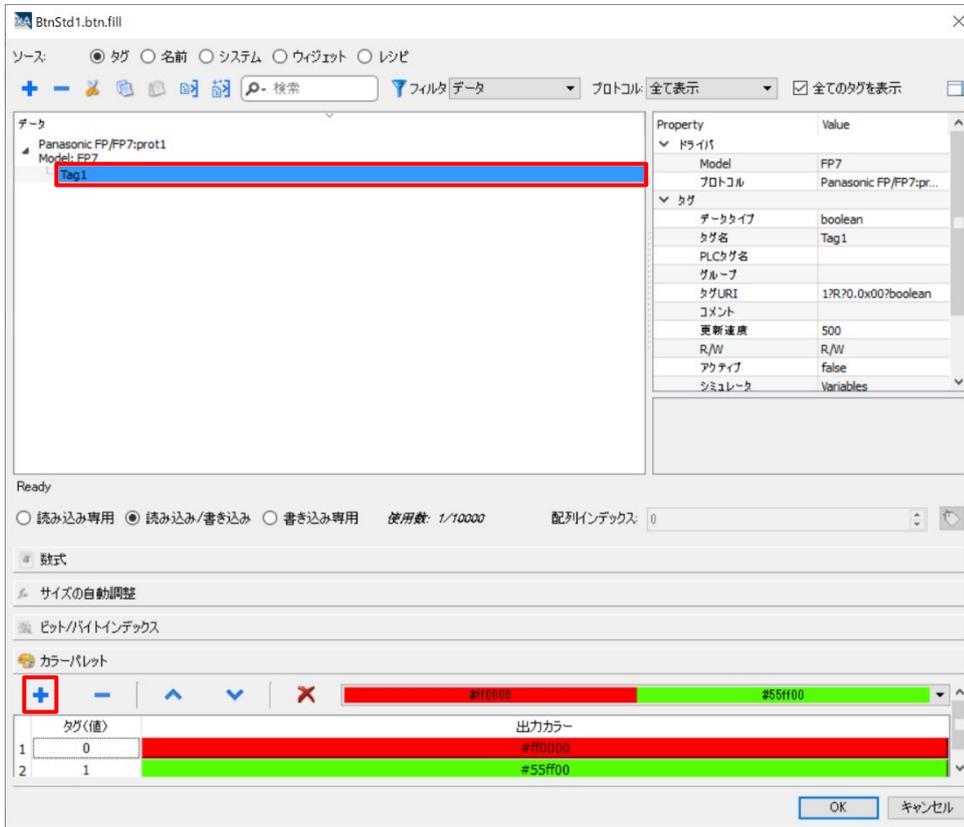
\$('Tag1')?'ON':'OFF'



ON/OFF で塗りつぶしの色を変更する方法

「プロパティ」-「塗りつぶしの色」右端の  をクリックして、下記ダイアログが表示します。

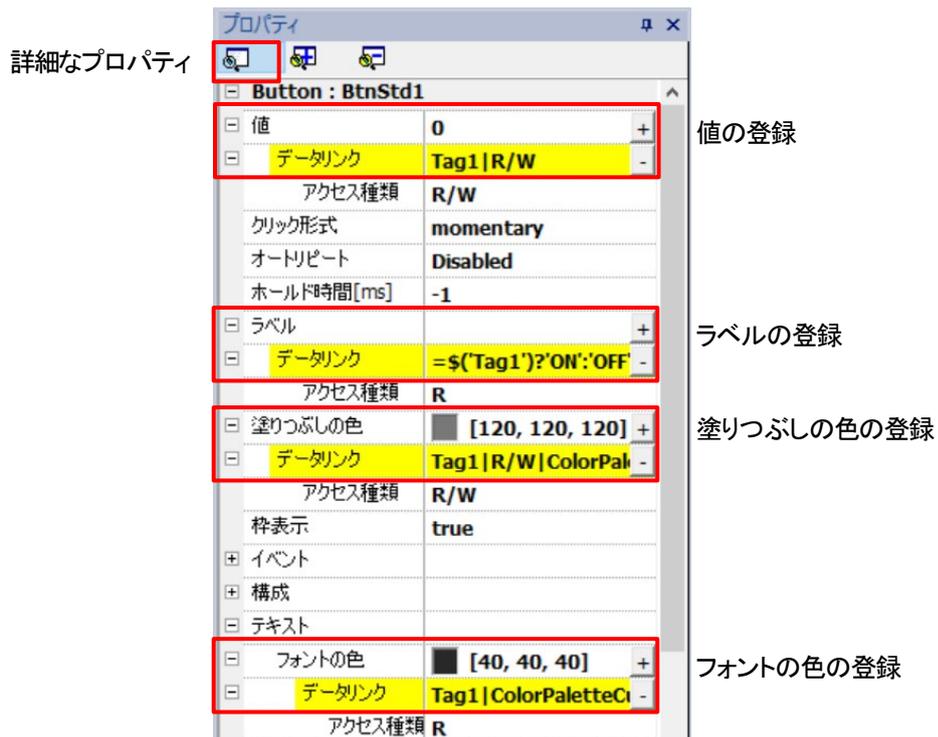
「Tag1」をクリックし、「カラーパレット」の  をクリックすることで、カラーパレットが追加されます。



ON/OFF で文字色を変更する方法

「プロパティ」-「詳細なプロパティ」を表示します。

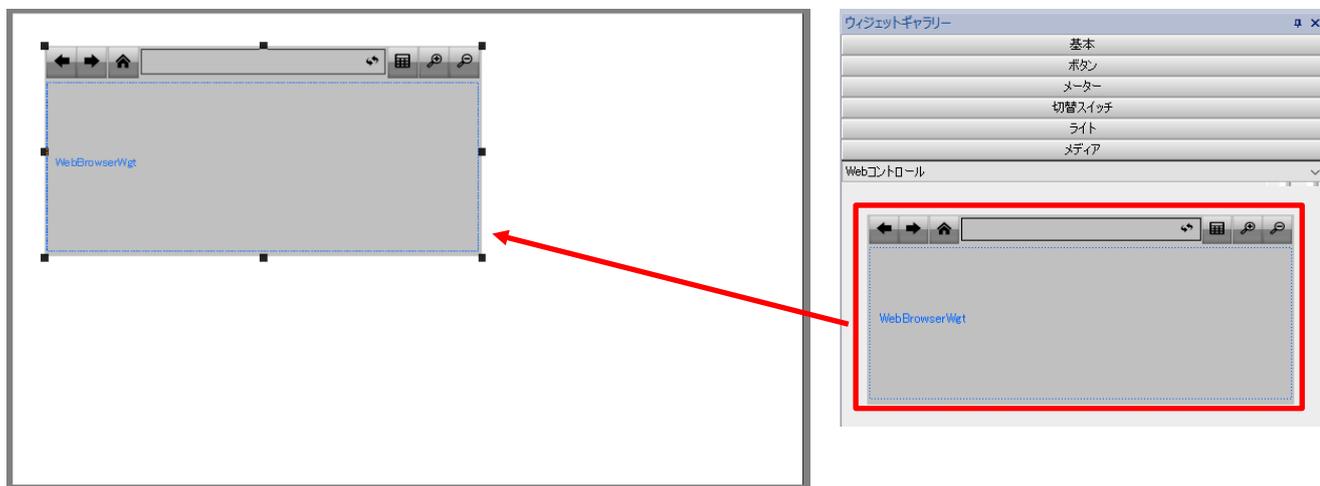
「プロパティ」-「テキスト」-「フォントの色」をクリックし、上記と同様に「Tag1」をクリック後、カラーパレットにて色を登録します。



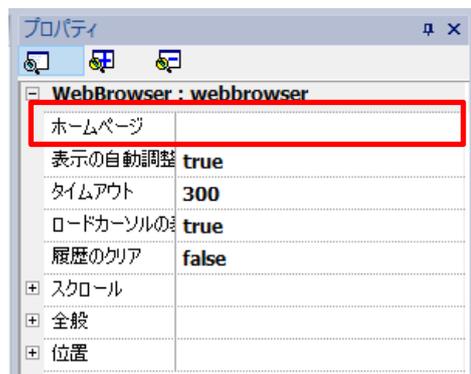
Web コントロール部品

Web コントロール部品を使用することで、WH 画面上に Web 画面を表示することができます。

「ウィジェットギャラリー」-「メディア」から「Web コントロール」を選択し、Web コントロール部品を作画画面に配置します。



「プロパティ」-「ホームページ」に入力したアドレスが、Web 画面のホーム画面の URL となります。



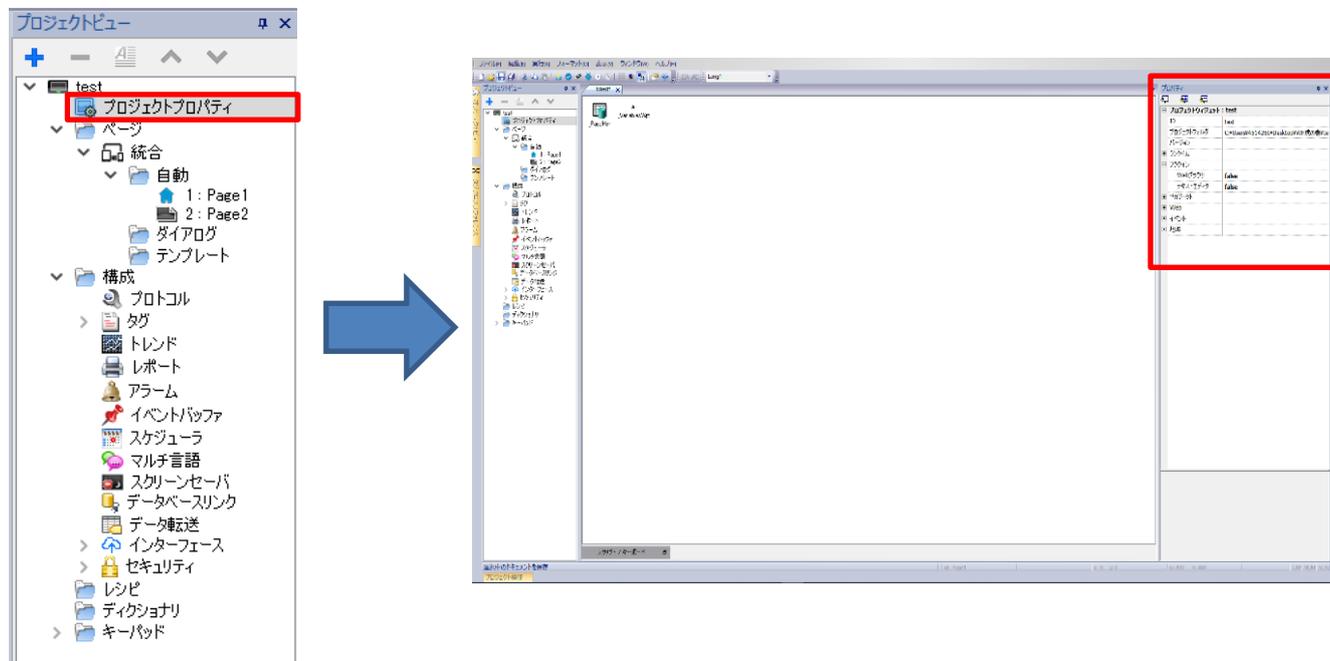
登録した Web コントロール部品は、下図の様に表示されます。



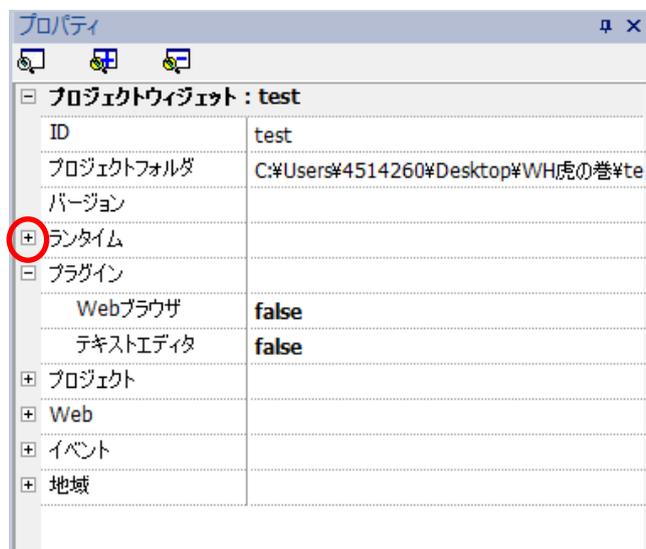
プラグインのインストール方法

WHシリーズでWebブラウザ、テキストエディタの機能を使用するには、プラグインのインストールが必要です。

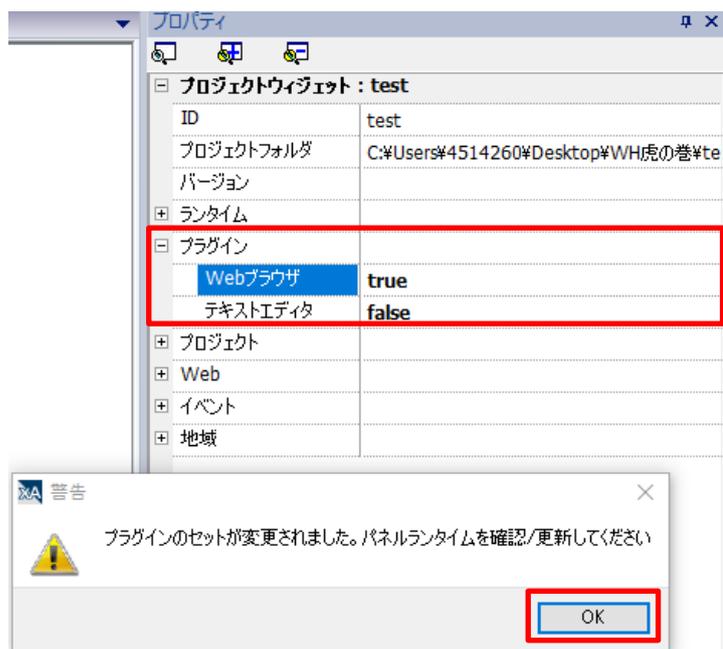
プロジェクトビューの「プロジェクトプロパティ」をダブルクリックすると、画面右側に「プロパティ」が出現します。



「プラグイン」の ⊕ をクリックします。



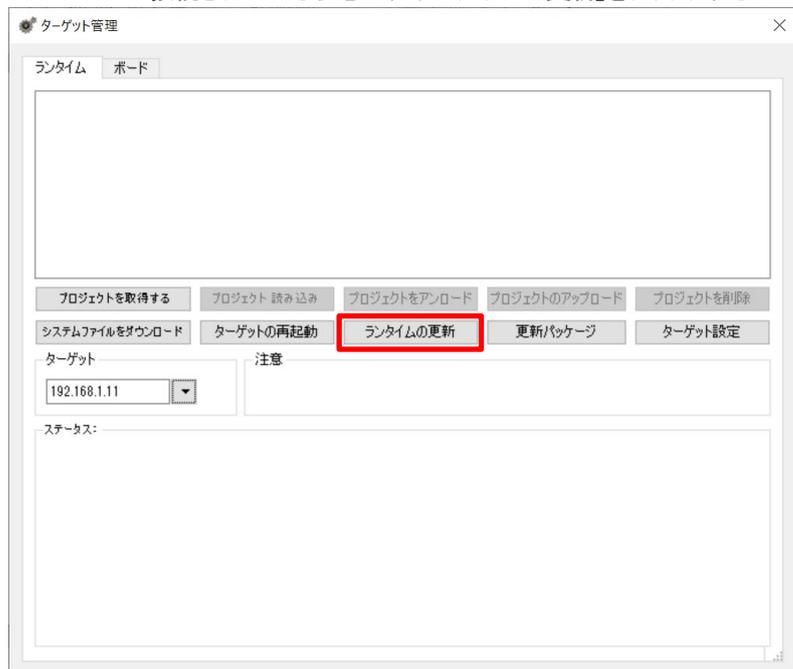
使用したい機能を true に変更し、ダイアログの「OK」をクリックします。



次に、メニューバーの「実行」-「デバイス管理」を選択します。

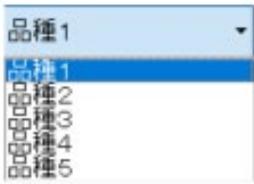


WH と PC が接続されている状態で、「ランタイムの更新」をクリックするとプラグインのインストールが開始されます。



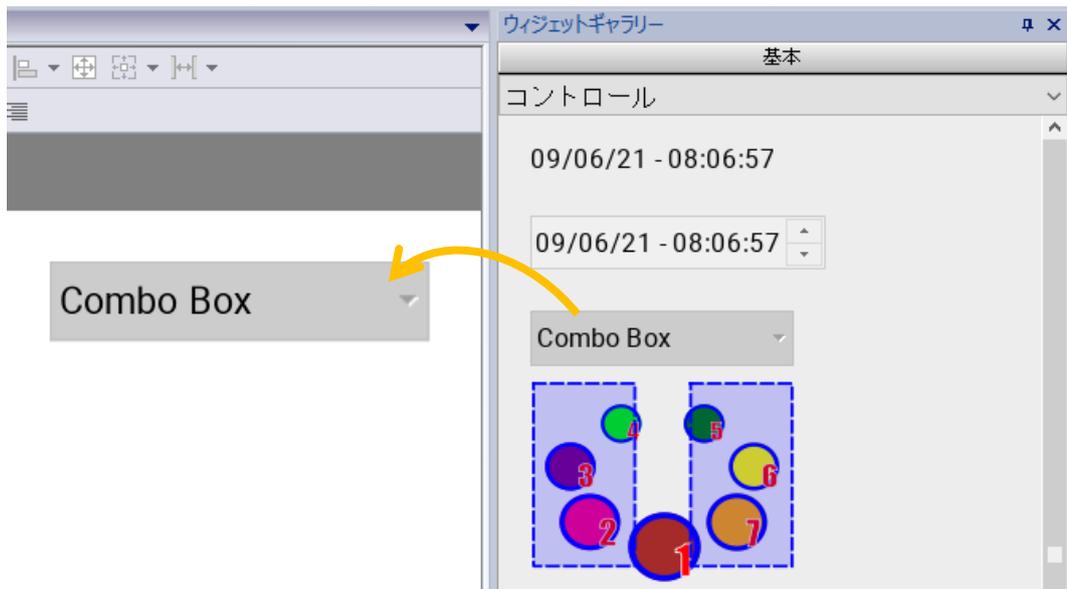
Combo Box

「Combo Box」を使用することで、プルダウンリストからの項目選択、値の変更をすることができます。



部品の配置と設定

「ウィジェットギャラリー」→「基本」→「コントロール」→「Combo Box」



プロパティ

「リスト」→「+」→「メッセージを追加/削除」

コンボボックス

インデックス	文字列リスト
0	品種0
1	品種1
2	品種2
3	品種3
4	品種4

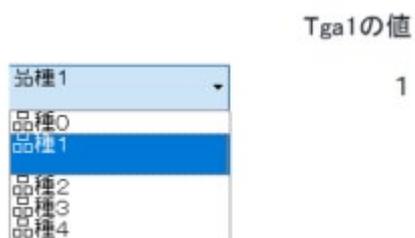
左図のようにコメントを登録します

+ をクリックして行を追加、
「文字列リスト」にコメントを入力します

プロパティ内 → インデックスに「Tag1(Short)」を割り付けます。



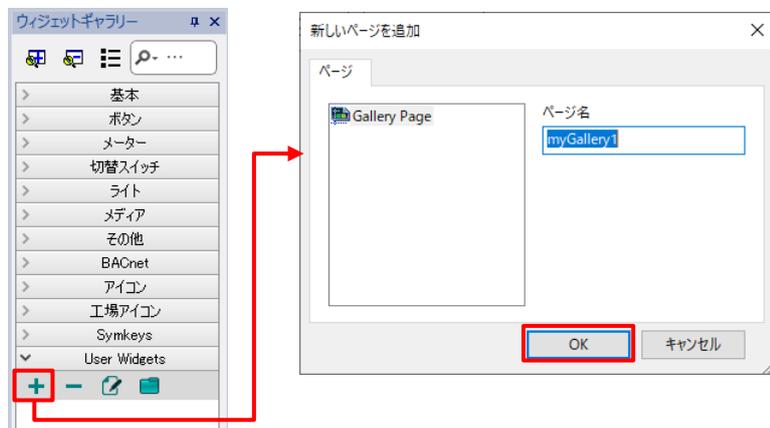
Combo Box のプルダウンリストから「品種」を切り替えることで、Tag1 の値を変更することができます。



User Widgets の作成

User Widgets を作成できます。

「ウィジェットギャラリー」から「User Widgets」を開き、**+** をクリックします。



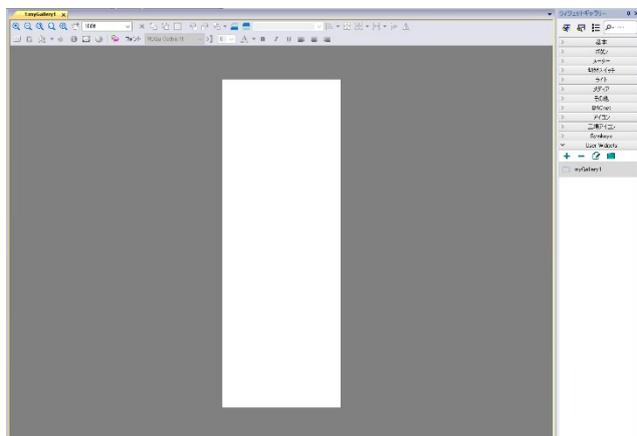
「新しいページ」のダイアログが開きます。「ページ名」から自由に名前入力ができます。

ここではデフォルトの「myGallery1」を使用します。

「OK」をクリックします。

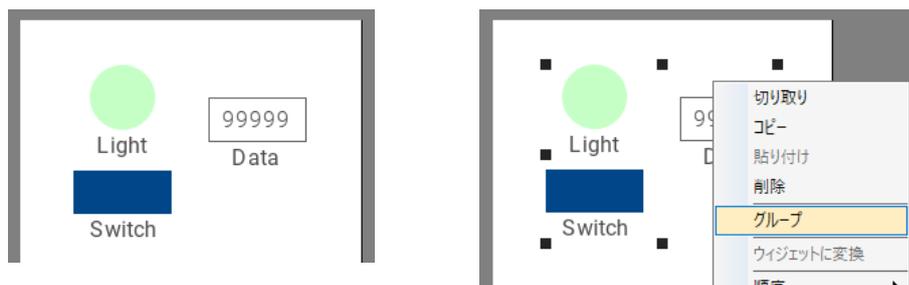
ここでの「ページ」はプロジェクト内で User Widgets を保存しておくフォルダに相当します。

編集画面が立ち上がるので、編集をします。



下例では 3 種類のウィジェットとラベルを設置しています。

ひとまとまりの部品として扱いたい場合は、グループ化を行ってください。

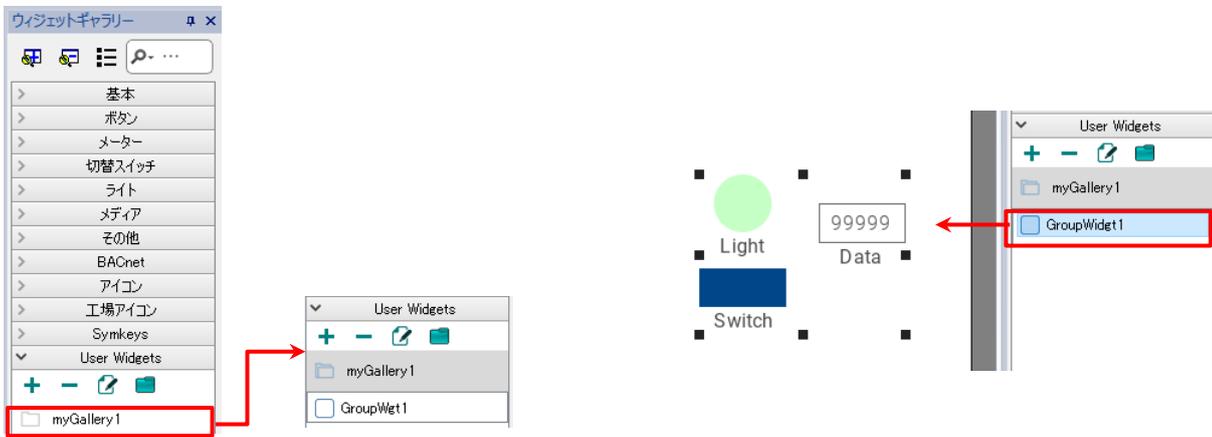


部品の名称を変更する場合は、「プロパティ」-「全般」の「ID」の変更を行います。

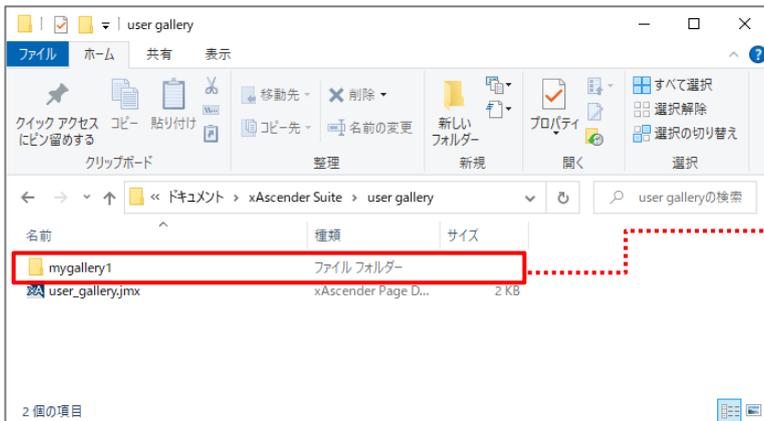


プロジェクトを保存することで作成した部品も保存されます。

「User Widgets」にフォルダ「myGallery1」があり、クリックすると作成した部品「GroupWgt1」が表示されます。「GroupWgt1」をクリックし、編集画面にドラッグ & ドロップすることで、プロジェクト内へ設置できます。



作成した部品は下図の様に、フォルダファイルとして保存されます。(本例ではフォルダ「myGallery1」)
格納先はデフォルトではパソコンの「Documents」-「xAscender Suite」-「user gallery」直下となっています。
このファイルフォルダを別パソコンにコピーすることで、作成した部品をコピー先のパソコンでも使用することができます。

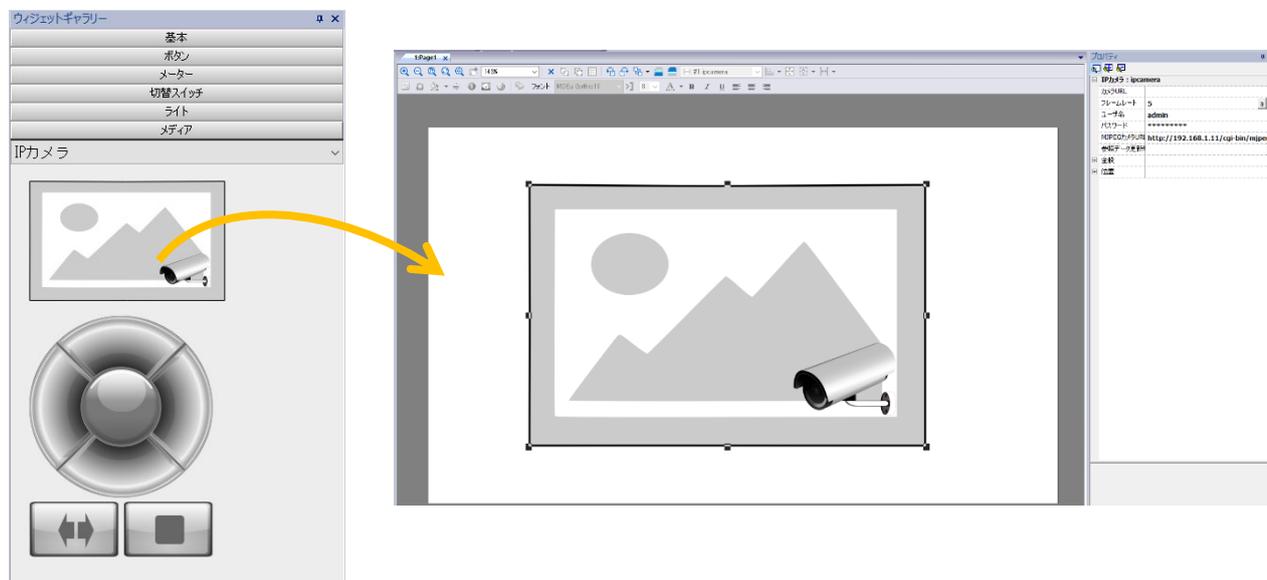


別 PC にコピーして使用可能

IP カメラ

WH 上にネットワークカメラの映像を表示させる方法、および IP カメラウィジェットの使用方法について説明します。

「ウィジェットギャラリー」→「メディア」→「IP カメラ」を選択し、編集画面に追加します。



プロパティ



カメラ URL:
JPEG 形式のカメラを使用する場合の URL

フレームレート:
1 秒間に表示する JPEG 画像の枚数

ユーザ名 & パスワード:
カメラにアクセスする際のユーザ名 & パスワード
(カメラ側で設定している場合のみ設定します)

MJPEG カメラ URL:
JPEG 形式のカメラを使用する場合の URL

※カメラ URL および MJPEG カメラ URL は xAscender Studio ヘルプ内「IP カメラウィジェット」を確認してください。
※MJPEG カメラ URL を使用する際、「カメラ URL」、「フレームレート」は無視されます。

上記の設定を行い、
IP カメラと WH を接続し、プロジェクトをダウンロードすることで、
WH 本体に IP カメラの映像を表示することができます。

アイコン付きボタンのアイコン変更方法 1 (ファイル名を複写)

アイコン付きボタンのアイコンを変更する方法の説明です。

アイコン付きボタンのアイコンをデフォルトから変更したい場合、ウィジェットギャラリーのアイコンタブには、多数のジャンルのアイコンが格納されているので、この中の任意アイコンに入れ替えて使用出来たら便利です。

円形(青)ボタン[アイコン付き]を画面に配置して、アイコンを入れ替えてみましょう。



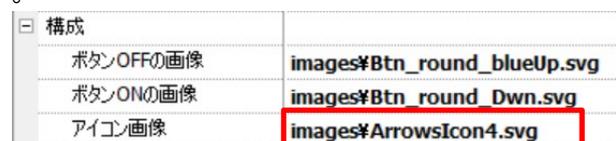
ここでは、ウィジェットギャラリーの[アイコン]タブ→[矢印]→[Arrows4]に入れ替えます。
Arrows4 を画面にドラッグしてプロパティを確認します。



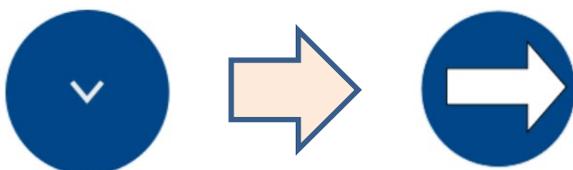
右図は、円形(青)ボタン[アイコン付き]の構成プロパティです。
アイコン画像のファイル名が表示されています。



先ほどドラッグした Arrows4 のプロパティ→参照画像のファイル名をコピーして
円形(青)ボタン[アイコン付き]の構成→アイコン画像に貼り付けます。
これで、アイコンの変更が完了しました。



アイコン画像のプロパティ変更と同時に円形(青)ボタン[アイコン付き]のアイコンが変更されます。



最後に不要であれば、Arrows4 のアイコンは、削除してください。

アイコン付きボタンのアイコン変更方法 2（ファイル一覧より選択）

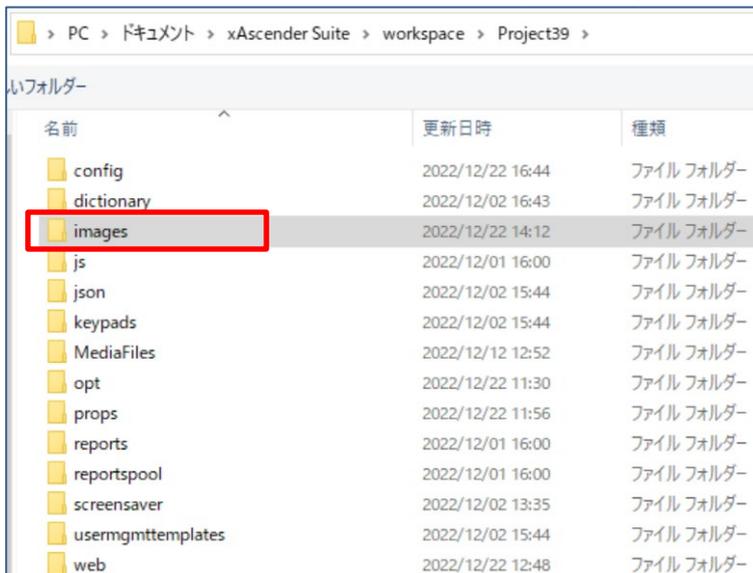
アイコン付きボタンのアイコンを変更する方法の説明です。
アイコン付きボタンのアイコンは、デフォルトのままでは使用しにくいです。

前ページで使用したアイコンは、プロジェクトの Images フォルダに格納されています。
他のアイコン付きボタン等で再利用する際は、構成→アイコン画像プロパティの右端の参照ボタンをクリックしてプロジェクト内の images フォルダを開いてファイルを選択することも可能です。

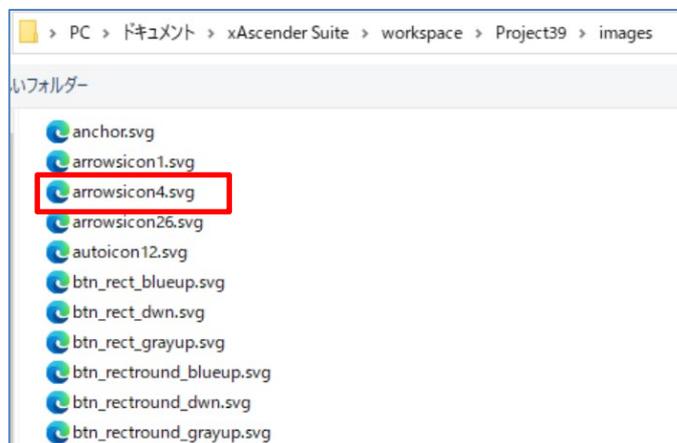
- ① アイコン入れ替え対象ウィジェットの参照ボタンをクリックします。



- ② プロジェクトのフォルダ xAscenderSuite→workspace→Project フォルダ→images フォルダを開きます。



- ③ アイコンファイルを選択します。ここでは、arrowsicon4.svg を選択しています。



エクスプローラに SVG ファイルのサムネイルを表示させる方法

前ページのアイコン付きボタンのアイコンを変更する方法 2 では、images フォルダに格納された SVG ファイルを選択する方法を説明しました。Windows のデフォルトのままでは、全てのアイコンファイルに同じアイコンが表示されており、ファイル名で識別するか、実際にファイルを開いてみて確認する必要があります。

ここでは、効率が悪いので、ファイルの一覧表示にアイコンのサムネイルを表示する方法を説明します。

svg-explorer-extension というフリーソフトのインストールが必要です。(ファイル名は、**svg_see_x64.exe**)

このソフトを使用すると、エクスプローラの機能を拡張してファイルを一覧する際に SVG ファイルのサムネイルを表示することが可能になるので、SVG ファイルの内容が簡単に確認でき選択できるようになります。

(注)フリーソフトの社内 PC へのインストールは、**フリーソフト導入申請**が必要です。

ダウンロード先 <https://github.com/tibold/svg-explorer-extension/releases>

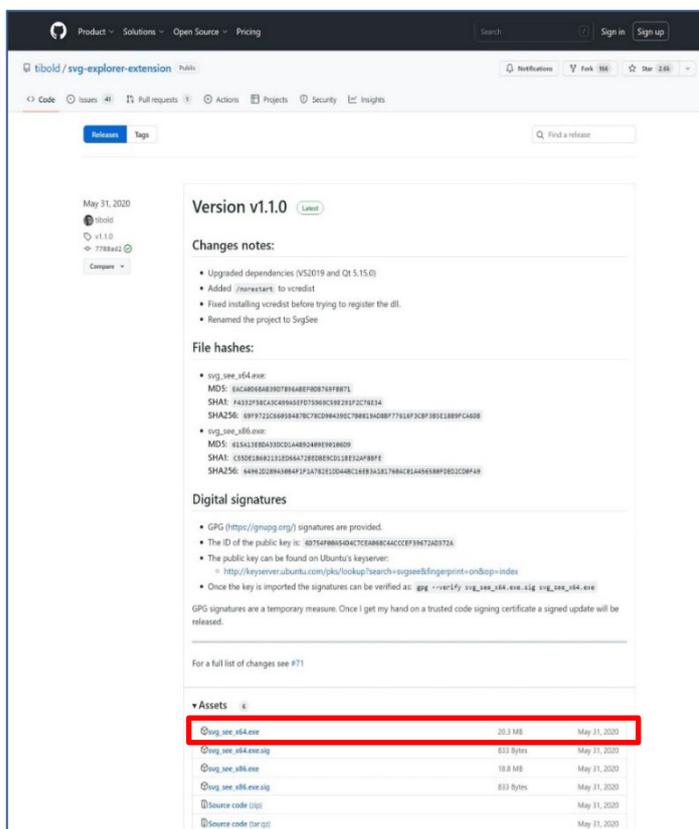
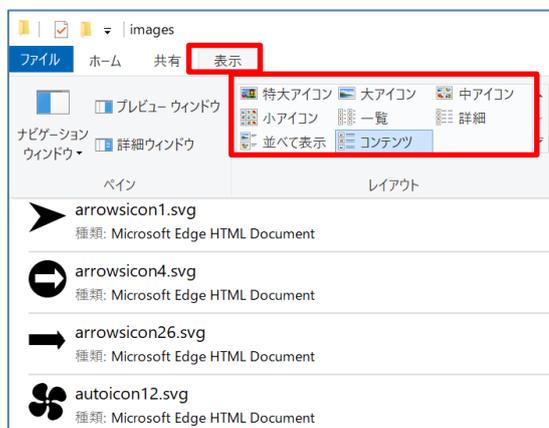
ダウンロードするファイル名 **svg_see_x64.exe**

Assets の先頭にあるファイル **svg_see_x64.exe** (右下図赤枠) をダウンロードして、インストールすると、インストール完了後から SVG のサムネイルがエクスプローラで表示されるようになります。

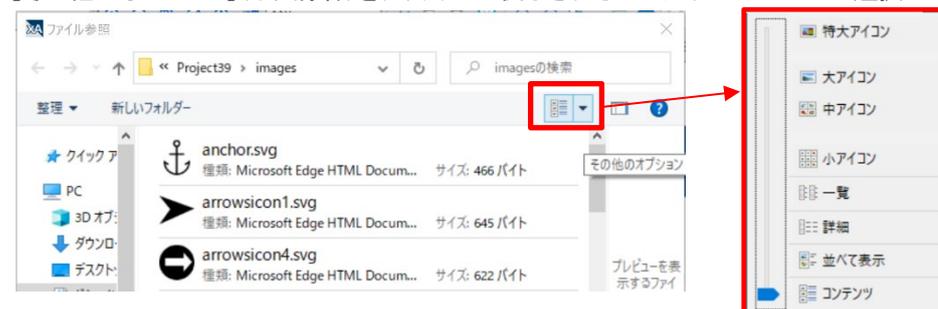
エクスプローラの表示モードは以下のいずれかを指定してください。

特大アイコン・大アイコン
中アイコン・並べて表示・コンテンツ

エクスプローラでの表示モード選択例
表示メニューを開いて設定します(下図赤枠)



表示メニューが表示されない場合は、**[その他のオプション]** (下図赤枠) をクリックして表示されるコンテキストメニューで選択してください。



メディアウィジェット表示時に自動再生させない設定

メディア部品を配置したページを表示させると自動的にビデオの再生が始まります。
自動的にビデオ再生を始めさせたくない場合の方法について説明します。

方法:

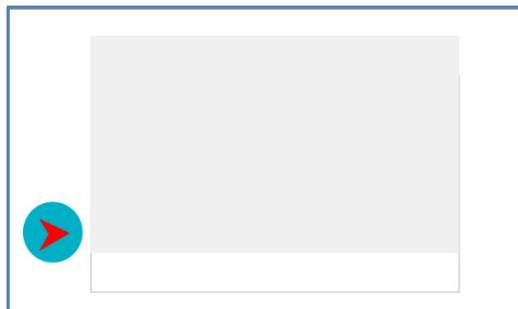
再生開始・停止用のボタンとメディア部品を配置します。

ボタンの値プロパティは、

boolean 型の内部変数 Tag1 とリンクしておきます。

クリック形式は、maintained に設定します。

値	1
データリンク	Tag1 R/W
アクセス種類	R/W
クリック形式	maintained



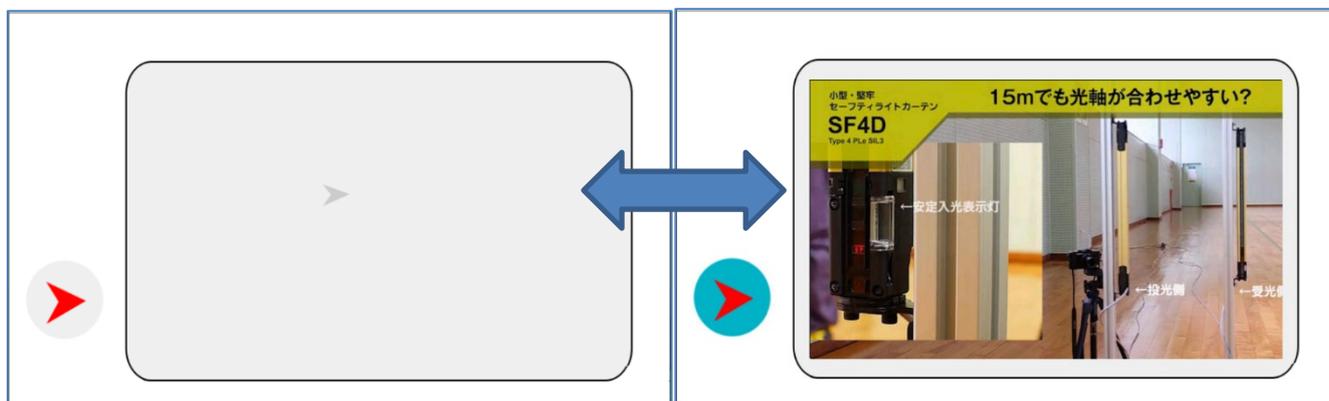
メディア部品のプロパティ

全般→表示を false に設定

上記のプロパティを boolean 型の内部変数 Tag1 とリンクしておきます。

表示	false
データリンク	Tag1

この設定によれば、メディア部品を配置したページを表示させても、最初は表示が無効になっているので再生はされません。
再生開始・停止用のボタンをクリックすると、リンクされた表示プロパティが TRUE になって再生を始めることができます。

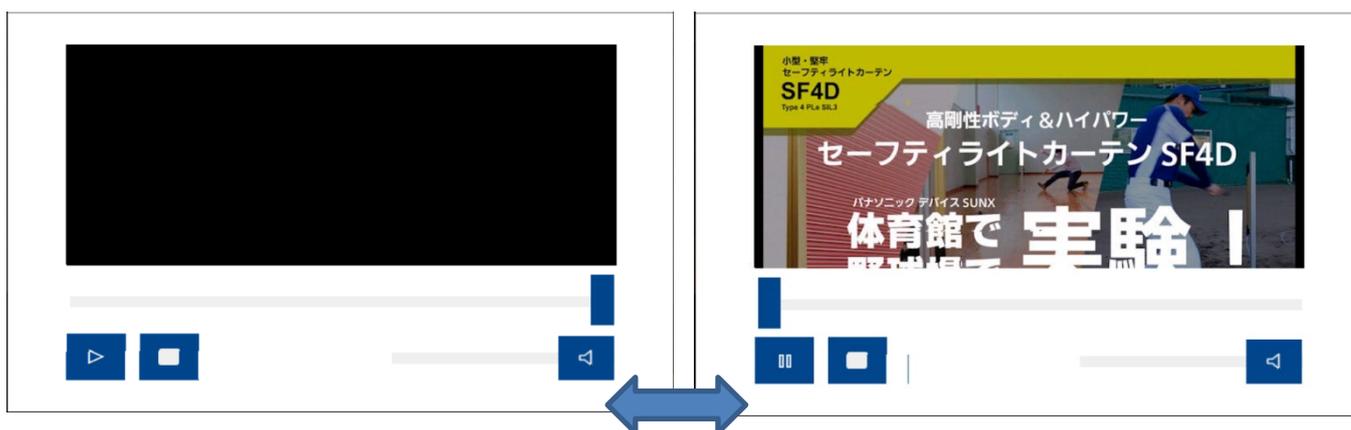
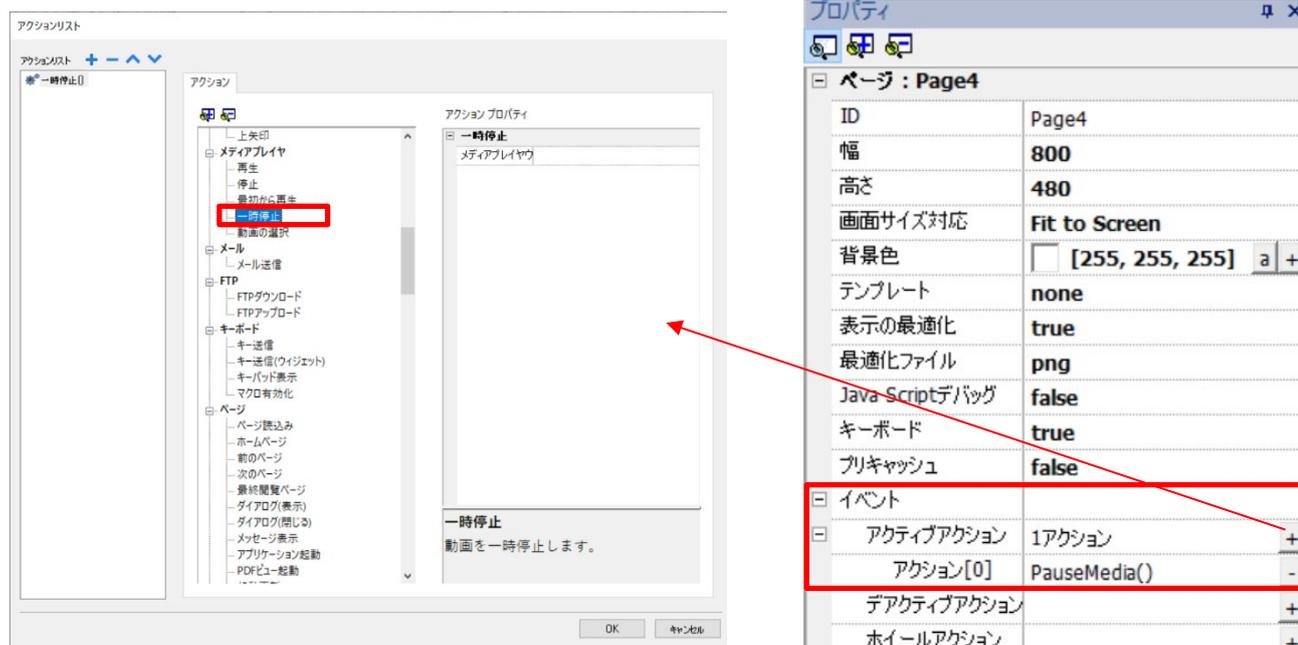


メディアプレイヤーウィジェット表示時に自動再生させない設定

メディアプレイヤーウィジェットの場合もメディアウィジェットと同様に、ページ表示と同時に自動的にビデオ再生が始まります。メディアプレイヤー部品の場合もメディア部品と同様の方法も可能ですが、メディアプレイヤーには、再生ボタンなどが最初から配置されていますので、再生開始停止ボタンを作成するのは、見栄えが良くありません。

メディアプレイヤー部品で、ページ表示時に再生を停止した状態で始める方法を説明します。

ページのアクティブアクションで、**メディアプレイヤー一時停止** アクション(下図)を設定することによって停止状態で始めることが可能です。



ラジオボタン作成方法 1 (PLC との通信タグを使用する場合)

ラジオボタンは、排他的な項目を並べて、1つの項目のみを選んでほしいときに、使われる入力形式です。

ラジオボタンは、項目の先頭に丸印が表示され、マウスでクリックするなどして選択します。

同時に選択状態にできるのは1つの項目のみで、ある項目を選択すると、他の項目は未選択状態に変わります。

ラジオボタンという呼び名は、ラジオの機械的な押しボタンに、仕組みが似ていることから名付けられました。

ラジオボタンの具体例

下記から好きなものを1つ選択してください。

- りんご
- バナナ
- キウイ
- パイナップル

WH でラジオボタンを構成する方法

ここでは、PLC との通信タグを使用して上記具体例のラジオボタンを作成します。

上記のくだもの選択画面を作成してください。○はすべてボタンで作成し、文字はラベルで作成します。

PLC との通信タグを以下のように設定します。その後、各○ボタンの値プロパティに bool 型のタグを順にリンクします。

変数名	データ型	PLC アドレス
g_bApple	boolean	R0
g_bBanana	boolean	R1
g_bKiwii	boolean	R2
g_bPinapple	boolean	R3
g_wSelection	unsignedShort	WR0

最後に各ボタンのクリックイベントに g_wSelection タグへのデータ書き込みアクションを登録します。

各ボタンでの書き込み値は、下記のように設定してください。

りんご	1	(R0 のみ 1 でその他は 0 のデータ)
バナナ	2	(R1 のみ 1 でその他は 0 のデータ)
キウイ	4	(R2 のみ 1 でその他は 0 のデータ)
パイナップル	8	(R3 のみ 1 でその他は 0 のデータ)

各ボタンクリック時の動作説明

この設定によると、各ボタンクリック時には、値プロパティにリンクした PLC のビットアドレスを書き込みし

さらにクリックアクションで、各ビットが含まれるワードアドレスにワードデータが書き込まれます。(書き込み順序不明)

このワードデータの内容が対応する果物のビットのみ 1、かつ他の果物のビットは 0 になっているため

クリックした果物のボタンのみが選択された状態で保持されます。

これで、ラジオボタンが作成できました。

ラジオボタン作成方法 2 (内部変数タグを使用する場合)

ここでは、内部変数タグを使用したラジオボタンの設定方法を説明します。

内部変数タグには同じメモリアドレスに bool 型と unsignedShort 型(16ビット)の変数を割り当てることができないので unsignedShort 型(16ビット)の変数に対して設定を行います。

PLC の unsignedShort 型通信タグを使用する場合も同様の設定になります。

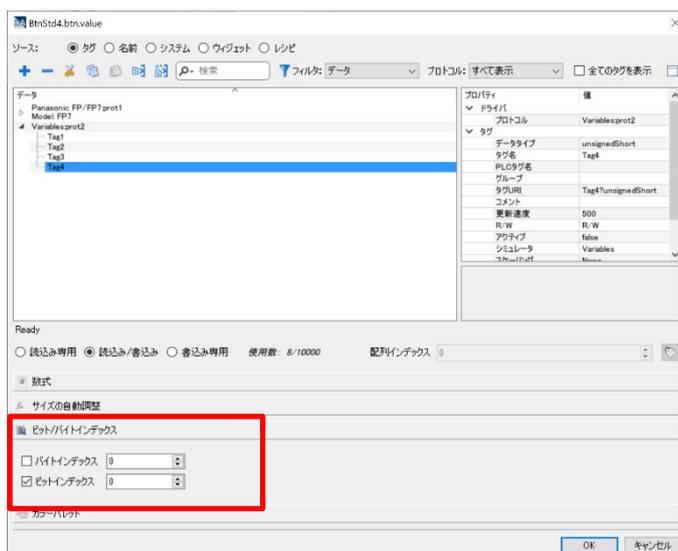
unsignedShort 型の変数内のビットデータに対してボタンの値(bool 型)をリンクする際の指定方法の説明となります。

タグのリンク設定画面で、ビット/バイトインデックスを

下図のように指定します。

つまり、ビットインデックスをチェックして

ビットインデックス番号を設定します。(最下位ビットは 0)



- | | | |
|------------------------------|-------------------|-------------|
| <input type="radio"/> りんご | に対しては、ビットインデックス 0 | |
| <input type="radio"/> バナナ | に対しては、1ビットインデックス | |
| <input type="radio"/> キウイ | に対しては、ビットインデックス 2 | |
| <input type="radio"/> パイナップル | に対しては、ビットインデックス 3 | をそれぞれ指定します。 |

最後に各ボタンのクリックイベントに Tag4 タグへのデータ書き込みアクションを登録します。

各ボタンでの書き込み値は、下記のように設定してください。

この設定は、前ページの設定と同じです。

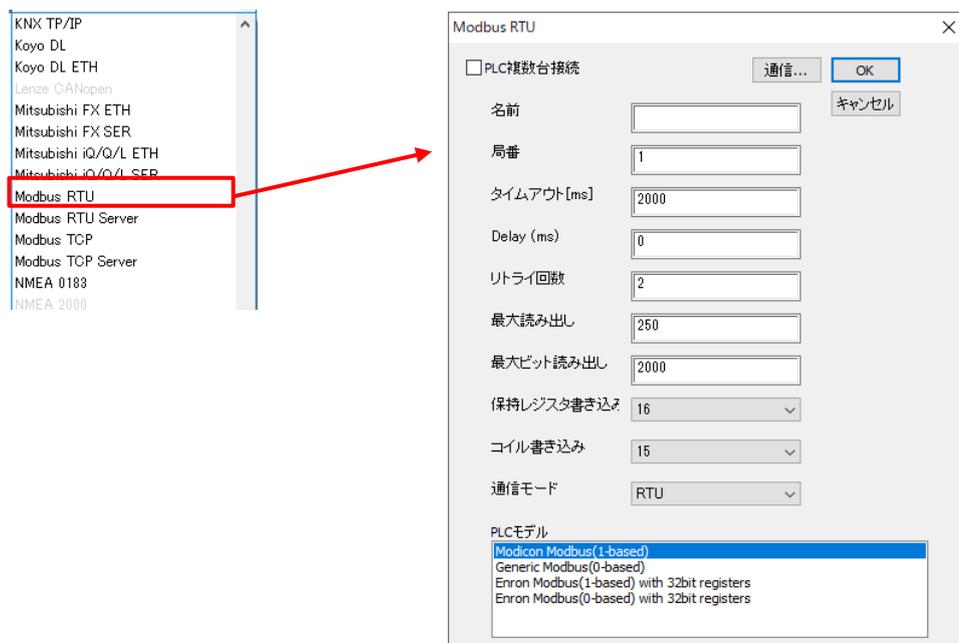
- | | | |
|--------|---|--------------------------|
| りんご | 1 | (R0 のみ 1 でその他は 0 のデータ) |
| バナナ | 2 | (R1 のみ 1 でその他は 0 のデータ) |
| キウイ | 4 | (R2 のみ 1 でその他は 0 のデータ) |
| パイナップル | 8 | (R3 のみ 1 でその他は 0 のデータ) |

4 - 通信設定

Modbus 通信時の設定

PLC 等の外部接続機器と、シリアル通信で接続をする場合の設定方法を説明します。

1. 「プロジェクトビュー」-「構成」-「プロトコル」-「PLC」欄のドロップダウンリストから Modbus 対応のプロトコルを選択し設定することができます。
ここでは例として「Modbus RTU」を使用する場合の設定を説明します。



名前	ネットワーク構成中のノードを識別する名前。
局番	スレーブデバイスの Modbus ノード。
タイムアウト[ms]	サーバーデバイスからの応答がない場合の、2 つの再試行間のミリ秒単位による時間の遅延。
遅延[ms]	前回受信されたフレームの最後と新しいリクエストの開始の間における時間の遅延 (ミリ秒単位)。
リトライ回数	通信エラー状態を報告する前に、コントローラへ特定のメッセージが送られる回数。
最大読み出し	データブロックリクエストの最大長 (バイト単位)。
最大ビット読み出し	ブロックリクエストの最大長 (ビット単位)。

保持レジスタ書き込み	保持レジスタへの書き込み時に使用するファンクションコードに、No.06 / No.16 から選択します。
コイル書き込み	入力コイルへの書き込み時に使用するファンクションコードに、No.05 / No.15 から選択します。
通信モード	RTU モード / ASCII モードを選択します。

PLC モデル

Modicon Modbus (1 ベース):	標準的な Modbus のプロトコルで、かつ、アドレス指定時にオフセット値として1が付与されます。
Generic Modbus (0 ベース):	標準的な Modbus のプロトコルです。
Enron Modbus (1 ベース):	32ビットレジスタに対応した Enron Modbus に対応したプロトコルで、かつアドレス指定時にオフセット値として1が付与されます。
Enron Modbus (0 ベース):	32ビット登録メモリ領域を持つ Generic Modbus の実装を拡張します。

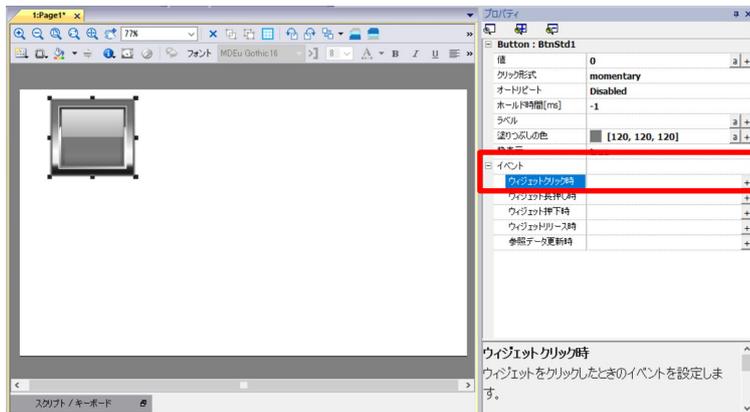
FTP クライアント機能の使用方法

ここでは、FTP クライアント機能の使用方法について説明します。

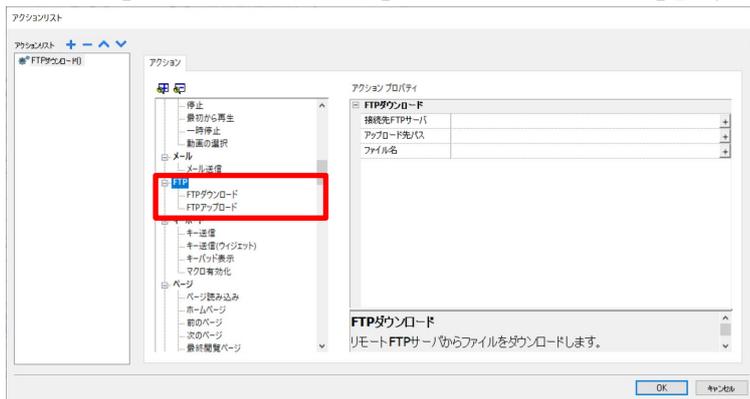
FTP クライアントを使用する場合は、各製品の「イベント」内で設定を行います。

※本記事では「ボタン」部品、「ウィジェットクリック時」での例を説明します。

「ボタン」部品を一つ追加し、「プロパティ」-「イベント」-「ウィジェットクリック時」右側の **+** をクリックします。



「アクション」内の「FTP ダウンロード」または「FTP アップロード」を選択します。



FTP ダウンロード(FTP サーバからファイルをダウンロード)

FTPダウンロード	
接続先FTPサーバ	+
アップロード先パス	+
ファイル名	+

FTP サーバの設定
ダウンロードするサーバ側のファイルパスを指定
保存先のファイルパスを指定

FTP アップロード(FTP サーバへファイルをアップロード)

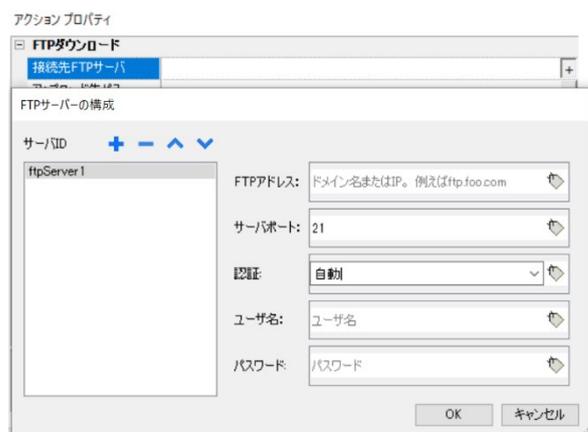
FTPアップロード	
接続先FTPサーバ	+
ファイル名	+
アップロード先パス	+

FTP サーバの設定
FTP サーバへアップロードするファイルパスを指定
FTP サーバ上の保存先パスを指定

※ファイル名およびアップロード先パスは右端の **+** からタグを紐づけることが可能です。

接続先 FTP サーバの設定

「接続先 FTP サーバ」の右端  をクリックすると詳細設定を行うダイアログが表示されます。



「接続先 FTP サーバ」右側の  をクリックし、FTP サーバの設定を行います。

FTP アドレス :FTP サーバ側の IP アドレスを指定

サーバポート :FTP サーバ側のポート番号を指定
(デフォルト:21)

認証 :自動/匿名(Anonymous)

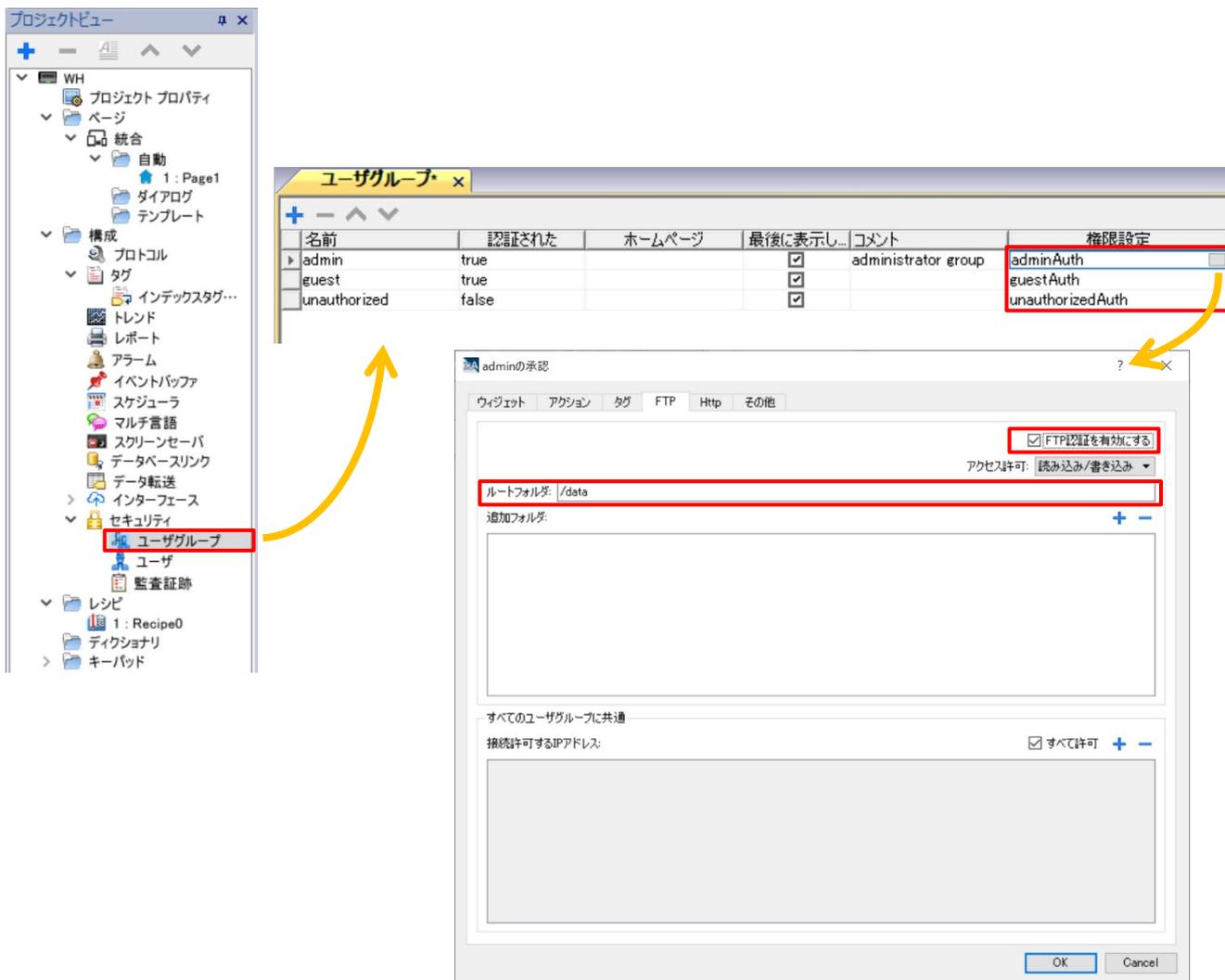
ユーザ名 :FTP サーバ側で設定したユーザ名

パスワード :FTP サーバ側で設定したパスワード

FTP サーバの有効化

xAscender Studio での FTP サーバ有効化方法

プロジェクトビューの「設定」-「セキュリティ」-「ユーザグループ」をダブルクリックし、「ユーザグループ」を開きます。「権限設定」右の をクリックし、「FTP」タブを選択し、「FTP 承認を有効にする」に を入れます。
ルートフォルダ : アクセス先を指定



FTP サーバのユーザ名/パスワード

プロジェクトビューの「設定」-「セキュリティ」-「ユーザ」内で、FTP サーバのユーザ名/パスワードを設定することができます。



※パスワードは、8文字以上でアルファベットの大きい文字・小さい文字・数字・記号を組み合わせる等第三者から推測されないような設定を推奨しております。

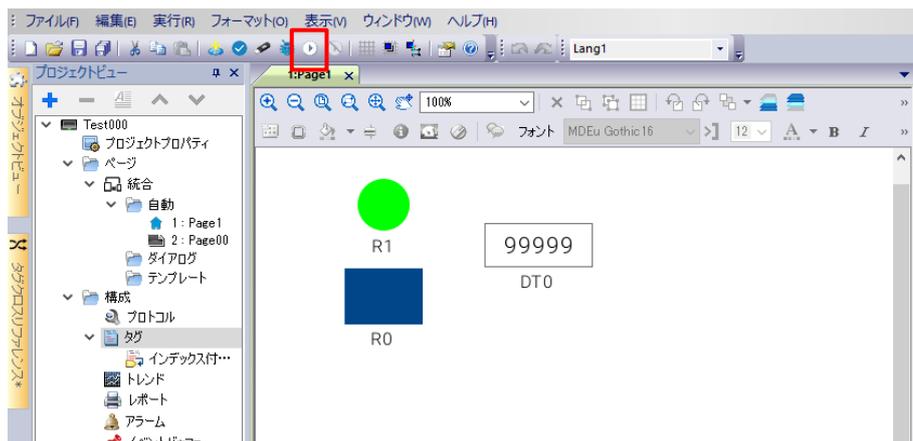
シミュレータでの PLC 連携

シミュレーション機能実行時に、PLC の値を参照することが可能です。

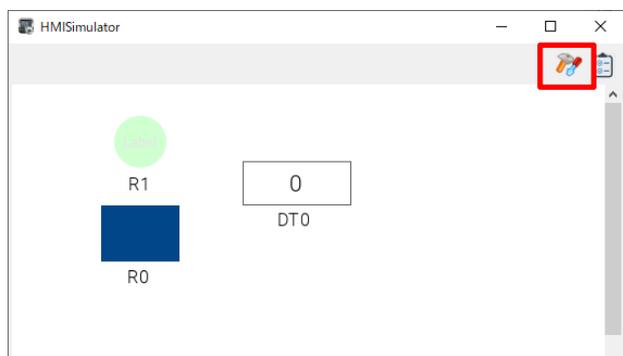
シミュレーション実行前に、シミュレーションを実行するパソコンと PLC が Lan ケーブルで接続されていることを確認してください。パソコン、PLC とともに、同一ネットワークになるように IP アドレスを設定してください。

例) パソコン:192.168.1.100、PLC:192.168.1.5

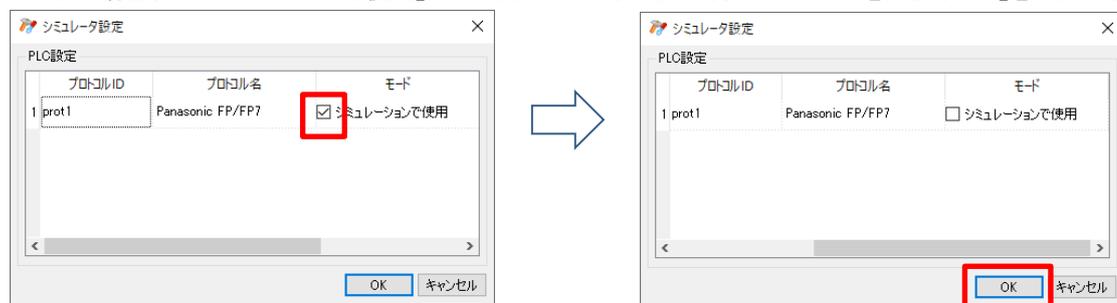
xAscender Studio からシミュレーションを実行してください。



シミュレーション画面が開きますので、 をクリックしてください。



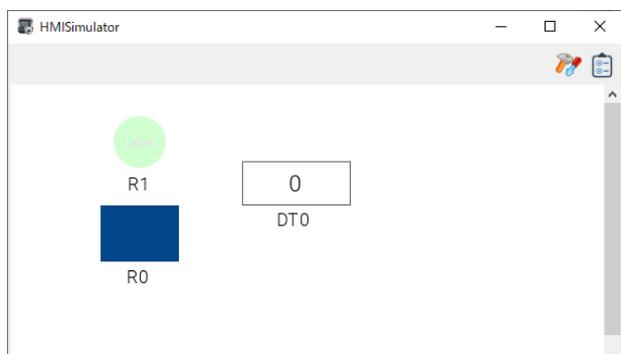
デフォルトの場合、「シミュレーションで使用」にチェックが入っているので、このチェックを外して「OK」をクリックしてください。



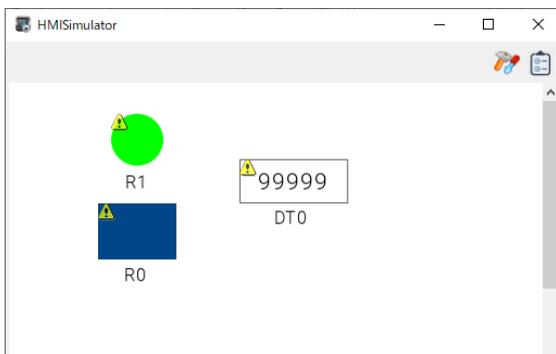
数秒間、読み込み画面が表示されたのち、シミュレーション画面が表示されます。

右図のように  が生じられている場合、パソコン-PC 間の接続ができていないため、通信設定を見直してください。

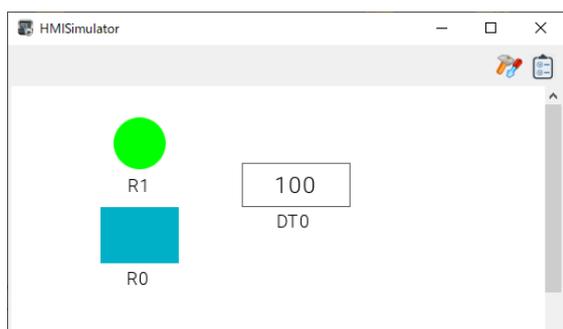
・正常に接続ができた場合



・接続ができていない場合



配置したウィジェットの操作、もしくは PLC からのデバイス操作を行い、シミュレーション画面と PLC が連動していることを確認してください。



ユーザモニタ 1			
絞り込み (Ctrl+F)			
	変数名	値	FPアドレス
1	R0	TRUE	R0
2	R1	TRUE	R1
3	DT0	100	DT0

通信対象機器構成が変動するシステムでの通信局無効指定

複数の設備で構成される製造ラインをモニタするシステムにおいて

ある時は、全ての設備を動作させるが、ある時には特定の設備は動作させない運用を行う場合を想定します。

このような生産設備の運転方法は、並列運転設備や直列運転の設備で生産計画の都合やメンテナンス上採用されることがあります。

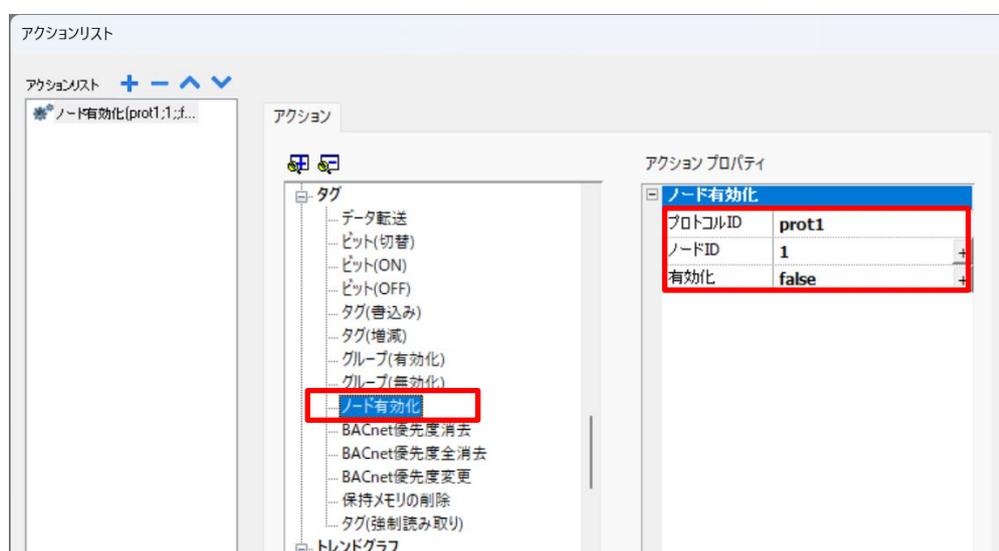
WH からそれら全ての設備に対して通信をおこなっていると、存在しない設備・PLC(電源が落ちている設備)に対しても通信を行い、通信の失敗と通信の再チェック処理が延々と行われ、システム変数には常に通信異常が報知されています。

このような場合に、存在しない設備・PLC(電源が落ちている設備)に対して**通信を行わない設定**を行っておれば、上記のような無駄な動作状態とシステム変数による通信異常の誤検知を避けることができます。

これには、ボタンなどのクリックアクションに、**タグアクション**→[**ノード有効化**]を設定し実行すれば、

対象ノードに対する通信や**通院異常検知を停止**させることができます。

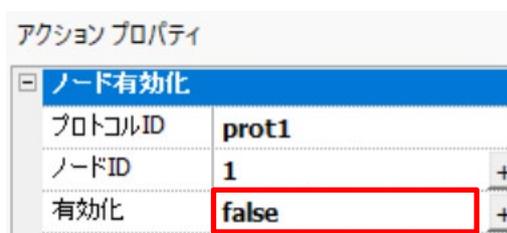
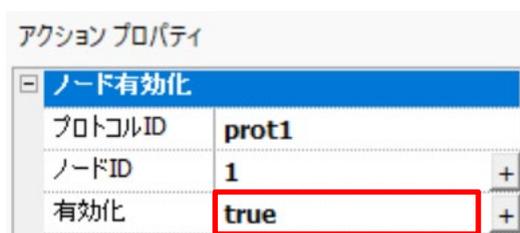
下図は、タグアクションのノード有効化の設定画面です。(下図赤枠)



有効化する場合も、無効化する場合も、いずれもアクション名は**ノード有効化**を選択します。

左下図は、**有効化の設定例**、右下図は、**無効化の設定例**です。

有効化する場合は、下図の**有効化**を **true** に設定し、無効化する場合は、下図の**有効化**を **false** に設定します。



プロトコル ID は、ドロップダウンリストに表示されるリストから適切なものを選択してください。

ノード ID も同様です。プロトコル ID の番号と同じ番号を選択します。

【具体的な例】

A, B, C 三台の設備(PLC)をモニタするシステムで、途中でCの設備を停止させ、AとBの2台のみで運転する場合切替用のボタンを設け、そのボタンのクリックでCの設備との通信を無効化するように設定します。

運用上は、Cの機械の電源を切断する前に、そのボタンをクリックして、通信を無効化します。

【動作確認例】

設備 A, B, C の生産実績をモニタしています。
設備 C の右側に設備の有効無効ラジオボタンを設けています。
下半分には、システム変数の
Protocol Communication status
Protocol Error Count
Protocol Error Message
をモニタするようにしています。
通信状態は、1で正常、その他のエラーは表示されていません。

A生産実績	<input type="text" value="3567"/>		
B生産実績	<input type="text" value="2872"/>		
C生産実績	<input type="text" value="8961"/>	有効 <input checked="" type="radio"/>	無効 <input type="radio"/>
Protocol Communication Status	<input type="text" value="1"/>	Protocol Error Count	<input type="text" value="0"/>
Protocol Error Message	<input type="text"/>		

次に、設備 C を無効にするラジオボタンをクリックします。
設備 C に対する通信や通信異常検知処理は停止しています。
C のモニタには通信をしていない状態表示の△が表示されています。

A生産実績	<input type="text" value="3567"/>		
B生産実績	<input type="text" value="2872"/>		
C生産実績	<input type="text" value="9421"/>	有効 <input type="radio"/>	無効 <input checked="" type="radio"/>
Protocol Communication Status	<input type="text" value="1"/>	Protocol Error Count	<input type="text" value="0"/>
Protocol Error Message	<input type="text"/>		

ここで生産設備 C の PLC からイーサネットケーブルを切断しても状態は、何も変わりません。**通信異常は検知されません。**

次に設備 C の PLC にイーサネットケーブルを接続した後、設備 C の有効ボタンをクリックすると、前画面と同様の状態に戻ります。

A生産実績	<input type="text" value="3567"/>		
B生産実績	<input type="text" value="2872"/>		
C生産実績	<input type="text" value="10351"/>	有効 <input checked="" type="radio"/>	無効 <input type="radio"/>
Protocol Communication Status	<input type="text" value="2"/>	Protocol Error Count	<input type="text" value="17"/>
Protocol Error Message	<input type="text" value="[prot1,PAFP;1] failed - Error: node is offline"/>		

次に設備 C が有効の状態、設備 C のイーサネットケーブルを切断してみてください。

ノード有効でかつ通信中のケーブルを切断すると**システム変数に通信異常状態が検知**・表示されます。
(右図赤枠)

5 - JavaScript 関連

WH での JavaScript の動作の仕組み

WH では、JavaScript でプログラムを作成することにより、ページ上の複数のウィジェット・プロパティやプロジェクト内の複数のタグデータ等を操作して、ウィジェット単体よりも複雑な処理を 1 回で行うことができます。以降の説明では、JavaScript のことを JS と略記します。

JS プログラムは、WH プロジェクト内の **ウィジェット等の操作イベントなどで起動**され実行されます。このような処理の仕組みは、**イベントドリブン**と呼ばれます。

そのため、あらかじめウィジェット等のイベントのアクションとして JS を割り当てて関数を作成しておく必要があります。

JS のプログラム作成

ウィジェット等の **イベントのアクションとして JS を選択**すると、ページ画面下部の JS エディタ内に **アクション関数**が定義されます。このアクション関数は、上記のイベントが発生した場合に呼び出されて実行されます。

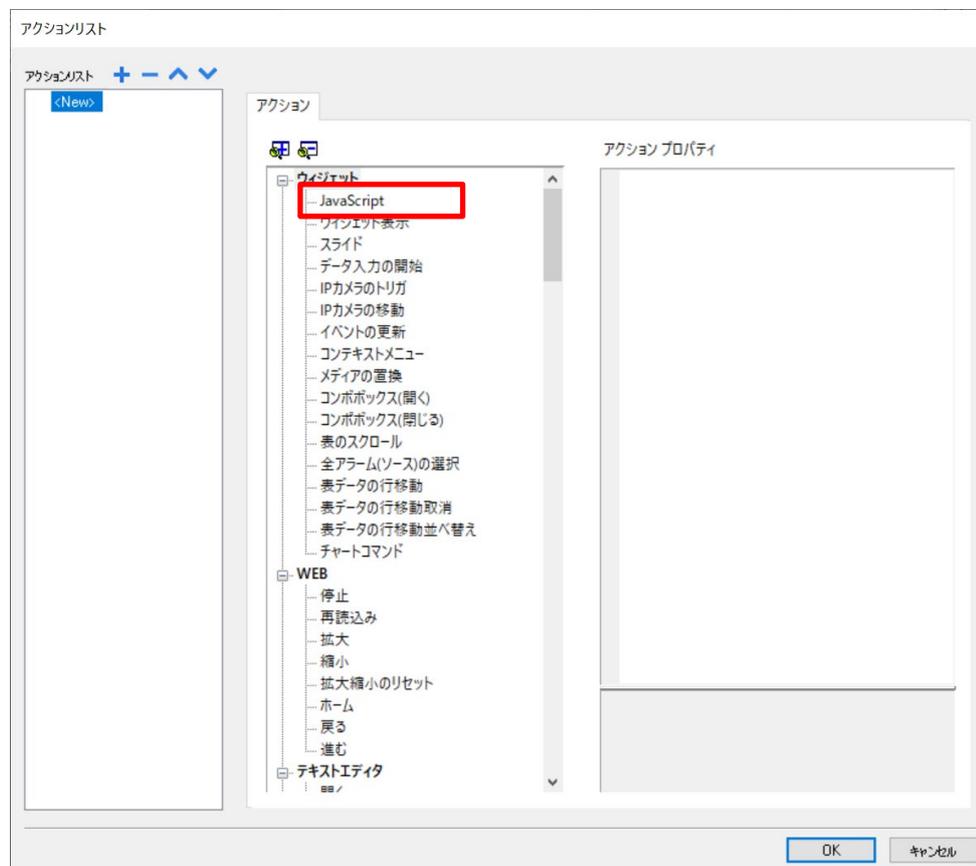
例えば、ボタンウィジェットの場合、

イベントプロパティとしてウィジェットクリック時～参照データ更新時の 5 つのイベントが定義されています。(下図)

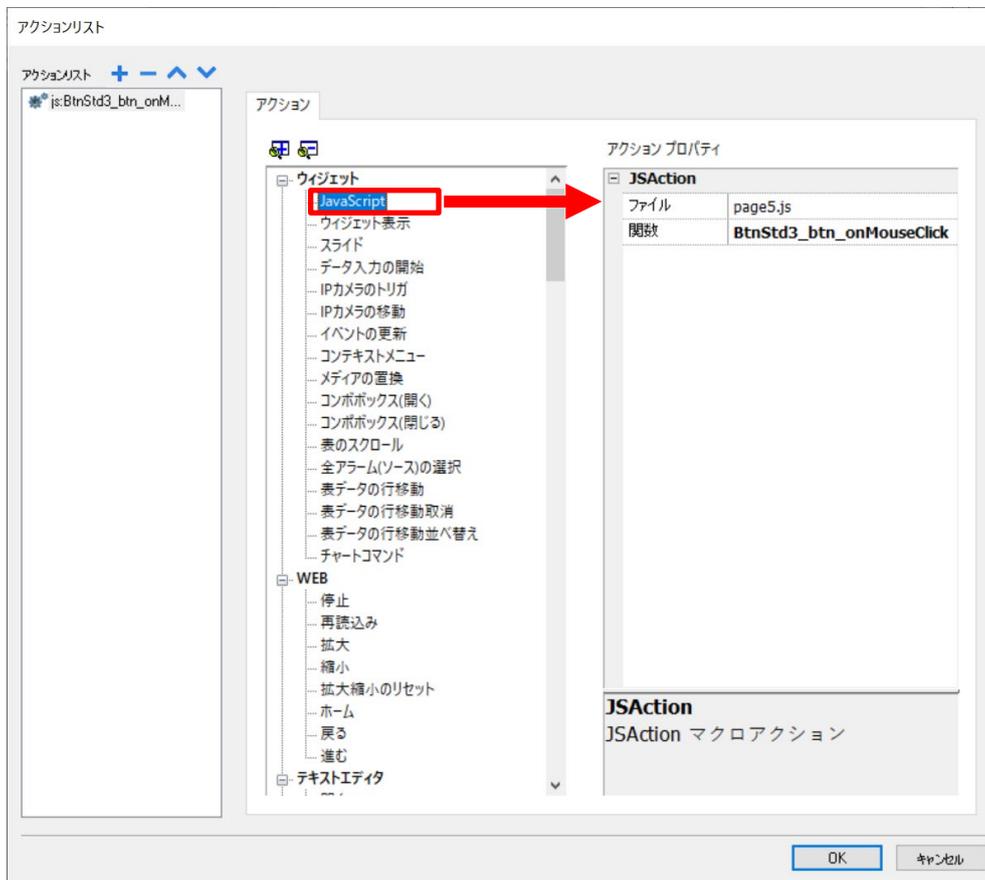
イベント	
ウィジェットクリック時	+
ウィジェット長押し時	+
ウィジェット押下時	+
ウィジェットリリース時	+
参照データ更新時	+

ウィジェットをクリック(タッチ)した時のイベントのアクションとして JS を選択します。

上図赤枠をクリックするとアクションリスト設定ダイアログが開きます。(下図)



[アクション]タブ→ウィジェット→JavaScript を選択する(下図赤枠)とアクションプロパティが表示されます。
[OK]をクリックします。



下図のようにページ下部に表示されるスクリプトエディタ内に BtnStd3 をクリックしたときに実行される関数が作成されました。関数名は、BtnStd3_btn_onMouseClicked です。{ } で挟まれた部分に処理プログラムを構成します。このままでも、このボタンをクリックしたときにこの関数は動作しますが、何もプログラムがないので、無処理となります。



オブジェクト種類と JavaScript による操作

オブジェクトとは、WH の画面を構成するソフトウェアの機能ごとのまとまりです。
各オブジェクトは、**プロパティとメソッド(関数)**を持っています。
JS では、各オブジェクトのメソッドを使用して、プロパティやタグを操作します。

オブジェクト	説明
Widget	ページ内のすべての要素(ウィジェット)の基本クラスです。
Page	現在の WH 表示ページを参照します。画面の最上位オブジェクトです。
Group	タググループを管理します。タグのグループに対して同じ処理を実行できます。
Project	プロジェクトウィジェットを定義します。プロジェクト唯一のウィジェットであり、project 変数を使用してタグ、アラーム、レンピ、スケジュール等のプロジェクトに関するデータを取得することができます。
State	制御された環境から取得した変数の状態を格納するクラスです。 状態値と取得タイムスタンプと、その品質を設定するフラグが含まれます。

JS では、上記の各オブジェクトのメソッド(関数)を使用して、各オブジェクトのプロパティにアクセスして画面の処理を行います。
JS でプログラムするには、各オブジェクトがどのようなメソッドを提供しているか知っている必要があります。

メソッド(関数)の呼び出しは、

`object.method()` の様に**オブジェクト名とメソッド名(関数名)**を「.」ドットでつないでいきます。

JavaScript の実行手順まとめ

<xAscender での準備>

- 1: ウィジェット等のアクションで JS 関数を指定します。
- 2: JS エディタにイベント関数が登録されます。
- 3: イベント関数内に JS でプログラムします。
このプログラムで、ページ内のウィジェットプロパティやプロジェクト内のタグ操作等を記述します。

<WH での動作>

- 1: ボタンタッチ等設定したアクションでイベントを発生させます。
- 2: イベント発生時に、イベント関数が呼び出されて実行されます。

JavaScript による 1 ショットタイマー使用例

JavaScript による 1 ショットタイマー処理の使用例です。

内容

ボタンをクリックすると、テキストボックスにメッセージの表示を行い、3 秒後に表示を消去します。
メッセージ表示中に再度ボタンをクリックすると、消去用タイマーを削除して、自動消去されなくなります。

プロジェクトの準備

・必要なウィジェット

1. テキストボックスにメッセージを表示するためのボタン
プロパティのクリック形式は maintained に設定してください。
ボタンのクリックイベントでテキストボックスにメッセージを表示します。
3 秒経過したらテキストボックスのメッセージを消去します。
3 秒以内に再度ボタンをクリックするとメッセージは消去されずに残ります。
2. メッセージを表示するためのテキストボックス(ラベルウィジェット)
必須設定項目=プロパティ→全般→表示の最適化→dynamic

・必要なタグ

なし

・画面の構成例



・ボタンクリックイベントで作成する JavaScript のプログラム例

```
var mytimer; // タイマーID 格納用変数
function BtnStd3_btn_onMouseClicked(me, eventInfo)
{
    if(me.value == 0){
        // BtnStd3 の値が 0 の時は、タイマー停止
        page.clearTimeout( mytimer );
    }
    else{
        // BtnStd3 の値が 1 の時は、メッセージを表示してタイマー開始
        var Message = "警告:3 秒後に消去されます。";
        var Label1 = page.getWidget( "label1" );
        var result = Label1.setProperty ( "text", Message );
        mytimer = page.setTimeout( "ClearMessage()",3000);
    }
}
// メッセージ消去関数
function ClearMessage(){
    var Message = "";
    var Label1 = page.getWidget( "label1" );
    var result = Label1.setProperty( "text", Message );
}
```

JavaScript による 1 ショットタイマー処理方法

JS で 1 ショットタイマー処理(ある時間経過後に 1 回のみ実行する処理のこと)をおこなう方法を説明します。

1 ショットタイマー処理をおこなうには、**Page オブジェクトのメソッド**を使用します。

1 ショットタイマー処理の開始・停止には、以下のメソッド(関数)を使用します。

- | | | |
|----------------|---------------------------|--------------------------|
| 1 ショットタイマー処理開始 | <code>setTimeout</code> | |
| 1 ショットタイマー処理停止 | <code>clearTimeout</code> | 関数の遅延実行をキャンセルするために使用します。 |

setTimeout と clearTimeout の使い方まとめ

- ① **setTimeout** 実行時に以下の内容を指定します。
 1. タイムアップ時に実行する関数 (コールバック関数) を指定
 2. **遅延時間**(繰り返しの時間)同時に、**遅延タイマーの ID 番号**を取得します。
この関数は、1 回実行するのみで指定した遅延時間後に指定したコールバック関数が実行されます。
別途、コールバック関数の作成が必要です。
- ② その後、指定時間経過後に指定した関数が実行されます。
関数が実行された後は、起動したタイマーは自動的に削除されますので、**clearTimeout** 関数を呼び出す必要はありません。
- ③ タイムアップ前(関数実行される前)にタイマーを停止して関数実行を取りやめる場合は、**clearTimeout** 関数を実行します。

setTimeout メソッドの呼び出し

`number page.setTimeout (functionName, delay)`

戻り値	タイマーID: 複数のタイマー識別用です。タイマー停止要求時に使用できます。
引数	説明
functionName	コールバック関数の指定 呼び出す関数の名前文字列または文字列を含む変数名
delay	遅延時間(ミリ秒)

clearTimeout メソッドの呼び出し

`void page.clearTimeout (timerID)`

戻り値	なし
引数	説明
timerID	停止するタイマーID を指定します。setTimeout 関数実行時に取得した値

JavaScript によるウィジェットプロパティ操作方法

JS でウィジェットプロパティを操作する方法を説明します。

ウィジェットプロパティを操作するには、**ウィジェットクラスオブジェクトのメソッド**を使用します。

ウィジェットプロパティの値取得(読み出し)・設定(書き込み)には以下のメソッド(関数)を使用します。

タグの内容取得: `getProperty`

タグの内容設定: `setProperty`

操作対象のウィジェットクラスオブジェクトは、**Page オブジェクトの `getWidget` メソッド**で取得します。

つまり、**ウィジェットプロパティへのアクセスは、以下の 2 段階の手順で実行**します。

1. 操作対象のウィジェットクラスのオブジェクトを取得 → `var object = page.getWidget` メソッド使用
2. 操作対象のプロパティへのアクセス: 1 で取得したウィジェットクラスオブジェクトのメソッドを使用
→ `object.getProperty` または `object.setProperty` を使用

また、ウィジェットプロパティを `setProperty` で変更する場合には、

変更対象ウィジェットのプロパティ**全般**→**表示の最適化**を

Dynamic に設定しておく必要があります。

(忘れやすいので、注意が必要です)

全般	
点滅	false
ID	label1
表示	true
不透明度	1
ロック	false
表示の最適化	dynamic

`getWidget` メソッドの呼び出し

`page.getWidget (wgtName)`

戻り値	指定したウィジェットオブジェクトが返ります。
引 数	説 明
wgtName	ウィジェット名の文字列または文字列を含む変数名。 文字列はダブルクォーテーションで囲みます。

使用例: Label1 ウィジェットのオブジェクトを Label1 に取得します。

```
var Label1 = page.getWidget ( "label1" );
```

`getProperty` メソッドの呼び出し

`object.getProperty (propertyName, [index])`

戻り値	指定したウィジェットのプロパティが返ります。
引 数	説 明
propertyName	取得するプロパティ名の文字列または文字列を含む変数名。 文字列はダブルクォーテーションで囲みます。
index	配列から取得する場合に要素のインデックスを指定。(デフォルト = 0)

使用例: Label1 ウィジェットの text プロパティを取得し label1text 変数にセットします。

Tag は、Page オブジェクトのメソッドで取得できますが、ウィジェットのプロパティは、

Page オブジェクトの `getWidget` メソッドで、**ウィジェットオブジェクト**を取得し、そのウィジェットオブジェクトのメソッドである `getProperty` メソッドを使用して取得します。

```
var Label1 = page.getWidget ( "label1" );  
var label1text = Label1.getProperty ( "text" );
```

setProperty メソッドの呼び出し

boolean object.setProperty (propertyName, value, [index])

戻り値	ブール値(True または False)を返し、プロパティ設定結果を示します。
引 数	説 明
propertyName	設定するプロパティ名の文字列または文字列を含む変数名。 文字列はダブルクォーテーションで囲みます。
value	設定する値
index	配列に設定する場合、要素のインデックス(デフォルト = 0)

使用例:

ボタンの値(1 / 0)に従って、Label1 ウィジェットの text プロパティの表示を OK または NG に切り替えます。

ボタンのクリック形式は、maintained に設定してください。

表示を変更するラベルの全般→表示の最適化プロパティは、dynamic に設定しておいてください。

ボタンのクリックイベント関数内に以下のプログラムを記述します。

```
if (me.value ==1) {  
    var label1text = "OK";    // true の時、OK  
}  
else{  
    label1text = "NG";        // false の時、NG  
};  
var Label1 = page.getWidget("label1");  
var result = Label1.setProperty ( "text",label1text );
```

JavaScript によるタイムインターバル使用例

JavaScript によるタイムインターバル処理の使用例です。

内容

ボタンをクリックすると、250 ミリ秒周期にタグの値を+1して更新し、再度ボタンをクリックすると動作を停止する。

プロジェクトの準備

・必要なウィジェット

1. タグ値のインクリメント(+1)の開始。停止用ボタン
プロパティのクリック形式は **maintained** に設定してください。
ボタンのクリックイベントでインターバルタイマを起動し、250 ミリ秒周期でタグの値を+1します。
再度ボタンをクリックすると、タグの値の更新を停止します。
2. タグのモニター用数値ウィジェット
値に tag1 を読み出しで割り当ててください。

・必要なタグ

内部変数タグ: Tag1 int 型を定義してください。

・画面の構成例



・ボタンクリックイベントで作成する JavaScript のプログラム例

```
var mytimer;          // タイマーID 格納用変数
function BtnStd1_btn_onMouseClicked(me, eventInfo)
{
    if(me.value == 0){
        // BtnStd1 の値が 0 の時は、タイマー停止
        page.clearInterval(mytimer);
    }
    else{
        // BtnStd1 の値が 1 の時は、タイマー開始
        mytimer=page.setInterval( "Tag1Increment()", 250);
    }
}
// Tag1 の内容を+1 する関数
function Tag1Increment(){
    var state = new State();
    var value = project.getTag("Tag1", state);    //Tag1 の値を value に読み出す
    value = value +1;                            //value を+1 する
    project.setTag("Tag1", value);              //Tag1 に value の値をセットする
}
```

JavaScript によるタイムインターバル処理方法

JS でタイムインターバル処理(一定周期で同じ処理を繰り返すこと)をおこなう方法を説明します。
タイムインターバル処理をおこなうには、**Page オブジェクトのメソッド**を使用します。
タイムインターバル処理の開始・停止には、以下のメソッド(関数)を使用します。

タイムインターバル処理開始 `setInterval`
タイムインターバル処理停止 `clearInterval`

setInterval と clearInterval の使い方まとめ

- ① `setInterval` 実行時に以下の内容を指定します。
 1. 繰り返し実行する関数 (コールバック関数) を指定
 2. **インターバル時間**(繰り返しの時間)同時に、繰り返し**タイマーの ID 番号**を取得します。
この関数は、1 回実行するのみで指定したコールバック関数が繰り返し実行されます。
別途、コールバック関数の作成が必要です。最初の関数実行も指定時間経過後になります。
- ② その後、指定時間経過後に指定した関数が繰り返し実行されます。
- ③ 繰り返し実行を停止したい場合(タイマーを停止したい場合)は、`clearInterval` を実行します。
その際、`setInterval` 実行時に取得したタイマーID を指定します。

setInterval メソッドの呼び出し

`number page.setInterval (functionName, interval)`

戻り値	タイマーID : 複数のタイマー識別用です。タイマー停止要求時に使用できます。
引 数	説 明
functionName	コールバック関数 の指定(繰り返し実行する関数) 呼び出す関数の名前文字列または文字列を含む変数名
interval	インターバル時間 、 msec 単位 で指定します。

コールバック関数とは、ある関数を呼び出す時に、引数に指定する別の関数のことです。
呼び出し元が用意したある関数を、呼び出し先の処理の中から、呼び出し返す処理を行うので
Callback (コールバック)と呼ばれます。`setInterval` 関数の場合、繰り返し実行する関数を指定します。

clearInterval メソッドの呼び出し

`void page.clearInterval (timerID)`

戻り値	なし
引 数	説 明
timerID	停止するタイマーID を指定します。 <code>setInterval</code> 関数実行時に取得した値

JavaScript によるラベル表示の折り返し表示方法

ラベルウィジェットに文字列を折り返して表示する場合、固定文字列であれば、xAscender のラベル部品をダブルクリックして専用エディタを開いて、任意の場所で enter キー入力することにより折り返すことが可能です。

値をタグにリンクして表示する場合は、折り返すことが困難です。制御文字を文字列に組み込んでも折り返さず文字列として表示してしまいます。

変動する文字列をラベル部品に任意の長さで折り返して表示したい場合、

JavaScript のプログラムで文字列の任意の部分で制御コードを組み込んでラベルに格納すると折り返して表示することが可能です。文字列を一定の長さで区切って縦に並べたい場合(折り返し表示したい場合)は、この方法で表示することが可能です。

具体例

PLC より区切り文字コンマで区切られた 5 つの果物の名前の文字列が受信され、

そのコンマを削除して果物の名前を 1 つのラベルウィジェットに上から順番に並べて表示する場合

受信文字列

Apple, Orange, Pineapple, Banana, Strawberry

文字列を受信するタグ

Tag5

表示するためのラベルウィジェット

必須設定項目=プロパティ→全般→表示の最適化→dynamic

この例では、処理を簡単にするためにラベルウィジェットへの表示は、表示用のボタンをクリックしたときに文字列受信タグ(Tag1)に受信している文字列をラベルに表示するようにします。



ボタンのクリックイベントプログラム例

```
function BtnStd1_btn_onMouseClicked(me, eventInfo)
{
    var state = new State();
    // Tag から文字列を取得
    var value = project.getTag("Tag5", state, 0);

    // 文字列のコンマを改行\r\nで置換
    var resultstr = value.replace(/,/g, "\r\n");

    // ラベルに複写
    var Label = page.getWidget("label1");
    var result = Label.setProperty("text", resultstr);
}
```

JavaScript によるタグ操作方法

JS でタグにアクセス(内容の読み出し・書き込み)する方法を説明します。

タグは、Project オブジェクトで管理されています。タグを操作するには、**Project オブジェクトのメソッド**を使用します。

タグの値の取得(読み出し)・設定(書き込み)には以下のメソッド(関数)を使用します。

タグの内容取得: getTag
タグの内容設定: setTag

getTag メソッドの呼び出し

Project.getTag (tagName, state, index, [callback], [forceRefresh])

戻り値	タグ値が返されます。 tagName で指定したタグが配列型で 引数 index = -1 の場合、配列が返されます。
引 数	説 明
tagName	タグ名の文字列または文字列を含む変数名。 文字列はダブルクォーテーションで囲みます。
state	読み出し結果を格納する変数を指定します。
index	通常(タグが配列タイプ以外)は 0 を指定します。 タグが配列タイプの場合はインデックス値を指定します。 インデックス値に-1 を指定時は、配列型のデータが読み出されます。
callback	コールバック関数名(非同期読み取りが必要な場合に指定)。デフォルトは、「」
forceRefresh	True = 実行中に直接デバイスから読み取ったタグの更新値です。 デフォルトは FALSE です。

使用例: Tag1 の内容を取得する場合。変数 tag1value に読み出しています。

```
var state = new State();  
var tag1value = project.getTag ( "Tag1", state );
```

setTag メソッドの呼び出し

number Project.setTag(tagName, tagValue, [index], [forceWrite])

戻り値	forceWrite が true の場合に、アクションの成功と失敗を示す整数値。 0 は成功を意味し、-1 は失敗を意味します。 forceWrite が false の場合、返される値は未定義です。
引 数	説 明
tagName	タグ名の文字列または文字列を含む変数名。 文字列はダブルクォーテーションで囲みます。
tagValue	書き込み値を含む変数(オブジェクト)
index	通常(タグが配列タイプ以外)は 0 を指定します。 タグが配列タイプの場合はインデックス値を指定します。 インデックス値に-1 を指定時は、配列型のデータを引き渡します。
forceWrite	タグを強制的に書き込みを有効にするブーリアン値で、 関数は、値が書き込まれるのを待ってから戻ります。デフォルト = false。

使用例: Tag2 の内容を設定する場合。変数 settingValue の値 100 を Tag2 に設定しています。

```
var settingValue= 100;  
project.setTag("Tag2", settingValue);
```

JavaScript によるラベル付きボタンの表示内容変更方法

ラベル付きのボタンの表示内容を JavaScript で変更する方法です。

OFF の時のラベルを設定しておき、クリックイベントに JavaScript 関数を指定します。

JS 関数内では、ボタンの値(0 または 1)によって表示する内容を切り替えます。

具体例:

ラベル付きボタンの値が 0 の時は、「開始」とラベル表示され、クリックして値が 1 になると「停止」と表示されるラベル付きボタン。

ラベル付きボタン (StdBtn1) の初期値は下記の様に設定しておいてください

ボタン初期値: 0

ボタンクリックモード: maintained

ラベル初期表示: 停止

また、ラベルのプロパティの全般→表示の最適化プロパティは、dynamic デフォルトの normal のままで変更できます。

(dynamic にしなくても変更できます。ラベルウィジェット単体では、dynamic に設定しなければ変更できません。)

動作の説明

1: ボタンをクリック 1 回目

値が 0→1 に変化

クリックイベント発生→下記の関数が呼び出し実行される

2: 関数内では、ボタンの値をチェックし

値が 1 なら停止

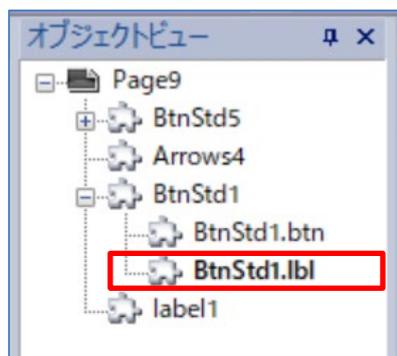
値が 0 なら開始 をラベルに表示します。

イベント関数例

```
function BtnStd1_btn_onMouseClicked(me, eventInfo)
{
    if (me.value == 1) {
        var label1text = "停止"; // true の時、停止
    }
    else {
        label1text = "開始"; // false の時、開始
    };
    var BtnStd1Label = page.getWidget("BtnStd1.lbl");
    var result = BtnStd1Label.setProperty ("text", label1text );
}
```

ラベル付きボタン BtnStd1 のラベルオブジェクト名は上記のように "BtnStd1.lbl" のように記述する必要があります。

詳細は、[オブジェクトビューを開く](#)と確認できます。(下図赤枠)



JavaScript による複数の内部変数タグ初期化方法 1 (ホームページ表示時初期化)

複数の内部変数を初期化したい場合に、**ページイベントのアクティブアクション**で **JavaScript 関数を起動**して初期化可能です。ページイベントのアクティブアクションは、ページを表示した場合に起動されます。

プロジェクト起動時だけでなく、ホームページでデータ変更後、別の画面を表示・操作してホームページに戻ってきた場合にも実行されて初期値に戻ってしまいますので、注意が必要です。

注)内部変数タグのみでなく、**PLC との通信タグに対しても有効**です。

(例)左下図のように Tag1, Tag2, Tag3 の 3 つの int 型内部変数タグを定義し

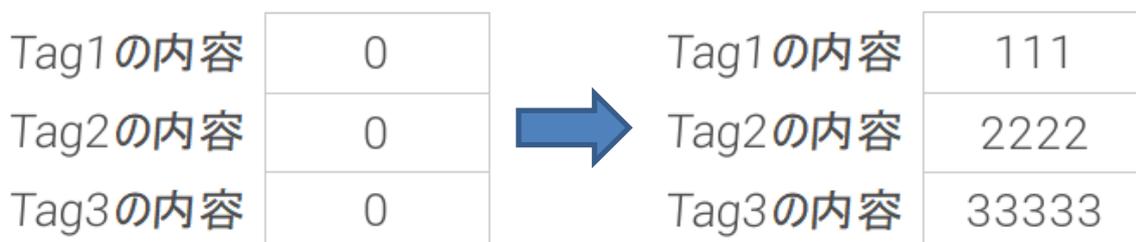
右下図のようにホームページのアクティブアクションで JavaScript 関数を起動して 3 つのタグを初期化します。

ホームページ表示時にそれぞれ初期化します。

The screenshot shows the software interface with two main panels. The left panel, titled 'タグ' (Tag), shows a tree view of 'Variables:prot1' containing 'Tag1', 'Tag2', and 'Tag3'. The right panel shows the configuration for 'Tag1' with properties like 'データタイプ' (int), 'タグ名' (Tag1), and 'タグURI' (Tag1 ?int). The rightmost panel shows the 'イベント' (Event) configuration, where the 'アクティブアクション' (Active Action) is set to '1アクション' (1 Action) with the script 'js:Page1_onActivate()'.

・ページアクティブアクションで作成する JavaScript のプログラム例

```
function Page1_onActivate(me, eventInfo)
{
    project.setTag("Tag1",111);
    project.setTag("Tag2",2222);
    project.setTag("Tag3",33333);
}
```



JavaScript による複数の内部変数タグ初期化方法 2（初期画面で初期化実行）

前頁の JavaScript による内部変数タグ初期化方法 1 では、ホームページの表示アクション時にデータを初期化していたのでホームページ内で、データ内容を変更後に別の画面に遷移し、再度ホーム画面に戻ってきた場合にも初期化が実行されるので変更したデータ内容が保持されませんでした。

ここでは、そのような場合にも変更されたデータが保持される方法を説明いたします。

注)内部変数タグのみでなく、PLC との通信タグに対しても有効です。

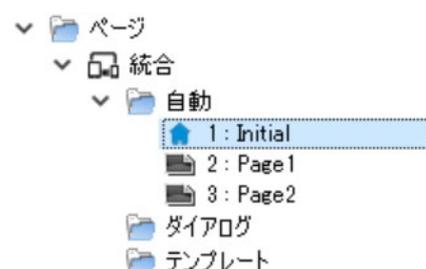
前頁と同様に Tag1, tag2, Tag3 の 3 つの変数をプロジェクト起動時に初期化する場合の例です。

前頁のプロジェクトをベースに変更する説明をします。

1. 右図のように Initial ページを作成し、ホームページに設定します。
アクティブアクションを JavaScript 関数起動で設定します。
関数を以下のように記述します。

```
function Initial_onActivate(me, eventInfo){
    // 3 つのタグに初期値設定
    project.setTag("Tag1",111);
    project.setTag("Tag2",2222);
    project.setTag("Tag3",33333);

    // 内部変数初期化完了 ページ 1 へ
    project.nextPage();
}
```



2. 前頁で説明した Page1 は、そのまま残して、アクティブアクションを削除します。
ページ 2 へ移行するボタンを配置します。
3. Page3 を作成し、Page1 へ戻るボタンを配置します。

この方法により、起動時は Initial ページが表示されますが、データ初期化後すぐに Page1 が表示されます。

Page1 では、Tag1, Tag2, Tag3 の初期データが表示されます。

ここで、データ部品を操作して、Tag1, Tag2, Tag3 の数値を任意の値に変更し、ボタンをクリックして Page3 に移行します。

Page3 から Page2 へ戻るボタンで Page2 に戻っても、Page2 で設定したデータはそのまま初期化されずに残ります。

タグの一括書き込み (JavaScript 機能使用・非サイクリック通信)

タグの一括読み出し (JavaScript 機能使用・非サイクリック通信) の項では、JavaScript プログラムにより、**手動更新に設定された配列変数タグに PLC から一括でデータ読み出し**を行いました。

ここでは、上記の一括転送により、読み出した PLC の値を JavaScript 内で定義した配列変数に複写し、その配列変数の値を加工 (+10000) して PLC のデータレジスタ DT10~ に一括書き込みする方法を説明します。

「タグ一括読み出し (JavaScript 機能使用・非サイクリック通信)」での具体例に以下の内容を追加します。

1. PLC への書き込み用の配列変数タグを定義します。

タグ名は g_auIWtTag、データ型 unsignedshort、PLC アドレスは DT10~DT14、要素数 5、R/W、手動更新

データ	タグURI	プロパティ	値
Panasonic FP/FP7:prot1 Model: FP7		▼ ドライバ	
		モデル	FP7
		プロトコル	Panasonic FP/FP7:prot1
		▼ タグ	
		データタイプ	unsignedShort-5
		タグ名	g_auIWtTag
		PLCタグ名	
		グループ	
		タグURI	1?DT?10?unsignedShort-5
		コメント	
		更新速度	手動
		R/W	R/W
		アクティブ	false
		シミュレータ	Variables
		スケーリング	None
		最小値	0
		最大値	65535

2. 画面に実行ボタンを配置し、イベントプロパティに JavaScript を設定します。

【画面構成例】

【JavaScript 処理手順】 (n=0-4)

① 一括読み出し DTn → g_auIRdTag[n]
 ② データ加工 g_auIRdTag[n] + 10000 → val[n]
 ③ 一括書き込み val[n] → g_auIWtTag[n] → DT1n

実行

読み出しタグ

g_auIRdTag[n]

99999

書き込みタグ

g_auIWtTag[n]

99999

読み出しタグのモニタ

項目: numeric3	
値	99999
データリンク	g_auIRdTag[0]
アクセス種類	R

書き込みタグのモニタ

項目: numeric2	
値	99999
データリンク	g_auIWtTag[0]
アクセス種類	R

JavaScript 実行手順の説明

JavaScript 実行用ボタン

イベント	
ウイジェットクリック時	1アクション
アクション[0]	js:BtnStd1_btn_onMouseClicked()
ウイジェット長押し時	
ウイジェット押下時	
ウイジェットリリース時	
参照データ更新時	

3. JavaScript 関数内に以下の記述を追加します。

```
function BtnStd1_btn_onMouseClicked(me, eventInfo)
{
    // タグ一括読み出し
    var state = new State();
    var value = project.getTag("g_auIRdTag", state, -1, "", true);

    var val = new Array(5);
    // 読み出しデータ加工
    for (var ptr = 0; ptr < 5; ptr++){
        // 読み出しデータ+10000 → val[]
        val[ptr] = project.getTag("g_auIRdTag", state, ptr)+10000;
    }
    // val 配列データを g_auIWtTag へ一括書き込み
    // 同時に PLC の DT10~DT14 にデータが書き込まれる
    var status = project.setTag("g_auIWtTag", val, -1, true);
}
```

【動作確認】

シミュレータを起動し、PLC と実通信を行って動作を確認します。

実行ボタンをクリックして、読み出しタグや書き込みタグのモニタ値を確認し、ツールソフトでもモニタ値を確認します。

以下のように動作確認ができれば OK です。

【JavaScript 処理手順】 (n=0-4)

① 一括読み出し DTn → g_auIRdTag[n]
 ② データ加工 g_auIRdTag[n] + 10000 → val[n]
 ③ 一括書き込み val[n] → g_auIWtTag[n] → DT1n

実行			
	777		10777
読み出しタグ	778	書き込みタグ	10778
g_auIRdTag[n]	779	g_auIWtTag[n]	10779
	780		10780
	781		10781

ユーザモニタ 2			
絞り込み (Ctrl+F)			
	変数名	値	FPアドレス
1	g_auIWtTag		
2	[0]	10777	DT10
3	[1]	10778	DT11
4	[2]	10779	DT12
5	[3]	10780	DT13
6	[4]	10781	DT14
7	g_uiTestDT0	777	DT0
8	g_uiTestDT1	778	DT1
9	g_uiTestDT2	779	DT2
10	g_uiTestDT3	780	DT3
11	g_uiTestDT4	781	DT4

【JavaScript と配列変数タグを使用するメリット】

JavaScript では、配列変数タグの全要素に対する読み出しや書き込みが 1 命令で実行できるので便利です。

タグの一括読み込み (JavaScript 機能使用・非サイクリック通信)

前項のデータ転送機能を使用したタグの一括読み込みでは、以下のような使いにくさがありました。

1. 通信用として登録したタグにはデータが読み出せない。つまり、直感的な動作とは異なる動作
注)これは一括読み出しする動作に関しては特に問題ではありません。
2. タグに配列変数を定義しても、データ転送エディタに登録する際は、各要素を1つずつ登録する必要がある。
つまり、多数のデータを一括で読み出す場合に登録に大変な手間がかかります。

ここでは、通信用のタグに配列変数を定義し、手動更新の設定を行い、その通信用タグにデータを一括読み出しすることを JavaScript で実行する方法について説明いたします。

【PLC 側の設定】

前項「タグの一括読み込み(データ転送機能使用・非サイクリック通信)」の設定と同じです。

【WH 側の設定】

① プロトコル設定

前項「タグの一括読み込み(データ転送機能使用・非サイクリック通信)」の設定と同じですが、Variables プロトコルは使用しません。

② タグ設定

1. 上記①-1 プロトコルに unsignedshort 型の 5 個の要素を持つ配列型タグを登録します。

タグ名は、**g_auiRdTag** で、PLC アドレスは DT0~DT4 の 5 ワード占有します。更新周期を「手動」に設定してください。
この 5 要素の配列型タグに PLC の DT0 から DT4 のデータを一括読み出して格納します。

Panasonic FP/FP7

Memory Type: DT - Data reg, Offset: 0, SubIndex: 0

Data Type: unsignedShort [], Arraysize: 5, 変換: | +/-

Unsignedshort 型の配列を指定。配列のサイズは、5

プロパティ	値
▼ ドライバ	
モデル	FP7
プロトコル	Panasonic FP/FP7:prot1
▼ タグ	
データタイプ	unsignedShort-5
タグ名	g_auiRdTag
PLCタグ名	
グループ	
タグURI	1?DT?0?unsignedShort-5
コメント	
更新速度	手動
R/W	R/W
アクティブ	false
シミュレータ	Variables
スケール	None
最小値	0
最大値	65535

③ ページ設定

1. `g_auIRdTag[0]`, `g_auIRdTag[1]`, `g_auIRdTag[2]`, `g_auIRdTag[3]`, `g_auIRdTag[4]` の値を表示するためのデータウィジェットを5つ定義します。
それぞれのデータウィジェットの値プロパティには、上記のタグを読み出し専用(RD)でデータリンクします。
2. データを一括読み出しするためのトリガボタンを配置します。イベントプロパティに JavaScript アクションを登録します。

DT0→	<code>g_auIRdTag[0]</code>	1029
DT1→	<code>g_auIRdTag[1]</code>	1030
DT2→	<code>g_auIRdTag[2]</code>	1031
DT3→	<code>g_auIRdTag[3]</code>	1032
DT4→	<code>g_auIRdTag[4]</code>	1033

タグ値一括読み出し

イベント	
ウィジェットクリック時	1アクション
アクション[0]	<code>js:BtnStd2_btn_onMouseClicked()</code>

プロパティ

項目	numeric3
値	99999
データリンク	<code>g_auIRdTag[0]</code>
アクセス種類	R
数値形式	Numeric

配列タグの要素番号は、ここで指定します。

配列インデックス 0

④ JavaScript のプログラム記述例

以下のような JavaScript のプログラムを記述します。

```
function BtnStd2_btn_onMouseClicked(me, eventInfo)
{
    var state = new State();
    var value = project.getTag("g_auIRdTag", state, -1, "", true);
}
```

一括読み出しボタンをクリックすることにより、このプログラムを実行して、`g_auIRdTag` の全要素データを読み出してセットします。JavaScript と配列変数タグを使用すると、1行で配列要素のすべてを読み出すことができるので、大変便利です。

【具体例の動作確認】

FPWIN-Pro での PLC 側のデータモニタの値と xAscender シミュレータ上の読み出し値を見比べて一括読み出しができていないことを確認します。

タグ値一括読み出しボタンをクリックして、下図のようなイメージで、データがタグに転送されることを確認してください。

DT0→	<code>g_auIRdTag[0]</code>	1029
DT1→	<code>g_auIRdTag[1]</code>	1030
DT2→	<code>g_auIRdTag[2]</code>	1031
DT3→	<code>g_auIRdTag[3]</code>	1032
DT4→	<code>g_auIRdTag[4]</code>	1033

タグ値一括読み出し

改訂履歴

発行日時	番号	改訂内容
2021年3月	gts242ja	初版
2023年3月	gts242ja-01	新規記事追加
2023年9月	gts242ja-02	新規記事追加

●技術に関するお問い合わせは

FAデバイス技術相談窓口



0120-394-205

※受付時間/9:00 ~ 17:00(12:00~13:00、弊社休業日を除く)

Webサイト industrial.panasonic.com/ac/

パナソニック インダストリー株式会社 産業デバイス事業部

〒574-0044 大阪府大東市諸福7丁目1番1号

© Panasonic Industry Co., Ltd. 2023

本書からの無断の複製はかたくお断りします。2023年9月

No.gts242ja-02