

A grayscale photograph of an industrial facility, possibly a power plant or refinery, featuring large storage tanks and piping. In the background, there are wind turbines and a range of mountains under a hazy sky. This image is framed within the central inverted triangle of the cover.

Student Workbook

Advanced Configuration

Trihedral®
A Delta Group Company



VTScada Advanced Configuration, Student Workbook
Copyright Trihedral Engineering Limited, 2/4/2025
All rights reserved.

Support: support@trihedral.com
Sales: sales@trihedral.com

Trihedral Engineering Limited
Head Office

1160 Bedford Highway
Suite 400
Bedford, Nova Scotia
Canada B4A 1C1
Phone: 902-835-1575
Toll Free: 800-463-2783
Fax: 902-835-0369

Trihedral UK Limited

Glover Pavilion, Campus 3
Aberdeen Science Park
Balgownie Drive, Aberdeen
UK, AB22 8GW
Phone: +44 (0) 1224 258910
Fax: +44 (0) 1224 258911

Trihedral Inc. ~ USA

7380 W Sand Lake Road,
Suite 100
Orlando, FL 32819
Phone: 407-888-8203
Fax: 407-888-8213

Trihedral Engineering Limited
Western Canada
403-921-5199

Trihedral Inc. Southern US
407-443- 3785
205-224-3977
205-612-6665

Trihedral Inc. Mid Atlantic US
937-470-2503

Trihedral Inc. Southwestern US
281-387-2939

Trihedral Inc, Central US
224-409-8221

Trihedral Inc, New England US
617-877-6660

Trihedral Inc, Western US
916-905-3343



Contents

Preface	1
1 A Short Review...	2
2 Edit Properties	13
Edit Properties with the Advanced Mode	14
Configuration Examples	17
Configuration File Structure	18
Workstation-Specific Properties	23
System Properties - Setup.ini	25
3 Expressions, Part One	28
Syntax Rules for Expressions	30
Where to Write Expressions	30
Use Tag Values in Expressions	33
Operators	35
Comparisons: Operators Functions	36
"Invalid" in Expressions and Results	38
Working in Steady State	40
Using Functions	42
Function Selection Dialog	44
Math in Expressions	48
Text Functions in Expressions	49
4 Design Your Own Tags	52
Add Properties to Context Tags	54
Automated Tag Configuration	58
From Context to Type	65
Modify a Custom Type	66
Start Tag Expressions	72
Multi-Write Tags	76
5 Create Widgets	79
Tag Widgets	82
Keep a Tidy Palette	86
Edit a Tag Widget	86
Linked Tag Properties	89

6 Parameterized Pages	98
Link Parameters to Objects in a Page	100
Setting Values for Parameters	103
Expressions in Page Titles	104
Widgets versus Parameterized Pages	106
Folders	108
7 Advanced Security Customization	112
Best Practices for Security	112
Create Accounts and Roles	115
Protect Pages and Output Tags	121
Rules for Privilege Scope	124
Read-Only Workstation	128
Administrative Settings	131
Automatic Sign Out Time Period	132
Password Options	133
Advanced	133
OpenID Connect Authentication	134
8 Filtering Tags, Alarms and Realms	138
Realm Filtering	139
Global Tag & Area Filtering	147
Tag Area Filtering	151
Tag Area Filtering Example	153
9 Reusable Application Layers	155
Inheritance Across Layers	159
Create an OEM Layer	162
Distribute OEM Layer Updates	167
10 Master & Subordinate Applications	169
Subordinate Application Tags	172
11 Sites & Maps	176
Sites List Display Options	177
Site Details Page	182
Site Details Configuration	184
Create a Custom Site Details Page	185
Site Map	190

Change the Map Source	194
Bulk Downloads of Map Tiles	196
Use Maps Without an Internet Connection	198
Custom Map Icons	199
Site Map Widget	204
12 Log, Note, and Report	206
Collecting Data	207
Edit Data	209
Symbols for Edited Values	211
Historian and Logger Configuration	211
Historian Data Storage	213
Historians and Multiple Servers	218
HDV Grid as a Reporting Tool	221
Reports Page	223
Report Tags	226
Run-Time Configuration of Report Tags	230
Excel Add-in for Data Retrieval	231
Settings for the Excel Add-in	233
Create and Manage Connections	233
Create Queries	237
Example Queries with Excel	247
SQL Queries	249
Configure an ODBC Server	251
SQL View	256
REST Queries - Notes & Examples	258
The Report Studio	262
Report Studio Tools	265
Configure Table Appearance and Data Aggregation	267
Configure Time Columns	268
Create Tag Queries	270
Create Alarm & Event Queries	275
Add Additional Columns [Optional]	278
Add Summary Rows	278
Add Decorative Elements to Reports	279

Define Parameterized Reports	282
Make Parameterized Phrase	287
13 Advanced Configuration Topics	289
Alarm on Low Disk Space	289
Alarm Database Groups	289
Customize Columns in Alarm Displays	291
Tag List Widget for Reports	296
Control Locks	299
Control Lock Security	303
Control Tokens	304
Recipes and Batch Processing	311
14 Expressions, Part Two	317
Other Tag Properties	317
Use Application Properties	320
Time and Date Functions in Expressions	322
Triggers in Expressions	325
Resetting functions in expressions	325
Noticing when a value changes - Watch()	326
Keep the Else's Under Control	326
Multilingual Expressions	327
15 Best Practices for Scaling Up Your System	332
Appendices	343
A Tag Tables	344
B Document Your Application	346
Create a Help Widget	346
C Languages	350
Phrase Editor	357
C Page Shuffling	360
PageShuffleList	360
PageShuffleEnable	360
PageShuffleDelay	361
PageShufflePauseDuration	361
PageShuffleRealms	361
Notes	363

E Index	366
----------------------	------------

Preface

Welcome to Trihedral's VTScada training program. This workbook was designed for use in a classroom, but is also suitable for self-directed courses.

This workbook will match the course outline, providing reference information and exercises. Space was left for you to add notes. Please feel comfortable about writing in this book. Throughout the course, you will have an opportunity to practice each topic as it is taught. The exercises are intended to give you a feel for the capabilities of VTScada and how to use the features in real-life situations.

Please ask questions. It often turns out that the best parts of a course are the discussions that follow on from a good question. Don't be shy about asking yours.

Course Objectives

This course is designed for VTScada developers who have experience creating applications and want to learn advanced skills and techniques. It assumed that you know the material in the introductory-level Operations and Configuration course.

In this course you will learn how to:

- Customize how VTScada looks and works using configuration properties.
- Write expressions to extend tags, widgets and pages.
- Design your own complex tag types.
- Design your own complex widgets.
- Build widgets for high performance HMI displays.
- Re-use pages by adding parameters.
- Fully explore VTScada's report generation options.
- Distribute your applications using OEM layers and ChangeSets.
- Learn to implement advanced security techniques including realms and read-only workstations.
- Create new feature using VTScada's scripting language.
(Or at least, see the fundamentals of that language.)

A few notes about this course:

A set of files (zipped) is included with the course. These include ChangeSets, sample code, and other files that you will need. Take a moment now to ensure that you have these. You are advised to unzip everything to a temporary folder on your hard drive for quick access.

A license key is provided in classroom courses, enabling all features of VTScada. This is a trial license, good only for a limited number of days. If you already have a license key that has not expired (trial or otherwise) then use that instead. If you don't have a license key, then the VTScadaLIGHT option works perfectly well for this course (although you may reach the 50-tag limit at the end of chapter 4 and need to delete a few I/O tags).

Note: This course uses a simulated PLC. From time to time, you may notice a few quirks that come from connecting to a simulation rather than a physical device.

1 A Short Review...

What VTScada is, what it does, and how it works.

VTScada is human-machine interface (HMI) software. Its sophisticated features allow you to create advanced applications for monitoring and controlling systems.

While VTScada is often found in the water supply and waste-water industries, it can be (and is being) used in any industry that requires supervisory control of, and data acquisition from, mechanical processes.

Note: Trihedral strongly recommends that participants take the Level 1 course (VTScada Operations and Configuration) before taking this one. Self-directed study is fine, especially if you study the modules and do the exercises in the [VTScada Academy](#).

At the beginning of this advanced course, it is assumed that you know enough about VTScada to be able to recreate the Level 2 application in short order if given a list of the tags and an image showing the page layout. It contains very little that is not covered in the Level 1 course.

The following chapter provides a review of VTScada components, with an emphasis on ways to customize each component. (The main focus of this course!) If there is any component that you are not sure how to use, this is the time to learn. Ask questions!



Figure 1-1 is typical of a VTScada application. It shows the start (a generator in this case) and the end (various monitoring readouts).

Between the equipment and the visual display, VTScada provides the following services:

- Communication protocols and drivers for a variety of devices.
- Animated widgets for real-time monitoring of processes.
- Output widgets to provide operator control.
- Scaling from raw data to meaningful display information.
- Data logging for reports.
- Configurable alarms to warn when values go outside allowed limits.

- A distributed version control system that allows application updates made on one workstation to be shared between many.
- Account-management services to provide security to an application.
- Internet, XML-based protocols to allow a wide range of remote access.
- And, much more.

Caution: If your VAM already has an instance of the Training Simulator, uninstall it *right now*. The simulation code is subject to change from one version to another and older versions are sometimes not compatible with the current version of VTScada.

Exercise 1-1 Setup...

Note: There are two ChangeSets to install, from two different locations. Pay close attention to the instructions.

1. Install VTScada now. You should run the installation program as an Administrator if you are able. (Do not run VTScada itself as an Administrator. That last instruction refers only to installation.)
 - Agree to the license terms.
 - Use maps downloaded as needed from OpenStreetMap.
 - Provide your name, company and the installation key.
(If you already have a key, with all features enabled, use that.)
 - Do not choose the VTScadaLIGHT option unless you do not have a key.
 - Do install into a new folder, especially if you already have VTScada on your computer. C:\VTScadaClass is suggested.
 - Choose to add to the start menu or create shortcuts as you prefer.
 - Choose to do a Complete installation
 - 64-bit is the best choice for most computers.
 - Do not install VTScada to run as a Windows service.
 - Your choice as to allowing reporting of usage statistics, unless you are installing as VTScadaLIGHT where usage statistics are mandatory.
2. A snapshot ChangeSet is provided with VTScada:
`C:\VTScada\Examples\V12.1 Training Simulator.snapshot.`
Install and run this now. (*Note the path - this is included with VTScada, not the course handouts.*)
3. As soon as the simulator application is running, minimize it.
You're going to get a head start with an application that's partially built.
4. Another ChangeSet is provided in the same zip file where you found this manual.
Level2.ChangeSet. Install and run this now.
All exercises should be done within the Level 2 application.

The behavior of the simulation has a few quirks, but the hardware configuration is realistic. You will use a TCP/IP tag and a Modbus Driver to communicate, exactly as if you had hardware using the Modbus protocol. I/O addresses are provided for the holding registers and input coils (with one holding register address using the /float suffix for the sake of the example).

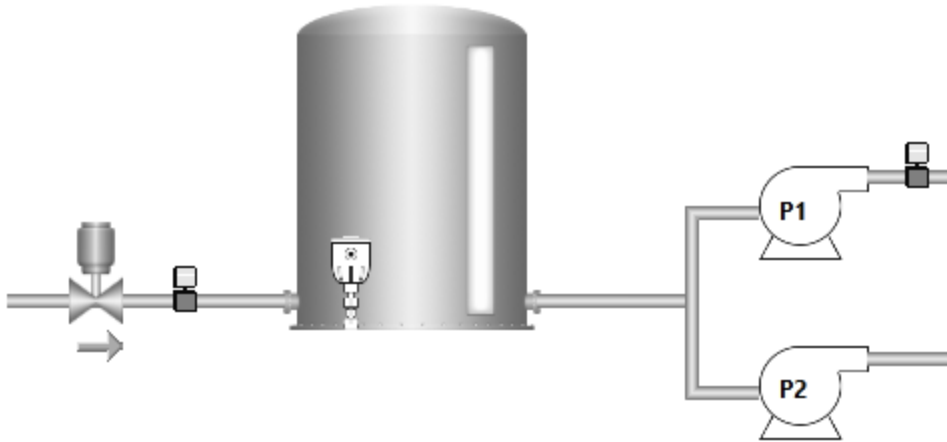


Figure 1-2 The simulated system

- Fluid flows into storage at a varying rate through the valve.
- The storage container is equipped with a level sensor.
- Pump P1 starts in automatic mode, switching on and off as the level reaches high and low setpoints, respectively.
- Pump P2 may be added.
- A number of I/O addresses are provided for monitoring and operator control.
(Refer to the appendix of this workbook for a table.)

The VAM (VTScada Application Manager)

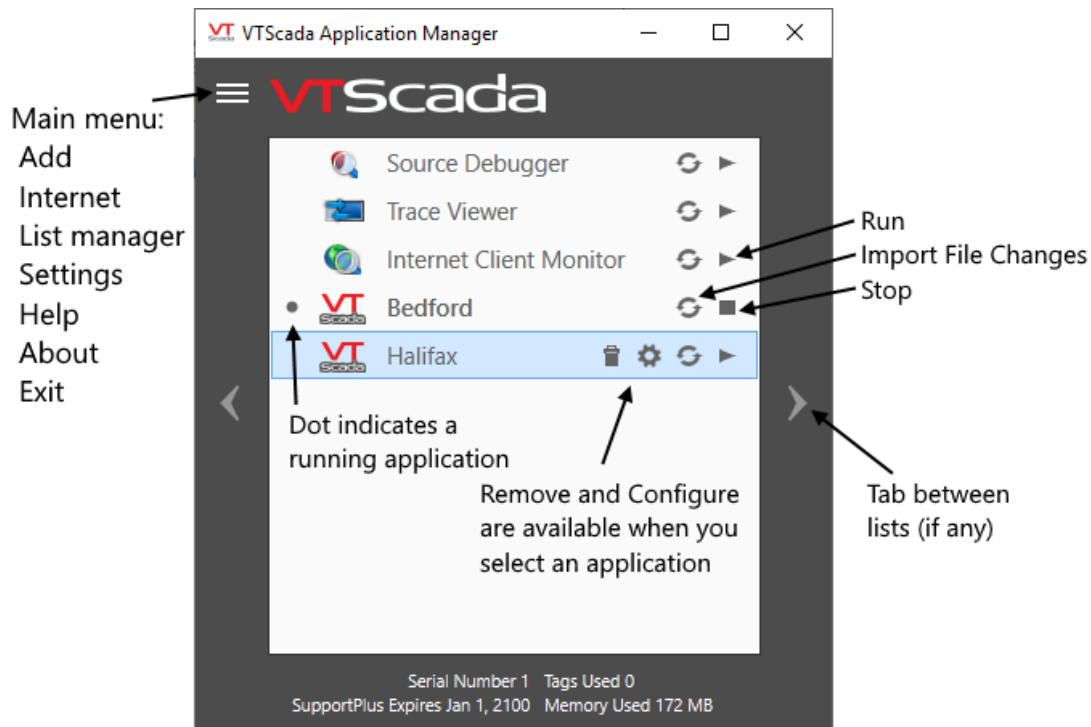


Figure 1-3 The VTScada Application Manager (VAM)

Everything begins here. Most of the applications shown in this example are installed with VTScada, and are used for analysis and debugging.

Opportunities for customization:

- Order of the applications. Click and drag to re-organize.
- List of applications shown. Remove items from the list without uninstalling them.
- Organize your applications into separate lists.
- Color theme. Will also apply to any application that doesn't have its own theme.
- Choice of language for the operator interface.
- Icon for each application. Set within each application's configuration dialog.
- Visibility of the VAM. If your application is configured to run automatically, you may wish to keep the VAM hidden to prevent tampering.
- Create, configure, start, stop, or delete applications.
- Set up the VTScada Thin Client Server. Allow remote access to applications.

Create new applications.

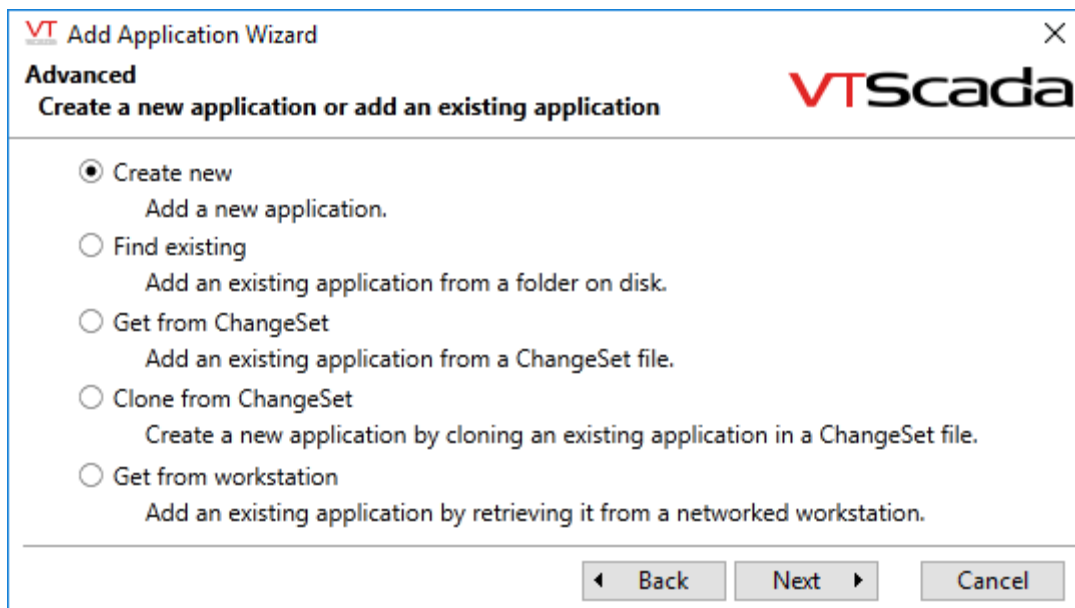


Figure 1-4 The advanced screen of the Add Application Wizard

You can create an application from scratch, or choose between four methods of installing an existing application on the server.

All new applications are built on top of an existing application, also referred to as an OEM layer. This may be one of your own devising, (as will be described during the course) or it may be the VTScada system layer. OEM layers can contribute tag definitions, device drivers, report formats and more to new applications. If you have developed a set of tools, specific to an industry, and are building applications for clients in that industry, having an OEM layer allows you to re-use those tools easily.

Opportunities for customization:

- OEM layer, and all that it can include.
- Customized lists.

- Visibility options.

Configure & Manage an Application

The Application Configuration dialog provides an enormous range of control for each application. The first item in the side menu, Edit Properties, give you access to hundreds of variables that control everything from display colors to the specifics of device driver communications.

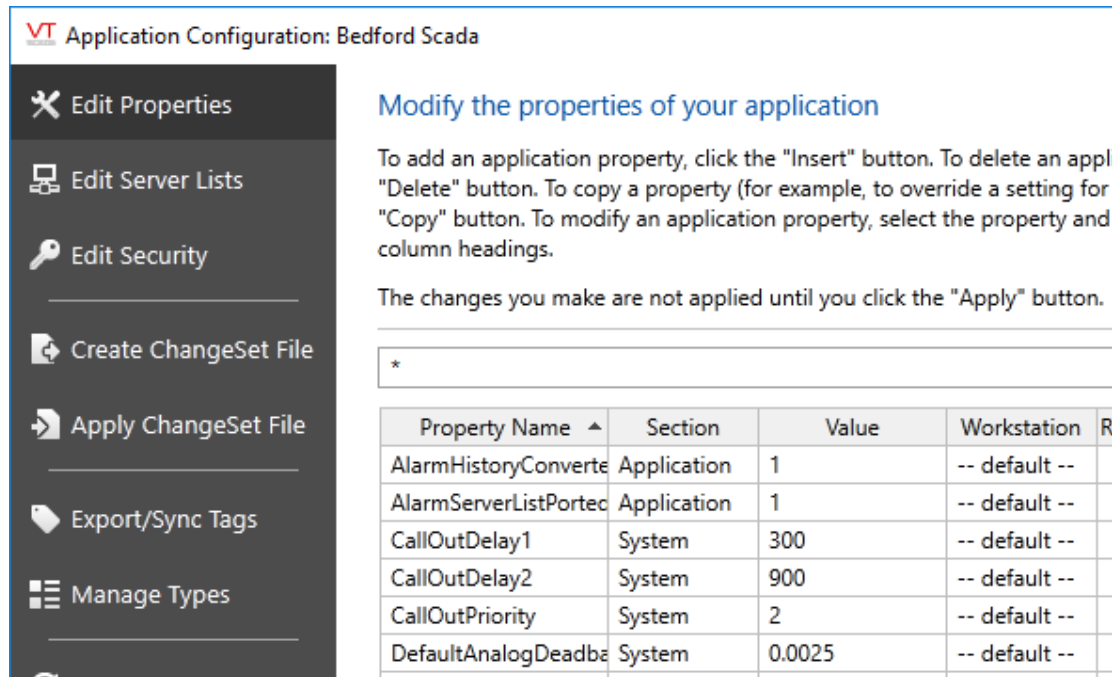


Figure 1-5 Detail from the advanced mode of the Edit Properties page of the Application Properties dialog

You can access and use the Application Configuration dialogs whether the application itself is running or stopped. Note only that for a small number of configuration tasks that involve tags, the application must be running for the tags to be available.

Opportunities for customization:

- Properties to control VTScada behavior.
- Properties to control application appearance.
- Change labels (alternate language versions).
- Create properties for application-specific settings.
- Configure security.
- Configure primary and backup servers.
- Off-line creation and editing of tags.
- Edit user-created tag types.
- Import (or discard) user-edits to source code.
- Version control.

The Idea Studio

Within the application, the primary tool for development and customization is the Idea Studio. This includes the following components:

- Palettes. There are three separate palettes: 1) Animated widgets that represent tag values. 2) Static images 3) Basic shape building-blocks. Used together, these provide equipment information, equipment control, and visual context to the display pages. Build the operator interface by dragging these to pages or to custom widgets.

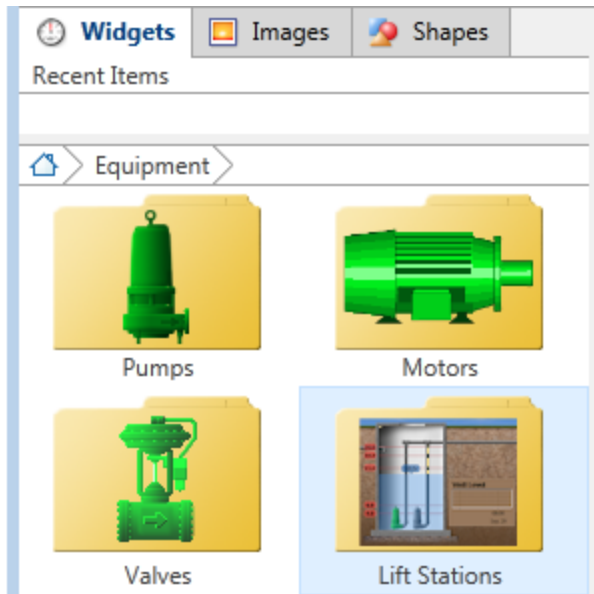


Figure 1-6 Sample widget palette

- Panels. There are three of these on the right of the screen, two of which open automatically in response to things you select on the screen. Panels provide a variety of tools to help you select what objects, or which aspects of selected objects, you are editing.
- File menus. Open, close, create and delete pages and custom widgets as required. Pages and custom widgets are the screens on which the operator interface is displayed.

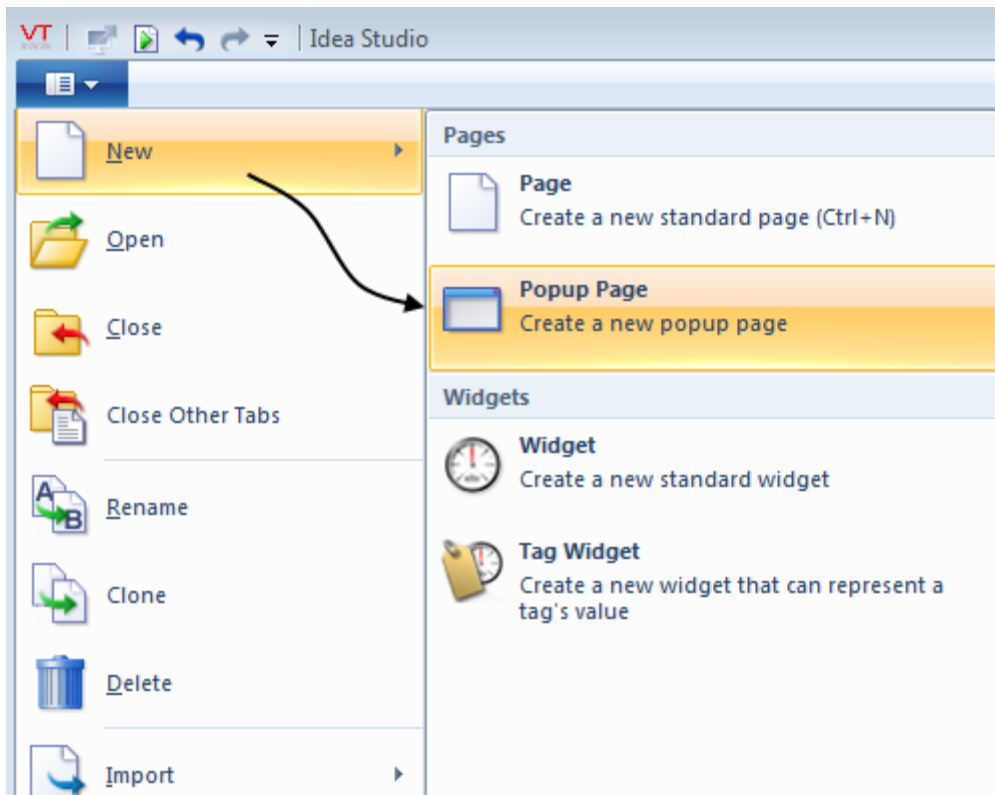


Figure 1-7 The file menu

- Toolbars. These provide the tools for adjusting the properties of each object you draw, as well as tools for selecting and organizing groups of objects.

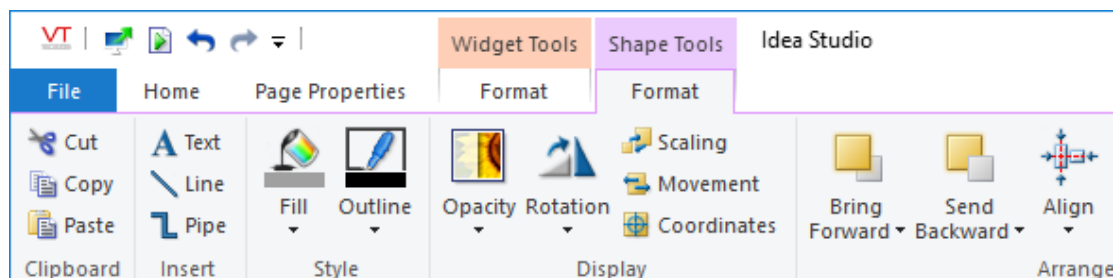


Figure 1-8 Sample formatting toolbar

- Display controls. Scale the display as required to see fine detail, or view a large page on a small monitor. Property-control toolbars are context sensitive, appearing only when an object of the matching type is selected.
- Quick-Access Toolbar. Store frequently-used tools in a toolbar that is always visible.

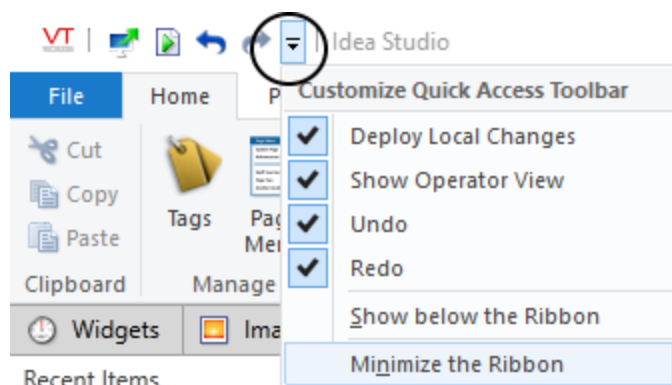


Figure 1-9 Customizing the Quick Access toolbar

Opportunities for customization:

- Palette organization.
- New pages & widgets.
- Control of object appearance & behavior.
- Create and configure the operator interface.
- Put frequently-used tools into the Quick Access toolbar.

The Tag Browser

Tags hold the configuration of the core elements of VTSkada. All communication with PLCs and RTUs is done via the tags that you create. All alarm configuration is stored within tags, as is all data logging configuration. Menu configuration for both pages and the Idea Studio palette is also stored in tags.

The Tag Browser is where you create and work with these tags. It contains tools to filter and sort the tag list so that you can quickly find the specific tags you need.

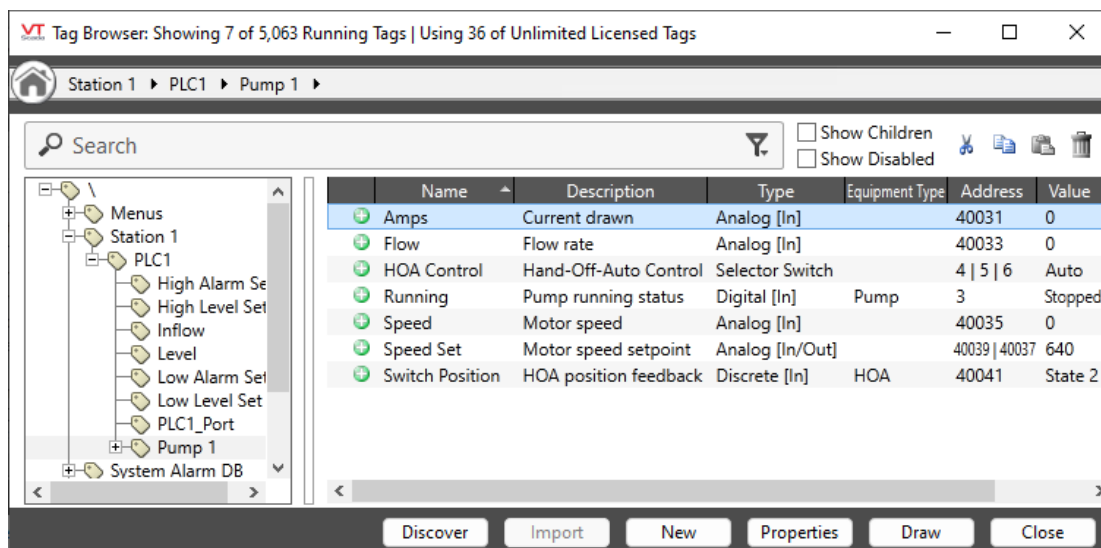


Figure 1-10 The Tag Browser

Tags are designed to be modular. To read a value from an address on a PLC, you will need a port tag holding the communication details (Serial port or TCP/IP), a driver tag to hold the communication protocol (Modbus, DNP3, etc.) and an I/O tag to hold the address register on the PLC, scaling and other details. Logging and alarm configuration is built into some I/O tags, but must be configured as separate tags for others.

When tags relate to each other, such as the tags in the communication chain, it is best to create them in a hierarchy as shown in Figure 1-10, above. In the example shown, Pump, Tank and Valve are user-defined types, built using Context tags.

Opportunities for customization:

- Create and configure tags.
- Create new types of tags.
- Use parameter expressions to automate tag configuration.
- Configure menus and palettes with Menu Item tags.
- Configure visual display standards with Style Settings tags.

Alarms

Alarms are used primarily to notify operators when operational values go outside set bounds. Alarms can also be configured to record operational events without raising any notification.

Alarm configuration is stored within tags. This can be either dedicated alarm tags, or alarm configuration within complex tags such as the I/O and Calculation.

The Alarm Page shows the status of current alarms, or alarm history. Operators use this page to review, acknowledge, shelve, or otherwise work with alarms. You can also create customized alarm notification displays so that operators can view and acknowledge a filtered list of alarms related to some portion of the application while continuing to monitor the related I/O.

The optional Alarm Notification System allows you to send alarms by phone, email or text message to a roster of operators. They can hear or view the alarms, and acknowledge them remotely.

Opportunities for customization:

- Create and configure alarms.
- Create customized alarm displays.
- Control which alarms go to which roster list of remote operators, and automatically switch rosters based on time of day or the date.
- Restrict access to alarms outside an operator's designated areas of concern.
- Create custom alarm lists.

Logging & Reports

An Historian, configured as a tag, will record data from selected tags to a data store. You can allow it to store data using Trihedral's proprietary format, or you can write to one of four supported database programs (not included with VTScada .)

Like alarm configuration, logging configuration is stored either within dedicated Logger tags, or within tags that have built-in logging capabilities.

You can view logged data on screen in a graph (the Historical Data Viewer - HDV) or it use it as the basis for a report created using VTScada's report generator. You can also export the raw data for use in a third party reporting program, or you can purchase the ODBC option and run SQL queries against VTScada's stored data.

Opportunities for customization:

- Rate and timing of logging.
- Destination of logged data - VTScada or 3rd party database.
- Customized display of logged data in the HDV.
- Customized reporting of logged data.

Security

Use privileges granted to user accounts to control access to VTScada features and to the pages and output tags that you create. To simplify management of security features, you can create roles, which are job descriptions having a certain set of privileges, then grant those roles to accounts. Privileges can be further limited by tag context, so that a single privilege will give operators access to one part of the operation but not another.

Opportunities for customization:

- Roles for job descriptions.
- Rules to limit access to defined contexts.
- Security groups combined with Tag Area Filtering to limit access to tag selection and alarms.
- Windows Security Integration.

Server Configuration

If you have more than one VTScada license and some of those licenses include the ability to configure a workstation as a server, then you can distribute an application across multiple workstations. Use this to configure redundancy so that the loss of a server has no impact on any function of the application. It also enables load distribution so that separate parts of a large application can use different workstations, maximizing the use of resources.

After building an application, you can configure it to start automatically, then re-install VTScada to run as a service instead of as an interactive desktop program.

Opportunities for customization:

- Primary and backup server lists.
- Shared server lists for drivers.
- Load distribution.
- Run VTScada as a service.

OEM Layers

OEM == Original Equipment Manufacturer. Originally intended so that hardware companies could create a set of VTScada tools that would be available in all their customer's applications.

This feature is in use and available to you as well, even if you do not manufacture and sell control equipment. If there is any chance that you might someday want to copy tags or other custom development work from one application to another, then your best course of action is to create those in an OEM layer right from the beginning.

You can use any VTScada application as the OEM layer for another application. The other application inherits all of the tags, pages, widgets, configuration properties, report definitions, and customized code in the underlying layer. Some of these features are copied to the new application and some are not copied but are available for use. The topic towards the end of the course discusses that idea in detail.

As an example, all VTScada applications are built on top of at least one OEM layer: The VTScada System Library. That application holds everything that you recognize in VTScada including the Tag Browser, all of the tag definitions, the Idea Studio, Application Configuration dialog, and more.

More about this later. For now, whenever you see the term "OEM layer" just think "underlying application".

2 Edit Properties

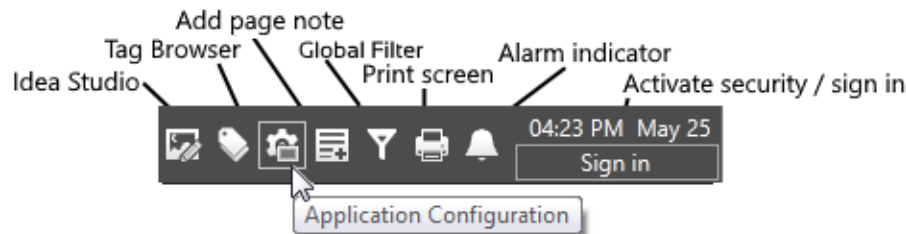
Properties control how VTScada looks and how it works. For example: if you would like to silence all alarm sounds while you're building an application you can set AlarmSoundDisable to 1. If you prefer that each new tag not start with the same configuration as the last one, set RememberNewTagParameters to 0. And if (by chance) you're using typed Modbus addressing, you need to consider whether ModiconTypedAddr1Offset needs to be set to 0 or 1.

The list goes on for over 1000 properties.

In the Application Properties dialog, only a few dozen properties are available in the Basic mode. Most can be found in the Advanced mode but not all. There are also...

- Hidden properties.
Most are hidden so that access can be better controlled. You can view these by opening a configuration file, but only an authorized user can import any changes.
- System properties.
Affecting the VTScada program itself rather than an application, these properties are stored in Setup.INI, found in the installation folder. Changes to that file do not take effect until VTScada itself is restarted.

Open the Application Configuration dialog



The Application Configuration dialog contains a large collection of tools in addition to the Edit Properties page.

Note: After security is enabled, you will need the Configure privilege to open this dialog.

Open the Application Configuration Dialog from the VAM:

1. In the VTScada Application Manager (VAM), select the application.
2. Select Application Configuration.

Open the Application Configuration Dialog from within an application:

1. While an application is running, select the Configure button.

Open the Application Configuration Dialog from within the Idea Studio:

1. Open the Idea Studio.
2. Expand the File menu.

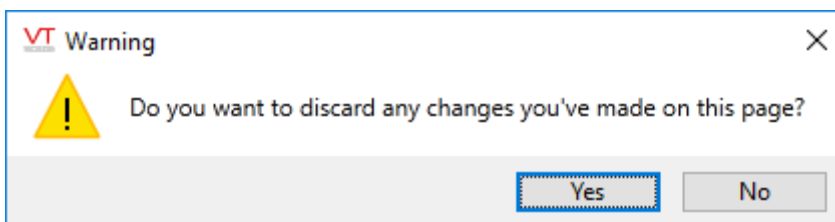
3. Select Application Configuration.

Tip: Keep a tidy development environment. If you leave the Application Properties dialog open while you (or others) work on other tasks, you can expect to see warnings that settings may have changed, even though the change may have been to a file that doesn't affect configuration properties. Also, while a tag properties dialog is open, certain other development tools will not open.

When moving from one development tool to another, unless you plan to return to the first immediately, it is better practice to close the first window before opening the second.

Unsaved Changes

If you make changes in the Application Configuration dialog, then navigate to a different page, or close the configuration window, you will see the following message.



Click Yes to discard the changes and continue navigating or No to retain the changes and continue working on the same page.

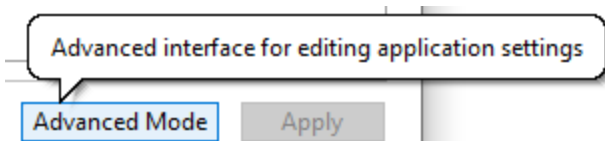
The Basic mode makes it easy to adjust the most frequently used properties, and that mode was covered in a lesson module for the introductory course.

Many more properties are available in the Advanced mode. Use the comments and the documentation to discover what each property controls. Properties shown in black font can be edited directly. Properties shown in gray font must be copied first, using the Copy button at the bottom of the list.

Edit Properties with the Advanced Mode

Many more properties can be found by clicking the Advanced Mode button at the lower-right of the Edit Properties dialog. This changes the view to a list of all properties in effect for this application, including those configured in OEM layers. These give you extensive control over how your application looks and works.

These notes do not attempt to describe each of the nearly 1000 properties available. You are encouraged to refer to the property reference section of the VTScada documentation to explore the options.



... Changes the display to...

Modify the properties of your application

To add an application property, click the "Insert" button. To delete an application property, select the property to delete and click the "Delete" button. To copy a property (for example, to override a setting for a particular workstation), select the property to copy and click the "Copy" button. To modify an application property, select the property and modify the property fields. You can sort by clicking on the column headings.

The changes you make are not applied until you click the "Apply" button.

☐ Hide OEM Properties

Property Name	Section	Value	Workstation	Restart	OEM	Comment
PortedLabelOverrides	Application	1	-- default --			
RosterDelay	System	10	-- default --			The delay between callouts to contacts on a roster lis
SQLQueryHideLegacyTables	System	1	-- default --			If TRUE prevents v11.0 style tag tables from showing up when querying the V1
UpgradedTagUniqueIds	Application	1	-- default --			
#DataLines	System	100	-- default --	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Number of communications display lines on the CommData page (maximum
ABBTotflowFailoverCount	System		-- default --	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Number of retries before failing over for all instances of the ABB Totflow dm
ABDF1IgnoreAckNak	System	1	-- default --		<input checked="" type="checkbox"/>	Ignore ACK/NAK responses on AB DF1 Full Duplex protoco
ABDF1SerialMacraac	System	A	-- default --		<input checked="" type="checkbox"/>	Maximum number of macraac in flight on AB DF1 Full Duplex protoco

Caution: Do not reassign any VTScada property for another use. All the default properties are needed for their intended purpose.

Caution: The Application Configuration dialog will add properties only to the .Dynamic file. Any property that belongs in .Startup must be added using a text editor and then the changes imported. ([Import File Changes Tool](#)) Existing properties in either file can be changed using the Application Configuration dialog.

Gray font / Black font?

Properties displayed in a black font are defined in the current application and can be edited there. Properties displayed in a faint gray font are defined in a lower layer. These OEM properties affect your application but cannot be modified within it.

To override a gray (OEM) property, select it, then use the Copy button to copy it to your application. After copying, you can change the local instance to the value you want. Changes made to that property in the underlying application will no longer affect your application.

The following tools are available to help you work with the property list in the advanced mode:

- Hide OEM Properties removes all but the local properties from view.
- Filter the list, by entering a portion of a property name and the * wildcard, then press enter or press the filter button. For example, enter Disp* to find most Display Manager properties.
- Sort the list by clicking on any column header. Reverse the sort order by clicking a second time.

Property Attributes

Every property has seven attributes, which can be seen in the seven table headings. These are:

Property Name	Simply, the name of the property. Most can be found in the Reference chapters of the VTScada documentation.
---------------	---

Section	Every property must belong to a section. These cannot be assigned randomly. VTSada will ignore any property definition that is located in the wrong section (with a few exceptions). Common sections are [System] and [Application].
Value	All properties are stored as text, including those written out as numbers. Some properties are set to blank. In most cases, blank means "use the default", which might mean, "don't use this feature".
Workstation	You can define properties whose value applies only to the named workstation. These will be saved in files named for the workstation. Auto-Deploy is an example of a workstation-specific property.
Restart	If a property has a check mark in this column, then any change to that property will require the application to be restarted before the change will take effect. Properties that require a restart are stored in the file Settings.Startup. All other properties are stored in the file Settings.Dynamic. Note that, as soon as you click the Apply Changes button to save a property that requires a restart, no further development work can be done in the application until that restart occurs. Therefore, choose your timing of such changes carefully.
OEM	Any property with a check mark in the OEM column is defined in that layer. There may or may not be a copy of the property in the current application layer. If so, that copy will override the value in the OEM layer.
Comment	An optional description of the property. When working directly with the configuration files, this must be stored on the line below the property definition or it will be taken as part of the value.

Change (or add) a property value (Advanced mode):

The general steps are as follows:

1. Open the Application Configuration dialog.
2. Open the Edit Properties page.
3. Find the property...
 - a. The properties used most often, can be found in one of the four tabs of the basic mode: Display, Alarms, Historical Data Viewer or Other.
 - b. For other properties, click the Advanced Mode button to view a list of all. The list can be sorted by name, type, and value.
 - c. Advanced Mode: If the property has not yet been set in the current application, copy it from the underlying OEM layer. OEM properties are shown in gray and have a check mark in the OEM column.
 - d. If the property is not listed, click the Insert button to add it.
4. Change the property's value.
5. Apply the changes.
You may need to deploy the change if *Automatically deploy local changes* is not selected in the "Other" tab of the basic mode.

Widgets to change a property value

If you need to change the value any property on a regular basis, it may be inconvenient to open the Application Properties menu. Instead, you can use an [Edit Property Checkbox](#) or an [Edit Property Field](#) to modify that property on any of your pages. This may be especially helpful when you need to allow an operator to change a property but do not want to grant Application Configuration rights to the operator's role or account.

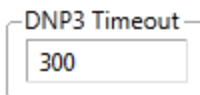
The Edit Property Checkbox widget provides a way for operators to switch a specific property on and off. You choose which property can be edited when configuring the widget. These widgets can allow access to certain properties that need to be updated, without granting operators full access to the Application Properties dialog.

If the property can have values other than zero and one, use the Edit Property Field widget.

☒ Echo Dialed Alarms

The Edit Property widget is used to allow operators to change an application property value. You choose which property can be edited when configuring the widget. These widgets can allow access to certain properties that need to be updated, without granting operators full access to the Application Properties dialog.

If the property can have values of only zero and one, use the Edit Property Checkbox widget.



Configuration Examples

It's easy to underestimate how much control VTScada gives you through configuration properties. A few examples might help to expand the horizon.

Alarm Properties

AlarmDatabaseGroups - If you have many Alarm Database tags, use this to create named groups that display alarms from multiple databases.

MinMuteDuration - Set the minimum time duration for alarm muting. (There is also a property for the maximum value.)

ApplyMuteSilencePerUser - Controls whether the Mute and Silence controls of the Alarm Page affect only the current user.

AlarmSpeechEnable - Speak alarms at the server instead of using the alarm tones.

AckAllAcksOnlyVisible - By default, the Acknowledge All tool in the Alarm Page and Alarm List will acknowledge all alarms that pass the filter currently in effect, including those on the next page of a scrolling display. When set to 1 (TRUE), only those alarms that are visible in the list will be acknowledged.

AlmPgStartList - Selects the default list of alarms to display when the Alarm page opens.

Security

MaxFailedLoginAttempts - How many attempts before the account is locked for X minutes.

MaxRateFailedLoginAttempts - How frequently attempts can be made before switching from the maximum attempt value used for operators to the maximum value used for automated cracking programs.

Tags

RememberNewTagParameters - Whether new instances default to the values from the last instance created or if every tag starts out blank.

DefaultAnalogDeadbandFractionOfFullScale - The default deadband when logging analog values.

ParmChangedColor - The green background when a tag parameter is changed.

ParmInfoDateFormat - How to display dates in the tag modification tooltip

Application Properties

AutoActivate - Activate and start are not the same thing. For example, an OEM layer must activate before the dependent layer can run, but the OEM layer itself does not run.

AutoStart - Run this application every time VTScada starts. Required when you run VTScada as a Windows service.

DoNotStart - Maybe your OEM layer isn't meant to run.

HideFromVAM - Maybe your OEM layer shouldn't even be seen.

NoSoftDriverFailure - Controls whether drivers should switch to backup servers on failed communications.

Logging

OperatorLogTemplate - controls what information is included when VTScada logs operational events.

HistorianConnectionRetryDelay - the number of seconds to wait before retrying a connection.

TraceUserConfigActions - Log operational events.

Exercise 2-1 A development page for all your applications

Configuration File Structure

Configuration properties are stored in one of the following files:

- Setup.INI. Located in the top level of the VTScada folders, these affect VTScada in general. Changes can be made only by editing this file using a text editor. Changes do not take effect until VTScada restarts.
- Settings.Dynamic. Located in your application folder. Properties that can be changed without needing to restart the application. These are the majority of the properties that appear in the advanced mode of the Edit Properties panel.
- Settings.Startup. Similar to Settings.Dynamic, except that changes to these properties take effect only when you restart your application.
- Workstation.Dynamic and Workstation.Startup. "Workstation" must be replaced by the name of the workstation where these properties will be in effect. Workstation-specific files are stored in a sub-folder of the application. For example:
C:\VTScada\BedfordDemo\WorkstationSettings\harvie-pc.dynamic

Note: Changes made directly to any of an application's configuration files are ignored until an authorized user runs the Import File Changes command.

All configuration files share the same structure:

```
[SECTION_NAME]
PropertyName = Value
    ; Comments
PropertyName = Value
    ; Comments

<HIDDEN_SECTION_NAME>
PropertyName = Value
    ; Comments
```

Rules:

- Section names are marked by square brackets.
- Hidden sections are marked by angle brackets.
- Every property must be stored in the appropriate section. Properties stored in the wrong section will be ignored.
- Property assignments take the form, Name = Value.
Text values are not enclosed in quotation marks. Quotation marks may be included as part of a label.
- Comments are both on a separate line, and are marked by a leading semi-colon.
- Attempting to add a comment after a value, and on the same line as the value, will break the property.

Properties that can be changed without a restart of the application must be stored in the .Dynamic file. Properties that are loaded only on application restart should be stored in the .Startup file. Moving a property from .Startup to .Dynamic does not result in it being able to change without restarting the application.

Hidden Property Sections

Some application properties can never be viewed or edited in the properties list. These are defined in one of the hidden sections within Settings.Dynamic or Settings.Startup. Security defaults (time-outs, group name delimiters, etc.) are examples. Alarm-Area Filtering (covered later in this course) is another.

Hidden sections are those whose section heading is enclosed in angle brackets. <SecurityManager-Admin> is one of these. As a developer, you should be aware of these sections because you may need to edit properties within them, although this tends to be rare.

Direct Editing of Settings.* (and other) Files

A user-copy of both the Settings.Dynamic and Settings.Startup files can be found in every application. For example, C:\VTScada\MyApp\Settings.Dynamic. As a general rule, use the VTScada dialogs to edit configuration properties. It is rarely necessary or convenient to work directly with the files. One exception to that is that changes can be made to the properties in a hidden section only by directly editing either the Settings.Dynamic or Settings.Startup files. Note that none of those changes will have any effect unless subsequently imported by someone whose account has the Edit Files privilege.

Before using the Import File Changes button in the VAM, you are advised to review those changes by using the Import/Export Files page of the Application Properties dialog.

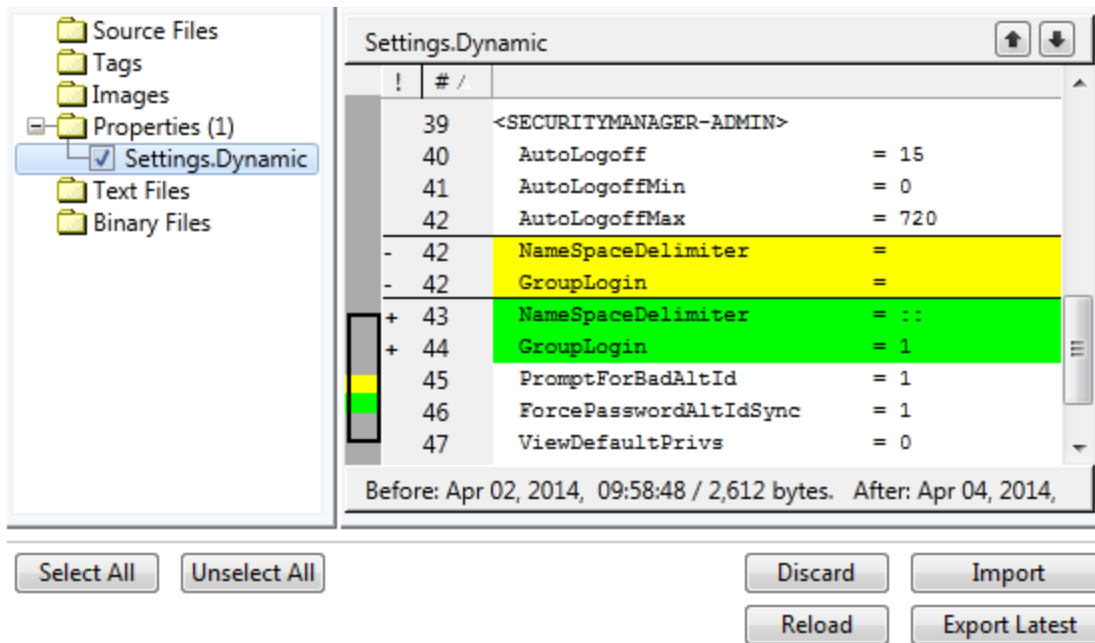


Figure 2-1 Current (unchanged in code) values shown in yellow. New (not yet imported) values shown in green.

User files have the following attributes:

- VTScada maintains them, writing updates whenever changes occur in the application. These updates are merged into the files rather than over-writing them.
- VTScada ignores changes made to user files until an authorized user imports those changes.
- You can force a write from VTScada to the user files at any time. You have the option of merging changes or of over-writing the user file with the current contents of the working file.

Other Configuration Files

While discussing files that you can edit directly, note that in addition to Settings.Dynamic and Settings.Startup, you may also edit the following:

- AppRoot.SRC – contains constant definitions and module declarations for your application.
- Page source files in the \Pages sub-folder - contains the source code defining every page in your application.
- Image files in the Bitmaps sub-folder – contains your additions to the application's library of graphic images.
- User-created widgets, pages and tags.
Just be aware that if you edit the source code for a user-defined tag, it will then count against your license limit.
- AlarmListFormats.XML Must be copied from the \VTScada\VTs folder to your application before editing. You can use this XML file to modify the columns in the provided alarm lists or to generate a completely customized list format. (Note that

while the file will set initial column widths, on-screen changes to the widths are stored with the user-account and do not reload from the XML file.)

User-files that you should never attempt to edit include:

- Accounts.Dynamic - plain text and intentionally indecipherable by humans. The encoded contents of this file are tied to the application it was generated within and cannot be transferred to other applications.
- Servers.XML – stores the network server configuration in an XML format. Use the Application Configuration tools to edit this rather than attempting to do so directly.

Exercise 2-2 Modify System Properties

To complete the following exercises, you will need to refer to the VTScada Help File.

Note: Each step is independent. Apply your changes and view the results after each. Following all the steps in this exercise, feel free to change the properties back to their original values.

1. Open the Application Configuration dialog and set the Edit Properties display to the advanced mode. (You should revert these changes after seeing the result.)
 - i. Copy DispMgrHoriz and set to 2.
 - ii. Copy DispMgrVert and set to 1.
2. As developers add tags, each subsequent instance of a type is automatically configured to be the same as the last. This can be convenient if you are adding a series of nearly identical tags, but may be inconvenient if it results in properties being incorrectly set the same from one tag to another. In this exercise, you will create a convenient way to switch the property controlling that feature, on or off.
 - i. Open the Idea Studio
 - ii. Create a new standard page. Name it `Properties`.
 - iii. Expand the Tools palette, then the Standard Library.
 - iv. Add an Edit Property Checkbox to the page.
 - v. Configure its properties as shown:

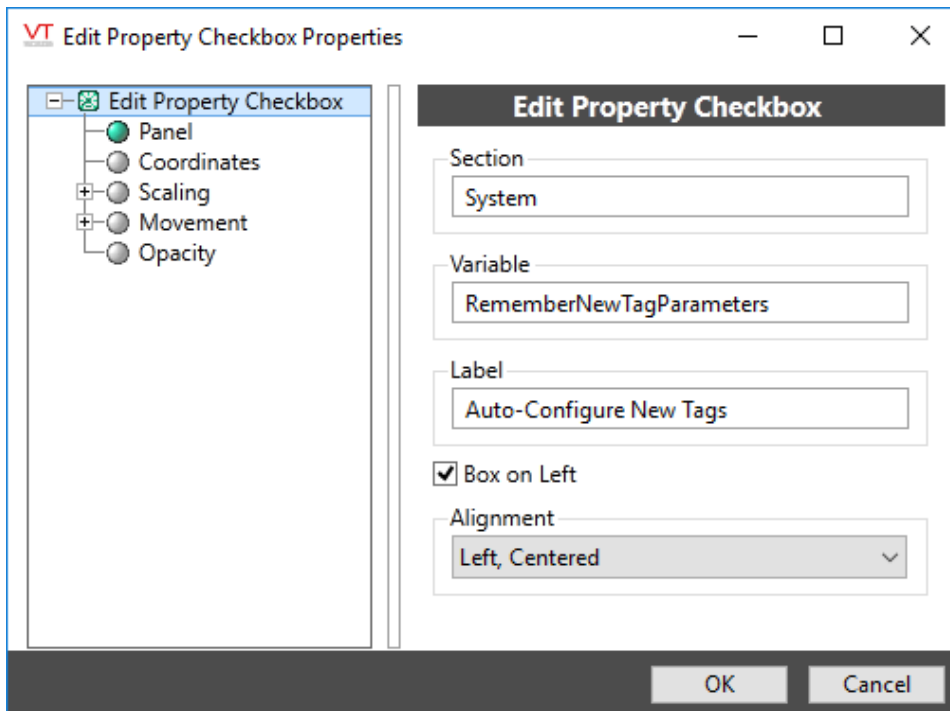


Figure 2-2 Toggle auto-configuration of new parameters

- vi. Close the Idea Studio by clicking the Operator View button.
 - vii. With the box selected, open the Tag Browser and add two Font tags at the root level. (Do not use copy / paste!)
The point is to test the 'Remember New Tag Parameters' feature. Configure the first tag with random values, then notice that all except the name will be used in the second tag. A side effect of the exercise is that you're going to learn about Font tags as you figure out how to configure one.
 - viii. Deselect the check box.
 - ix. Create a third Font tag.
Note that the fields are not pre-configured this time.
 - x. Delete your Font tags and close the Tag Browser.
3. Your manager has requested a strict limit on the number of pop-up pages that can be opened at the same time.
- i. Set the limit so that no more than two pop-up pages can be opened at a time. (Hint: search on *pop* in advanced mode.)
 - ii. Set the pop-up behavior so that if a subsequent pop-up is opened, earlier ones will close automatically.
 - iii. Test this by right-clicking in the text version of the page menu to open VTScada system pages as pop-ups.
(A system page is any built-in VTScada page.)

Workstation-Specific Properties

Any property can be assigned a value that will apply only to the instance of VTScada running on a specific workstation. You might use this feature to set display properties or security features that will be in effect only for a given workstation. Workstation-specific properties are mandatory when configuring Tag Area Filtering or Alarm Area Filtering. And, in combination with expressions, these can also be helpful for certain driver configurations such as Modbus Plus PLC addressing, where the link path to a device can vary from server to server.

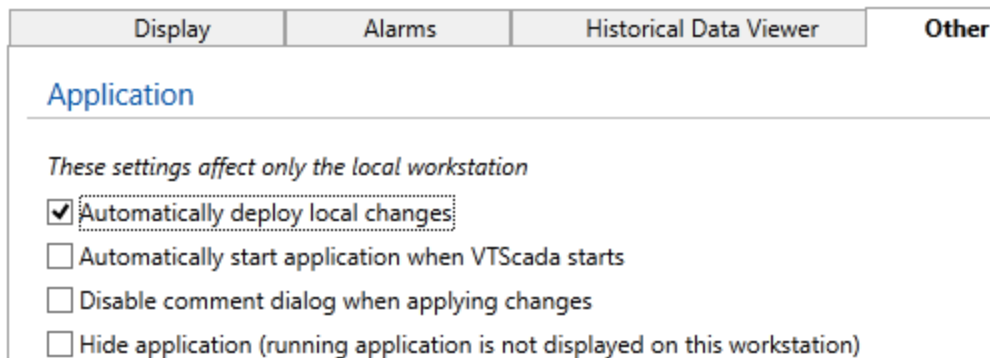


Figure 2-3 Workstation-specific properties in the Application Configuration dialog, Other tab

You can create a workstation-specific version of any property, not just the four shown here. Workstation-specific properties are stored in a file named after the workstation where they will apply and having an extension of either .Startup or .Dynamic. These are located in the WorkstationSettings sub-folder of your application.

Caution: The Application Configuration dialog will add properties only to the .Dynamic file. Any property that belongs in .Startup must be added using a text editor and then the changes imported. ([Import File Changes Tool](#)) Existing properties in either file can be changed using the Application Configuration dialog.

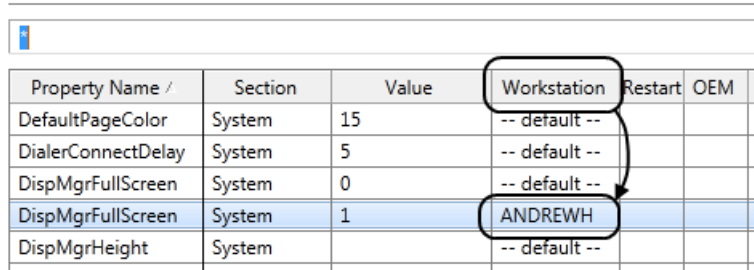
For thin client connections, the name of any remote machine making a client connection cannot be known. You cannot create a workstation-specific setting for any particular remote client. You can, however, create workstation-specific settings for all Internet client connections to a specific VTScada Thin Client Server by designating the name of the computer hosting the VTScada Thin Client Server. This may be especially useful if you have one server for Internet connections from within your trusted network and another in a DMZ (demilitarized zone) for connections from the Internet in general and you wish to have separate rules for each.

Tip: On a related theme, if you want to set display properties that apply only to thin clients, consider one or more [Realm Display Setup Tags](#)

Modify the properties of your application

To add an application property, click the "Insert" button. To delete an application property, click the "Delete" button. To copy a property (for example, to override a setting for a particular workstation), click the "Copy" button. To modify an application property, select the property and modify the column headings.

The changes you make are not applied until you click the "Apply" button.



Property Name /	Section	Value	Workstation	Restart	OEM
DefaultPageColor	System	15	-- default --		
DialerConnectDelay	System	5	-- default --		
DispMgrFullScreen	System	0	-- default --		
DispMgrFullScreen	System	1	ANDREWH		
DispMgrHeight	System		-- default --		

In this example, the setting for the variable, `DispMgrFullScreen` will apply only to the VTScada installation running on the workstation, ANDREWH. (`DispMgrResizable` must also be set to 0.)

Workstation-specific properties are stored in files named after the workstation to which they apply. For the example given, there will be a file named, "ANDREWH.Dynamic" in the sub-folder, "WorkstationSettings". (C:\VTScada\BedfordScada\WorkstationSettings\ANDREWH.Dynamic")

The structure of the Workstation files is identical to that of the Settings.Startup and Settings.Dynamic files. All of the same rules for section names, comments, etc. apply.

Workstation-specific properties for hidden sections

Properties stored in hidden sections cannot be created or modified using the Application Configuration dialogs. You must work directly in the Settings.Dynamic file for these.

As an example, perhaps you wish to set the property [RemoteAuthMethodsDisallowed](#) to "usernamepassword", but only for a VTScada Thin Client Server located in your organization's DMZ. The process is simpler if you already have at least one workstation-specific property defined for that machine. Therefore, the recommended process is as follows:

1. Use the Application Properties page to insert a new property with a name that does not match any existing property. (Perhaps, "nada".)
2. When defining this property, set the workstation to the name of the machine where you want create a workstation-specific hidden property.
3. Upon saving the new property, the WorkstationSettings folder will be created and a workstation-specific version of the Settings.Dynamic file will be created within it. You could rename this to `.Startup` if the property you are setting loads only on application restart.
4. Open this file with a text editor and add the appropriate hidden-section label and the properties with values for that workstation.
5. Optionally, delete the property you created in step 1 of this process.
6. Use the Import File Changes button in the VAM to import the updated version.

System Properties - Setup.ini

There is another properties file to know about: Setup.INI. This contains properties that apply to VTScada as a whole, rather than to any individual application¹. Some of these are intended for the use of system integrators, to allow them to customize the overall appearance of VTScada. Others affect VTScada operations such as the use of XML features or the response to pending UPS battery failure.

Note: Changes you make to Setup.ini will not be over-written by subsequent updates of VTScada. This differs from the behavior of VTScada prior to the release of version 12.

Note: Comments (if provided) must be on a following line, and must begin with a semi-colon.

Settings within Setup.INI will be read only when VTScada starts. The following sections are included:

[APPS]

A listing of the applications in the VAM. It's best left for VTScada to manage, especially if you have custom lists.

[OEM]

Properties that you can use to customize VTScada. These include the following:

NoSplash

When set to 1 (true) the initial splash screen animation will not run when VTScada starts.

HideWAM

When set to 1, the VAM will not display. Use with caution! This can be helpful when an application has been set to auto-start and you want to ensure that no-one at the customer's site has access to the VAM. Before using, make certain that one application has been set to auto-start and that it contains a security account that possesses the privilege, Application Manager View.

Note: "HideWAM" is a carry-over from the days when it was the "Web Application Manager" not the "VTScada Application Manager". It still exists for backward-compatibility, but you are advised to use "HideVAM" for custom code that will change the setting.

[System]

System-wide properties including the following examples apply to your UPS:

```
OrderlyShutdown          = 1
; Enable / Disable orderly shutdown
ShutdownOnLowBattery     = 0
; Shutdown VTScada when windows reports battery as "low"
LowBatteryPercent        = 10
; Shutdown VTScada when battery % is less than this amount
```

¹As usual, exceptions apply. The section, [LAYER] contains properties that relate to applications. If defined only in Setup.INI, then these properties will apply to all applications, but any can be added to the [SYSTEM] section of an application's Settings.Dynamic file to override that property in a specific application.

2 Edit Properties

```
LowBatteryTime          = 15
; Shutdown VTScada when battery time (minutes) is less than this
```

[Layer]

Properties that can be copied to an application, but will otherwise affect all applications. Most relate to synchronization timing or behavior of the version control repository. Notable properties in this section include:

```
AutomaticDeploy = 1
; All changes are automatically deployed when this flag is non-zero
RepositoryCommentMinLen = 0
; Minimum number of characters required for repository comments
RepositoryCommentDisable = 0
; TRUE to disable prompting for repository comments
```

[Remote]

Properties that relate to the VTScada Internet Connection, including the folder path to the monitor log file.

[Themes]

Theme definition codes. Modify the existing themes or create your own. Each theme definition used the following format:

Theme = ThemeName, Hue Offset, Saturation, Brightness, Contrast

For example:

```
Theme = Grey, 0,0,1.1,1
Theme = Navy,-15,2,0.7,1
```

To create a new theme, you will need a paint program that can represent hue as a wheel, where 0 is blue. The hue value for the theme is the offset clockwise or counter-clockwise around the hue wheel.

[Trace]

Properties related to data collection for debugging.

[SQL_*]

Data type definitions for supported SQL programs

[TextFileExtensions]

[FileExtensionClasses]

FileExtensionClasses and TextFileExtensions work together to help define which files within an application folder can be added to the file manifest and what type of file each is, based on the extension.

[CriticalConfigurationFiles]

Files that may not be removed from the manifest.

[Clients-AdditionalAllowedOrigins]

Relevant to those using the VTScada Anywhere Client or the Excel Add-in. See: [Domain Aliases \(CORS\)](#)

Add any domain names or IPs that are used in connection URLs and that are not already included in the server list for the VTScada Thin Client. Set each domain name to 1.

```
[Clients-AdditionalAllowedOrigins]  
myservice.com = 1
```

[Clients-AdditionalAllowedHosts]

Specify a white list of hosts for which VTScada will accept HTTP requests. If populated with any host name(s), VTScada will accept requests only for the names in the list. If not populated, VTScada will accept requests for any host name.

For example, your server might be known as "A, A.domain.com, or B.domain.com", but you might want to allow connections only to "A.domain.com"

The list should be populated as shown in the previous example for the [Clients-AdditionalAllowedOrigins] list.

```
HostName = 1.
```

[Mapping-FontsToFontFiles]

Maps fonts used in VTScada to file files.

[Mapping-TimezonesIANAToWindows]

Maps timezone names used in VTScada to Windows timezone names.

[HTTP-Unauthenticated]

Resources accessible via HTTP without authentication

[StandardServerLists]

Server lists to choose from when adding a new service in the Edit Server Lists panel.

[SlippyMapRemoteTileSourceX]

Where X is a number that must increase incrementally from 1 for each defined tile source. Defines the location and other required information for tiles to be used in a slippy map.

3 Expressions, Part One

In this chapter, you will get a preview of VTScada programming. You will learn how to compare values from two or more tags, create complex tags that self-configure, build triggers, configure alarm suppression, display context-relevant operational instructions, and more. All of these tasks are achieved with expressions.

An expression is "any calculation that returns a result". In more practical terms, an expression is something that...

- Can combine or compare multiple tag values for monitoring or reporting.
- Can signal a need for control actions based on any set of system conditions.
- Can consider the time, date, signed-in operator, system status, etc.
- Can extend the capabilities of VTScada.

Why Create an Expression?

Many reasons. With an expression you can:

- Create a custom trigger for an alarm or event.
- Perform calculations using values from several I/O tags.
- Configure alarm suppression rules.
- Create context-sensitive operator instructions, telling them what to do in the current situation.
- Create custom tag types with many I/O child tags, all of which configure themselves based on settings in one parent tag.
- Hide controls based on a user's security privileges.

And much, much more.

Why Not Create an Expression?

As the adage says, "don't reinvent the wheel". All too often, people end up calling technical support for help with an expression only to be told that they could have simply used the Trigger tag, the Multi-Write, or one of the tags from the Analytics group. Always take a moment to look for existing tools before attempting to build one from scratch.

Also: After you create an expression, it becomes your job to maintain it. Someone will need to update tag references, etc. whenever the application is changed.

General steps to create an expression:

In any VTScada tag configuration field that has the options, Constant, Expression, Tag...

Figure 3-1 Expressions are frequently used for tag configuration

1. Click the Expression option to select it.



2. Click the expression editor button.
3. Enter an expression into the editor window.

Figure 3-2 The expression editor window can be re-sized as needed.

4. Click OK to save your work and return to the tag configuration.

Tip: The maximum length of an expression is 65535 characters.
If your expression approaches this limit, you should consider whether there might be a shorter way to write it.

Syntax Rules for Expressions

Note: You cannot save an expression that contains a syntax error. For example: unbalanced parenthesis "2 + (2/3", or using an operator without an operand "2 + ". It is perfectly legal to save an expression that will have an INVALID result, such as any number divided by zero.

Syntax and notes for expressions:

- Operators are symbols such as plus and minus signs. Operands are the things being operated on by the operators. In the expression $2 + 2$, the digits "2" are operands and the "+" is an operator.
- Spaces between operators and operands.
These are not required but they are recommended. The spaces will increase the clarity of your code and help you avoid errors.
- Extra spaces are ignored.
- Line breaks are ignored, except that they count as a space.
- There is a precedence to the order in which mathematic operations are performed. For example, multiplication before addition. You can use parenthesis to improve clarity and to control the order of the operations. $((2 + 3) * 5)$.
- Close what you open; parenthesis (...), quotation marks "...", etc.
- Text must be enclosed in double quotation marks.
- To display a quotation mark in text, use a doubled set of quotation marks:
"The computer said, ""Hello World""."
- Text not enclosed in quotation marks is taken as the name of a variable. $2 + X$.
If the variable cannot be seen in the current scope of the expression you will get an error message.
- VTScaDa text, such as tag descriptions, etc. is stored in phrases and referenced by phrase identifier key values. If you query a tag's description parameter, you will get the phrase identifier key not the text of the description. See: [Multilingual Expressions](#)
- You cannot declare variables in expressions or assign values to variables. But you can use existing ones such as tag values and application properties.
- FALSE is zero. TRUE is any numeric non-zero, usually 1.
- "Invalid" is a clearly defined thing in VTScaDa. It means that there is no value available. Examples include the value of an I/O tag when communications are lost, division by zero, and any variable that is declared but has not had a value assigned.
[Invalid](#)
Invalid is neither TRUE nor FALSE. Any calculation with an Invalid operand will return Invalid as a result.

Where to Write Expressions

Expressions can be added to tags and to widgets.

An expression can be used anywhere that you see the option: Constant / Expression / Tag.

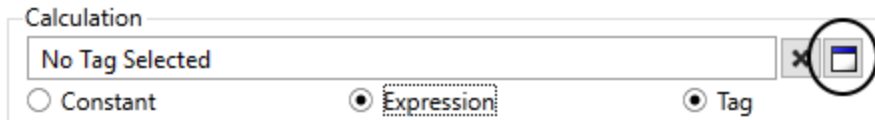


Figure 3-3 Constant / Expression / Tag in a tag configuration parameter

You must open the Expression Editor (button circled in preceding figure) to start typing an expression or to use the function editor.

Tip: When you open the Expression Editor, you may notice a link to the Function Selection Dialog. That's a great tool, but for your first few expressions it's better to type them directly, especially if you make a mistake or two along the way. The experience you gain in the process will make it easier to create more complex expressions later.

Exercise 3-1 Create a Calculation tag

1. Open the Tag Browser and create an I/O and Calculations tag at the top level.
2. Name it `Demo Calc`
3. Set the data type to Analog with the Calculation option selected.
4. Open the Calculation tab.
5. Select the Expression option, then click to open the Expression Editor.
6. Type the following expression to calculate the area of a circle with radius 5.

```
\pi * pow(5, 2)
```

7. Click OK in the expression editor to save your work.
You will not be allowed to save the expression if it contains a typo.
8. Deselect the Questionable Data property on the Quality tab.
9. Open the Display tab and set the digits after decimal to `2`.
10. Close the properties dialog.
11. Draw the I/O tag on the Overview page using a Numeric Value widget.
12. Configure the widget to show two decimal points.

Note: You might have been tempted to look for a function to format the expression to round to two decimal points. It is usually better to calculate mathematic results with full accuracy and use display parameters to adjust how it's presented.

Expressions can be added to any tag parameter:

For any tag parameter, but especially those that don't have the option shown in the preceding figure, you can right-click to create a parameter expression. These are slightly different from the last as they're designed to evaluate only when the tag starts.

If you add an expression for the tag's name parameter, you create what is called a Start Tag expression - one that controls whether the tag should start rather than one that changes the name.



Figure 3-4 Adding a parameter expression

Parameter expressions give you the option to deselect the option, "Optimize to only evaluate at tag initialization". Deselect that option only if you are certain that you need the parameter to change dynamically while the application runs. Be warned that a non-optimized tag parameter expression will cause that tag (and all of its child tags) to restart every time the value changes. This is not at all the same as an expression in a parameter that gives you the option of Constant / Expression / Tag.

Caution: Parameter expressions add extra demands of CPU time and memory. This is especially true for non-optimized tag parameter expressions. The extra load is very small but can add up in larger systems to impair performance. Use where needed, but only where needed.

Expressions can be used in graphic object properties

For any property of any object drawn in the Idea Studio, you can use an expression.

VT Rectangle Properties

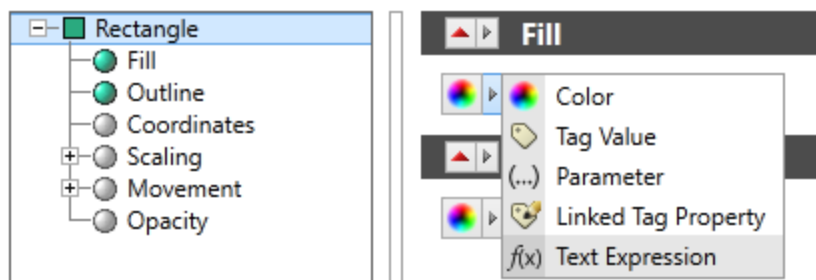


Figure 3-5 Setting a rectangle's fill color using an expression

The version of the expression editor that opens with Idea Studio object properties differs from the one in tags. Here, you can type directly in the field rather than opening the editor. Also, you get a tag-picker, allowing you to open a new instance of the Tag Browser and choose a tag for use within your expression.

Expressions can be used in page titles

The title of any page can be configured using an expression to reflect the content when the page opens. In most cases, this is used with parameterized pages and the expression will use one of the page parameters. If the page parameter refers to a tag, then you can use any property of that tag. For example, if you had a parameter named PumpSelection, holding a Pump Status tag then, your page title expression might be, "PumpSelection\ShortName". For anything more complicated you will need to use text handling functions.

We will revisit this at the end of the chapter on parameterized pages.

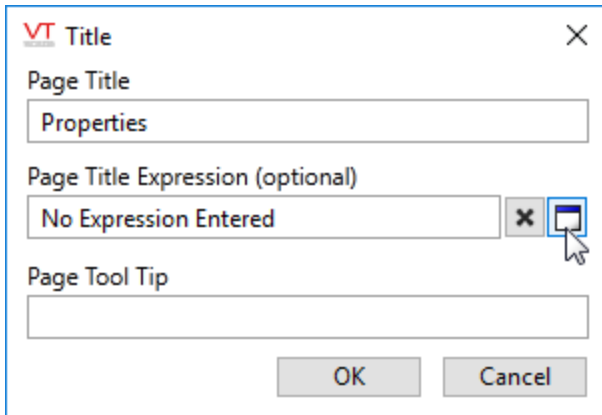


Figure 3-6 Opening the Expression Editor to set a page title

Use Tag Values in Expressions

Note: This information applies to expressions created within the expression editor. It does not apply to VTScada script code created in modules such as custom-coded reports.

In expressions, tag names are enclosed in square brackets: [Level]. If you want to see a property of that tag, you must say which one. For example, [Level]\Value, [Level]\ShortName, [Level]\Area. Because (usually) you want the tag's value, VTScada offers a shortcut. If no property is specified, VTScada assumes that you want the tag's current value. ([Level] == [Level]\Value. Note the location of the backslash relative to the bracket. Switching the order is a common mistake.)

Only the expression editor that opens from object properties within the Idea Studio will provide a Tag Browser button to help you locate and select a tag. In all other cases, you must type in the name of the tag that you want to use.

Note: You can copy the hierarchy of a tag's name from its properties dialog. Highlight the hierarchy and press Ctrl-C. (Right-clicking to open a dialog for copy/paste won't work.)

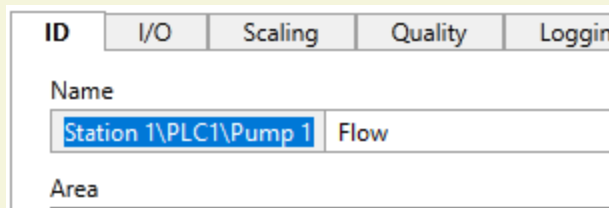


Figure 3-7 Copying the hierarchy of a tag name

You can also use an expression in the Help ID field to obtain the full name:

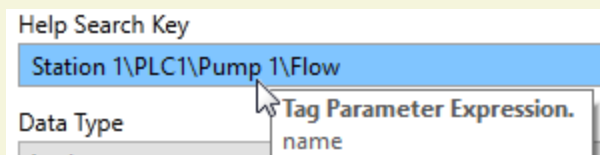


Figure 3-8 Copying the full tag name, from the expression, NAME

If the tag isn't nearby in the hierarchy, you will need to tell VTScada where to find it.

For example, given the following tag structure:

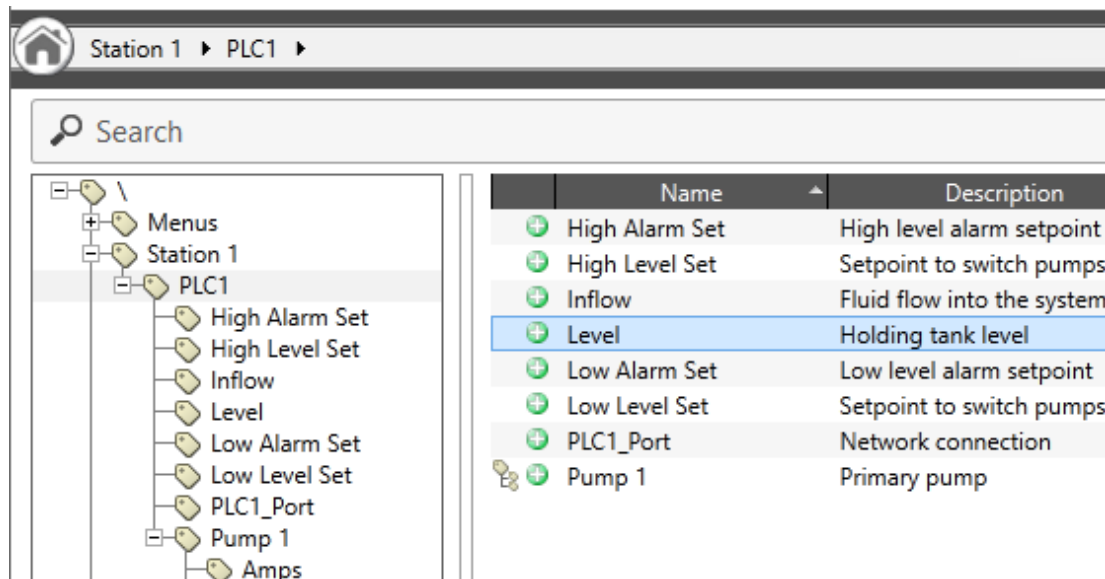


Figure 3-9 Expression to be created in \Station 1\PLC1 using Level

If you are creating an expression in a tag that's a child of PLC1 and that uses the value of Level, you would write it as [Level], as shown:

```
[Level] * 2 / 3 ; two thirds of the value
```

If you want to refer to a tag that is not an immediate sibling in the tag hierarchy, you will need to provide the address for where to find it.

Let's say that there is also a Station 2 with a level, and you want SiteCalc to show the average of the two:

```
(([Level] + [Station 2\PLC1\Level]) / 2 ; the average of two tank values.
```

Level within Station 1 can be found just by its name, but you must provide a path to the tag named Level in Station 2.

The following tools are available to help you specify tags in the hierarchy. All of these are used inside the square brackets that denote a tag name.

\	Divider between parent and child tag names.
..\	Forces VTScada to start searching one level up the hierarchy tree. Always use this when the property you want to refer to in a parent might also exist in the current tag.
Child\GrandChild	To reference a value (or other property) from a child of the current tag, start with the name of the child tag, then a backslash between each subsequent child name.
*TagType ..*TagType	Ancestor Relative Path, used when selecting a parent tag of a given type. For example, if you want a calculation tag to refer to the first parent that is a driver, use [*Driver]. (*) If you want it to refer to the first parent that has a numeric value, you can use [..*Numeric]. The ..\ portion is necessary to prevent the calculation tag from finding itself.

<>	Absolute path. The tag name must start immediately below the root level of the tag hierarchy.
----	---

(*) Types you are likely to use most often include: *Port, *Driver, and *Numeric.

There are advantages and disadvantages to each method. For each situation, VTScada uses what is most *likely* to be useful in that situation. "Likely" doesn't mean "always".

The Scope Function and Tags

VTScada handles tag addresses differently in different situations. In a tag's configuration, other tags are usually identified using relative addresses, [Tag Name]. In the Idea Studio, widgets link to tags using absolute addresses [<Full Tag Name>].

If you are writing a VTScada script code module, you can't use the square bracket shortcut. Instead, you must use the scope function:

```
Scope(VTSDB, "Full Tag Name or GUID", TRUE)\Attribute
```

If providing a full tag name, do not include the brackets. Providing the tag's GUID is equivalent to the short form, [< >]. "Attribute" is not a keyword here; use \Value or \Description, etc. If you don't provide an attribute, then the Scope function returns a link to the tag object itself, not the value.

Note: If adding expressions to tags in an exported Excel file, you cannot use relative references such as [TagName]. All existing expressions are exported using the full syntax and with certain characters doubled: "Scope(Self, "TagName")\\Value". You must do the same.

Exercise 3-2 Expressions That Use Tag Values

1. Create an I/O And Calculations Tag that shows the difference between the Inflow rate and the Pump 1 flow rate.

Operators

Operators are symbols used to perform an operation, comparison, or mathematical function such as addition and subtraction. All operators can be used in all expressions.

Spaces

You are advised to separate operators and operands with a space. For example:

```
2 + 3
```

By doing so, you make your code easier to read and make syntax errors less likely. The exception to this recommendation is for operators that come before or after their operands, such as pre-increment and post-decrement. These should never be separated from their operands.

```
A = ++B { Example valid only for script code, not expressions. };
```

Operator Priority in Statements

The order in which the operators are executed is significant. Consider the expression:

```
1 + 2 * 3
```

If operators were evaluated from left to right, the value of this expression would be 9, but in reality the value is 7. This is because VTScada, like all programming languages, assigns varying priorities to operators. Operators with higher priority are done first. Multiplication has a higher priority than addition, so in the previous example the correct result is 7.

If the addition were intended to be done before the multiplication, the expression could be written as:

```
(1 + 2) * 3
```

Parentheses () have the highest priority of all operators and force the expression within them to be evaluated first.

Some operators have equivalent priorities, such as addition and subtraction. In these cases the evaluation proceeds from left to right.

Note: If you have any doubt as to the order of execution within an expression, add parentheses to state (or control) the order explicitly. They cost nothing in terms of speed or memory requirements.

Operators and Invalid

Any expression that contains an Invalid will return Invalid unless PickValid() is used to substitute a valid value. There are two exceptions to this rule: AND (&&) and OR (||).

Boolean logic dictates that anything AND 0 is always a 0. Likewise, anything OR 1 must return 1.

Operators and Tag Expressions

All operators may be used in script code. Several are not available to, or not appropriate within, expressions written in tags and Idea Studio properties. For example:

- The assignment operator, =, and any variation thereof such as add equals, cannot be used in a tag expression.
- In a tag expression, square brackets are used to indicate tag names, and therefore do not work as array index operators.
- The address and dereference operators are not meant to be used in tag expressions.
- Pre- and Post- increment and decrement operators are not available for tag expressions.
- Shift left and Shift right.

For a list of the VTScada operators, refer to this topic in the online documentation: [Operators](#)

Comparisons: Operators || Functions

Note: Boolean. Noun, always capitalized. A binary variable having two possible values, either true or false. Named for George Boole.

You will often want to choose between actions based on the current situation. For example, if the room is too cold, increase the heat, otherwise reduce it. If the system is down for maintenance, disable the alarms, otherwise leave them enabled.

Typically, test expressions use conditional operators such as:

>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
==	Equal to
!=	Not equal to

Multiple Comparisons

&&	And
	Or

The function to choose between two options based on a comparison is `IfElse` and there are two ways to write it:

```
IfElse(Comparison-Expression, Expression if TRUE, Expression if FALSE)
```

```
Comparison-Expression ? Expression if TRUE : Expression if FALSE
```

Functionally, these are identical. It's your choice as to which form you prefer to use, but in general the `IfElse` form is recommended when there is a choice between running one expression and another:

```
IfElse([Pressure] > \Setpoint), {Do this... }, {Otherwise do this})
```

or returning the appropriate message:

```
Concat("The pressure is ", [Pressure] > \Setpoint ? "high" : "low")
```

Any expression that will return either a 0 (FALSE) or 1 (TRUE), can be considered to be a test expression. (Any numeric other than zero is considered TRUE, but 1 is typically used by convention).

Tip: If your goal is a form of trigger, there's no need to write `IfElse(Comparison-Expression, TRUE, FALSE)`. Simply write `Comparison-Expression`. The result of the comparison is returned as TRUE or FALSE, therefore you don't need to write extra code to return those values.

Exercise 3-3 Situational Message or Instructions

You can use the `MultiText` widget to display a message that varies according to a changing value. But, what if it also makes a difference whether equipment is running or an alarm is active? Further, what if you want the text of the message to include data calculated from two or more other tags? For this, you're going to need an expression.

1. Open the Idea Studio.
2. Working on the Overview or Station Status page, draw text (plain text from the ribbon, not a text widget) on the page.
3. Open the Edit Text dialog and change the data source to Expression.



4. Create an expression so that when the Level value is more than 50%, the message is "Level exceeds safe amount". When the Level is less than 50% the message should be "Safe level".
5. [Optional] Adjust the message to tell your operators the percentage by which the level exceeds the safe amount.

If you knew the math behind the system, you could create a message that tells operators how to adjust the pump or valve to balance the flow rate, or predict when the tank will empty or fill.

"Invalid" in Expressions and Results

"Invalid" is a term used to signify "no possible result". For example, trying to divide a number by zero. If an I/O tag cannot connect to hardware, its value will be Invalid. Any variable that is declared but not initialized with a value, will start out as Invalid.

Invalid is not an error message. It's a perfectly normal value that you can expect to see frequently. It is a unique data type that is not TRUE or FALSE, numeric or text. VTScada will never write an Invalid to I/O. (That last sentence is an important detail.) Invalids guard the system from performing control actions based upon erroneous or bad information. And any calculation that includes an invalid value produces an invalid result (with very few exceptions).

Your expressions need to be able to handle the situation when tags or calculations return a value of Invalid. As an example, suppose that you are trying to calculate a running total instead of using the VTScada Totalizer tag. If the tag value you are watching goes to Invalid, your calculation's result would become Invalid as well unless you have code to say otherwise. To avoid this situation, several functions are available:

PickValid()

This takes a list of parameters and returns the first one that has a valid value. In most expressions that include a tag value, you should protect against an Invalid result by wrapping it in a PickValid. For example:

```
PickValid(X, 0)
```

If X, whatever that is, has a valid value it will be used. If it doesn't, then 0 will be used.

Valid()

This tests whether a variable holds a valid value.

```
valid(x)
```

If X has a valid value then this will return TRUE (1). Otherwise, FALSE (0).

True()

Returns a 1 or 0 depending on whether the parameter evaluates to TRUE or FALSE.
Always returns 0 if the parameter is Invalid.

False()

Returns a 1 or 0 depending on whether the parameter evaluates to FALSE or TRUE. Like True(), this always returns 0 if the parameter is Invalid.

Exercise 3-4 Using PickValid

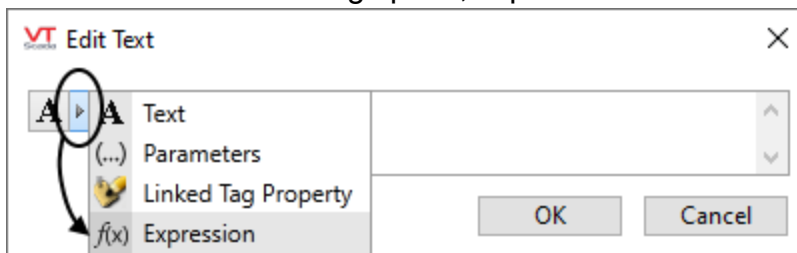
When a VTScada application starts, it takes a few moments to establish communications and read tag values. This is true even when communicating with a simulator rather than hardware. The following illustrates the point and the importance of using PickValid when referring to a tag value in your expressions.

Preparation:

1. Open the Application Configuration dialog, then Display tab of the Edit Properties page.
(This assumes that you are viewing properties using the basic mode rather than the advanced mode.)
2. Set the First page at startup property to use the Overview page.
3. Apply the change, then close the Application Configuration dialog.

Now for the expression:

1. Open the Idea Studio to the Overview page.
2. Delete everything.
3. Add text.
Just plain text from the toolbar, not a widget.)
4. When the Edit Text dialog opens, expand the list of data sources



5. Select Expression.
6. Select the Tag Browser button that appears within the Edit Text dialog.
7. Select the tag, Level
[<Station 1\PLC1\Level>]
8. Edit the expression to look exactly as follows:

```
PickValid([<Station 1\PLC1\Level>], "Not Valid")
```

Note that there are three parenthesis after the PickValid: ([<
9. Close the Edit Text dialog.
10. The text should show the current level of the tank.
11. Stop and restart the application, watching the text carefully as soon as the page loads.

Bonus question: what would you have seen during startup if the expression used the tag name without wrapping it in a PickValid? If unsure, try it and find out.

Working in Steady State

Most of the expressions that you create in tags and graphics work in steady state. For example, the code behind any page in VTScada. If none of the tag values change, nothing on the page will change. Deep in the VTScada engine, code is watching for changes to occur, but at the level of your page, all the code is simply sitting there. It does nothing, and takes no CPU time until a change occurs, such as an input tag reporting a new value. This triggers a linked widget to update, showing that value. Only the code for one tag and then one widget is processed. This is an important part of what makes VTScada so efficient. If you write code for a new display feature, it is extremely unlikely that you would need to write a loop to repeatedly check for new values.

Note that there are ways to get into trouble with steady state. For example, perhaps you might create two I/O and Calculation tags, both as analog calculations. You name them Calc1 and Calc2.

In Calc1, you suppose that you have created an expression that uses some other analog tag's value and also the value of Calc2. In Calc2, you have an expression that uses the value from Calc1. If you were working in an ordinary programming language, this would not be a problem. Your code would update Calc1, then Calc2 and then stop. But in steady state, expressions are triggered whenever any of the components change. What will happen in VTScada is:

1. The analog tag changes value.
2. This triggers the expression in Calc1.
3. The change in Calc1's value triggers the expression in Calc2 and so Calc2's value changes.
4. This triggers the expression in Calc1.
5. And, so on...

The process will never stop on its own, therefore you must make sure that you never create circular references like the one in this example.

Exercise 3-5 Pegging a CPU

(Do not do this experiment within a production application.)

The following demonstrates how poorly-designed expressions can cause trouble:

1. Create two numeric calculation tags. Name the first, "Calc1" and the second, "Calc2".
2. Enter the following expression for Calc1:

```
[Calc2] + 1
```

3. Enter the following expression for Calc2:

```
PickValid([Calc1] + 1, 0)
```

4. Let the system run for a few moments while you check the CPU statistics using the Windows Task Manager.
5. Delete the tags.
In an extreme case, you might need to use the Windows Task Manager to stop VTScada.

Break the Steady-State Rule for Expressions

The rule is that you must use Steady State functions in your expressions, excepting optimized Tag Parameter expressions. What if you need a script function, like `CurrentTime`? Then you can take advantage of a useful hack.

The Workstation Status driver tag provides a way to process script functions and display them in a steady-state page. It was given this feature originally because the memory information from the operating system is not an accurate reflection of memory in use. To obtain an accurate value, it was necessary to use the `VTScada Memory()` function, which runs only in script mode. A side benefit of this enhancement to the Workstation Status driver is that you can now use nearly any script-mode function with the Workstation Status tag and retrieve a result. Additionally, that result will update on a set schedule, almost as if it were running in Steady State.

For further information, refer to: [Workstation Status Driver I/O Addressing](#)

Exercise 3-6 Run a script function with a Workstation driver

1. In the Tag Browser, create a Workstation driver tag at the root level named Server Status.
2. As a child of Server Status, create a String I/O tag named MemUsage.
3. The I/O address for this tag will be:
Expression:Memory()
(type exactly as shown)
4. Change the scan interval to 5 and save this tag.
5. Wait five seconds for the first value to show.
6. Review the description of the `Memory()` function in the documentation.
7. Create a second String I/O tag, this time with the address:
Expression:CurrentTime()
8. Review the description of the `CurrentTime()` function in the documentation.

Using Functions

Functions perform tasks such as returning the absolute value of a number or retrieving data from the Historian. In the VTScada documentation, each function description includes information that you will need to use the function correctly. Look for the following:

Library Name

Not all functions are part of a library, but for those that are, you must **scope**¹ to the library to use the function. Use the backslash operator before the library name and the dot operator before the function: `System.Bevel()`. The backslash scope operator means (roughly) "keep looking until you find this" while the dot scope operator means "look only within this specific library".

Returns

Used most often by functions that are subroutines. For many functions (example: `GetTagHistory`) data is returned through parameters that are passed by reference, rather than as a return value from the function.

Usage Rules - Script or Steady State

VTScada code runs in two modes: Script or Steady State. Many functions will work in only one mode. The "Usage" line in each function description tells you the mode where the function can be used.

Note: Just because a function can be used in a given situation, does not mean that it should be. For example:

- * Graphics functions work in steady state but it makes no sense to use one in a Calculation tag's expression. Optimized tag parameter expressions can use script-only functions, but you would never use the `speak` function there.
- * `MatchKeys` will capture keystrokes only when used in a window or page, not in a service or Calculation tag.

Usage Examples. If you are writing...

General Expressions (Calc. tags)

If you are writing an expression for a Calculation tag, or anywhere that you have the option "Constant / Expression / Tag". Similarly for expressions for image and shape parameters when defined in the Idea Studio.



Figure 3-10 Constant / Expression / Tag in a tag configuration parameter

If the function is marked as "Script Only" then you cannot use it here.

¹v. Find or provide path to a function or variable. n. The path to a function or variable.

If the function works in Steady State, then it will compile when used in a Calc tag expression, but it may or may not be useful there. For example, any of the functions that display a graphic object are not useful as a tag parameter.

Tag Parameter Expressions - Optimized

Only functions that can be used in Script may be used for optimized tag parameter expressions. These expressions are evaluated as the tag is initialized, either on start-up or whenever the tag or one of its parents is re-initialized. You cannot use Steady State-only functions in this situation.

Tag Parameter Expressions - Not Optimized

Only functions that can be used in both Script and Steady State may be used for non-optimized tag parameter expressions. These expressions are re-evaluated whenever any of the values referenced by the expression change. This means that the tag (and all of its child tags) will refresh each time a parameter value changes. (A somewhat heavy process of running code that assembles parameter values, and detects and reacts to parameter changes.) You cannot use Script-only functions in this situation.

Page Code, Services, Reports, etc.

These are full VTScada modules, declared in the application's AppRoot file. The full VTScada language and function list can be used in accordance with the rules for a module.

Function Groups

A general classification of functions. Many belong to more than one group, and some are difficult to classify as a member of any. May be helpful when trying to find similar functions.

Related to

A list of similar or complementary functions.

Format and Parameters

The format line for each function description provides an example of how the function is to be used. Optional parameters are shown within square brackets in the format example.

You may copy the format example for use within your code, replacing parameters with values as required.

```
System.Bevel(X1, Y1, X2, Y2 [,Title, AlignTitle, Color])
```

Note: If your application predates version 11.2, the Settings.Startup property LocalScopeSyntax = 0 will prevent the use of the dot scope operator. If you are sure that you have not used a dot in any tag, widget, page or other name, you can set LocalScopeSyntax to 1 to allow this operator.

Note the use of commas to separate parameters when more than one is required. If a function does not require parameters, you may omit the parenthesis, although this is discouraged as a matter of practice. If a function has optional parameters, you can omit them from the call. If you need to provide only the n^{th} optional parameter, use Invalid as a placeholder for each optional parameter you are not specifying, filling from left to right.

Most parameters to functions can be expressions including other functions. While there is no limit to how deeply these can be nested, you should avoid making your code difficult to follow by nesting too many levels deep.

Parameter Descriptions

A description of each parameter used by the function, identifying whether it is required, the data type, and relevant notes.

Exercise 3-7 Practice Finding Functions

1. The function, `CurrentTime`, returns a UTC timestamp for the instant that it is called. But, can you use it in the kind of expression that you create in an I/O and Calculations tag?
2. What function might you use instead, if all you need is the time of day in a format that the computer will understand?
3. What function might you use if you need to trigger some action after a pump has been running for one minute?

Function Selection Dialog

Use this to browse a portion of the VTSkada function library for code to use in your expressions. The Function Selection dialog is available from any VTSkada Expression Editor dialog.

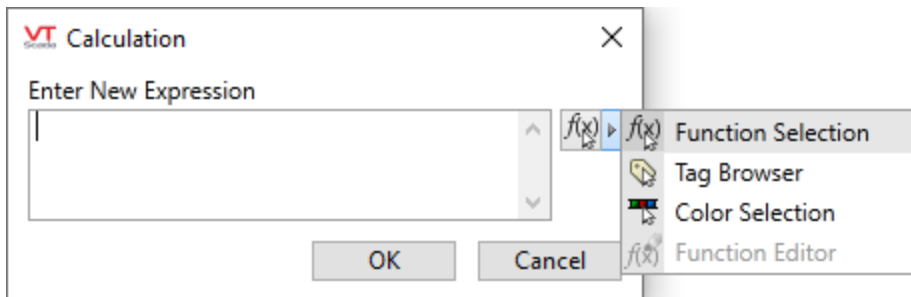


Figure 3-11 Options depend on where the expression is being created and therefore the value that should be returned.

This example is meant to return a numeric (analog) result.

Function Selection versus Function Editor

- *Function Selection* helps you create expressions. Use this to find and choose both operators and functions. Everything that you add is always appended to what you have already written.
- *Function Editor* helps you edit the parameters of the functions and operators you have already added to your expression. Use this after you have an expression containing at least one function or operator, in order to change that part of the expression.

Tip: While the Function Selector is helpful, you still need to know how expressions work and have a clear idea of the expression you are attempting to build. It is also helpful to look up the description of each function and its parameters in the VTScada documentation while editing.

* Press F1 while selecting parameters, to see the description of the function you are configuring (as well as suggestions for alternatives).

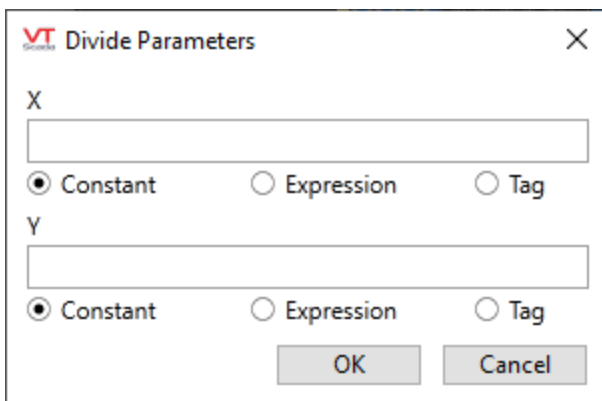
* Advanced developers can create their own functions for use with the Function Selector. See [Add Functions to Your Application's Library](#)

Function Editor / Parameter Assignment

To use the Function Selector:

1. Begin creating an expression by opening the expression editor.
2. Place the cursor in your expression where you want to add a function or operator.
3. Open the Function Selector. (Refer to the first image in this topic.)
4. Choose your function or operator, then click OK.

If parameters are required (as they almost always are) the function editor dialog will open. The title of this dialog varies because it always includes the selected function followed by the word "Parameters". In the following example, the selected operator was "Divide" (/).

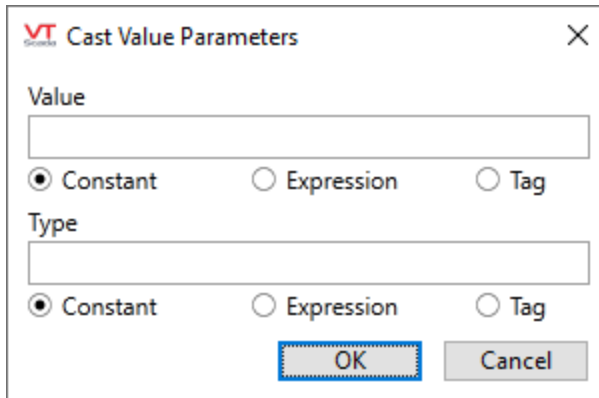


Tip: Plan to press F1 at this step. Doing so will open the VTScada documentation to the relevant function or operator, where you can learn more about what is needed for each parameter.

The expression editor will try to simplify your expression as much as possible before saving it. For example, if you type something like `Max(1,2,3,4)`, the expression will be interpreted as a constant "4" and the expression editor will not allow editing.

Example:

Suppose that your intent is to create an expression that returns the integer portion of tag's floating point value. For this, you could use either the `Int` function, or the `Cast` function. For the purposes of this example, let's assume that you decide to use `Cast`.

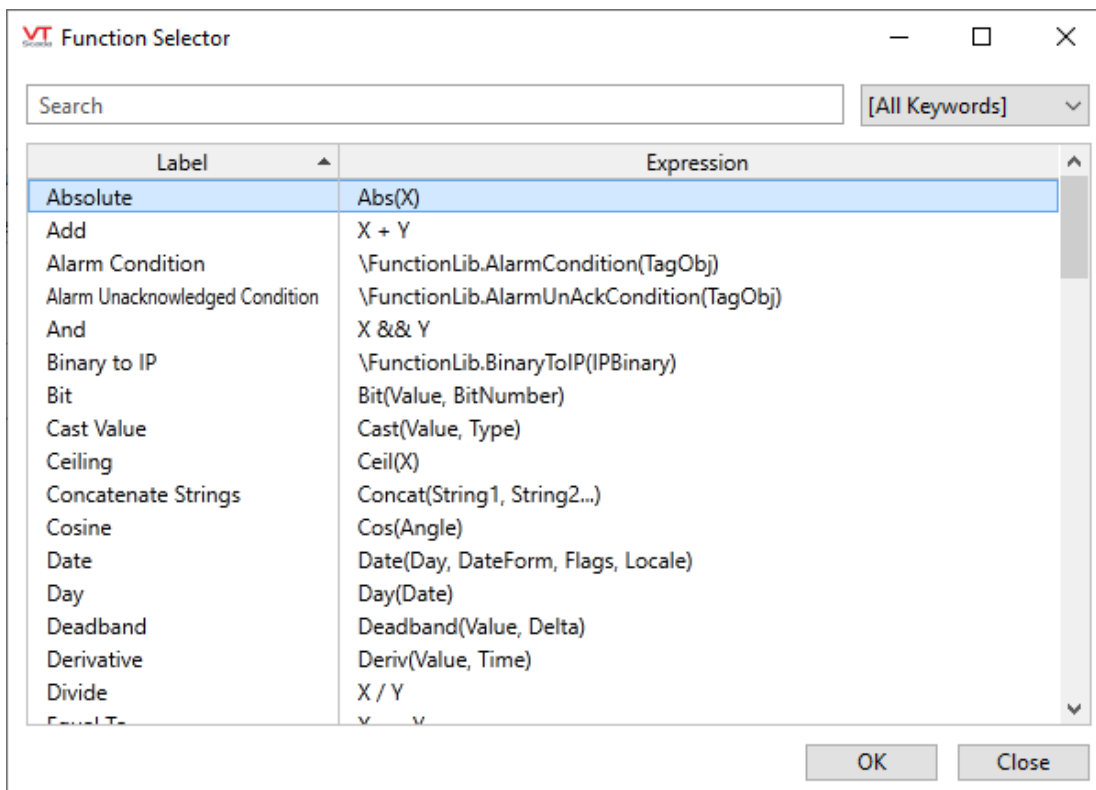


The dialog box is titled "Cast Value Parameters" with a close button (X) in the top right corner. It contains two sections: "Value" and "Type". Each section has a text input field and three radio buttons labeled "Constant", "Expression", and "Tag". In the "Value" section, the "Tag" radio button is selected. In the "Type" section, the "Constant" radio button is selected. At the bottom, there are "OK" and "Cancel" buttons. The "OK" button is highlighted with a blue dashed border.

For the "Value" parameter, you will select the tag. But, what should you provide for the "Type" parameter? You will need to refer to the documentation of this function to discover that the Type must be a number, and further, that number must be 1 in order to cast to an integer. Pressing "F1" will open the relevant documentation topic.

Note: When adding text parameters, do not type quotation marks around the text. For functions with optional parameters, beware of telling the dialog that you want to include an extra parameter and then leaving it unspecified. "Invalid" is the likely result.

Function Selector Description



The dialog box is titled "Function Selector" with standard window controls (minimize, maximize, close) in the top right corner. It features a search field with the placeholder text "Search" and a dropdown menu currently showing "[All Keywords]". Below the search field is a table with two columns: "Label" and "Expression". The table lists various functions, with "Absolute" and its expression "Abs(X)" currently selected. At the bottom right, there are "OK" and "Close" buttons.

Label	Expression
Absolute	Abs(X)
Add	X + Y
Alarm Condition	\FunctionLib.AlarmCondition(TagObj)
Alarm Unacknowledged Condition	\FunctionLib.AlarmUnAckCondition(TagObj)
And	X && Y
Binary to IP	\FunctionLib.BinaryToIP(IPBinary)
Bit	Bit(Value, BitNumber)
Cast Value	Cast(Value, Type)
Ceiling	Ceil(X)
Concatenate Strings	Concat(String1, String2...)
Cosine	Cos(Angle)
Date	Date(Day, DateForm, Flags, Locale)
Day	Day(Date)
Deadband	Deadband(Value, Delta)
Derivative	Deriv(Value, Time)
Divide	X / Y
Equal To	X == Y

Search

Type a function (or part of a function) into the search field and press the Enter key on your keyboard. Wild cards are supported and are assumed if not provided.

Filter

The list is prefiltered to functions and operators that can be used in the context of where you are creating your expression. For example, if your expression will run in steady state, script-only functions such as CurrentTime are removed from the list.

You can further filter the list by setting the drop-down selection to any of:

All Keywords

All functions and operators that can be used in the current context.

Alarm

Check whether a given tag has active or unacknowledged alarms.

Counters

Only one: Trigger Counter - watch equipment starts.

Date and Time

Calculate time and date in many ways.

Logical

A large variety of functions to check the status of your application.

Math

Generic, Boolean, and Trigonometric functions.

Mouse

Two functions that return the location of the pointer within the application.

Network

Translate a TCP/IP address to text.

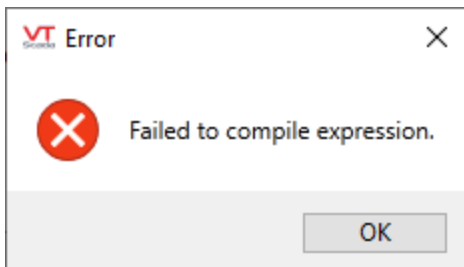
Text

Manipulate and examine text.

Variable

Cast one variable type to another, or specify the scope of an object.

Error: Failed to compile expression



You will see this or a similar message when there is a syntax error in your expression. Refer to the [Syntax Rules for Expressions](#)

Tip: The dialog indicates only syntax errors. Errors in logic can be saved and will run, but return the wrong result. Write your expressions with care.

Exercise 3-8 Practice using the Function Selector

1. Create a new I/O and Calculations tag named Level Steps.
2. Ensure that the data type is Analog and select the Calculation option on the ID tab.
3. Configure an expression using the Function Selector to use the Deadband function, setting the Delta value to 2. (Hint: Don't forget that you can use F1 to open the documentation.)

(You might want to disable alarm sounds before proceeding with the next part of this exercise. As a reminder, you can find a control for that in the Alarms tab of the Edit Properties dialog.)

1. Edit the tag, Level, to add a low level alarm. This should have a priority of "Warning" and a setpoint of 10.
2. Modify the tag, Inflow, to add a similar low level alarm with a setpoint of 25.
3. Create a new I/O and Calculations tag named Station Alarms
4. Set the data type to Digital and select the Calculation option.
5. Use the function selector to create an expression that will be TRUE whenever there is an active alarm on either Level or Inflow. (You will need the OR operator, which is written as two vertical bars: `||`)

After ensuring that your expression works, feel free to disable those two alarms before proceeding.

Math in Expressions

The following are a few of the mathematic functions available to your expressions. For a complete list, see Math functions in the VTScada Function Reference. (All math functions can be used in any expression.)

Max(X, Y, Z, ...)	Returns the variable having the largest value.
Pow(X, Y)	Returns the value of X raised to the power of Y.
Sqrt(X)	Returns the square root of the value in X.
Int(X)	Returns the number with any digits following the decimal point truncated.
Sin(X)	Returns the trigonometric sine of X, where X is measured in radians.
Cos(X)	Returns the trigonometric cosine of X.

Exercise 3-9 Use Math Functions

1. Find and then use a function that will return the number 1000000 as a string with embedded commas (1,000,000).
2. The pump speed ranges from 0 to 1200, defaulting to 640. Find and then use a single function that returns 640 as a percentage of the possible range. (i.e. expressed as a percent of the range between 0 and 100.)

Text Functions in Expressions

Expressions can be used to display calculated text as well as numeric values. For example, you might use the `Concat()` function to join the value of a tag or the result of a calculation to a sentence.

A few of the string handling functions in VTScada are as follows. See [String and Buffer Functions](#) for complete descriptions of these and other functions. (Note: many string functions cannot be used in Steady State, and thus cannot be used in an expression.)

<code>\GetPhrase & \GetParmPhrase</code>	<p>Note the backslash that begins each of these functions.</p> <p>Given a phrase key, or a parameterized structure (<code>\ParmPhrase</code>) of phrase keys, this will return the matching phrase(s) in the current language.</p>
<code>Concat(a, b, c...)</code>	<p>Concatenates any number of sub-strings into one sentence. Works in only one language at a time.</p> <pre>Concat("Level of Tank 1: ", [TankLevel_1], "%")</pre> <p>Returns (for example): "Level of Tank 1: 30.35552%"</p> <pre>Concat(" Viewing Station: ", StationNumber)</pre> <p>Example shows how to set the title of a parameterized page, where <code>StationNumber</code> is a text or numeric parameter of that page.</p>
<code>Format(width, precision, value)</code>	<p>Turns a numeric value into a text string, having the specified width and precision (number of decimal points).</p> <pre>Format(5, 2, [TankLevel_1])</pre> <p>Returns (for example): "30.36"</p>
<code>Replace(sentence, start, length, find, replace)</code>	<p>Searches the sentence, beginning at the <code>Start</code> character and continuing for <code>Length</code> characters, looking for every instance of <code>Find</code> and replacing it with <code>Replace</code>. Note that character counting begins with 0.</p> <pre>Replace("This is good", 1, 12, "is", "was")</pre> <p>Returns: "Thwas was good"</p> <p>(Note that every instance of "is" is replaced. This may have unintended consequences.)</p>
<code>SubStr(sentence, start, length)</code>	<p>Returns a substring of <code>Sentence</code>, beginning with the <code>Start</code> character and running for <code>Length</code> characters.</p> <pre>Substr("on a Halifax pier", 5, 7)</pre> <p>Returns: "Halifax"</p>

Examples: Text Functions in Expressions

Concat

One of the most frequently used functions is `Concat()`. This combines several bits of text together into one sentence. If any of the parameters is a number, it will be converted to text for use in the sentence.

`Concat()` can take any number of parameters. Don't forget to include spaces unless you want to join parameters together into a single word. Parameters may be expressions. Numeric values will be translated into strings.


```
Concat("This ", "and ", 3/4, " of that.")
```

returns "This and 0.75 of that."

Tip: As of version 12, you should create \ParmPhrase structures rather than use Concat to generate all user-interface text.

StrLen

Returns the length of a text string measured as the number of bytes. This can be useful to know when you need to find some portion of text.

```
StrLen("Welcome to VTScada")
```

will return 18.

SubStr

Returns a portion of a text string, starting at a specified number of characters from the left. Be careful: counting starts at zero.

Providing the length (number of characters) to return is optional. If you don't, you'll get everything from the n^{th} character to the end.

This function has the form:

```
SubStr(String, Start[, Length])
```

Those square brackets have a meaning: they enclose parameters that are optional. Do not type the square brackets when writing a function in your expressions.

```
SubStr("Good morning to you.", 5)
```

Will return, "morning to you."

```
SubStr("Good morning to you.", 5, 7)
```

Will return, "morning"

Locate

This function locates a substring within a longer line of text, returning the location of that substring as the number of bytes from the beginning. Counting begins at zero, therefore if the substring is at the beginning of the longer string, 0 is returned. If no match is found, then -1 is returned.

You can specify a start point, which is useful if the substring occurs several times and you want to find all of them. You just have to run the function several times, starting one character to the right of each successive match.

You can also specify whether you want a case sensitive match or not. (Case sensitive is the default.)

```
Locate("Big haystack with a needle", 0, "needle")
```

returns 20

```
Locate("Big haystack with a needle", 0, "Needle")
```

returns -1

```
Locate("Big haystack with a needle", 0, "Needle", 1)
```

returns 20

Exercise 3-10 Practice with text expressions

Typically, someone will want to use a text function to specify a given tag name on the fly within an expression. Something like...

```
Concat("[AlarmPriority", x, "]\Description")
```

...to retrieve the description of a given Alarm Priority tag. This doesn't work.

The expression will return the tag name and stop at that point, not continuing on to evaluate that tag's properties.

Fortunately, there's another way to specify the name of a tag. You can use the Scope() function, which takes any expression that evaluates to a tag name. So, for example,

```
Scope(VTSDB, "AlarmPriority0")\Description
```

The quotation marks are around the tag name to tell VTScada that this is text, not the name of a variable. If you use a function that returns text (like Concat does), you don't need to wrap extra quotation marks around it. It's text.

```
Concat("AlarmPriority", x)
```

Assuming that x is a variable containing a number from 0 to 4, this will return the name of an AlarmPriority tag as *text*, which is what the LocalScope function needs for a parameter.

However, there's still a problem... What you'll get won't be the description of the tag, it will be the phrase identifier key for the description. You'll need to do a look-up on that key to get the description by calling \GetPhrase(). (Note the backslash in front of \GetPhrase.)

1. Create an I/O tag named X using the Discrete data type and with a Scaled Process Data Max set to 4. No I/O device or I/O address will be used.
2. Draw X on the Overview page as a Numeric Entry widget.
3. Create another I/O tag using the String Calculation data type.
4. Give it an expression that returns the description of any specified Alarm Priority tag.
Your expression should use [X], created in step 1, to get the priority number.
5. Draw the tag on the Overview page using a Draw Text widget, placing it near the Numeric Entry widget.
6. Switch to operator mode and put numbers 0 through 4 in the numeric input widget.

4 Design Your Own Tags

"Alarm", "I/O and Calculation", "Counter" are examples of tag types supplied with VTScada. You can easily create your own types to join this list, such as "LiftStation", "Generator" etc. Each one of your new types may contain as many child tags as required to describe the matching equipment.

To create a new type of tag, start with a Context tag. Any Context tag with a legal word in the Type field can be turned into a new type template. All child tags of that Context will be included in the new template. The Type property value becomes the name of the new type. Within your application (and any applications that use this one as an OEM layer), the new type will be available alongside the standard tag types.

The following images illustrate the process.

The screenshot shows the 'New Context Properties' dialog box with the 'ID' tab selected. The 'Name' field is 'LP001', 'Area' is 'Zone A', 'Description' is 'Lift including inputs, controls, logging and alarms', and 'Type' is 'LiftPump'. The 'Type' field is circled in red. The 'OK' and 'Cancel' buttons are at the bottom right.

Figure 4-1 Step 1: Create a Context with a Type property.
Add all the relevant child tags.

Tip: Name the type something that you can guarantee will be unique to your application. One way to do this is use compound words like AceCustomPump. An error message will be displayed if the type name cannot be used.

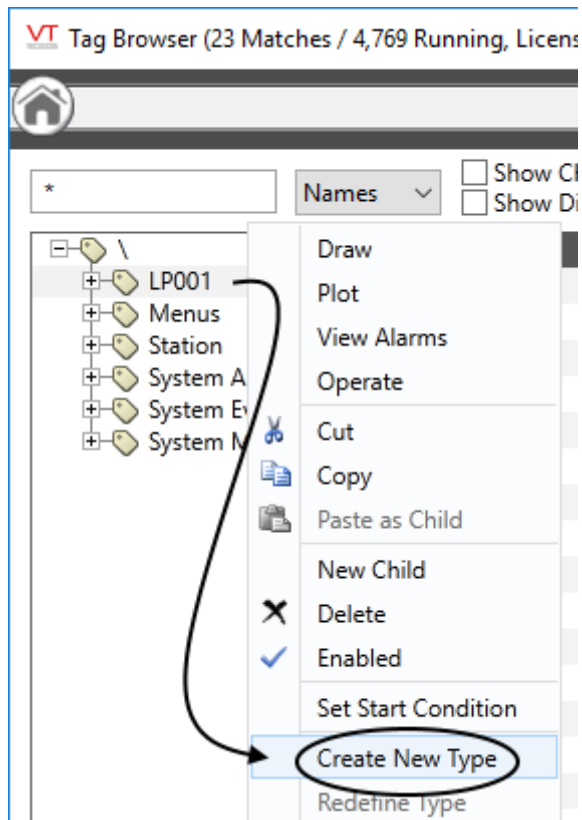


Figure 4-2 Step 2: Create New Type

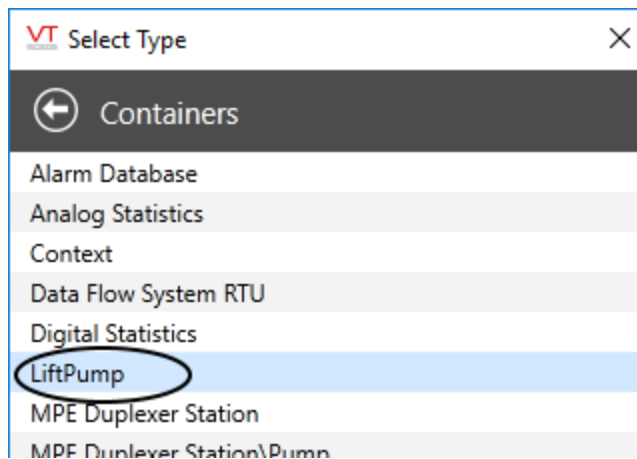


Figure 4-3 Step 3: Your custom type is now available when adding new tags.

New copies of your custom type can be created as easily as any standard VTScada tag type. If the type definition includes child tags, as is normally the case, then those will be created at the same time. Note that you can build on this by creating your own widgets that display all the child tags in one object.

Plan ahead and do less work later.

Before following the steps just described, take time to build the tag so that as much work as possible is done once and never needs to be repeated. For example, configuration fields in the child tags can and should use parameter expressions (that you create) so that they will be configured automatically using information from the parent tag.

When adding new instances of your custom tag type, developers should only need to configure the parent type. All child tags should be configured automatically using parameter expressions. The expression can be complex, or it can be as simple as copying a value from the parent tag to the child. The goal is to keep all of the configuration in one place / tag.

If some child tags will not be needed in every instance of the type, then use Start Tag expressions. These are similar to parameter expressions, but are stored in the tag's name field and control whether the tag will be enabled or disabled. For example, in a lift station that can have up to three pumps, all three will exist in the type definition, but two will be controlled by Start Tag expressions and be enabled only according to the number of pumps configured in the parent station.

Tip: If there's a chance that you will want to use the new type in other applications, build the type in an OEM layer below your application rather than directly in the application. It's much easier to distribute feature updates via OEM layers than to export tag types. ([Reusable Application Layers](#))

Custom configuration

Some (not all!) configuration properties that relate to tags can be inherited from a parent tag instead of using global settings from the Application Configuration dialog. For example, you can have Roster and Alarm tags with their own delays and call-out priority settings. Alarm message templates can be designed that differ from one part of the application to another. Drivers can have differing configuration if (for example) you have some Modbus devices whose addressing begins at 0 and others whose addressing begins at 1.

Improve the design

You can modify the design of the new type by making changes to one instance, then issuing an "Redefine Type" command on it in the Tag Browser. For example, you might add a Context tag having two child tags: an Analog Input, and a Logger storing the Input's data. Later you decide to add an Alarm to the structure. By running a Redefine Type command, all new *and existing* instances of that type in the current application will include those child tags.

Tip: Never redefine a working instance of your custom type. Always create a temporary copy to modify, then run redefine on that. This avoids the danger of having local overrides become part of the type definition, changing all existing and new instances.

Add Properties to Context Tags

In nearly all cases, the Context tag that will form the basis of your new type definition will hold a set of properties relevant to that new type.

These can serve many purposes, as follows:

Site Properties

Property Name	Value	Comment
Latitude		Latitude (decimal degrees)
Longitude		Longitude (decimal degrees)
InitZoom		Map Zoom Level
CustomDetailsPage		Custom Details Page
CustomMapIconParm		Custom Map Icon
SiteListDisplay	0	Site List Display

Figure 4-4 Properties after clicking Add Site Properties

All Context tags and user-defined types based on Context tags will be included in Site Lists, whether you add these properties or not. But, you must add Latitude and Longitude properties before the tag can be drawn on a map and other site properties before you can set configuration options such as excluding the tag from a Site List or setting a custom details page or map icon.

Connector Properties

If you want to represent a pipe, power line, or other conduit between two sites use the Add Connector Properties button.

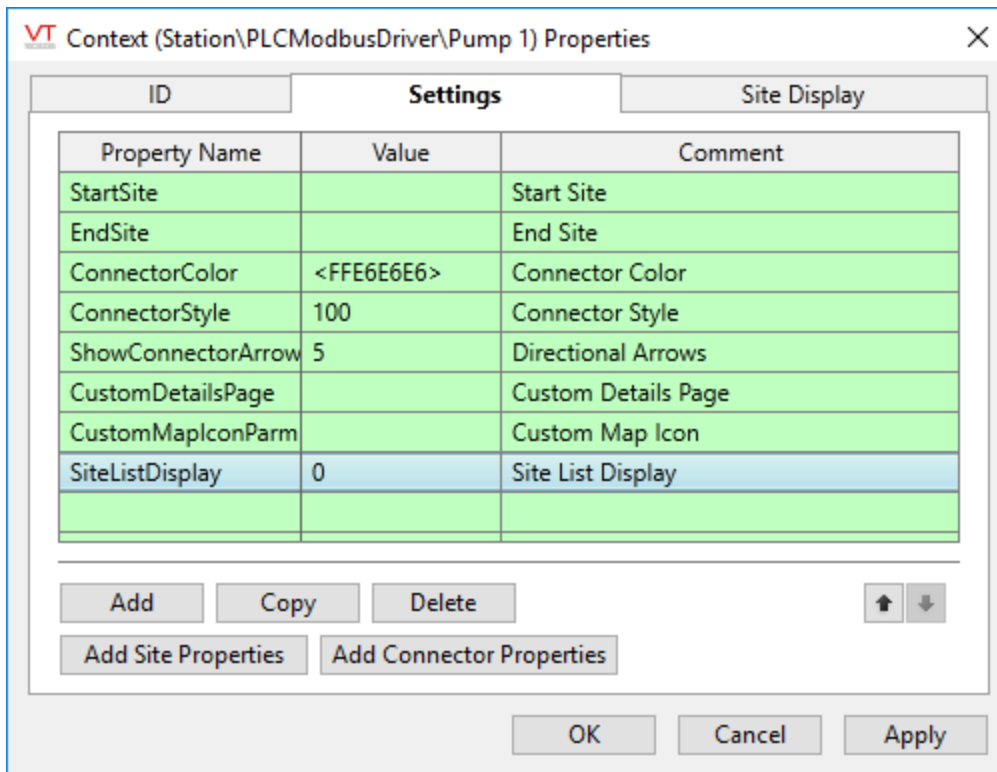


Figure 4-5 Properties after clicking Add Connector Properties

Connectors link to sites that you create with latitude and longitude properties, and are drawn on a map as a straight line with arrows. Use Connectors when the connection between two sites is monitored by I/O. You would not use a Connector when all that's needed is a line between two points.

Custom Properties

You can add any properties that you want to a Context tag for use in your custom type. There is no limit, although a long list may become difficult to use. These properties may hold all the information required to fully describe the site or object. Child tags can use tag parameter expressions to inherit any of these properties for use in their own configuration. As stated earlier, it should rarely be necessary to configure any child tag of a custom type. All typical configuration should be done only in the parent tag of the structure. This may mean duplicating properties from the children in the parent.

Figure 4-6 Adding a property to a Context tag

The option to specify that the value is translatable text is important. If selected, whatever you type will be stored in the languages file as a new phrase, and the matching phrase key will be stored in the parameter. Do not select for any value that must be used as-is, such as an I/O address.

After the context is turned into a type, the panel shown in the following figure will change. The overall appearance will resemble the ID tab, but with an entry for every property. (A scroll bar is provided if needed.) The Property Name will vanish, replaced by the Comment, which becomes the title for the field. Values can be edited in exactly the same way that Description and Help ID properties can be edited on the ID tab. Those who learn the VTScada programming language can redesign the properties dialog as they like, noting that after doing so the tag will be counted against the number permitted by your license.

Property Name	Value	Comment
PumpCount	1	Number of pumps
TCPAddress	127.0.0.1	TCP/IP Address
Port	505	Port Number
IOBase	0	I/O Base Address
LastMaintenance	2000/02/29	Last Maintenance Date

Figure 4-7 Assorted custom properties

Exercise 4-1 Add Properties to a Context

One goal for this exercise is to be able to control how pumps are shown in Site Lists. To do that, you will need two site properties, but clicking Add Site Properties will give you more than you need. Thus, several properties will be deleted immediately after you add them.

1. Using the Tag Browser, find the tag Pump 1 then open its properties dialog.
2. In the Type field of the ID tab, add `CustomPump`.
3. Open the Settings tab.
4. Click the Add Site Properties button.
5. Delete the properties, Latitude, Longitude, InitZoom and CustomMapIconParm. You should be left with two properties: CustomDetailsPage and SiteListDisplay. More will be said about these soon.
6. Click the Add button, then fill in the Add Property dialog as follows. (see Figure 4-6, on the previous page
Property Name: `IO_Offset`
Value: `0`
Label: `I/O Address Offset`
7. Click OK to close the properties dialog.

You will use these properties (and more) in future exercises.

Automated Tag Configuration

Any parameter(*) of a tag can be configured using an expression rather than a static value. This is an extremely powerful feature that you can use to create tags that configure themselves based on where they are placed in the application. Also use it for tags that rely on data from the remote device for their configuration.

(*) The name parameter works differently. An expression here can be used to enable or disable the tag, but cannot change the name.

If you have created a tag as a child of another tag, then you have seen this feature in action. The Area field of every new tag will copy the parent tag's area unless you apply an override to set a new value. If you move or copy a tag to a new parent, its area field will change to match the new parent. The expressions described here are most effective when used in parent-child tag structures, where they can use information stored in an ancestor.

Note: When using expressions for parameters, it is up to you to ensure that the result is valid and that it makes sense for the parameter. For example, an I/O address of either INVALID or "Hello World" is unlikely to produce a desired result.

Tip: Before writing your first parameter expressions, review the notes in the chapter, [Expressions](#). In particular, take note of the advice in the topic, [Keep the Else's under control](#).

Identify Parameters with Expressions:

- Parameter fields that contain expressions can be identified by their blue shading.
- If you override an expression with a new value, the field will have a yellow shading.

The screenshot shows a configuration dialog with several fields. The 'Area' field is a dropdown menu with 'Northern' selected. The 'Description' field is highlighted in blue and contains the text 'Primary pump in the Northern region'. A context menu is open over the 'Description' field, displaying the text 'Tag Parameter Expression. \ParmPhrase("EU43y4gag", Area)' and 'Modification not yet saved'. Below the 'Description' field is the 'Help Search Key' field, which is empty. At the bottom, there is a 'Type' field.

Figure 4-8 "FmoockKh", in this application, is the key for a parameterized phrase:
 "Primary pump in the %0 region".
 "EU43y4gag" will not exist as a key in your application.

The context menu that appears when you right-click on a tag's configuration field will vary according to what the field is and how it was created.

Create, Edit, Remove, or Override an Expression

For all of the these actions, right-click on the parameter in the tag's configuration dialog. Your choices will vary depending on whether you add the expression to the name parameter (start conditions only), area parameter (which can inherit from its parent), and whether the tag is part of a user-defined type (where changes are considered overrides).

Tip: (1) "Reset to Default", seen in all the following examples, is not about expressions but rather for resetting a parameter to its default value if there was one. For example: in a driver tag, the port configuration defaults to [*Port] (ancestor relative path) but you might have changed it to absolute path. "Reset to Default" gives you an easy way to change it back.

Tip: (2) The ability to right-click on a parameter to add an expression does not apply to parameters that provide the options Constant - Expression - Tag. For those, use the tools provided instead of the right-click menu.

The screenshot shows a configuration dialog with 'Name' and 'Area' fields. The 'Name' field contains 'Tank level' and is highlighted in green. A context menu is open over the 'Name' field, displaying the options 'Add Start Condition', 'Remove Start Condition', and 'Reset to Default'. The 'Area' field contains 'Chlorination' and is also highlighted in green.

Figure 4-9 Name field.
 Add or remove start conditions

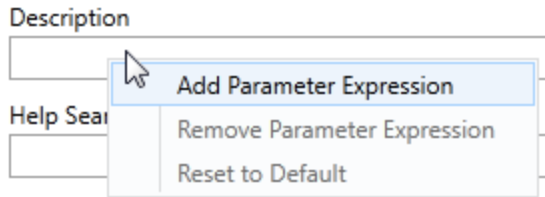


Figure 4-10 Other fields, no existing code to override.
Add or remove parameter expressions.

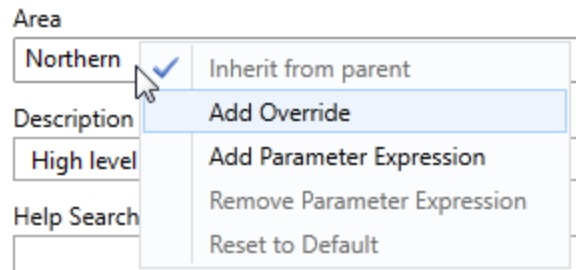
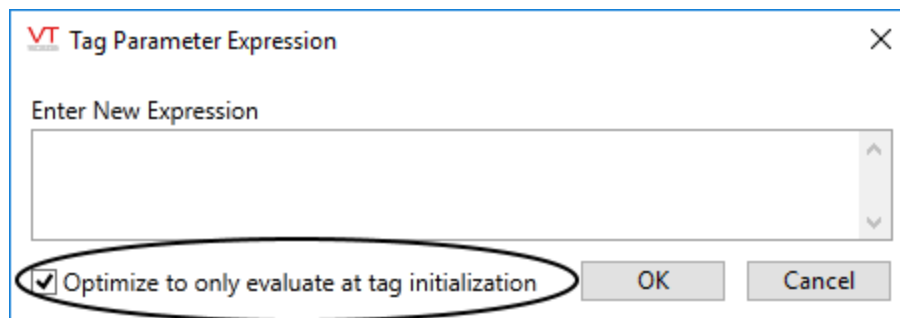


Figure 4-11 Other fields, with code to override (child tags of a user-defined type)
Inherit from parent (Area parameter only)
Add or remove override

Optimized Expressions

When using an expression to configure a parameter value, you can choose whether the value should be evaluated only when the tag starts (and re-starts), or whenever there is a change to any value used by the expression. Non-optimized tag expressions are intended for situations where configuration values must be obtained from equipment, which is reachable only after the application starts.



Caution: Do not deselect the optimization option unless there is a clear need to do so. Misuse of non-optimized tag expressions can have undesirable side-effects. In particular, note that configuring a tag so that its properties change dynamically may have an impact on logged values or cause unintended alarms. Every time that the non-optimized parameter expression changes, the tag will restart, as will all of its descendants, just as if you changed it in the Tag Browser. Be certain that variables within the expression will not change frequently.

A parameter expression that refers to tags outside the current hierarchy must use a non-optimized expression. There is no way to predict which hierarchy of tags will start first, therefore you must allow the expression to re-evaluate if a tag that it refers to *in a different hierarchy* starts later than the current tag.

When optimization is selected:

- The parameter value will be evaluated only when the tag is created, when the application re-starts, or when you explicitly change any property of this tag or a parent using the Tag Browser.
The expression will not be re-evaluated during normal operations.
- Your selection of functions is limited to those that can be evaluated in script mode.

When optimization is not selected:

- Expressions can use information obtained from hardware after tags have initialized and I/O operations begin.
This is the primary purpose of non-optimized parameter expressions.
- The expression will be re-evaluated whenever any of the parameters used in the expression change.
Be careful to consider all the possible effects of this fact.
- When a parameter changes, this tag and all of its child tags will restart, slowing the application temporarily and increasing the load on the CPU.
- Each non-optimized tag expression requires extra RAM, roughly equivalent to half that required for a typical tag.
- Your selection of functions is limited to those that can be evaluated in both script and steady-state.

Identifying properties

Do not add quotation marks to property names. Quotation marks should be used only to indicate literal text.

To read the a tag property other than it's value, follow the square brackets around the name with a backslash and then the name of the property.

```
[SomeTagName] \ShortName
```

To access the value of a property in a parent tag, preface the name of the property with two dots and a backslash (..\). VTScada will search upwards through the Parent-Child tree to find the first instance of a matching property name. You can specify the number of levels up at which to start by repeating the ..\ code.

Your expressions can use relative tag addresses, VTScada application properties and other system variables. Refer to topics in the scripting chapters beginning with [Use Tag Values in Expressions](#).

Note: Referring to text properties such as Description, Area or Engineering Units? You should use \ParmPhrase(), \GetPhrase() or \GetParmPhrase() rather than Concat(). See: [Multilingual Expressions](#).

Start Tag Expressions

An expression in the name field of a tag will not change the tag's name. Rather, this expression will control whether the tag starts or not. Examples of this can be seen in the various Station Tags, which may be configured with a varying number of pumps for each station. This is done by tying the number of pumps in the overall station tag configuration to an expression in the name field of each pump. If the number is 1, only the first pump tag will start and the station tag will have one pump. If the number is 2, then two pump tags will start.

The expression used for a start condition must evaluate to TRUE (usually defined as 1) or FALSE (defined as zero).

For example, the MultiSmart, and MPE station tags include an option to set the number of pumps in the station.

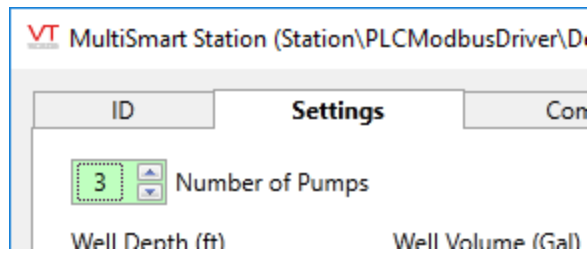


Figure 4-12 The number of pumps in the station tag can vary.

The tag structure varies according to the number you choose:

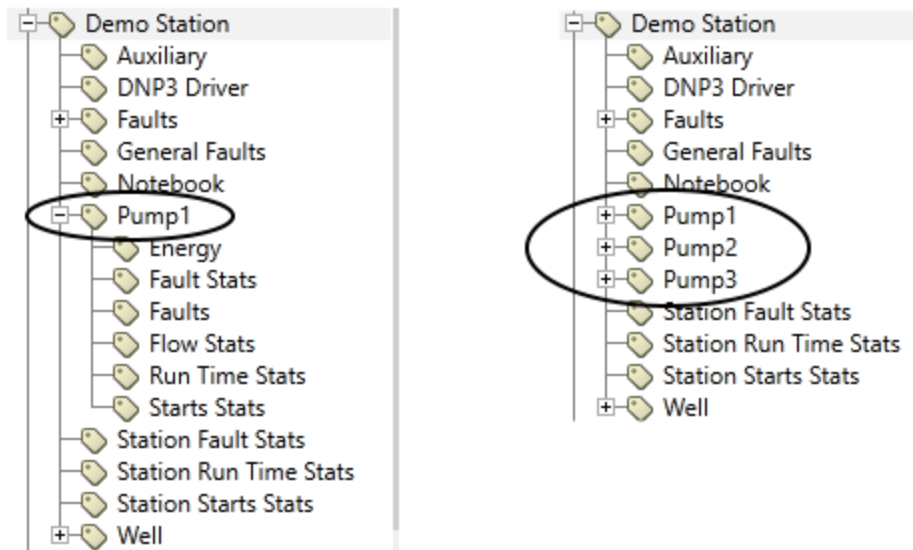


Figure 4-13 On the left, one pump. On the right, three pumps configured.

Changing the selector doesn't change the number of pump tags *created*. It changes the number *started*. Select the Show Disabled option in the Tag Browser to reveal the difference.

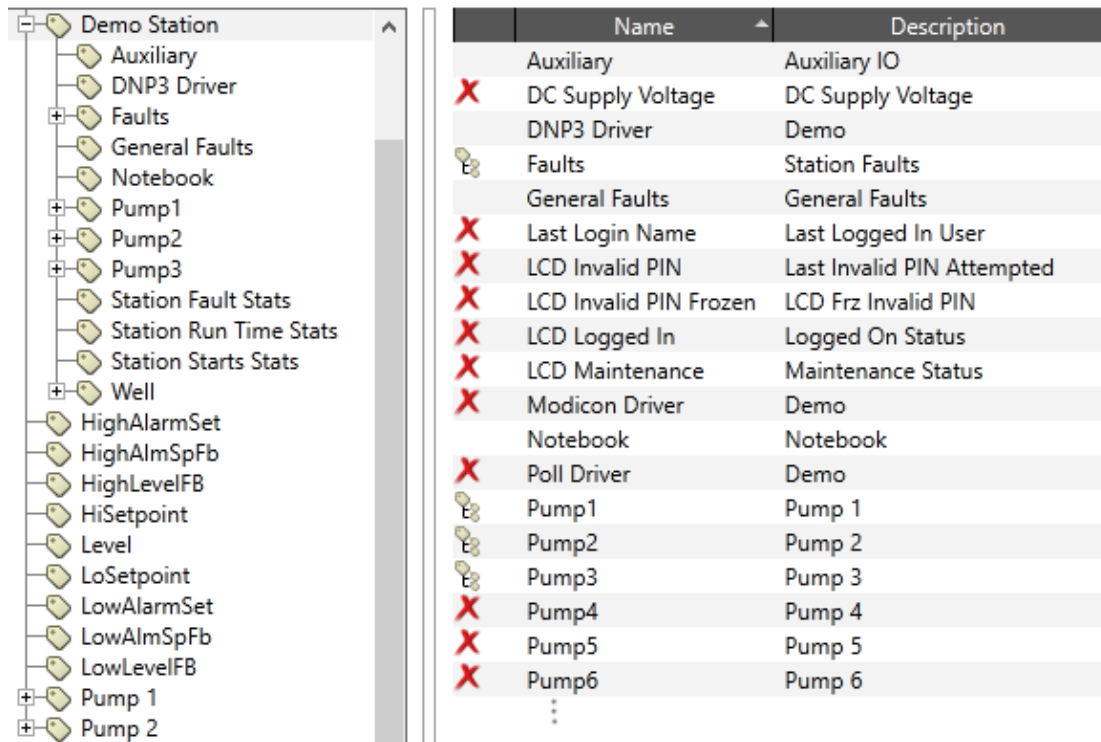


Figure 4-14 The Multismart always has the same number of pumps. The choice is how many to start/en-able.

To use this feature, add a Start Tag Expression to the name parameter of a tag.

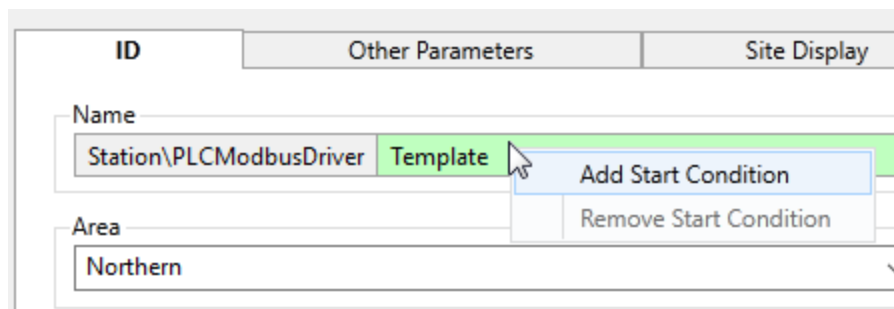


Figure 4-15 The method is the same as adding a Tag Parameter Expression.

In the Tag Parameter Expression editor, whatever expression you create must ultimately return a Boolean result, meaning either zero or numeric non-zero (usually, one). If the expression works out to true at start-up then the tag starts. If false, it doesn't.

Tip: Start tag expressions are always optimized. They run only when the application starts or when a parent tag is redefined, which causes all child tags to restart. Ensure that the conditional within your start tag expression can be evaluated in both of those situations.

Overrides

Having made use of an expression to control a tag's configuration, you can choose to override that value in a given child tag. For example, a child tag which would otherwise inherit the area value "North County" might be assigned the area "South Street". When a value that would otherwise be based on a calculation is changed, this is called an "override". Assigning a value to a field that has no expression does not override anything and is simply called "assigning a value". The difference can be seen in the color that the field will turn: orange for overrides, and green for simple assignments.

A tag that has an override is marked with a  in the Tag Browser.

Overrides are both common and useful. Do not assume that the symbol means that you must take action to "clean up" the override in some way.

Exercise 4-2 Add flexibility to Pump 1

If you have Excel or Access, you might save time by exporting then synchronizing your tags. Do steps 1 and 2 within the tag browser for one tag before exporting so that you have an example to copy. If you don't have Excel or would rather not use it in this case, continue on with the Tag Browser.

Whether you export the tags or work entirely within the Tag Browser, remember that copy and paste are faster than typing. Doing steps 1 and 2 together for each tag may be faster than doing step 1 for all, followed by step 2 for all.

1. For every child tag of Pump 1, add a tag parameter expression to the description to combine the name of the parent with the purpose of the child tag.

This example won't translate well, but for the purpose of this unilingual exercise it's faster than generating a set of new phrases.

For example: `\ParmPhrase("%0 %1", ..\ShortName, " flow rate")`

Don't forget the leading backslash.

2. For every child tag of Pump 1, add a tag parameter expression to the I/O address, adding the property `..\IO_Offset` to the current address.

For example:



Figure 4-16 Old address to new address

Note: The Selector Switch has three digital addresses. You'll need an expression for the address in each of the switch positions.

From Context to Type

After you have:

- Added child tags below your Context tag to monitor and control all of the I/O processes, alarms, logging of a device.
- Created properties in the Context tag to fully describe the device.
- Created parameter expressions within the child tags so that configuration is done automatically, using information stored in one place - the Context tag.

Then it's time to turn the Context tag into a new type. This is not mandatory. You could copy the Context tag and it's full structure of child tags for every new instance of the device. But, turning the Context into a type makes many other tasks easier.

Note: You will need the Manage Tag Types security privilege to perform the tasks described here.

The steps to create a tag type are as follows.

1. Ensure that your Context tag has a value in the Type field.
 - The Type value will become the name of the new tag type.
 - This value must be a single word and must be valid for use as tag type name. It must not match any existing tag type. If this condition is not met, an error message will be displayed.
 - Type values can have a maximum of 31 characters.
2. Right-click on the Context tag in the Tag Browser to open its context menu.
3. Select the option, Create New Type.

A dialog will open to confirm that you wish to proceed with this action. This same dialog provides an opportunity to control which of the built-in widgets will be associated with the new type. Some of these are not optional, and cannot be deselected. You may choose to create custom widgets at a later date for your new tag type.

5. Click, OK.

The new tag type is added to your application, and the selected context tag is converted to be an instance of this type.

Troubleshooting:

- An error dialog reports that the type name is not valid.

There must be a type name, which is a single word.
- An error dialog reports that the name exists, but you have not used that word elsewhere.

Name conflicts with other parts of VTScada can happen. The names, "Pump" and "Calculation" for example are both taken.

Exercise 4-3 Turn a Context tag into a Type.

1. In the Tag Browser, right-click on the tag, Pump 1, then click Create New Type from the pop-up menu.
2. In the Create New Type dialog that opens, click OK.

That's all there is to it. Much of the work that makes this feature so powerful was done earlier when you added parameter expressions to the child tags. The benefit comes when you want to create new pumps.

Exercise 4-4 Create a second pump

1. Right-click on PLC1, then select New Child.
2. Expand the All Tag Types group.
3. Find and select CustomPump.
4. Name the new tag, Pump 2
5. Set the description to Secondary pump
6. Open the Other Parameters tab.
7. Enter 20 for the I/O Address Offset.
8. Click OK to close the properties dialog, then examine the results in the Tag Browser.

Pump 2 should have the same set of child tags as Pump 1, but with descriptions that refer to Pump 2 and I/O addresses that are 20 greater than those under Pump 1. All tags should show a value, most of which will be 0 because Pump 2 defaults to the Off switch position.

Not only is it easy to create new pumps now, it's also easy to make changes. If the I/O addressing changes to add 100 instead of 20 then you need change only one field in one tag and all the child I/O will be updated automatically.

Recall that in the Tag Parameter Expression editor, you left the option selected to "Optimize to only evaluate at tag initialization". Whenever you edit the properties of a parent tag, all of the child tags re-start, but only when you edit those properties in the Tag Browser.

This example works well because the I/O addressing follows a pattern. But, a variation could work nearly as well if I/O addresses were random. In that case, you would create a property for each I/O address in the parent Context (CurrentAddr, FaultAddr, ModeAddr...). Each child tag would use a one-word expression to refer to the appropriate property. (..\CurrentAddr, ..\FaultAddr, ..\ModeAddr, etc.) The advantage is that you would be able to assign all the I/O on one screen using a list, rather than needing to open and configure every child tag individually.

What if the addresses use letters as well as numbers? Then you create expressions with Concat ().

Modify a Custom Type

It is inevitable that you will want to modify a custom tag type that you have created. You might decide to change the parameters in the parent. Or, you might decide to add, edit or remove child tags. Or, perhaps you just want to change a few things about one instance. Each type of change has its own procedure.

Note: If security is enabled, the privilege, "Manage Types", must be granted before you will be able to proceed. This privilege is not granted to any role by default.

Override a property of a single child tag instance.

If one instance of a custom type needs to have slightly different properties from all the rest, you will override that instance. For example, perhaps it's a characteristic of the primary pump that it is prone to overheat when running faster than 1000rpm, but the secondary pump doesn't have this problem. You might modify the speed I/O tag in the primary pump like so:

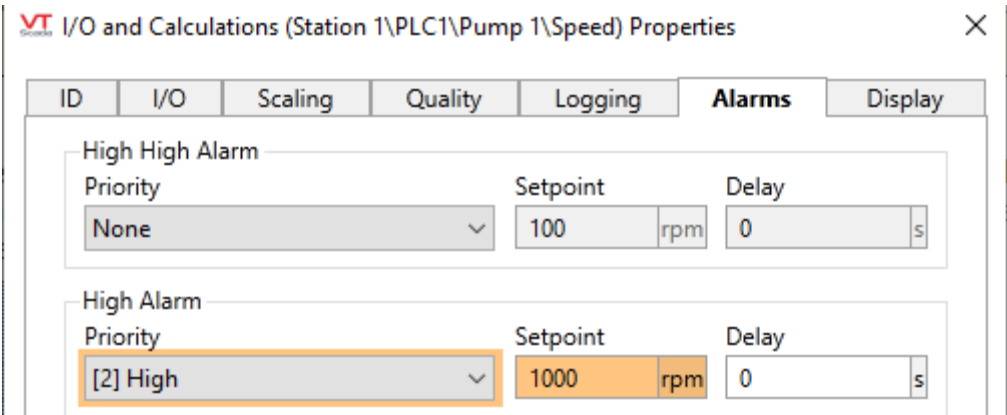


Figure 4-17 The orange color means that you are overriding default values.

All parameters in all child tags of a user-defined type are set automatically using whatever value was in that field when the type was created. You're allowed to override those values whenever and wherever you need. The orange color serves only as a reminder: "this value is an override, not a default".

In the Tag Browser, you'll get a similar warning:

HUA position	Pump 1 - Demonstration - HUA position	Discrete [In]
Running	Pump 1 - Demonstration - Running	Digital [In]
★ Speed	Pump 1 - Demonstration - Speed	Analog [In]

Figure 4-18 The star on the orange dot indicates that this tag contains at least one parameter override.

If you were to add a new child tag in Pump 1, it would show with a green plus symbol:

	Name	Description	Type	Equipment Type	Address
	Current	Pump 1 - Demonstration - Current	Analog [In]		40031
	Fault	Pump 1 - Demonstration - Fault	Digital [In]		7
+	Flow	Pump 1 - Demonstration - Flow	Analog [In]		40033

Figure 4-19 "This child tag was added explicitly by a user"

If you delete a tag, then it's deleted. No symbol will show that it ever existed.

In all cases, these overrides affect only Pump 1. Nothing has yet changed Pump 2 or any other new CustomPump that you create. For that, you can...

Redefine a Type Definition

You can right-click on the parent tag of the structure, Pump 1, and from the menu choose Redefine Type. Instantly, all of the overrides within Pump 1 will be made part of the CustomPump definition. Pump 2 (and all subsequent pumps) will gain a warning alarm on the speed and a new demo tag.

Caution: There's a risk that the instance contains overrides that are meant only for that one object, not for all. Take extreme care when using this command.

When you decide to modify a user defined type definition, the recommended procedure is to:

1. Create a temporary instance of that type. Call it Template or something equally obvious.
2. Override the child tags as required. Add new tags, modify parameters, even delete (or better, disable) obsolete tags.
3. Run Redefine Type on that temporary instance.
4. Delete the temporary instance.

All the current pumps and all news ones will use the redefined type definition with one very important exception:

All existing tags keep their overrides.

Therefore, in the case of a conflict between a parameter that you change in the type definition and a parameter that you overrode in one particular pump, you always keep the local override.

Exercise 4-5 Redefine a Type

To begin, you'll create a one-off change to Pump 1.

1. Using the Tag Browser, open the properties dialog for Pump 1\Speed.
2. Open the Logging tab and set the deadband value to 2.
3. Add a high alarm as shown

The screenshot shows a dialog box titled "I/O and Calculations (Station 1\PLC1\Pump 1\Speed) Properties". It has several tabs: ID, I/O, Scaling, Quality, Logging, Alarms, and Display. The "Alarms" tab is active. Under this tab, there are two alarm configuration sections. The first is "High High Alarm" with a priority of "None", a setpoint of "100 rpm", and a delay of "0 s". The second is "High Alarm", which is highlighted with an orange border. In this section, the priority is set to "[2] High" (also highlighted with an orange box), the setpoint is "1000 rpm", and the delay is "0 s".

4. Click OK to save the properties.
There should now be a star beside this tag in main window of the Tag Browser.
You won't see the star in the left-panel navigation window.

Next, you'll create a temporary pump instance and modify that.

1. Add a new CustomPump as a child of PLC1
2. Name this tag, `Template`.
Do not set any property other than the name.
3. Within Template, open the properties of the tag, Speed.
4. In the Logging tab, set the deadband to `1`.
5. Click OK to close this properties dialog.
6. Add a new I/O and Calculations tag as a child of Template to monitor the fault status of Pump 1.
Name the tag `Fault` and set the type to `Digital`. The address is `7`. Be sure to use expressions for the description and read address, matching those in the other I/O tags. Optionally, you might set an alarm, triggered by this tag changing to state 1.

"Fault" will be required by a later exercise. Be sure to both create and draw it.

7. Right-click on Template, then Redefine Type.
8. Read the warning message that appears, then click OK.
The star vanishes because your changes are no longer overrides. They're now part of the type definition.
9. Take note: Normally, you should delete the template at this point, but it will be needed in the next exercise. Leave it be for now.
10. Open the tag, Speed, in Pump 1 and examine the deadband in the Logging tab, then do the same in Pump 2.
Is the result as you expected?

Change the Parameter List

After turning a Context tag into a new type, the Settings tab becomes the Other Parameters tab. The ability to add and edit parameters has vanished. You can still work with the parameter list, but now you must use the Manage Types page in the Application Configuration dialog.

Modify a Type using external databases

Each Type has the potential to launch child tags when a tag of that Type is instantiated. These Type descendants can be created, modified, and deleted using external tag databases via this panel.

To do this, you must first select the Type and export the descendant tags to an external database. After modifying them, you can synchronize your application with the external database to apply the changes you have made. Note that the database created is marked or export so that when synchronizing it will automatically be applied to the correct Type - no selection is required for the synchronization step. As part of the synchronization process, the external database must either be updated to stay in sync with your application, or deleted.

To edit the properties of a Type or to remove a Type it must be declared by this application. A Type can only be removed if the application running and there are no running instances of the Type.

You must have the Tag Add/Modify/Delete privileges to remove or edit a Type, and to import Type descendant tags.

Output Type:

☒ Use a Microsoft Excel spreadsheet file
☐ Use a Microsoft Access database file
☐ Use an ODBC data source

Clean Up Options:

☒ Update the file after applying changes (increases the time it takes to sync tags)
☐ Delete existing file after applying changes

Other options:

☐ Open file after export (Microsoft Excel and Microsoft Access only)
☒ Mark empty sheets as hidden (Microsoft Excel only)
☒ Exclude tags created automatically and not changed

☐ Show OEM types

Select a Type to export, edit or remove. Selection not required for import.

Type	# of Descendants	Removable	Editable
CustomPump	9		<input checked="" type="checkbox"/>

Remove

Edit

Sync

Export

Figure 4-20 Manage Types

All of the user-defined types in the current application will be listed. You have two options:

- Export and synchronize just the tags within the type. Use this to modify the child tags outside VTScada.
- Edit the parameter list of the type definition.

Our focus here is the second of those two options. It's usually the easier method.

Select a type, then click the Edit button to open the Edit Properties page. Here you can work with the parameters of the type, just as you could in the original Context. In fact, this is slightly more powerful: you can use this page to delete the HelpKey parameter if you are not using it.

Edit the properties of "CustomPump"

You may add, delete and rearrange parameters. Parameter labels are used in the Type's Config Folder.

Parameter Name	Parameter Label	Text
Name	Name	<input type="checkbox"/>
Area	Area	<input type="checkbox"/>
Description	Description	<input type="checkbox"/>
HelpKey	Help Search Key	<input type="checkbox"/>
IO_Offset	I/O Address Offset	<input type="checkbox"/>
CustomDetailsPage	Custom Details Page	<input type="checkbox"/>
SiteListDisplay	Site List Display	<input type="checkbox"/>

Figure 4-21 Detail from editing a type's properties

Work directly in the list to change properties. Use the Add button to create new properties. Note that you can provide only a name and label when doing so. There is no way to set a default value here for your parameter.

The Translate option refers to the value and means "This property is text".

Besides providing a way to edit the parameters, this is useful as a reminder for the names that you gave each parameter. After the Context has been turned into a type, only the label will show, making it difficult to build an expression that uses the parameter if you forgot the name.

Exercise 4-6 Manage type properties

In the next topic of this chapter you will see Start Tag expressions. To prepare for an exercise that will use those, you will need another parameter to use in the expression. This is a perfect reason to modify the parameter list of the CustomPump.

1. Ensure that the Tag Browser is closed.
2. Open the Application Configuration dialog.
3. Select the Manage Types page.
4. Select your tag, CustomPump, then select Edit.
5. Delete the HelpKey property.
6. Add a new parameter named `HasFlowMeter`
(All one word)
7. For the label, type: `Has flow meter`
8. Note that you cannot set a default value.
9. Click OK to save.
10. Ensure that HasFlowMeter is just below IO_Offset. Move it if needed.
11. Select Apply.
12. Provide a suitable comment when prompted.
13. Close the Application Properties dialog then open the Tag Browser.
14. Open the properties of Pump 1 to the Other Parameters tab.

15. Set the value of the new property, Has flow meter, to 1
 16. Do the same for Pump 2, but enter a zero.
 17. Close the Tag Browser.
- You'll put these properties to work after the next lesson.

Start Tag Expressions

You can add a parameter expression to every parameter within a tag's properties dialog. In all cases but one, these enable you to set the value of the parameter based on conditions at start. The exception is the name parameter because you cannot use an expression to assign the name. Instead, an expression on this parameter is used to control whether the tag should start.

For example, the MultiSmart, and MPE station tags include an option to set the number of pumps in the station.

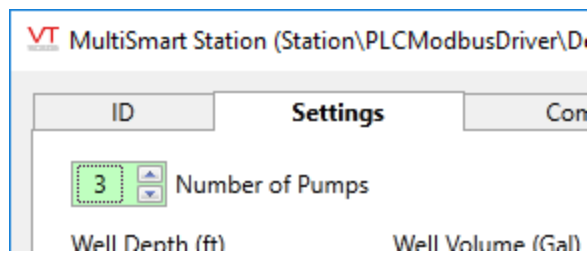


Figure 4-22 The number of pumps in the station tag can vary from 1 to 6

The tag structure varies according to the number you choose:

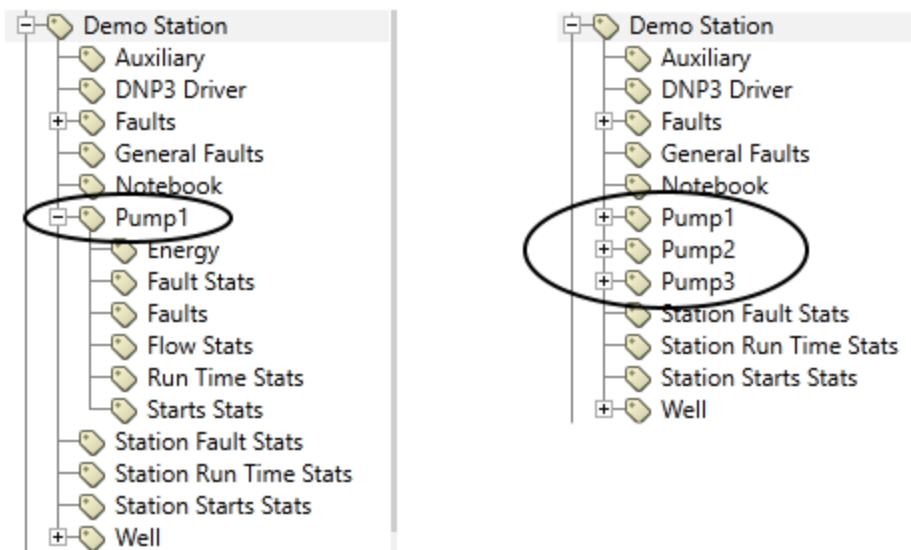


Figure 4-23 On the left, one pump. On the right, three pumps configured.

Changing the selector doesn't change the number of pump tags *created*. It changes the number *started*. Selecting the Show Disabled option in the Tag Browser reveals the difference.

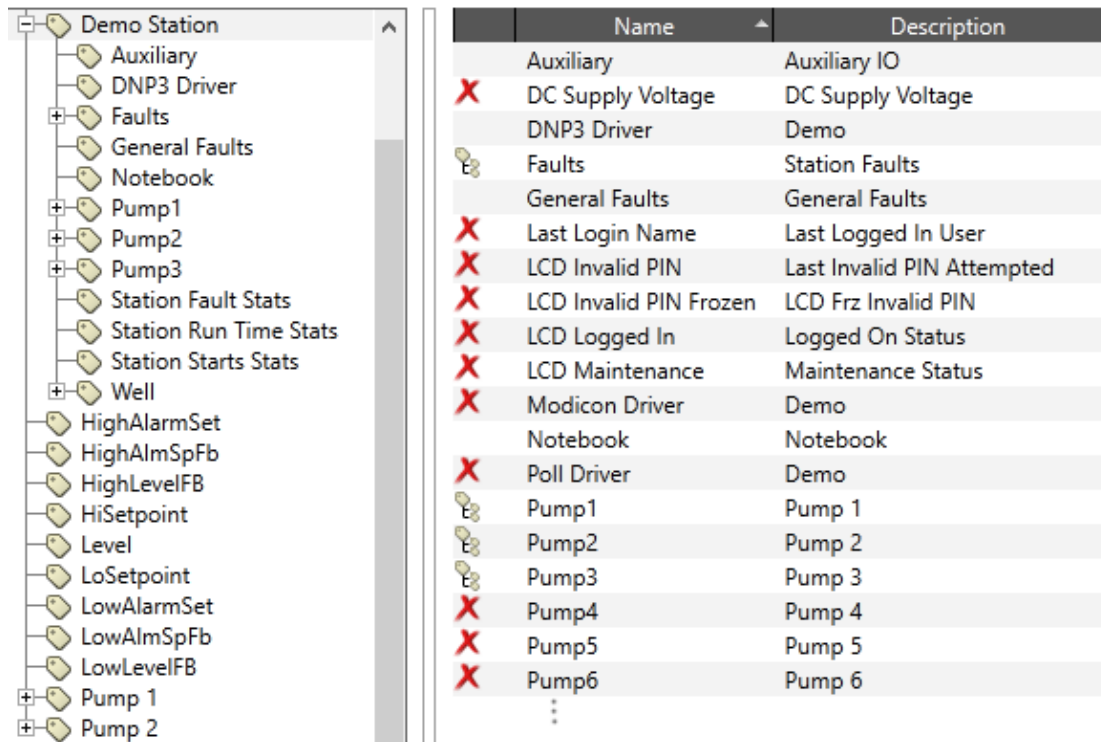


Figure 4-24 The Multismart always has six pumps. The choice is how many to start/enable.

To use this feature, add a Start Tag Expression to the name parameter of a tag.

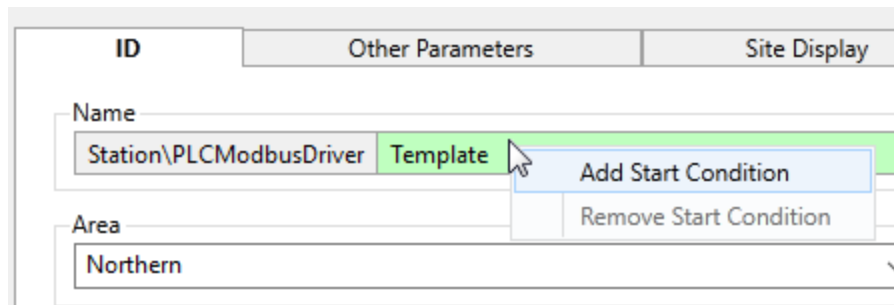


Figure 4-25 The method is the same as adding a Tag Parameter Expression.

In the Tag Parameter Expression editor, whatever expression you create must ultimately return a Boolean result, meaning either zero or one. If the expression works out to true at start-up then the tag starts. If false, it doesn't.

Exercise 4-7 Create a tag with a start condition

1. Open the Tag Browser and navigate to Template under PLC1.
2. Open the properties of Template.
3. In the tab, Settings, set HasFlowMeter to 1.
4. Close the properties dialog.
5. Open the properties of the child tag, Flow.
6. Right-click on the name, then choose Add Start Condition from the menu.

7. For the expression, enter:

```
PickValid(..\HasFlowMeter, 0)
```

8. Click OK to finish.
9. Right-click on Template.
10. Select Redefine Type.
11. Click OK when asked to confirm this action.
12. Delete the tag, Template.
13. Examine Pump 1 and Pump 2. The flow rate (Flow) should be present in Pump 1 but disabled in Pump 2.

Exercise 4-8 Practice making custom types

Using what you have learned in this chapter, you will create a new type using the Station 1 context. Design your stations so that some can have one pump while others have two. (You just finished an exercise that shows how to achieve that.)

Before using the Create New Type command on the top level context, you should do the following:

- Change the type property to `StationType`
- Ensure that site properties have been added. Keep all the site properties.
- Remember that the goal is to do all configuration in one place. A new station would need to be told the TCP/IP address for its port. Also, everything that you currently configure within a CustomPump (such as the I/O_Offset address and Has Flow Meter) should be given an expression referring to a new property that you will create in the parent Station.
- Add new properties within Station for the following. Write down each property name as you create it.
 - TCP/IP address of the station. (default, same as for the current TCP port)
 - Pump 1 I/O offset value. (initial value: 0)
 - Pump 2 I/O offset value. (initial value: 20)
 - Pump 1 has flow meter (initial value: 1)
 - Pump 2 has flow meter (initial value: 0)
 - Number of pumps (initial value: 2)
- In the child tags, PLC1_Port, Pump 1 and Pump 2, add expressions to use the properties you created in the parent context. No math required. Your expressions simply need to refer to the parent property name (example: `..\Pump1_IO_Offset`). It is not necessary or desirable to redefine the pump types after doing this.
- For Pump 2, add a Start Tag expression so that this pump will run only if the number of pumps in the station is greater than 1.

After doing the above, turn the Station into a new type.

1. Create a new instance, Station 2 in the area, West.
The IP address is the same as for Station 1, 127.0.0.1 (See following note.)
The number of pumps should be set to 1. Therefore, there is no need to configure anything related to Pump 2.
Pump 1 configuration should be the same as for Station 1 with an I/O offset of 0 and a 1 for Has flow meter.

Note: In normal operations, every station would have a unique IP address. But you are connecting to a simulator, and it is listening only on the one address. For each new station, use the same IP address. The simulator will launch a new, independent virtual station each time, keyed to the unique ID of the tag.

This also explains why you will see setpoints resetting to defaults. Each time you restart the tag a new simulated station is created

Connectors

A site holds information about equipment at a location. A connector holds information about something that runs between two sites. This could be a pipe, a transmission line, a railway...

A connector is based on a Context tag and therefore can have any properties that you want to configure. But, rather than having latitude and longitude properties, it will have properties to store site 1 and site 2, which are two station tags that the connector connects.

Connectors are drawn as straight lines between the sites. You can configure several properties of that line, but you can't draw it as a curve and it will always be just one straight line.

Exercise 4-9 Bonus Exercise: Experiment with Connectors

After you have two stations add each to a map as follows:

1. Open the Sites page.
2. Click once on Station 1.
The Site Details page for Station 1 should open.
3. In the map portion of the Site Details page, click the Update Site Location button.
4. Click somewhere on the map to place the site.
5. Back up in the sites list to do the same for Station 2.

In the next set of steps, you will create the connector:

1. Add a new Context tag named StationConnect at the root level of the tag hierarchy.
Give it an area and description.
2. In the Other Parameters tab, click the Add Connector Properties button.
3. In the Site Display tab, select Station 1 as the Start Site and Station 2 as the End Site.
4. Click OK and view the result on a Site Map.
5. Reopen the properties of StationConnect and experiment with the Connector Style, Directional Arrows and Connector Color properties in the Site Display tab.

Multi-Write Tags

Not counted towards your tag license limit.

The MultiWrite tag will write predefined values to a list of selected tags upon triggering. This can be used to place a plant into a state of operation or to quickly shut a plant down in one step.

Tip: Trihedral technical support frequently receives requests for help with scripting when people want to send a control signal in response to a trigger. The response is almost always to use a Multi-Write tag in place of a script.

Reference Notes:

Up to 100 output and memory tags can be controlled by a single MultiWrite tag, with a defined value set to be written to each. The trigger for the write may be any of a manual button press, a tag that changes state from false to true, or an expression that evaluates to true.

Each value to be written will be checked to ensure that it is valid before it is written. Invalid writes will be ignored. All valid values will be written, regardless of whether they have changed since the last write.

Should 100 tags not be enough, it is possible for one MultiWrite tag to trigger another upon finishing its write sequence.

An event will be recorded in the event history whenever a MultiWrite is triggered.

If triggering a Digital Control or Digital Output with a pulse duration, use care that you do not attempt to send a pulsed zero.

Set Value Buttons and Multi-Write tags.

This tag will work with the Set Value Button Widget but support is limited:

- Value to Write must be set to 1 in the widget for use with a Multi-Write tag. No other value will work.
- Multi-write tags cannot act as feedback tags. To use Select Before Operate in the Set Value, a separate I/O tag must be configured as the feedback tag and used as the linked tag in the Execute / Cancel buttons.

MultiWrite properties Write List tab

The Write List displays the output or memory tags that will be written to, and controls what value will be written to each. The outputs will be written in the order in which they are displayed.

ID	Output Tag	Value
1	[CommChannel\High Alarm Set] High a	Click to Add
2	[CommChannel\Low Setpoint] Low leve	Click to Add
3	Click to Add	
4		

You can add or edit any tag in the list by clicking on a row. You must select a tag before configuring the value to be written to it.

The Tag and Value input fields will be activated when a row is selected, except that rows must be filled in sequence from #1. You cannot add a tag to row three before row two has been completed.

The grid displays the name and description of each tag selected to be written to. The Value column shows what will be written to each tag when the MultiWrite is triggered. The value may be any of a constant, an expression or another tag's value. In the case where another tag's value is to be written, that tag's name will be display in the Value column.

Note that, the display may show a tag's Unique ID rather than its name when the tag is not selected. The description will always be shown for every tag, to help avoid confusion.

To add a tag to the list, select the next available row, then click on the Tag Browser button. Select or create an output tag. After you have specified the tag to write to, you can provide a value to write using the Value field.

To remove a tag from the list, select its row, then click on the X button in the lower-right corner of the window.

The arrow buttons below the grid may be used to move a selected row up or down in the list, thus controlling the order in which output tags are written to.

MultiWrite properties Activation tab

The Activation tab provides a way for MultiWrite tags to be controlled automatically.

Activation Trigger

Any tag or expression that will change from a false (0) to a true (1 or any non-zero number) can be used to trigger the write.

Privilege

Select a custom security privilege from this drop down to limit the operation of this control to only those operators who have been granted the matching security privilege.

Log an event on automatic multi-write

Selected by default. You can choose to not log events when multi-writes occur in response to the activation trigger. Choose this if you have a multi-write operation that triggers frequently.

An operator pressing the Multi-Write Button or Hotbox will always cause in a new event to be logged.

5 Create Widgets

Anything that you have drawn can be grouped into a new custom widget. There are three kinds, any one of which may be best depending on your purpose.

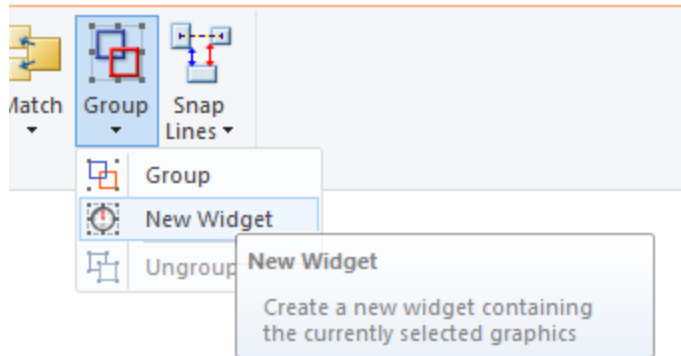


Figure 5-1 Detail from the Idea Studio, home ribbon

This can be as simple as a grouping used to make alignment easier on a page, or as complex as the components of the MultiSmart and MPE Site pages.

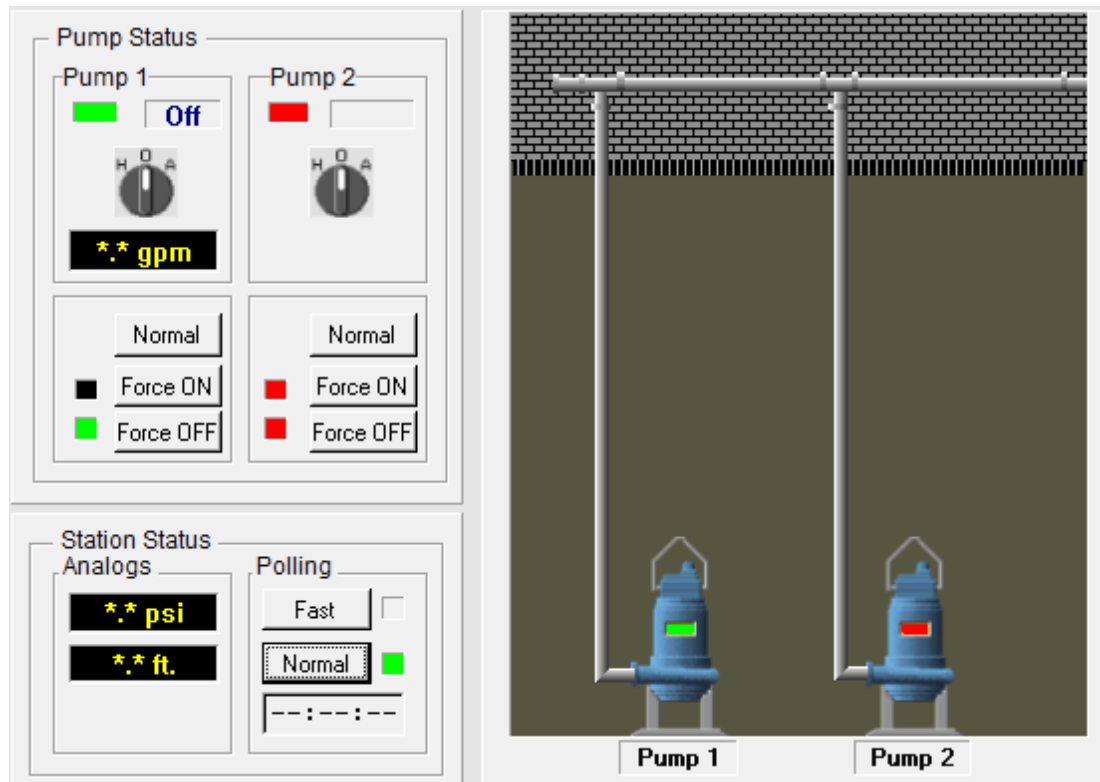


Figure 5-2 A single widget, with many components.

There are three classes of custom widget, listed in order of increasing features and capabilities:

Group

- Best choice for moving and aligning a set of elements while working in the Idea Studio.
- Worst choice for creating a library of shapes.
- You cannot assign the name.
- Cannot be added to a palette.
- Cannot be linked to a tag, even if the elements within the group are tag-linked widgets.
- If ungrouped, the file remains in your Widgets folder, but cannot be used (no user-assigned name).

Plain Widget (Simply labeled "Widget" in the dialogs)

- Elements may include one or more existing widgets in addition to shapes or images.
- Must be given a name.
- Automatically added to the palette.
- Tag links, if any, are done through parameters rather than directly to the widget.
- Does not become a native widget for any tag.

Tag Widget

- At least one existing widget must be included in the selection set.
- All of the included widgets must be linked to tags before grouping.
- Must be given a name.
- Automatically added to the palette.
- Becomes a native widget for a tag type, which is typically the common parent of all in the selection. (Other types can be added to the list.)

When creating either a plain widget or a tag widget, you will see the New Widget dialog.

Note: Widget names cannot match the short name of any tag.

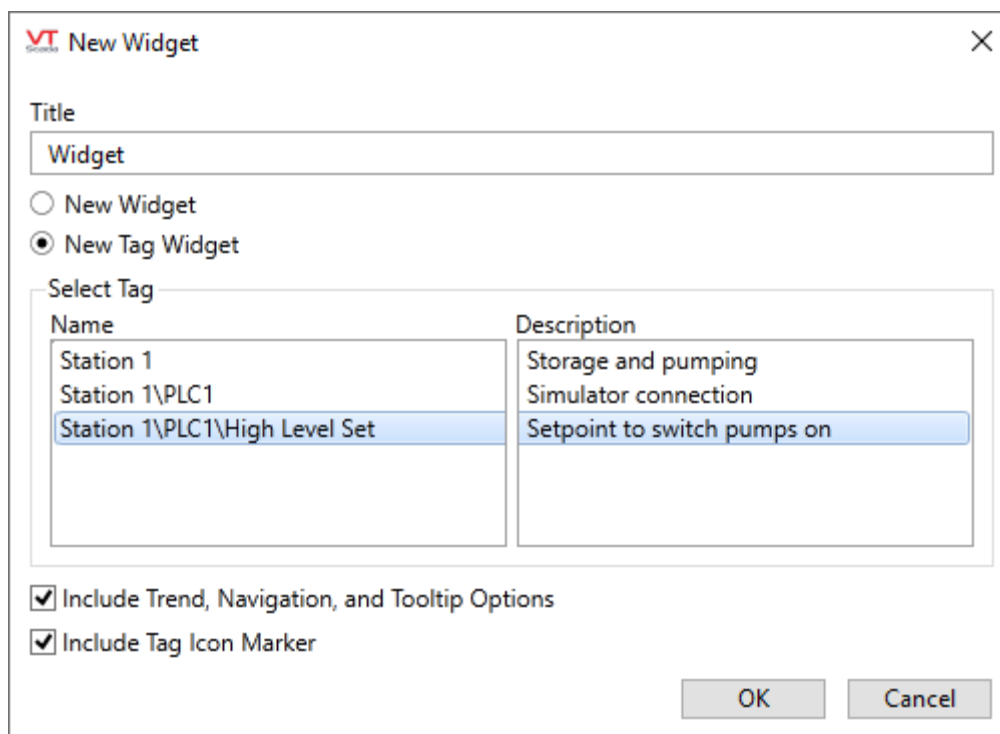


Figure 5-3 The New Widget dialog

New widgets are added to the top level of the Widgets palette and also to the list of most recently used widgets. Each time that you create a new widget, you should take a moment to reorganize the palette, moving the menu item into a folder of your choice. By keeping the palette organized, you will save time later when looking for a widget that you want to use.

Any individual widget can be broken apart into its components by selecting the instance, then clicking "ungroup" in either the formatting toolbar or the right-click menu. If your intent is to change the appearance of a widget, you should edit the widget definition rather than following a process of ungroup - edit - regroup.

Also, while you can nest groups into widgets into other widgets, you are advised not to. It's simply easier to manage groups and widgets that are not nested than those that are.

Widget definitions can be opened for editing the same way that a page is opened. Use the Open command in the Idea Studio's file menu to open the widget by name or select any instance in a page and click Edit, either in the toolbar or the right-click menu. Your changes will affect all instances of the widget or group throughout the application.

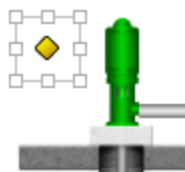


Figure 5-4 A Tag Icon Marker

You can choose whether to add a [Tag Icon Marker](#) to your widget. The icon (yellow blob) will not be visible to operators unless you take steps to make use of it, which usually means adding a parameter named "Questionable" to your context tag.

You have the option of giving every new widget, both plain and tag, its own parameters for operator interaction. In most cases, you probably won't want those. If your widget includes a control widget, a click on one is a click on both. If you include the Trend, Navigation, and Tooltip controls, and if you want to operate a control within the widget without having the Trend window for your widget opening, you are advised to open the new widget's properties dialog and select Disable Trend. In many cases it is easier not to include the operator interaction tools.

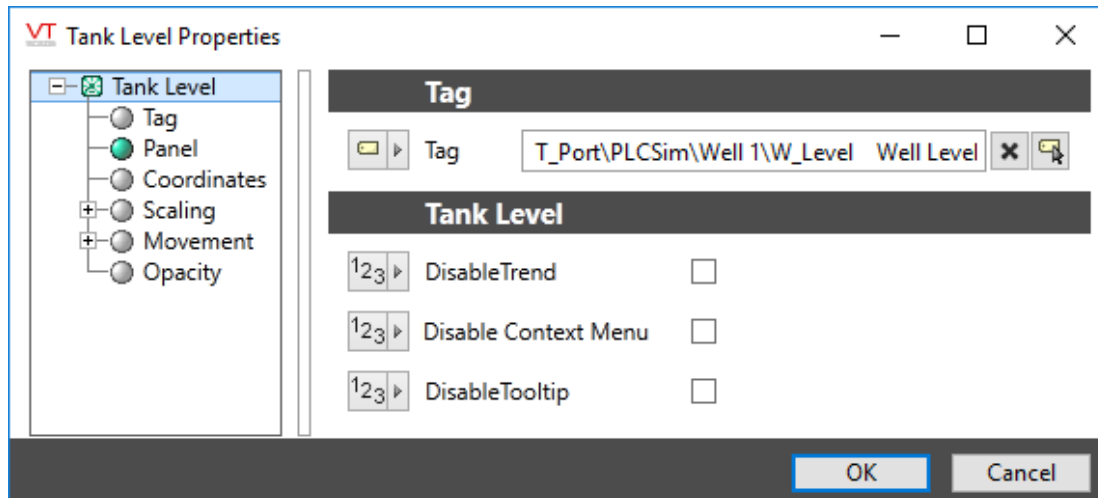


Figure 5-5 Operator interaction parameters. It's your choice whether to include these.

A widget can be given a background color, but this will be displayed only in the Idea Studio and in the palette. When drawn on a page, the widget's background color will not be included.

Application security privileges cannot be selected within widgets. To protect output controls, apply the security privileges to output tags or to pages.

Widget Rotation

Tag Widgets

Most widgets in the Idea Studio palette are tag widgets. VTScada makes it easy for you to create your own, extending the options available to create a user interface that is customized for your industry or application.

Tag widgets have the following features:

- Tag widgets are designed to be linked to a single instance of one or more types of tag. They are native representations of tag values, available when you click "Draw" on a tag selected in the Tag Browser.
- The tag to which a tag widget is linked may be a parent tag. When properly configured, all of the child tags will be assigned to components of the widget automatically. This enables you to draw an entire station or complex equipment with a single widget.
- Tag widgets may be given parameters, but this is uncommon.

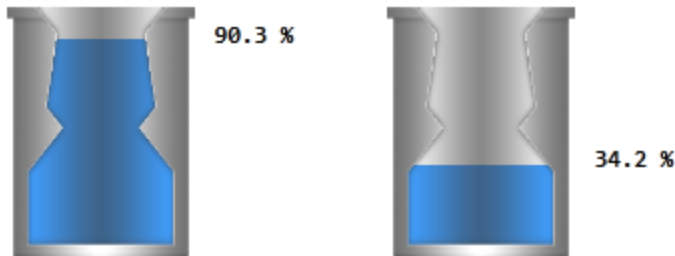
Example:

Figure 5-6 Tank Level with bar and floating text

This example of a user-created tag widget, Tank Level, is designed to represent analog values as a tank filling with fluid. The numeric display rises with the fluid level. Tank Level has become a native widget for use by any analog tag within the application.

The easiest way to create a new widget is to draw the components then group the pieces. Further editing can be done if required. All existing widgets that will go into the new widget must be linked to tags before being grouped.

Tip: Want to create your own version of the Tank Level? Here's how. Feel free to vary these instructions as appropriate for your application.

1. Draw an Analog I/O tag (or Analog Status) as a Color Fill widget.
2. Place a tank cut-out image above it and scale both so the Color Fill exactly matches the cut-out.
3. Draw the same tag as a Numeric Value widget, *placed to align with the bottom of the tank.*
!! Do not move this widget up or down after deciding on its placement !!
4. Check the Color Fill widget's coordinates to obtain its height in pixels. (Use for Y in the following expression)
5. Open the Numeric Value's properties and configure the Vertical Movement property as shown, where "Your tank height" is the number from the last step and the tag is the same one used to draw this widget. *Do not move the widget up or down after completing this step!*
6. Group the Color Fill, the tank and the Numeric Value as a new Tag Widget, changing the name, but otherwise accepting all defaults.

Exercise 5-1 A new widget

In a recent exercise you copied two widgets related to the status of a pump, then updated the links in the copies. This is a perfect candidate for a new widget.

1. Open the Idea Studio to the page, Station Status.
2. Select the Equipment widget and the fault indicator widget that are linked to Pump 1.

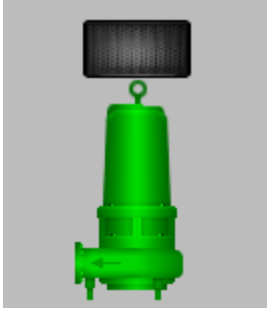


Figure 5-7 Example shows fault as an indicator - you might have drawn an Alarm Priority Box.

1. Right-click on either to open the pop-up menu.
2. Expand the Group option.
If "Group" is not available as an option, you have only one object selected. Ensure that both objects remain in the selection set.
3. Select New Widget from the menu.
4. In the New Widget dialogue, set the name to `Pump State` and ensure that the selected option is New Tag Widget, not New Widget.
DO NOT CLICK OK - LEAVE THE DIALOG OPEN WHILE READING THE FOLLOWING.

In the Select Tag portion of the New Widget dialog, Pump 1 should be the default tag for the new widget.

Tag Widgets with more than one tag should be designed to link to the closest tag in the tree that is a parent to all the tags displayed in the widget and also makes sense for what the widget represents.

The two widgets you choose both represent tags within a single pump, therefore a pump context makes the most sense in this example. (Note: you might want to check the Tag Links panel of the Idea Studio to ensure that both selected widgets are linked to child tags of the same pump. If not, fix that before proceeding.)

VT New Widget

Title: Pump State

☐ New Widget
☒ New Tag Widget

Select Tag

Name	Description
Station 1	Storage and pumping
Station 1\PLC1	Simulator connection
Station 1\PLC1\Pump 1	Primary pump
Station 1\PLC1\Pump 1\Fault	Pump 1 fault
Station 1\PLC1\Pump 1\Running	Pump running status

☐ Include Trend, Navigation, and Tooltip Options
☐ Include Tag Icon Marker

OK Cancel

- In the New Widget dialog, ensure that Pump 1 is selected.
 If so, then click OK.
 If not, stop and reselect the objects to ensure that only those linked to child tags of Pump 1 are included.
 Your new widget has been created and is almost ready to use.

Tip: The lesson of that last step is NOT that you should always select the next higher or lower tag in the hierarchy for your widgets. Rather, it's that you should always think about the choice and select the most appropriate tag. In most cases, the default will be the best choice.

Exercise 5-2 Use the new widget

- In the widgets palette, make sure that you are looking at the top level of the palette (i.e. no folder is open) then scroll down to the bottom of the list.
 You should find the Pump State widget.
- Drag this to the page.
- Link to Pump 2.
- Switch to Operator View to examine the results.

Exercise 5-3 More Widget practice

1. Open the page, Pump Controls in the Idea Studio.
2. Group all of the pump controls and the text into a new tag widget named wPumpControls.
The "w" stands for "widget" and you'll see why it's useful soon enough.
3. In the properties dialog for the new widget, ensure that two "Include..." options are deselected.

Keep a Tidy Palette

Every new widget that you create will be added to the top level of the palette. You don't have to create very many before this becomes disorganized.

Try to develop a habit of maintaining a tidy palette. You should create folders for your custom widgets, moving new Menu Item tags to those folders. These can be at the top level of the palette, or within existing folders.

If you find that you use the same dozen or so VTScada widgets for most of your work, you can save time by creating a folder named Favorites, to which you can copy your most frequently-used palette menu items.

Tip: You can change the order of folders in the palette by working with their Menu Item entries in the Tag Browser.

Tip: Note that you can select multiple tags in the Tag Browser's main window by holding down the Ctrl key while selecting. Use this to copy, move, or delete several tags at once.

Exercise 5-4 Organize your palette

1. Open the Idea Studio and ensure that the Widgets palette is open to the top level.
2. Right-click on the Pump State widget, then select New from the menu that opens.
3. In the New Item dialog, choose Folder.
4. Give the new folder the title, `Custom Widgets` and select OK.
5. Right-click on Pump State in the palette, then choose Cut.
6. Right-click on Custom Widgets, then choose Paste as Child.
7. Move the wPumpControls widget too.

Edit a Tag Widget

After creating a custom widget you will likely want to change it.

At the code level, a widget is very similar to a page. Both can be edited within the Idea Studio using exactly the same tools. In fact, it can be difficult to tell whether you are editing a page or a widget. The easiest way to check is to note the icon used in the tabs at the bottom of the Idea Studio.

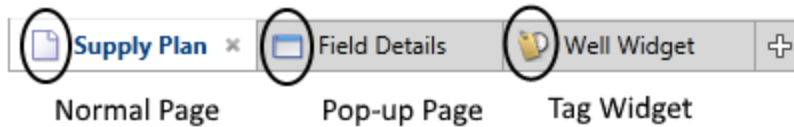


Figure 5-8 There's an icon for each type of page and widget

That and the toolbar ribbon...

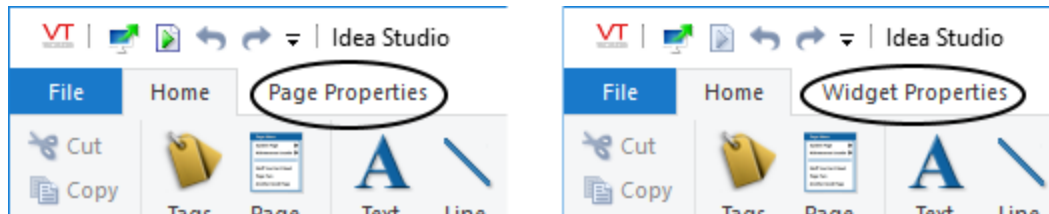


Figure 5-9 Don't ignore the ribbon title

There are two ways to open a widget for editing in the Idea Studio:

- Use the Open command, then select the widget.
The Open command is available from the File menu, the plus button, and the keyboard combination Ctrl-O.
- Right-click on a custom widget within a page then choose Edit from the pop-up menu.

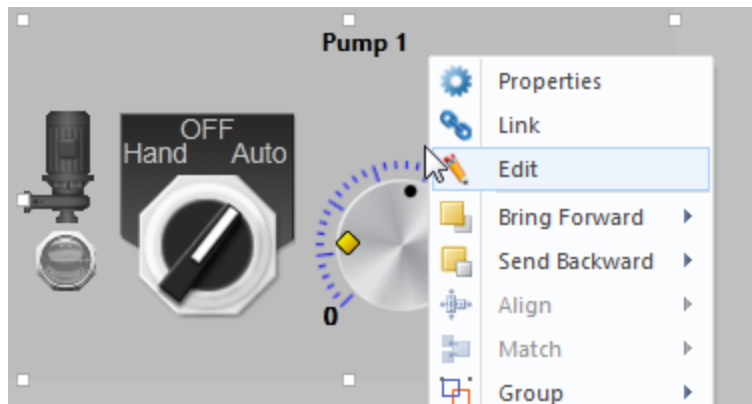


Figure 5-10 Right-click menu for a custom widget, within a page, within the Idea Studio.

Note: When editing one of your custom widgets, everything you do goes into effect immediately, changing all instances of that widget on all pages.

Reference Point

The reference for every widget is the upper left corner. This is 0,0 in the Idea Studio's coordinate system. Do not move the pieces of a widget away from this corner unless you want every instance of that widget to jump to a new location.

Equally important: Never scale the pieces of a widget while editing unless you want every instance to scale. Results can be unexpected.

Tools in the Widget Properties Ribbon

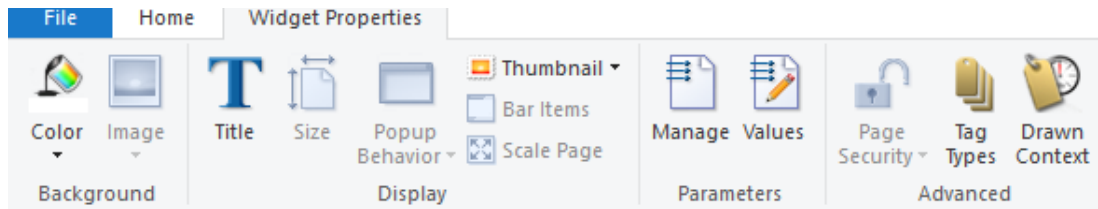


Figure 5-11 Tools in the Widget Properties ribbon

Many of the sections are shared with the Page Properties ribbon, with options that do not apply to widgets disabled. As shown, you can adjust the background color (this doesn't affect the drawn widget, but can be useful if your widget will normally be placed on a dark background) or change the title if the original name isn't suitable.

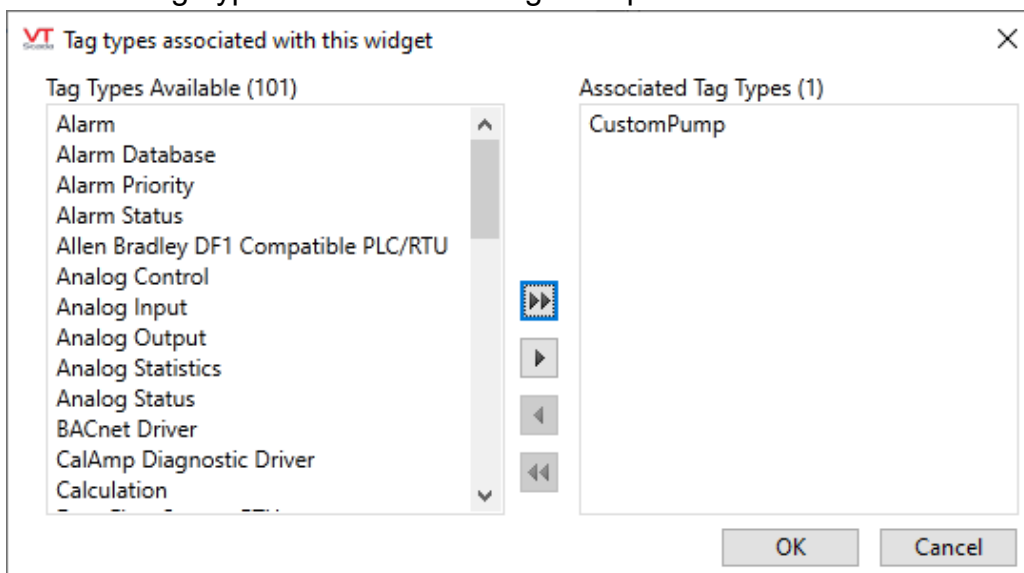
Of particular note are the tools Tag Types and Drawn Context.

Tag Types

A new tag widget can be linked only to tags like the one to which the original was linked. It is standard practice to expand this list to include similar types. For example, if the original was linked to an Analog Status tag, you may also wish to be able to link new instances of the widget to Analog Input, Calculation, and other tag with numeric values.

To link your widget to other tag types:

1. Within the Idea Studio, open the widget for editing.
Use the Open command in the file menu, press Control-O, or right-click on a drawn instance of a widget and select Edit from the pop-up context menu.
2. Select the Widget Properties ribbon.
3. Click the Tag Types button in the Widget Properties ribbon.



The Edit Parameters dialog opens.

4. Use the arrows to fill the column on the right with all the tags that you might want to link to your widget.

Troubleshooting:

- The Widget Properties ribbon is not visible.

The Widget Properties ribbon is visible only when editing a widget. Use the Open command to open a widget for editing.

Drawn Context

The Drawn Context is used only within the Idea Studio. It does not create a permanent link to any tag so far as the definition of the widget is concerned. It does allow you to select one tag to be used to provide values for display within the Idea Studio. This is extremely helpful as it provides context for everything you see as you edit the widget.

Linked Tag Properties

The objects you selected when creating a tag widget were linked to tags. After grouping, the link changes from a specific tag to a type of parameter called a "linked tag property". This is a very powerful concept, which gets it's own chapter.

Linked Tag Properties

Widgets are linked to tags. Properties of those tags can therefore be accessed and displayed in the widget.

In the majority of cases, the only property that is used is the tag's value, but you can link any property of any tag to any display property of anything within any tag widget that you create.

For example, you could:

- Set the color of an object based on Area - Objects in the East area will be green and objects in the West area will be purple.
- Tie the opacity of an object to a child tag's Start Tag expression. If the tag doesn't start, the object doesn't show. (Setting a widget's opacity to zero is roughly equivalent to disabling that widget. The code for the widget just doesn't run.)
- Change the scale of an object based on an analog tag's scaled max property.
- Create a label to match the name, area, and description of the linked tag.

If you examine the properties of any child widget, you'll discover that the data source selector has a new symbol, as circled in the following image:

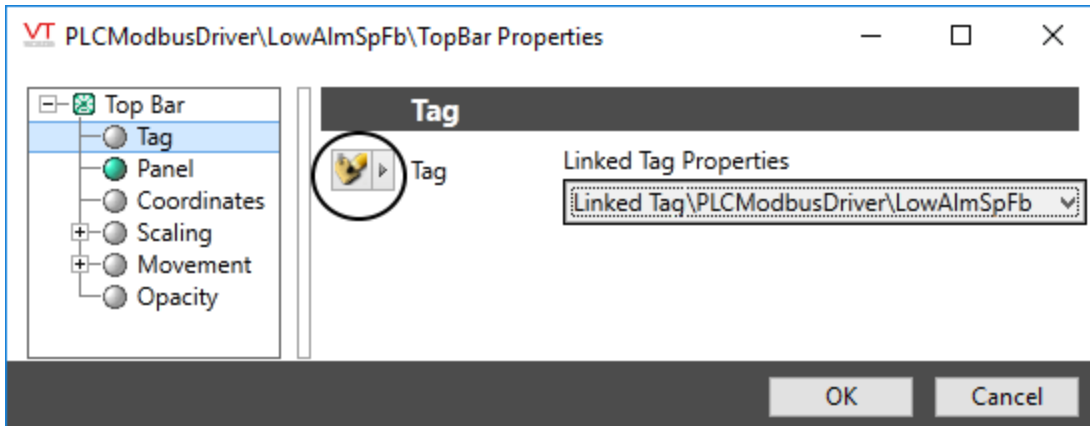


Figure 5-12 Note the symbol used for the data source. Tag names shown here are generic examples.

Suppose that you decide to add a new VTScada widget to your custom tag widget. Perhaps it's a Numeric Value so that an Analog Status is displayed as a number as well as a moving indicator. You can do this simply by dragging a Numeric Value to the widget, just as if you were drawing one on a page. *But, you absolutely must not link that new widget to any tag.*

Instead:

1. Open its properties dialog and click the arrow for the data source selection:

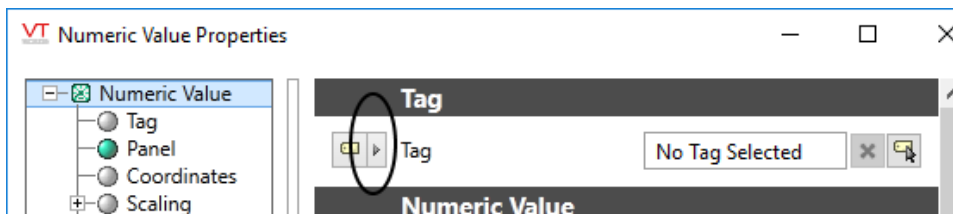


Figure 5-13 The data source selection arrow.

2. Choose the option, Linked Tag Property

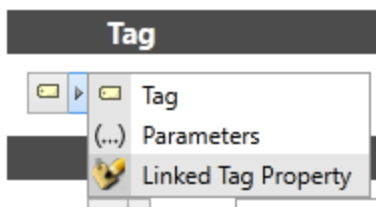


Figure 5-14 Selecting the Linked Tag Property data source

3. Expand the Linked Tag Property selection and navigate through the child tags to select the one you want. In this example the "Linked Tag" is a custom type and the Analog Status is a child of a child of that type.

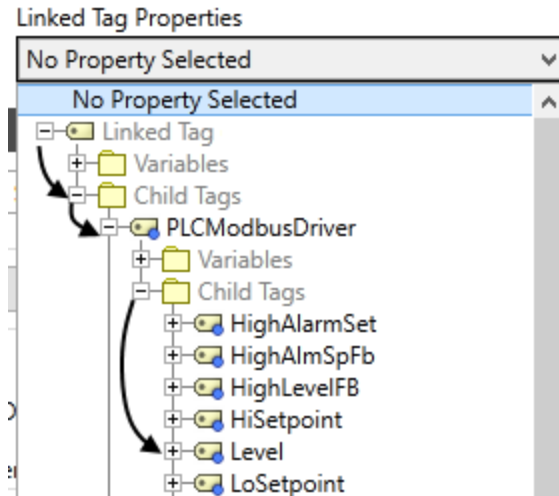


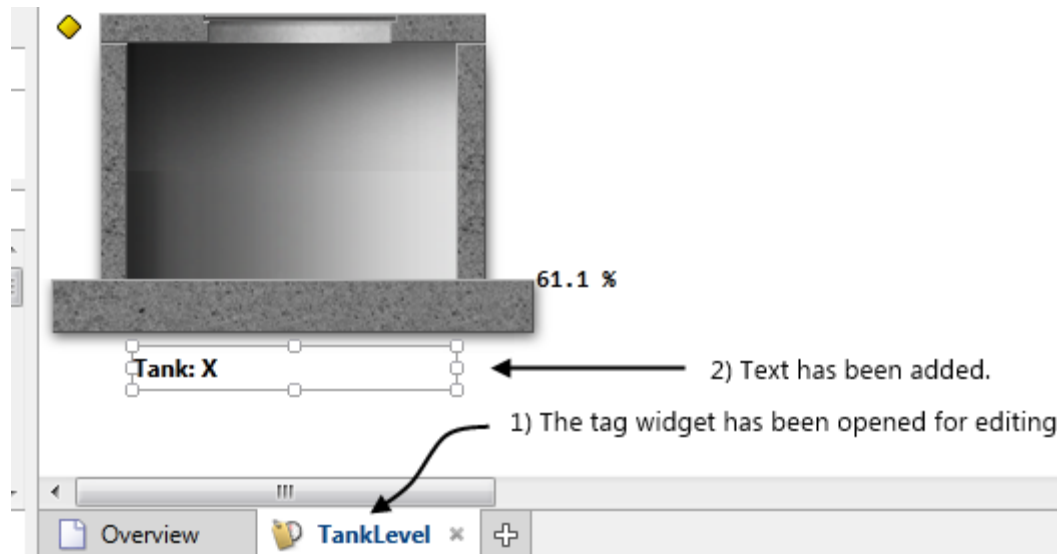
Figure 5-15 Selecting a child tag of the linked type.
The tag names shown are generic examples and do not match the exercise.

The result is that the widget can be used for the Level of any instance of the custom tag structure.

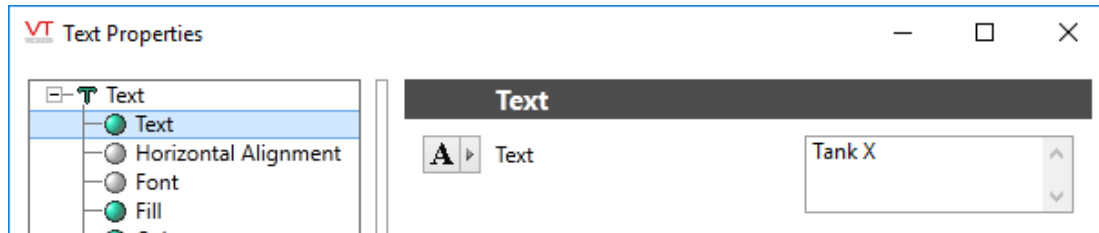
One common example is to display the tag name or description as a label within the widget. You might also choose to display the area value, create a calculation that uses the tag's minimum and maximum scaled values, or add the engineering units to the display.

Example: Configure a text label to display the description field of the linked tag.

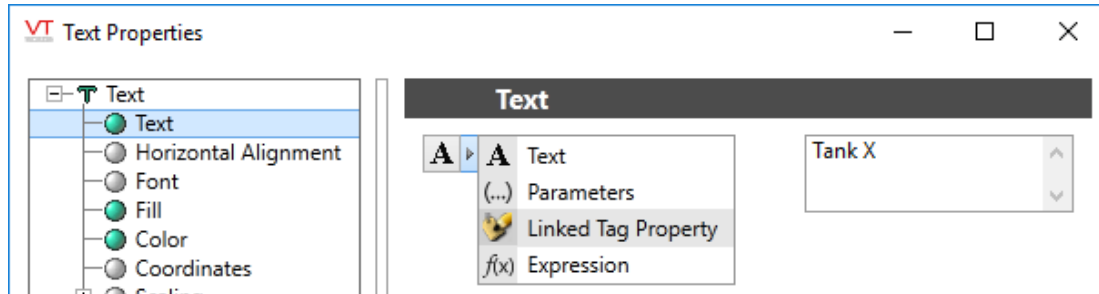
Prerequisites: This procedure applies only to tag widgets. The tag widget must be open for editing in the Idea Studio. Text has been added to the widget as a label.



1. Open the properties dialog of the text label.



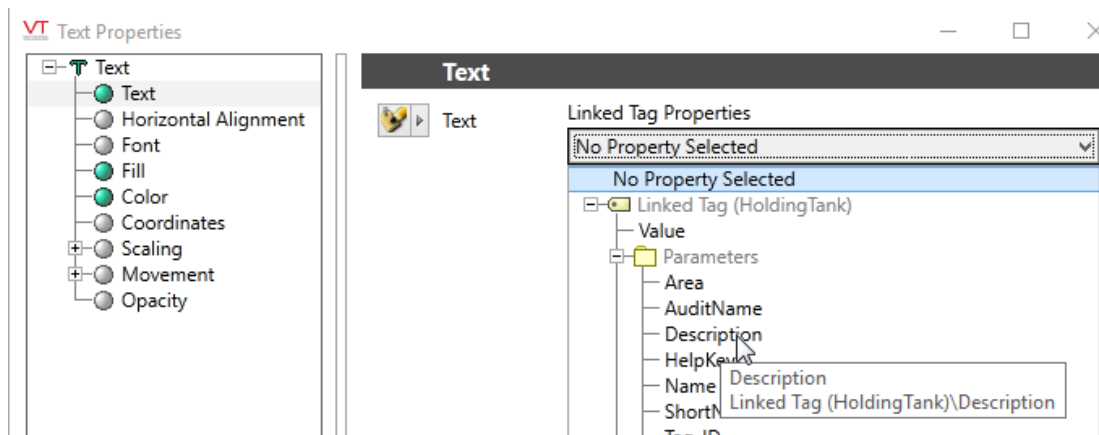
- Click the button as indicated. This provides access to a selection of data sources for the text to display.



- Choose the option, Linked Tag Property.

The text input field will be replaced with a drop-down list of tag properties. The list of possible properties comes from the list of types that the widget can be linked to.

- Expand the list to find and select Description.



Each instance of the widget will now show the description field of the tag to which it is linked. Note that this works, even though you did not use an expression with `\GetPhrase()`.

You might want the property to be only part of the label, or to create an expression that uses the property. After following the procedure described in the example, you can change the data source to Expression. You will find that the expression for the linked tag property is "Root\Property_Name". Use this in an expression as required.

Note: Take care: this information is for widgets linked to tags, not for parameter expressions in general. Here, "Root" refers to the current Drawn Tag Context. Use only when working with properties of a tag widget.

For example, to display the current tag's area value as part of the label "Tank in area: X", the process is:

1. Create a parameterized phrase. (Multilingual Expressions, on page 327)

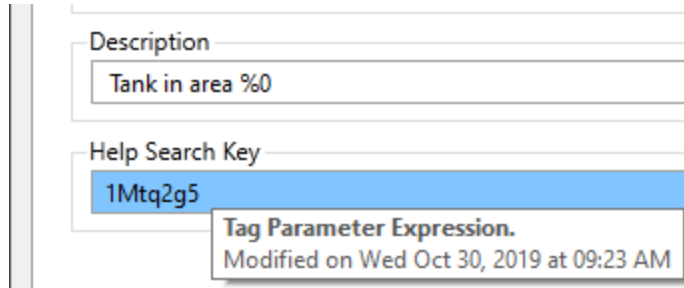


Figure 5-16 The expression is "Description", which returns the phrase key for the text shown in the Description parameter.

2. Use that phrase key in your expression and hand in the root tag's Area property for the parameter.

```
\GetParmPhrase("1Mtq2g5", \Root\Area)
```

Exercise 5-5 Edit the Pump Controls widget

The examples in the preceding text anticipate step 1 in this exercise, therefore detailed instructions are not provided.

1. Within the Idea Studio, add a Numeric Value widget to the wPumpControls widget, configuring it so that it will show the Flow rate from the linked pump. Careful! You should be editing wPumpControls, not adding the text below it on the Pump Controls page.

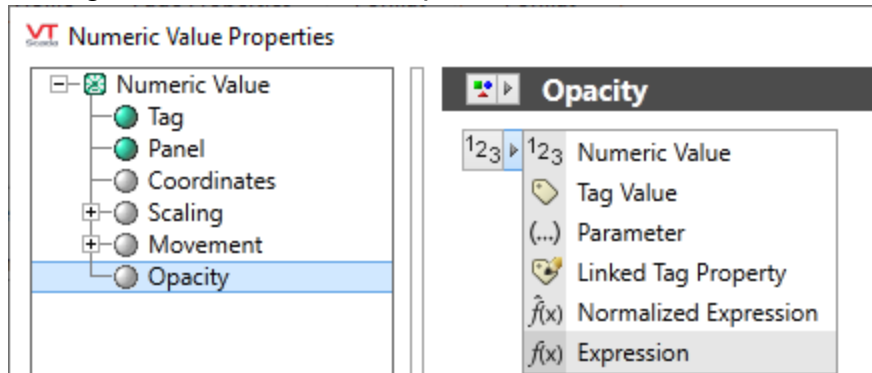
Not all pumps have a flow meter. Therefore, the controls page for a pump without this feature should not show a numeric value for a sensor that doesn't exist.

Just as you were able to set a Start Tag expression for a tag, you can tell a widget not to display if the linked tag hasn't started.

Tip: When viewed in the expression editor, "Linked Tag" is written as "\Root".

1. Open the properties dialog of the Numeric Value you added in the previous step.
2. Select the Opacity property.

3. Change the data source to Expression



4. In the expression editor, replace the 1 with the following:
`Pickvalid(\Root\Flow\Started, 0)`
5. Select OK to close the properties dialog.
6. Using the tabs at the bottom of the Idea Studio, close wPumpControls.
 (That's "close", not "delete" !)

Let's test your work...

7. Switch to / Open the page, Station Status, in the Idea Studio.
8. Drag the wPumpControls widget to the page twice.
9. Link one to Pump 1 and the other to Pump 2.
 You will fix that label in the next chapter.
10. Ensure that the Disable options are selected for both.
11. Switch to Operator View.
12. Operate the pumps.

Exercise 5-6 Create a label to match the linked tag

1. Open the Idea Studio to the wPumpControls widget.
2. Replace the text, "Pump 1" with a title that identifies the linked tag.
 The tools you will need are provided in the preceding text.
 You are free to choose which property or properties of the linked tag to use for your title.

Opacity Versus Linked Tag Properties

You were able to hide the Numeric Value widget based on whether the Flow had started because it is a child of a parent tag that is still there and running. Also, you were working with an I/O tag, which has a Started property.

Trying to do the same thing with the overall widget for your CustomPump (as an example) would be more complicated. First, user-defined types do not have a Started property or variable. Instead, you would need to refer to the Tag_Start_Condition property. But if the tag doesn't start, then that's neither 0 or 1 - the tag effectively doesn't exist at all. What you're likely to get instead is an unlinked widget, not an invisible one.

Exercise 5-7 Bonus exercise: Create a "step button"

Meaning, "a button that will increase (or decrease) a value by 1". VTScada does not include a step button widget, but you can easily create your own. In addition to the widget, you'll need two [Multi-Write tags](#) that you will create as children of the output tag for which you want to add step buttons. Multi-write tags do not count against your license.

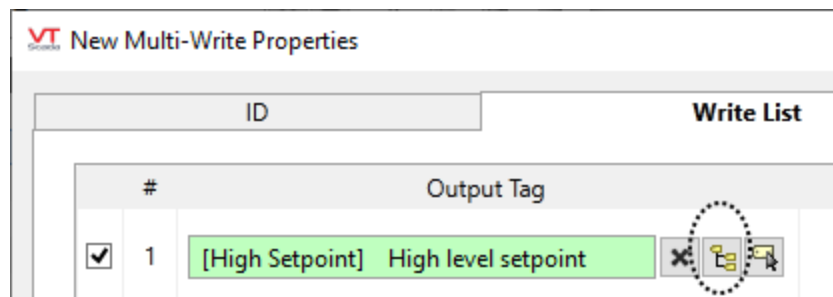
Preparation:

An I/O tag that writes an analog value, and for which you want to add step buttons.

This example uses the High Level Setpoint tag, found in the V12.1 Training Simulator.

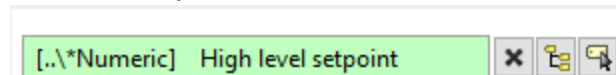
Part 1. Create two MultiWrite tags:

1. Add a MultiWrite tag as a child of your I/O tag.
In this example, it is named Step Up.
2. In the Write List tab of the MultiWrite, select the parent tag.



The purpose of the next two steps is to make this tag generic, so that it can be copied to any output tag where it will work without additional configuration.

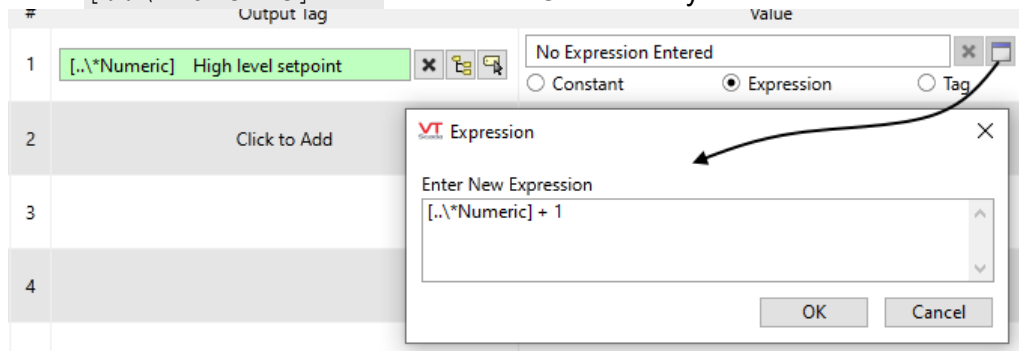
3. Select the path button (circled with a dotted line in previous image).
4. Select the option, Ancestor Relative Path.



The path should now read [..*Numeric], followed by the description of the tag you are using as the parent.

5. Click once in the Value column.
6. Select the option, Expression.
7. Open the expression editor.

8. Enter `[..*Numeric] + 1` then select OK to save your work.



9. Apply your changes and close the tag configuration dialog.
10. Copy Step Up
11. Paste it as a child of the High Setpoint, I/O tag, changing the name to Step Down
12. Edit the properties of Step Down to change the expression to subtract 1 instead of adding.
`[..*Numeric] - 1`

Step Up and Step Down can now be copied as children of any output tag for which want to add Step Buttons. The link to the new parent will be automatic thanks to the use of Ancestor Relative Path.

The next set of steps will create the basic output control plus two step buttons and save them as a widget for re-use.

Part 2: Create a Widget:

1. Draw the output tag as a Numeric Entry widget.
2. Adjust the appearance of that widget as you see fit but *do not* select a font for the label and *do not* provide a title.
3. Draw Step Up as a MultiWrite Button, placing it beside the Numeric Entry.
4. Change the label of the button to +
5. Draw Step Down as a MultiWrite Button, placing it below the button for Step Up.
6. Change the label of the button to -



7. Adjust the appearance as you see fit.

8. Group as a Tag Widget.
In this example, the widget is named Step Controls.

VT New Widget

Title
Step Controls

☐ New Widget
☒ New Tag Widget

Select Tag

Name	Description
Station 1	Storage and pumping
Station 1\PLC1	Simulator connection
Station 1\PLC1\High Level Set	Setpoint to switch pumps on
Station 1\PLC1\High Level Set\Step Down	Decrement by 1
Station 1\PLC1\High Level Set\Step Up	Increment by 1

☐ Include Trend, Navigation, and Tooltip Options
☐ Include Tag Icon Marker

OK Cancel

9. Ensure that the default tag selection is the original output tag.
If not, then ensure that both step controls are linked to MultiWrite buttons that are children of that tag and that no widget is configured to use a font or any other identifiable tag.
10. Select OK to finish.
11. Open the properties of the new widget.
12. Select the Disable Trend option.

Suggested Enhancement: Add text to the widget as a label. Use Linked Tag Properties so that the text shows the ShortName of the output tag.

6 Parameterized Pages

A parameterized page is one that can be reused for similar equipment installations. For example, your application may have multiple lift stations, each housing the same (or similar) equipment. The design of each page shows the equipment and differs only in which set of tags is being shown in a particular station.

Rather than create a separate page for each station, create one. But, link the widgets to parameters in the page rather than tags in the application. Navigational links to the page control which set of tags is used for the parameters when the page is opened via a particular link.

Tip: Pages can have a maximum of 100 parameters. If your page requires more than 10, you are encouraged to re-think your design. Perhaps Tag Widgets would be a better solution?

In the following example, a single control page was created, but it can be used for both pump 1 and pump 2 because the widgets are linked to parameters and a different set of tags is used for the parameters in each case. The page title uses a parameter within an expression so that operators will know which pump is being controlled.

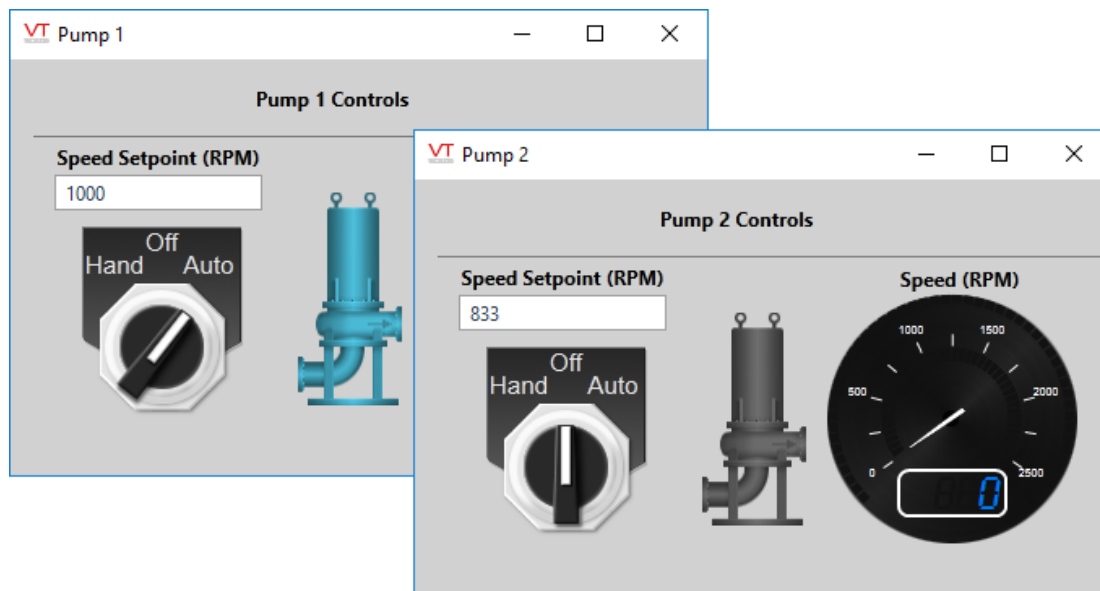


Figure 6-1 A single page, opened twice, showing tags from two separate pumps.

Parameters are typically linked to tags, but this is not a requirement. Parameters can be defined to hold numbers or text as well as tags. In all cases, the navigational link to the page must be configured to supply the value or tag to be used by each parameter.

Add parameters to a page using the Manage Parameters button in the ribbon, as shown. This button opens the Edit Parameters dialog, where you can view, edit or add parameters.

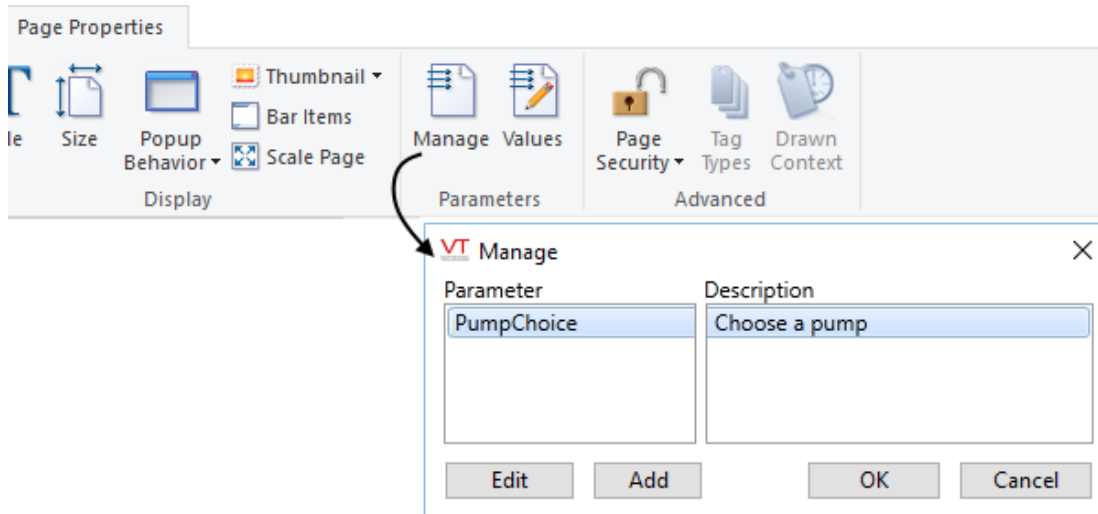


Figure 6-2 Manage parameters, button and dialog from the Page Properties ribbon

Clicking Add opens the following dialog, where you can configure the new parameter.

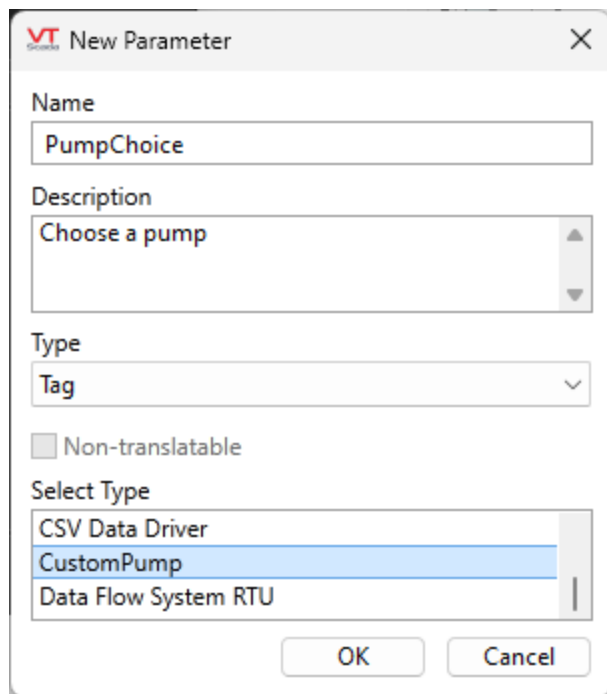


Figure 6-3 Adding a tag parameter to a page.
The type is a user-defined tag in this example

The example shows a new parameter being configured. Recommended practice is to ensure that the parameter name is unique, avoiding any conflict with tags or other named objects. The description field should guide developers or operators when selecting the tag or value for the parameter. If the parameter is of type tag, you would typically select which types can be used, thereby guiding later selection of tags to types that are appropriate. Use the Ctrl key to select more than type. Type selection is optional; if none are chosen then tags of all types will be available for use with the parameter.

The same dialog is used when editing existing parameters.

Parameterized pages and plain widgets use parameters in the same way and share configuration dialogs.

Only text parameters can be marked as non-translatable. (Other types are never translated.) Select this for text that should never be translated, such as I/O addresses.

Removing Parameters:

By design, there are no tools in the user interface to delete parameters. The easiest way to remove one is to use the [Version Control](#) system to find the change where the parameter was added to the page and reverse that. Otherwise, if you are certain that a parameter is not needed and that it cannot be edited to use for a new purpose, then experienced developers can remove parameters by editing the page's source code, then importing the changed file.

Exercise 6-1 Create a parameterized page

1. Open the Idea Studio.
 2. Open the page, Pump Controls
 3. Open the Manage (parameters) dialog and create a parameter as shown in Figure 6-2 and Figure 6-3
 4. Click OK to save your work.
- You won't use this parameter until the next exercise.

Link Parameters to Objects in a Page

Nearly every property of nearly every object that can be drawn on a page can use a page parameter as its data source.

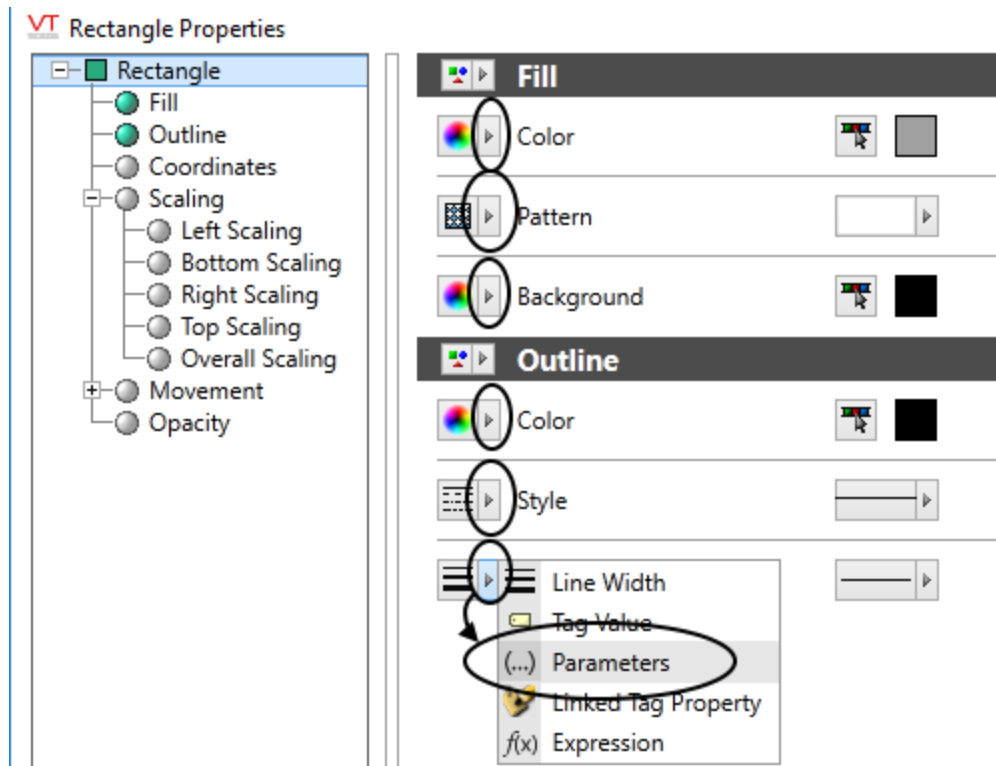


Figure 6-4 Parameters can also be used for scaling, movement and opacity.

Or, for example, text. Given a page that contains a line of text and also a text parameter...

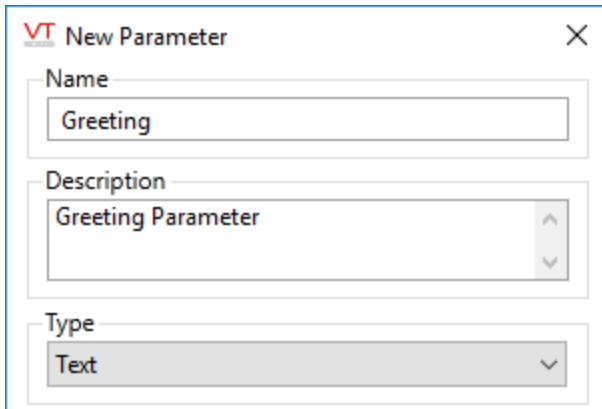


Figure 6-5 A parameter of type, Text.

You can open the Edit dialog for the text and choose an alternate source for the words that are to be displayed.

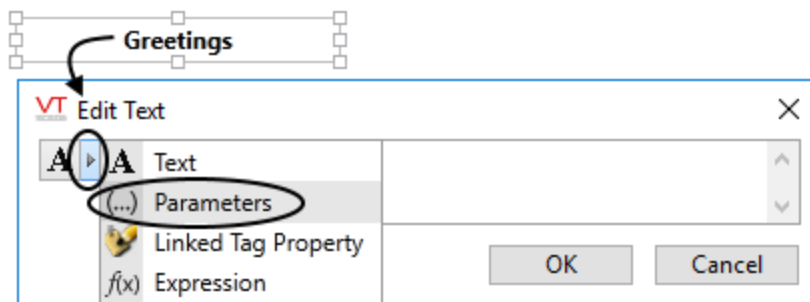


Figure 6-6 Expanding the data source, then choosing the option, Parameters

When the data source is set to Parameter, a drop-down selection of the parameters in this page will be available.

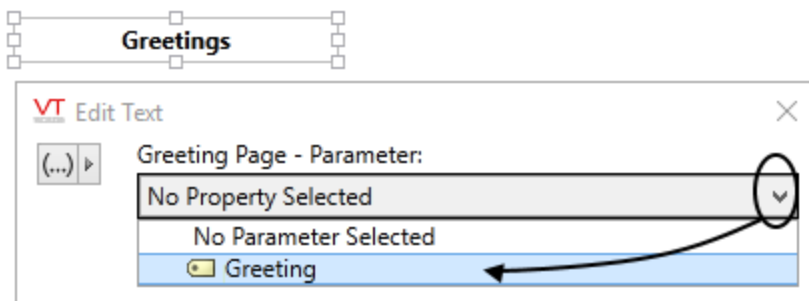


Figure 6-7 Selecting the specific parameter for the text.

The text will now display whatever is provided for that parameter when the page is opened.

Tag Parameters

If the parameter is a tag, then you have the option of selecting parameters, variables, or child tags with their parameters and variables.

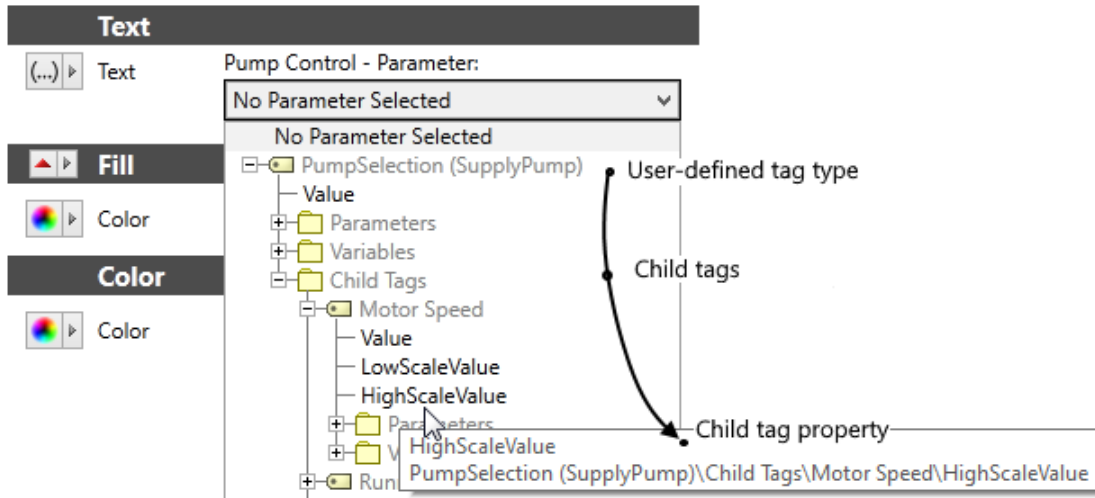


Figure 6-8 This parameter is a user-defined type with child tags.
Text in the page uses a property (*HighScaleValue*) from one of those child tags.

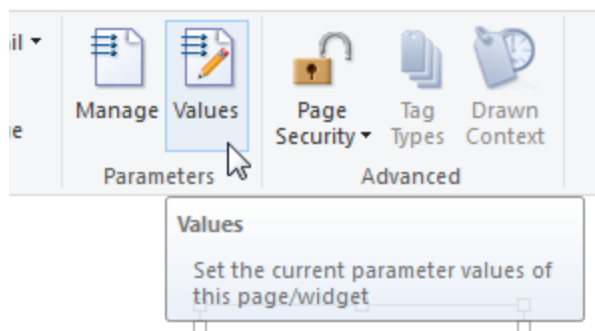
Parameter Values within the Idea Studio

If you happened to try the steps described in the previous example, your text would now look like the following in the Idea Studio:



Figure 6-9 Text with a blank parameter.

While working in the Idea Studio, the parameter has no value. You are advised to provide parameter values that are just for use within the Idea Studio so that you can see objects as operators will. To do so, click the Values button in the Page Properties ribbon:



The values you set in the resulting dialog do not become defaults. They are for use only within the Idea Studio. But there, they are very helpful in making the page look realistic and ensuring that text doesn't disappear.

Exercise 6-2 Link a parameter to a widget

1. Continuing to work in the page Pump Controls, ensure that an instance of the widget `wPumpControls` is displayed on this page.
2. Open the properties dialog for the widget and change the data source from Tag to Parameter. (Similar to Figure 6-6)

3. Using the drop-down that appears, select the parameter PumpChoice.
4. Ensure that the three Disable parameters are enabled (set to 1).

Note that the widget is showing indicators that it is unlinked. This is expected at this stage.

5. Use the Values tool in the Page Properties ribbon to select a pump for pPumpChoice.

Setting Values for Parameters

The value for each parameter in a page comes from the link to that page. This can be the Menu Item tag or it can be a page hotbox or page button widget.

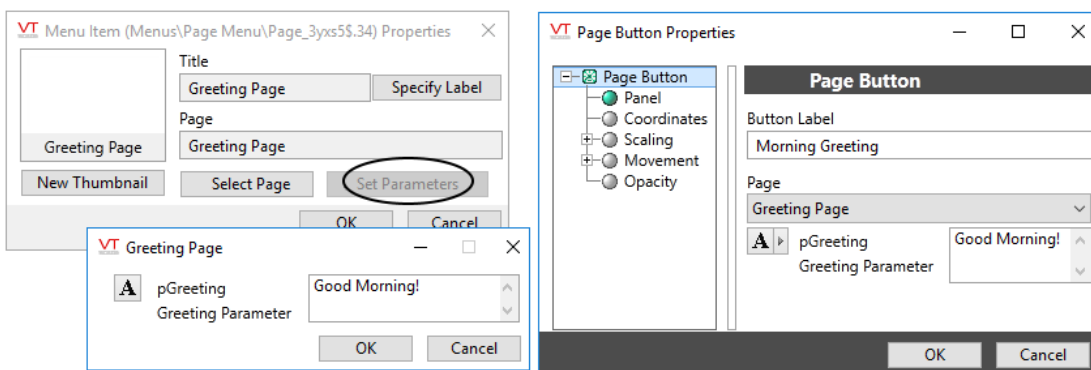


Figure 6-10 Setting a parameter value in a page Menu Item and in a page button widget

If you create a link to the page but don't provide values for the parameters then your operators will be prompted every time they open that page.

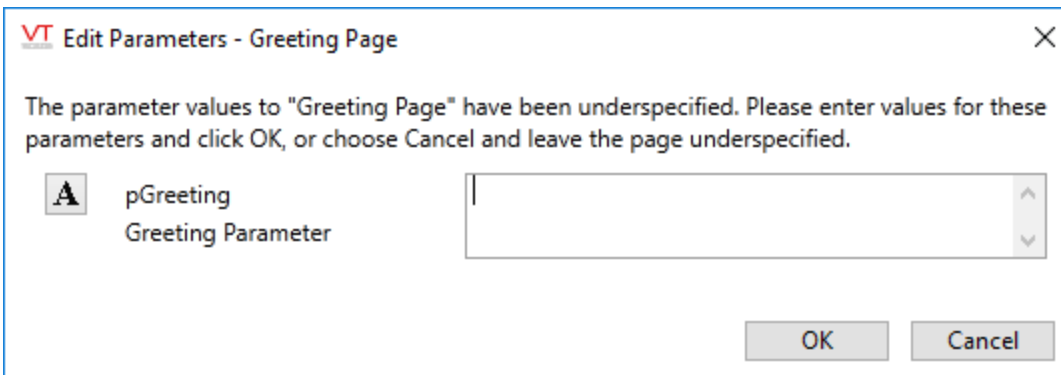


Figure 6-11 "Underspecified" is programmer-ish for "value not provided"

As a general rule you should provide values for all the parameters in a page in every navigational link that you give to operators.

Note: At the same time that you edit the navigation links to add parameter values, it is a good idea to change the label or title of the link so that operators will know what to expect when they open the page.

Note: Advanced VTScada developers can use `DecodeParms` to query the parameter values of existing MenuItem tags.

Exercise 6-3 Provide navigation links to the pump controls page

1. Working in the Idea Studio, open the Station 1 page.
You are about to move all pump control to pop-up pages rather than have the controls on the main page. This is often done for security and to reduce the chance of unintentional control actions.
2. Locate the Page Button below Pump 1.

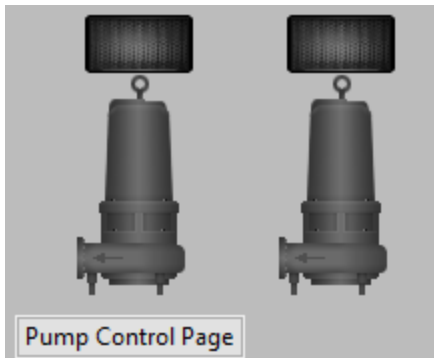


Figure 6-12 The approximate location for the Page Button.

3. Open the button's properties dialog.
4. Change the label to `Pump 1`
5. Select the page, Pump Controls.
6. When the parameter choices appear, select Pump 1 under Station 1 for PumpChoice.
7. If you have a pSubTitle parameter, provide useful text such as `Primary Pump Controls`
8. Copy and edit to add a button for Pump 2.
You may need to move the Pump 2 widgets to make room.
This will open the same page, but use Pump 2 under Station 1.
If you have a sub-title parameter, be creative.
9. Switch to operator mode, open the pages and operate the pumps.

Expressions in Page Titles

If you use parameterized pages, you should make sure that operators know which tags they are working with when they open that page. You could do this by using parameterized text at the top of the page. You can also do this by adding an expression to the page title that will use a parameter or a value from the linked tag.

The page title can be edited using a tool in the page properties ribbon:

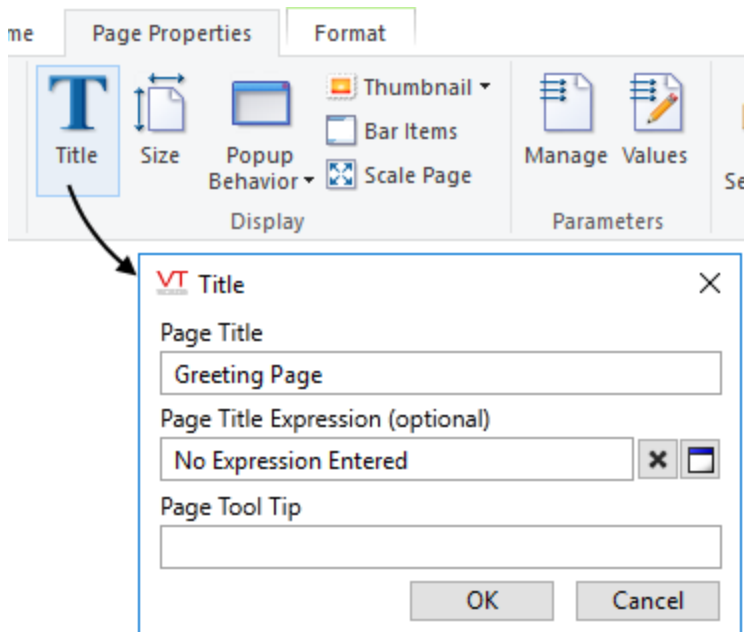


Figure 6-13 Editing the page title

If your page has a text parameter and you want to use that in the title expression, just type the name of that parameter. For example:

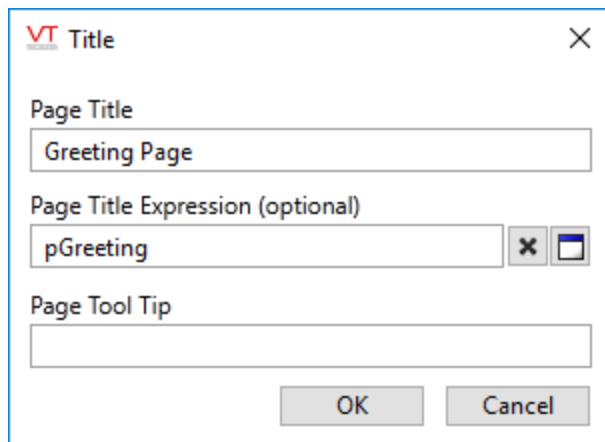


Figure 6-14 Using a text parameter for the page title.

If your page has a tag parameter and you want to use the name or description or some other property of that tag, then do so by adding a backslash after the parameter and the name of the tag property that you want to use. If the tag in question is a parent, you can use any property of any child tag within the structure.

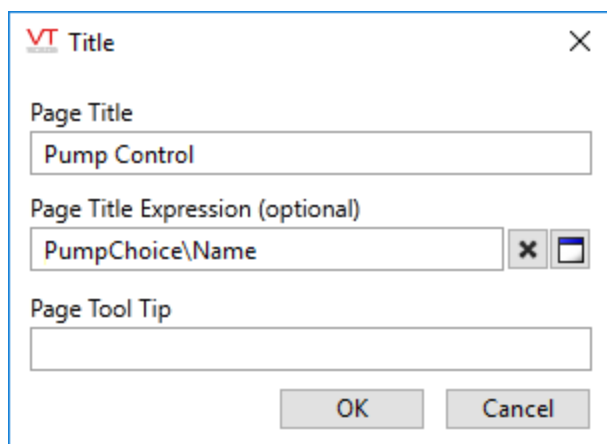


Figure 6-15 Using the short name of whatever tag is specified for the parameter *pPumpChoice*

Exercise 6-4 Create an expression for the Pump Control page

1. Using the preceding figure as a guide, create an expression for the Pump Control page's title that will show the description property of the chosen pump.

Widgets versus Parameterized Pages

Widgets and pages share a number of features:

- Both are created using the same tools of the Idea Studio.
- Both can display tag values and other contextual information.

They also have important differences:

- You can navigate to pages, but not to widgets.
Widgets can only be seen when drawn on a page.
- Widgets can be linked to tags and can make use of linked tag properties.
You can't link a tag to a page the same way that you can link a tag to a widget. You could write expressions in your pages to achieve a similar result, but this is more difficult.

When creating custom widgets, parameterized pages, and also custom tag types, your goal should always be to reduce repetitive work. If you have only two stations, then just create the tags for the two stations. Create a page for each station, copying objects from the first to the second where convenient.

But if you will have 20 or more stations, and each station can have two or ten pumping systems or other equipment, then create a custom tag for all the I/O in the equipment and create a tag for a station.

Move as much configuration work as possible to the top. For tags, this means creating parameters in your Context tag for everything that needs to be configured in the child tags. Then use expressions in the child tags to pull information from the parent context, performing math or string concatenation where useful.

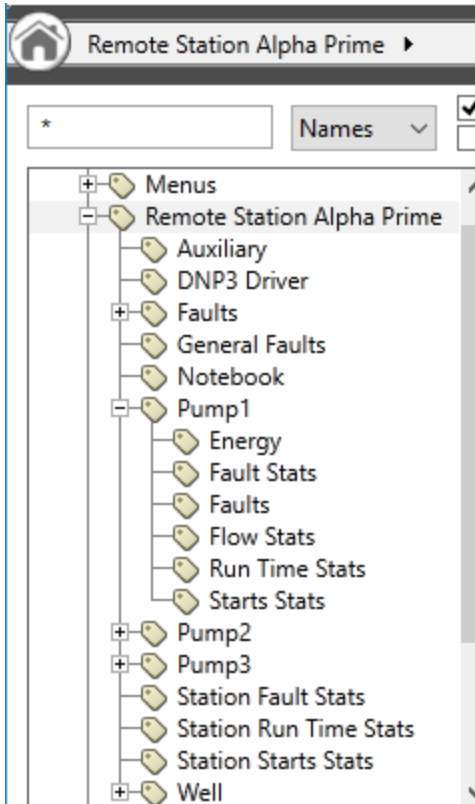


Figure 6-16 All the configuration for all these child tags is done within the parent, tag.

For widgets, it means creating a custom widget that can be linked to one of your user-defined tags and that represents some or all of the I/O within that tag.

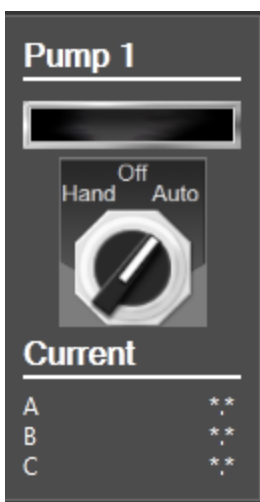


Figure 6-17 The widget shows five I/O, but links to only one tag.

For pages, it means creating a parameterized page that displays one or more of your custom widgets, passing the appropriate tag for the widget(s) through parameters when the page is opened.

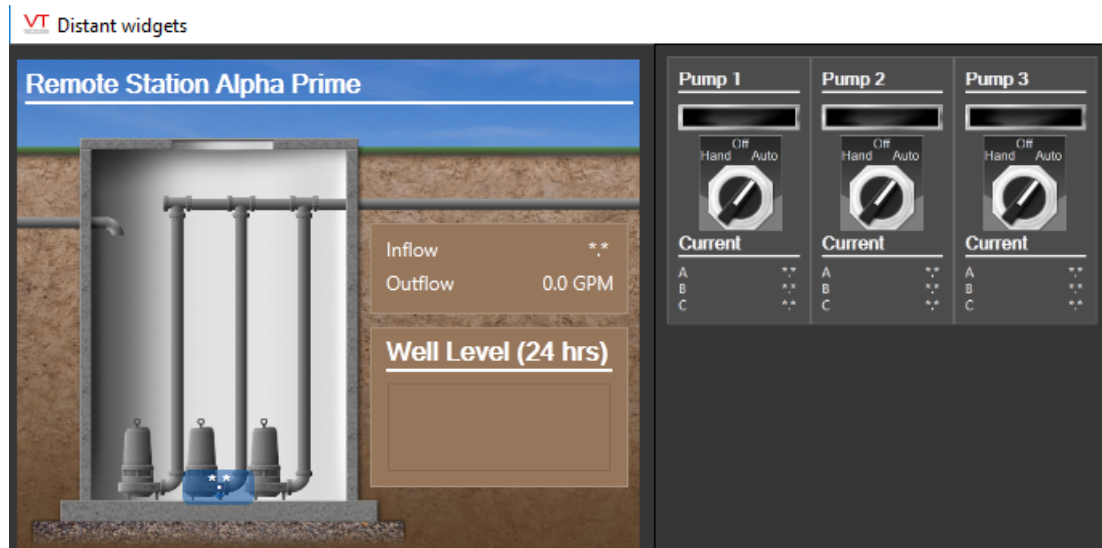


Figure 6-18 This page has one parameter for the station, not 18 for the I/O shown.

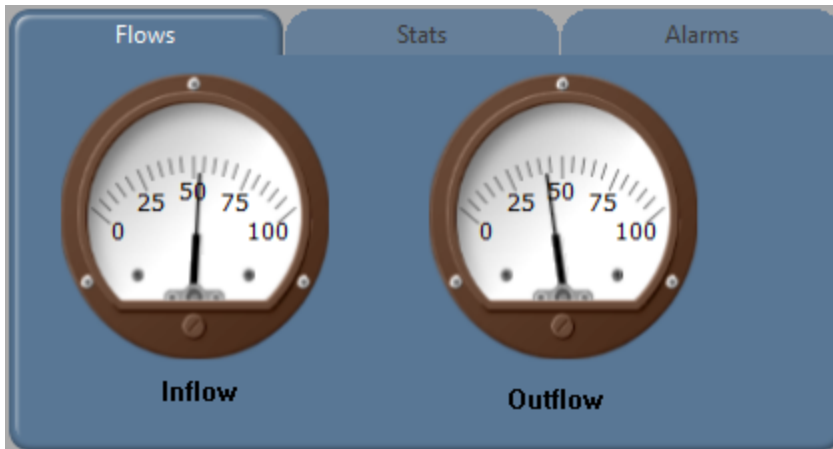
Exercise 6-5 Create a Station Widget

1. Open the page Station Status in the Idea Studio.
2. If anything that is NOT related to Station 1 is visible on this page, delete it now.
3. Select all the text and all widgets that relate to Station 1.
4. Group them into a new Tag Widget, ensuring that the selected tag is Station 1, not Station 1\PLC1.

Folders

The Folders widget creates a user-interface object with multiple tabs where each tab has a separate display of system monitoring and control objects. The tabs are used to select which frame to display.

Three styles of folder are provided in the palette, but these are all variations on the same widget.



(See Folders, later in this topic.)

A folder may have from one to fifteen tabs. You have extensive control over the appearance of the folder and the tabs within it.

The content for each tab's frame must come from an existing object in your application. These may be pages, tag widgets, or plain widgets. You do not draw objects on a folder as you would on a page.

There are two parts to a folder's configuration: The first defines the overall style of the folder. The second controls the content and the appearance of the individual tabs.

Note: If your pages include tabbed folders, then the Unique Key property of the folder must be set in order for tabs to function when viewed using a Mobile Internet Client.

Note: As a general rule, define the tab setup first, then adjust the style to your preferences.

Exercise 6-6 Tabbed pump monitoring

1. Open the Idea Studio and create a new standard page.
Name it `Pump Monitoring`
Add it to the page menu.
2. Draw a tabbed folder on your new page.
Decorations >> Folders >> Frame Tabs
3. Open the properties dialog of the Folder widget you just drew.
4. Open the Tab Setup tab.
Always start with Tab Setup when configuring a folder. The style can wait until the function has been defined.
5. Set the number of tabs to `2`.
6. For tab 1, change the label to `Station #1, Pump 1`
7. In the Contents section, select Page.
8. Select the page, Pump Controls.
9. Open the parameters dialog and choose pump 1 of station 1 for PumpChoice.
10. Repeat for tab 2, using the Pump Controls page again, but using the second pump.
Don't forget to set an appropriate title for each tab.

Switch to operator view and experiment with the tabs.

Exercise 6-7 Optional folder exercise: A folder inside a tag widget, inside a parameterized page

Do this only if you have extra time and think that it may be useful to your own applications. (Structures similar to the one built in this exercise are in use in several customer locations.) Only the general outline of the steps will be provided as part of the exercise is to work out the details. This exercise includes a relatively advanced use of expressions.

Part 1: Create the tag widget with folder:

1. Start by using the File >> New command in the Idea Studio to create a new, empty tag widget.
2. When prompted to select the associated tag types, choose StationType.
3. Drag one of the folders to the screen, placing it near the top left.
4. In the Folder properties, tab setup, change the number of tabs to 2.
5. For tab 1 contents, choose Tag Widget.
6. Change the data source to Linked Tag Property.
7. In the Linked Tag Properties drop-down, navigate to find and select Linked Tag (Station Type) >> Child Tags >> PLC1 >> Child Tags >> Pump 1.
8. Select the widget, wPumpControls.
9. Change the label to Pump 1 Controls.
10. For tab 2, do the same but select Pump 2.

Part 2: Create the parameterized page

1. Create a new pop-up page.
2. Create a tag parameter in that page, selecting the StationType as the tag.
3. Drag the widget you created in part 1 to the page.
4. Link that widget to the parameter.
5. Ensure that all of the Disable parameters are selected.
6. Open the Overview page (or create a new one).
7. Draw two Page Button widgets. Both will open your pop-up page, with one button passing Station 1 as the parameter and the other passing Station 2.
8. Test your buttons.

There is only one problem with this arrangement: there is no way to disable the tab for Pump 2 in Station 2, using tools available in the Idea Studio. A bit of scripting, not much more advanced than you've used with Calc tags, will fix this.

In the source code for your widget will be a GUITransform that draws the folder. It will look like:

```
GUITransform(2, 402, 502, 2,
    1, 1, 1, 1, 1 { Scaling },
    0, 0 { Movement },
    1, 0 { Visibility, Reserved },
    0, 0, 0 { Selectability },
    Scope(Code, "Library", TRUE)\FolderDM(2, 2, Invalid, 15, 232, \ParameterSet(... etc...
```

The very first parameter of the Folder function (starting with \FolderDM...) sets the number of tabs: \FolderDM(2. Therefore, you could replace the 2 with an expression that checks whether Pump 2 started and is therefore Valid. If it did, set the number of tabs to 2 and if it didn't set them to 1.

7 Advanced Security Customization

The following components of the VTScada security system are described in this chapter.

Account	The user name, password, privilege set, and other properties assigned to each user.
Role	A set of privileges granted to a named job description. Roles may be granted to other roles, thereby making it easy to build up a privilege set.
Rule	A privilege, or role, assigned to an account, and optionally restricted by a tag (usually the parent of a set of tags) or by a workstation name. If restricted by a tag, the privilege(s) are in effect only for that tag and its descendents, and denied for all other tags. If restricted by a workstation, the privilege(s) are in effect only when the user is signed in at the named workstation. Note: The majority of VTScada general privileges should not be restricted by a tag-scope rule. Doing so will effectively deny that privilege.
General privilege	A VTScada-based privilege, which applies to features built into VTScada.
Custom privilege	A user-created privilege, which may be applied to control access to pages and to output tags.
Realm-Area Filtering	This takes the three components... <ul style="list-style-type: none"> • security group (a name assigned to a group of accounts, not to be confused with a role) • the area property of tags • a realm, which in this case is the name given to a set of area values. ... and allows you to create a rule that denies certain access to tags of a given area, for users in a certain group. Tag access is denied for the Tag Browser, the Alarm page and alarm lists, the Historical Data Viewer and for Reports. Users may still open a page and access the tags there, unless you also make use of custom privileges, configured to complement the realm-area filtering rules.

Realm-area filtering is used for applications that span separate regions, zones, departments or other designations, and where managers want to prevent operators in one zone from having any access to data, alarms, etc in another zone.

Also, you can use expressions to check who is currently signed in at a workstation and whether their account possesses any given privilege. You can use this to disable features within a page, based on who is signed in.

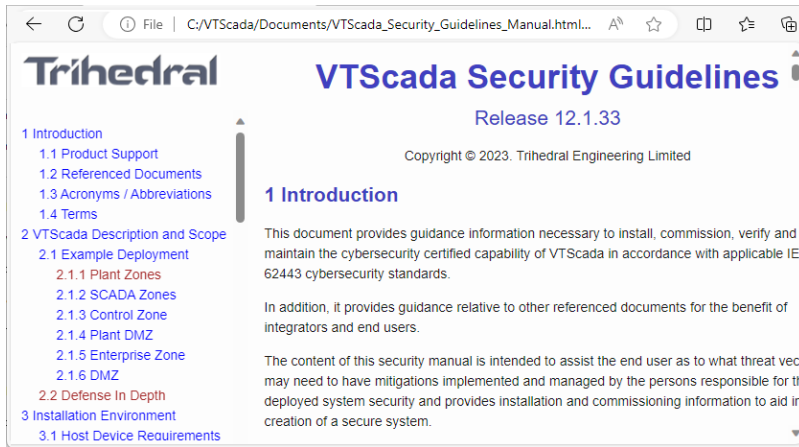
Best Practices for Security

SCADA Security Guidelines Manual

A best-practices guide is included with every copy of VTScada. This is a stand-alone document (not part of these help files), which can be found in the Documents sub-folder of your installation folder.

(C:\VTScada\Documents\VTScada_Security_Guidelines_Manual.html)

The manual provides guidance for installing, commissioning, verifying and maintaining the cybersecurity-certified capability of VTScada in accordance with applicable IEC 62443 cybersecurity standards.



Do not assume that your site is too small to be worth hacking.

All SCADA systems are targeted by hackers. You are running a SCADA system. Your site is a target.

Design your pages and tag structures with security in mind.

You can save a large amount of work, and greatly reduce the number of custom privileges you will need by creating a well-organized set of roles and corresponding rule scopes.

Enforce strong passwords.

Use the available options to enforce strong passwords (minimum length, combination of letters, numbers and other characters). Advise operators against re-using passwords for multiple applications.

Whenever there is ever a security update for your version of VTScada, apply it as soon as possible.

Typically, vulnerabilities are published a short time after a security update is distributed. Hackers will immediately seek to exploit that vulnerability on sites that have failed to protect themselves by applying the update.

Protect your control objects with custom privileges so only designated operators may use them.

This will limit the number of operators who will have access to controls and pages.

Place operator controls on pop-up pages.

By placing controls on a pop-up page, you reduce the chances of an operator accidentally issuing a control action. Also, because access to a pop-up page can be restricted using a custom privilege, it is possible to restrict access to many output tags with one privilege on one page rather than many privileges on many tags.

Use care when granting the Thin Client Access privilege and configuring a VTScada Thin Client Server.

VTScada cannot secure the networks between the remote client and the server.

Thin clients transmit the user credentials (username and password) using Basic Authentication, which is a simple non-encrypted Base 64 encoding of "username:password", and which is easily decoded by capturing network traffic.

It is essential that you use an X.509 certificate (commonly referred to as an SSL certificate*) to secure the communications from packet sniffing software connected to a local machine or switch. Do not overlook the possibility that attacks might originate from within your trusted network.

The use of a VPN is a reasonable second choice.

*Transport Layer Security (TLS) replaced Secure Socket Layer (SSL) security years ago.

If allowing Thin Client access, test the TLS connection using 3rd party tools, looking for weak ciphers, etc.

Smaller sites that do not have a dedicated IT department may need to find a contractor to help with this.

Do not grant privileges to the Logged Off account. (Exceptions may apply in rare circumstances.)

Do not grant unnecessary privileges to any account or role.

Configure the VAM to be hidden while the application runs to prevent access to the various diagnostic utilities by unauthorized persons.

This action is recommended for all sites. Use the "Other" tab of the Edit Properties page of the Application Configuration dialog to hide the VAM from all who do not have the required privilege, while the application runs.

Secure the Source Debugger and other diagnostic applications

This step is urged for sites that may prefer not to hide the VAM.

Before running the Import File Changes tool in the VAM, review the list of changes that will be imported using the Import/Export files tool of the Application Configuration dialog.

Note that the Import/Export tool will not add new files to your application. For that, you must use the File Manifest.

Consider running VTScada as a Windows™ service and using only thin clients to access applications.

This option provides maximum control over the choice of account under which VTScada runs, its permissions, and user-access to applications.

Use Windows security techniques to prevent unauthorized persons from accessing the VTScada program directory. Keep the workstations that are running VTScada in a secure location.

All other security measures are in vain if someone can destroy your application by deleting files or taking a hammer to the server.

Create Accounts and Roles

Note: If your application uses Windows Security Integration, refer to implementation notes in the topic: [Windows Security Integration](#)

Use the Accounts dialog to create and manage all accounts and roles. Your account must have the Manager privilege. The following instructions are for creating an account, but apply identically to creating a role, excepting that roles do not have passwords, alternate identification or automatic sign off periods.

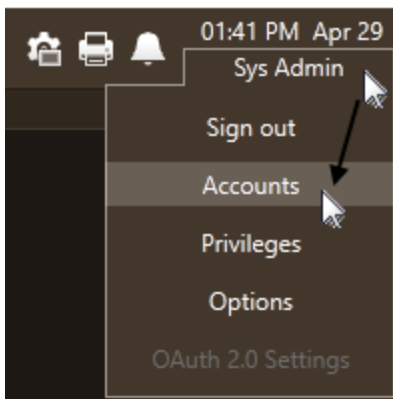


Figure 7-1 Reopen the Accounts Dialog (one of several ways)

Recommended practice is to assign privileges to roles, then assign roles to accounts.

Figure 7-2 The Accounts dialog. To open for the first time, click Sign In and choose to activate security.

There are several ways to create an account or role.

Note: If a manager is a member of a security realm, any new user accounts and roles created by that manager will automatically belong to the same realm. In this case, you must enter new user names as simply "UserName" rather than "GroupDelimiterUserName". Managers who are members of a realm cannot see accounts or roles that are not also members of the same realm.

Use a manager account that is not a member of a realm when creating roles, unless you need roles that exist only for a specific realm.

Note: If an asterisk is visible beside the name of an account or role, the changes have not been saved. Use care when closing the dialog.

Naming Rules for Accounts and Passwords

- Account names cannot begin or end with spaces.
- Do not use the following characters in VTScada account names:

" \ / [] : | < > + = ; , ? * @

- The following characters are not allowed in Windows account names: (spaces are accepted but not advised)

" / \ [] : ; | = , + * ? < >

Caution: Use care with special characters in account names and passwords. While valuable from a security standpoint, some characters may cause problems with certain Alarm Notification devices.

* As an example, it has been reported that many symbols other than letters and numbers in a password will not work with the Twilio® interface. To be safe, use only alpha-numeric symbols in account passwords.

Create a new account or role:

1. Right-click in the list of accounts and select "Add User" or click the Plus button above the list of accounts.
The same tools are found in the Roles section, immediately below the Accounts section.

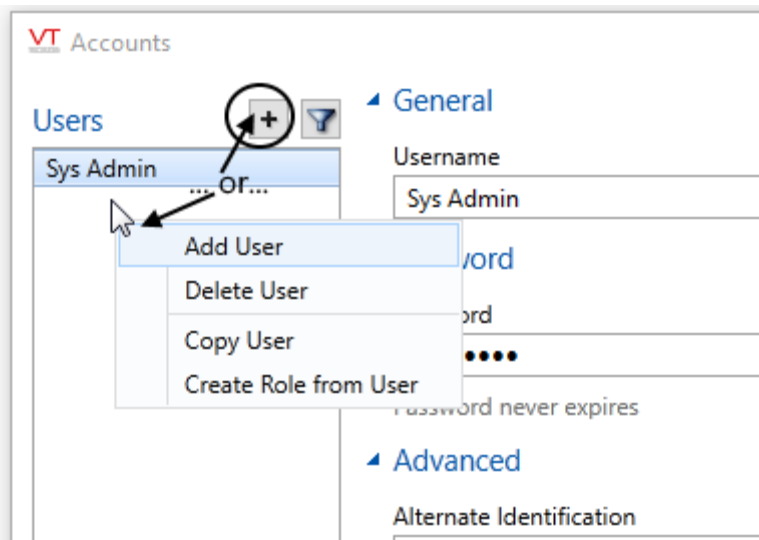


Figure 7-3 Either technique works to create a new account or role.

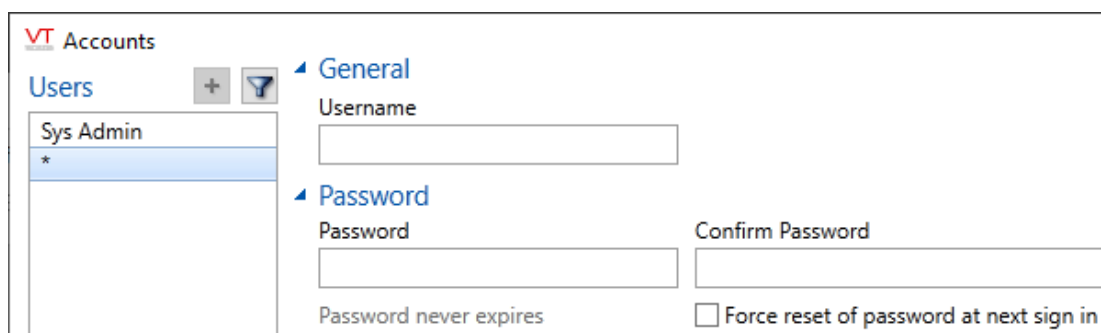


Figure 7-4 An asterisk appears, marking the place for the new account or role.

The asterisk is used to mark any account that has unsaved changes. You can switch between accounts to compare settings without losing changes made so far, and without losing track of which account has unsaved changes.

2. Enter a name for the new user.
Similarly, enter a name if creating a new role.

3. Enter and confirm a password.
Does not apply to roles.

You might wish to make this password a temporary, generic password and have the user change it when they first sign in. Users do not need the Account Modify privilege to change an expiring password, but they will need that privilege to change their password at other times.

Copy an existing account or role:

Creates a clone of the account or role, differing only in name and (in the case of accounts) password. Useful when creating a series of similar accounts or roles.

1. Click once to select the account to copy.
2. Right-click to open the menu, then click "Copy User" or click the Copy button.

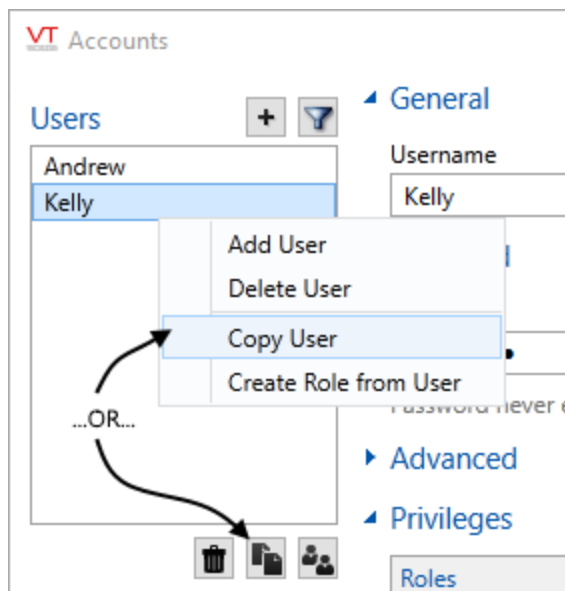


Figure 7-5 Again, the same applies if copying a role.

An asterisk will appear in the user list, marking a place for the new account or role.

3. Enter a name for the new account or role.
4. Enter and confirm a password.
Applies only to accounts.

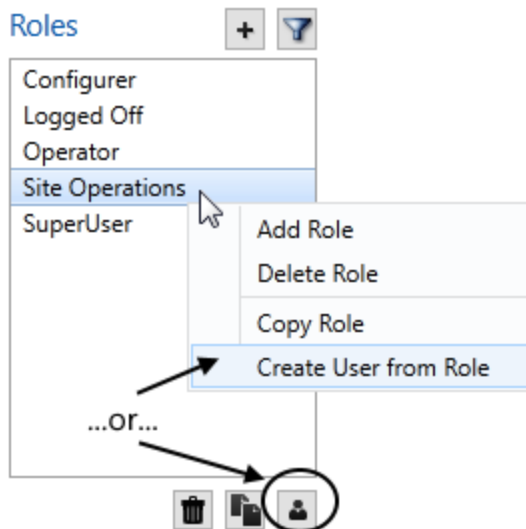
The new account or role will have all the privileges and other settings of the old account or role.

For accounts only, if the original used Alternate Identification, you will need to create a unique password for the new account.

Copy a role to a new user

Creates a new account having all the privileges that belonged to the role. This is not recommended as it is far more efficient to create a new account using one of the preceding methods, then assign the role.

1. Click once to select the role to copy.
2. Right-click on the role to copy and from the menu, select "Create User From Role" or, select the role then click the Copy button at the bottom of the list.



An asterisk will appear in the account list, marking the place where the new account is being added.

3. Enter an Account Name for the new user.
4. Enter and confirm a password.

Copy an account to a role

Creates a new role, having all the privileges that belonged to the account. This is useful if you originally assigned privileges to the account and now wish to manage rights using roles instead.

1. Click once to select the account to copy.
2. Right-click on the account to copy and from the menu, select "Create Role From User" or, select the account then click the Copy button at the bottom of the list.

An asterisk will appear in the user list, marking the place where the new role is being added.

3. Enter a name for the new role and apply changes.

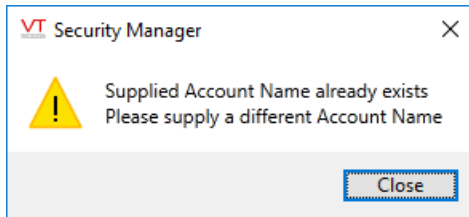
Delete an account or role

Select the account or the role then click the trash can (delete) button below the list. Note that nothing is deleted until you press Apply. Deleted roles will remain visible in other parts of the dialog until the change is applied.

If you delete the last account that possesses the Manager privilege, your application will return to the unsecured state. You will be warned.

Duplicate User Names

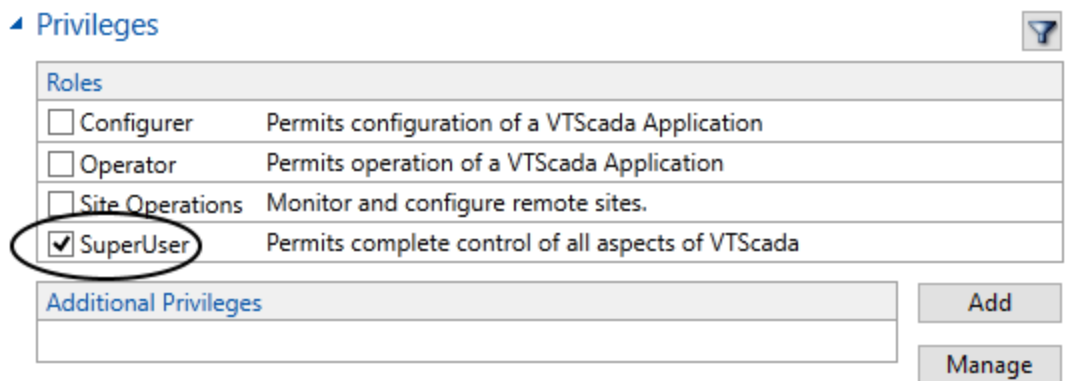
VTScada will not allow two accounts with the same name. The following warning dialog is displayed and the second account not saved.

**Troubleshooting:**

- Unable to open the Accounts dialog.
Your account does not include the privileges required to use this feature.
- The New and Copy features will not work
Your account does not include the privileges required to use these features.
- A "Discard Changes" dialog appears whenever I try to close the dialog.
Click Apply before trying to close the dialog
- Wrong name given to the account or role.
Click on the Name field, type a new name, then click, "Apply".

Exercise 7-1 Create accounts

1. Activate security for the application.
2. Name the first account after yourself, and create a password that you are likely to remember.
3. In the list of Roles, select SuperUser. (Figure 7-6)

*Figure 7-6 Choose a Role*

1. Deselect all the additional privileges.
These come with the SuperUser role.
2. Check that you remember your password, then click Apply.
3. Sign in.
4. Reopen the Accounts dialog.
5. Create a new role (careful ! new role, not new account) named "Course Priv-
ileges".

6. Assign the SuperUser role to that account and click Apply.
7. Open the role, Course Privileges.
8. Click the Add button to the right of Additional Privileges.
9. Grant all the privileges listed.
10. Remove the SuperUser role from Course Privileges.
11. Click Apply.
12. Open your account by clicking on it in the list of users.
13. Add the Course Privileges role to your account.
14. Expand the Advanced section if it is not already visible.
15. Change the automatic sign off period for your account to 60 minutes.
16. Create two more accounts.
Name the first, `NorthOperator` and the second `WestOperator`.
For the purpose of this exercise, give both accounts a simple password, such as "a". Don't do that in a real application.
17. Grant the Operator role to both accounts.
18. Sign out.
19. Sign in using one of the operator accounts and note what you can or cannot do in VTSkada with this account. Try the features in various built-in pages.

Protect Pages and Output Tags

With one exception, Control Outputs(*), the general privileges protect only what is built into VTSkada, not anything that you create. You must create custom privileges to protect what you build.

(*) The general privilege, Control Outputs, is applied to every tag by default, but is not shown in any tag's Properties dialog. If a tag is not otherwise protected by a custom privilege of your creation, then it is protected by Control Outputs instead (but not both).

These privileges are referred to as "custom privileges" because they are always custom-built for an application. (In earlier versions, they were known as "application privileges")

A custom privilege applied to an output tag will prevent an unauthorized user from operating the associated controls, but still allow them to see the tag. A custom privilege applied to a page will prevent an unauthorized user from opening the page, thereby hiding its contents from view.

Tip: As a suggestion, ensure that the start page for your application is one that may be viewed freely when no-one is signed in.

Users with the Administrator privilege can create custom privileges within the Privileges dialog. You can also use this dialog to suppress general privileges so that they cannot be seen in the Accounts dialog, thus preventing anyone (including yourself) from granting that privilege to an account or role while it is suppressed.

You can create hundreds of privileges if necessary. But, before creating large numbers of privileges, you should consider whether rules that limit a privilege by scope or workstation might help you keep the number to a manageable level.

For programmers: Every general privileges has a negative index number starting at -1. User privileges have an positive index number, starting at 16. The index number can be seen in the Privileges dialog, available from the security menu.

Only users who possess the Administrator privilege can add new privileges.

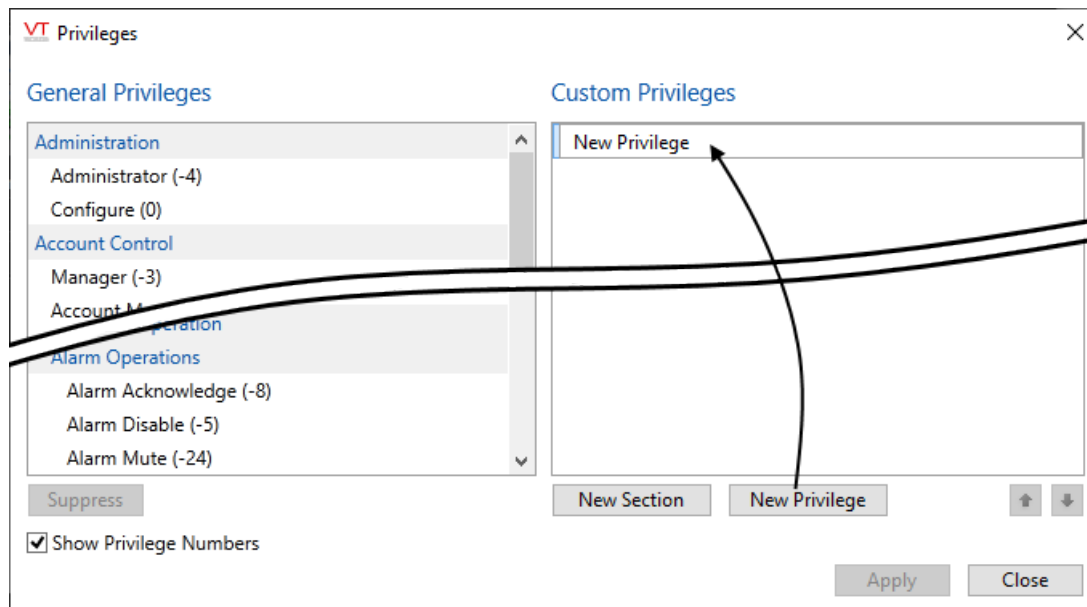
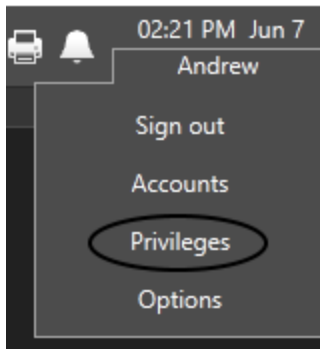


Figure 7-7 Section of the Privileges dialog with numbers shown.

For instructions to suppress general privileges, see *Suppress or Reveal Privileges*

Exercise 7-2 Add new privileges

If the currently signed-in user account is not yours, sign out, then sign in as yourself. Both the Idea Studio and the Tag Browser should be closed.

1. Select your username to open the menu.
2. Select Privileges.
The Privileges dialog opens.
3. Select New Privilege.
A new privilege with a default name will be added to the list.
4. Change the name of that privilege to `Pump Operation`
Press <enter> after typing.

5. Add a second privilege named `Pump Page`
6. Select Apply.
7. Select Close.

Now your application has custom privileges. But they don't protect anything until applied to tags or pages. You'll do that in the next set of steps:

1. Open the Tag Browser
2. Navigate to display all the tags below Station 1, Pump 1.
3. Open the properties of the pump control Selector Switch tag, Pump Control.
4. Select the Merit tab.
5. In the Privilege drop-down, select Pump Operation.
6. Select OK to save and close.
7. Open the properties of Speed Set (assuming that you created that tag).
8. Open the I/O tab.
9. In the Privilege drop-down, select Pump Operation.
10. Select OK to save and close.

The pump's controls are now protected. Go ahead and try to operate it. You'll get a message telling you that you don't have the Pump Operation privilege. But maybe you want to block unauthorized users from even monitoring the pump, not just operating it.

1. Close the Tag Browser if it is open.
2. Open the Idea Studio.
3. Open the Pump Controls page within the Idea Studio.
4. Select the Page Properties ribbon.
5. Expand the Page Security tool.
6. Select the Pump Page privilege.
The page immediately vanishes. Your account doesn't have the privilege required to view or edit it.
7. Close the Idea Studio.
8. The page cannot be seen or opened by anyone who does not have the Pump Page privilege.

In the final set of steps, you'll grant that privilege to the operator. You could grant the privilege directly, but that's regarded as poor practice. Instead, you'll take a few extra steps to do the job well.

1. Open the Accounts Dialog.
2. Select the Add New Role button. (A plus sign above the list).
3. Name the new role `Northern Operation`
4. Set the description to `View and operate pumps`
5. Select the Add button beside Additional Privileges.
The Add Privileges dialog opens.
6. Select both the Pump Operation privilege and the Pump Page privilege.
7. Select OK to close the Add Privileges dialog.
8. Select the Apply button in the Accounts dialog.
The Northern Operation role was temporary until that last step.
You could not use it before applying the changes that created it.

9. Select the NorthOperator account.
10. Add the role you just created to that account's list of roles.
11. Repeat for your own account.

Privileges are not granted automatically to any user account, including your own.

New section

Create named dividers within your list of custom privileges. This can be helpful for organizing privileges by purpose so that they are easier to find. The structure you create will also be used in the Add Privileges dialog when granting privileges to accounts or roles.

Privileges and sections can be reorganized using the up / down arrows at the bottom right of the list.

Sections can be indented so that they appear as sub-sections, using the left / right arrows on the selected section.



Figure 7-8 Two custom privileges, each in its own section.

Rename or delete privileges

You can rename a custom privilege by double-clicking on it within the Privileges dialog (or by selecting it and clicking the pencil icon) to open the editing field, then typing over the old name. Because applications privileges are tied to tags, pages and accounts by an identifying number, changing the name has no other effect.

There is no delete button for the list of privileges. If one becomes obsolete, rename it and use it for a new purpose, or rename it to "unused".

Troubleshooting:

- Difficulty entering the new privilege name.

Ensure that you click within the name entry field before starting to type, and press <Enter> or <tab> after typing the name.

Rules for Privilege Scope

A "rule" is defined as a limit placed on a privilege. The operator may have the privilege of operating a pump, but only pumps under one context (station or site) within the Tag Browser. Or, only when signed-in at a certain workstation.

Tag Scope Rules

Security rules are especially useful when you have organized your tags into parent-child hierarchies that group similar parts of the application together. For example, a city utility may have grouped all of the tags for the eastern half of the city under one Context tag named EasternZone. All of the tags for the western side are grouped under a Context tag named WesternZone. For operators who work in the EasternZone, you can restrict tag-related privileges within their job description role to apply only to tags in that zone, even though all tags are protected by a single privilege.

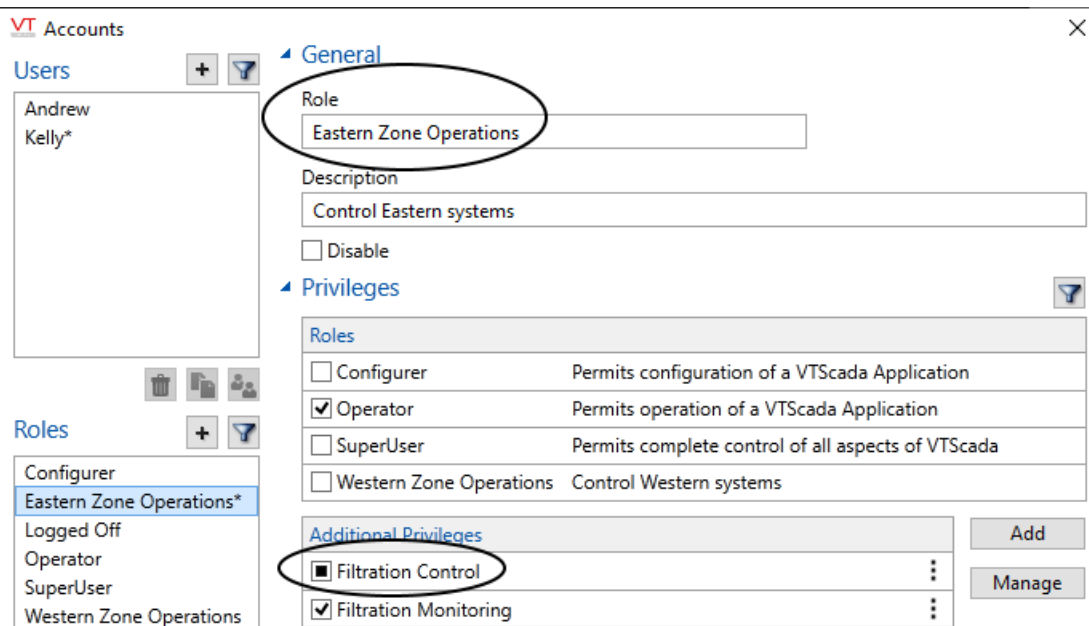


Figure 7-9 Detail from the Accounts dialog, showing with one privilege subject to tag scope rule. The square in the selection box of Filtration Control indicates that it is limited by a rule.

The example in the previous figure shows an example "Eastern Zone Operations" role. The role contains two custom privileges, Filtration Control and Filtration Monitoring. Filtration control is meant to be applied to I/O tags and is therefore limited by a scope rule to tags in the Eastern Zone context. (Examples follow, showing how the rule is applied.) The custom privilege, Filtration Monitoring, is meant to be applied to pages and therefore is not limited by a tag-scope rule.

Use the Manage Rules dialog (following figure) both to add and to remove rules. Removing the privilege (then re-adding it) is an inefficient way to remove rules.

Caution: Apply tag-based rules only to custom privileges or to the tag-related general privileges, Questionable and Manual Data. Limiting a general privilege such as Alarm Page Access to a tag is the same as denying the privilege.

Steps to apply rule-scope:

1. Find the privilege in the list of Additional Privileges.
If the privilege has not been granted to the account or role, add it. ([Assign Privileges](#))
2. Expand the menu for that privilege as shown:

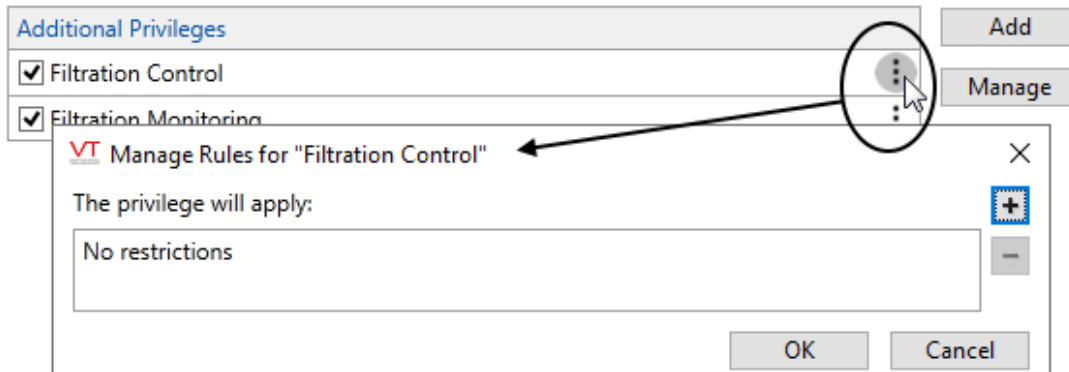


Figure 7-10 Step 1 of adding a rule to a privilege.

3. In the Manage Rules dialog, click the plus button to open the New Rule dialog.

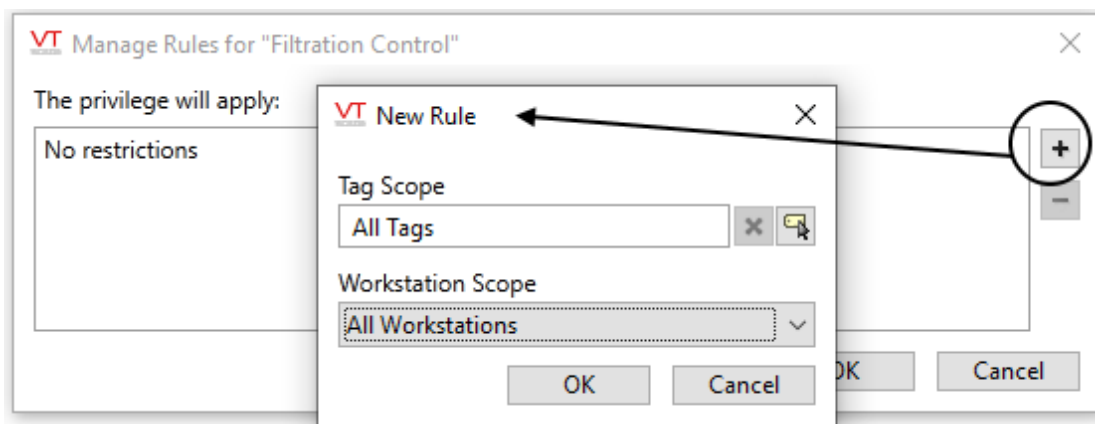


Figure 7-11 Use the New Rule dialog for both Tag Scope rules and Workstation Scope.

4. In the New Rule dialog, use the Tag Selection button to open the Tag Browser.
5. Select the tag (or better, the parent context) for which the rule is to apply.
6. Optionally, select more tags for which the rule should apply.
7. Click OK through all dialog boxes to exit.
8. Click Apply in the Accounts dialog to save your work.

The square instead of a check mark indicates that the privilege is granted conditionally.

Workstation Rules

You can also create a rule such that the privilege is valid only when the user is signed in on a named workstation. For example, if you have created a Manager account, with permissions to modify user accounts, you may wish to restrict that privilege so that it may only be used at a given workstation. Even if someone were to guess the manager's password, they would not be able to modify accounts unless they were also at that person's workstation.

Caution: Take care that the workstation you select is or will be available. Don't lock yourself out!

Workstation rules are not intended for use with Internet or Mobile client connections. It is not possible to determine the name of the remote device. The rule scope will apply to the VTScada Thin Client Server, affecting all connections.

The steps to apply a workstation rule to a privilege are the same as those to apply a tag scope rule, excepting only that you will choose one or more workstations instead of one or more tags.

Exercise 7-3 Custom privileges and scope-limited rules

This assumes that you have two pumps, as created in an earlier exercise.

If not, do the following. Otherwise, skip to the next set of instructions.

1. In the Tag Browser, navigate to Pump 1.
2. Right-click on Pump 1 and select Copy from the menu.
3. Right click on the parent driver tag (PLC1) and select Paste as Child from the menu.
4. In the warning dialog that appears, change the name from "Pump 1" to "Pump 2" *and press tab or enter.*
5. Click OK to save.
The I/O addresses are the same, but that doesn't matter for the sake of this exercise on security privileges.
6. Open the Idea Studio to the Pump Controls page.
7. Copy and paste the widgets for the HOA switch and the Speed Set, then link the new instances to the tags under Pump 2.

In the first set of steps, you will ensure that both pumps are protected by the same privilege, Pump Operation.

1. Open the Tag Browser.
2. Expand the tag tree until you can view the child tags of Pump 2.
3. Assign the Pump Operation privilege to both the selector switch and the speed control tags.
4. Close the Tag Browser.
5. Confirm that you can start and stop both pumps.

Next, you will apply a rule scope to the Pump Operation privilege.

1. Open the Accounts dialog.
2. Select the Northern Operation role in the list of roles.
3. In the list of Additional Privileges for that role, identify the Pump Operation privilege.
4. Expand the triple-dot menu to the right of that privilege.
The Manage Rules dialog opens.
5. Select the New Rule tool.
(The button labeled with a + sign.)
The New Rule dialog opens.
6. Select the Tag Browser button beside Tag Scope / All Tags.
An instance of the Tag Browser opens.
7. Navigate to find and select Pump 1.
The Tag Browser closes.
8. Select OK.

The final step is to test the rule.

1. Continuing to work in the Accounts dialog, select your user account.
2. Ensure that it has been granted the Northern Operation role.
3. Select the NorthOperator account
4. Ensure that it has been granted the Northern Operation role.
5. Apply your changes.
6. Close the Accounts dialog.
7. Try operating each pump. You should be able to operate Pump 1 but not Pump 2.

Bonus Exercise:

1. Copy the Northern Operation role to a new role called Western Operation.
2. In the Western Operation role, change the rule limiting the scope of Pump Operation from Pump 1 to Pump 2.
3. Apply your changes.
4. Configure your users such that one has the Northern Operation Role but not the Western and another user has roles Western... but not Northern...
5. Signing in as each user in turn, ensure that your ability to operate pumps is as expected.
6. If not, then either the tag is not protected as you expected, or the operator has more privileges than you expected. Look carefully to discover which case applies.

Read-Only Workstation

Caution: The default system privilege mask for a read-only workstation does not grant the configuration privilege or the edit files privilege. *If you configure the workstation you are using to be read-only, you will have no means to do further configuration at that workstation, or even to reverse that change.* Your only recourse will be to move to another workstation on your system and use the [Version Control](#) system to reverse the change.

*** DEFINING A WORKSTATION TO BE READ-ONLY ***

*** IS BEST DONE FROM ANOTHER WORKSTATION. ***

(A remote connection using a VIC or Anywhere client does not count as working at another workstation. Do not proceed unless your application has a [Client / Server Configuration](#))

You can configure a workstation to have read-only access to an application, regardless of the privileges assigned to the logged-in user at that workstation.

This is especially useful for workstations located in unsecured areas or for a server providing VTScada Thin Client access. If the workstation configured as read-only is also a VTScada Thin Client Server, then all VTScada Thin Client connections to that server will have read-only access.

Configuration of what can be done from a read-only workstation is done using three workstation-specific properties:

When set, the station can display I/O data but not write to hardware. No operator will be able to write to hardware from this station, regardless of their privileges, and without exception. This setting places no other restriction on the signed-on operator and they will be able to do all other tasks within the application according to their privilege set.

Relevant only on workstations where `ReadOnlyStation` is set. Does not apply otherwise. This is a bitwise value that controls which system privileges are enabled at a read-only workstation. By default, only the following privileges are granted. (Assuming that the signed-in operator has also been granted these privileges.)

- Account Modify
- Application Stop
- Thin Client Access
- Alarm Page Access
- History Page Access
- Page Note Hide
- Sites Page Access
- Maps Page Access
- Global Tag & Area Filter
- Recipe Page Access
- Remote Tag Value/History Retrieve
- Services Page Access
- Parameter View
- Remote Data Access

[StationMaskApp](#)

Note: The read-only workstation setting takes precedence over all security privileges related to writing I/O. If set for a particular workstation, then all I/O write access is denied. Attempts to operate any control will result in the Access Denied dialog box being displayed. There are no exceptions to this rule. There are no methods to enable partial write access at this workstation, or to grant write access to some users but not others at this workstation.

Note that you can also create [Rules for Privilege Scope](#) so that an operator's privileges are in effect only at named workstations. This provides more flexibility in controlling who can do what from where, but at the cost of not providing simple, blanket-coverage that applies to all operators.

To configure a workstation as read-only, add the following line to the [System] section of that computer's Workstation.Dynamic file, and import the file into the application's working set.

```
[System]
ReadOnlyStation = 1
```

Note: a Workstation.Dynamic file is not named "Workstation.Dynamic". Substitute the name of the computer to which the configuration variables should apply, for the word "workstation". You can have a different workstation. dynamic file for each workstation in your network.

Note: To add or change application properties, you must have the Configuration privilege or the Edit Files privilege. Note that this means that anyone with those privileges can make changes to these settings.

Steps to define a read-only workstation:

1. Open the Application Configuration dialog.
2. Click on the Insert button

The Add Property dialog will open.

3. Set the property name to `ReadOnlyStation`
4. The section should remain as `System`
5. Set the value to 1.
6. Select the workstation where this will apply.

In most cases, it is unlikely that you want your current workstation to become read-only.

7. Enter a comment, describing the new property.

Comments will be stored on the line below the property in the Workstation.Dynamic file.

8. Select OK

The dialog closes. Note that the new property is not saved until you apply your changes.

9. Select Apply.

The Comment dialog will appear.

10. Type a comment into the Comments dialog and click OK.

This comment is for the VTScada version control system and should explain why the new property is being added, unlike the earlier comment that explains what the property does.

Station Masks

A mask is a series of bits (0's and 1's) that are compared to a template using a bitwise AND. Every bit has a unique meaning. If the same bit is set in both the template and the mask, the meaning assigned to that bit is enabled. Typically, all bits are set to 1 in the template.

Numbering is zero-based and bits are counted from right to left. For example, a mask that is one-byte wide (eight bits) might be written as the following:

```
StationMaskSys = 00000010;
```

In this example, only the second bit (bit number 1 - Account View privilege) is set. (The bit furthest to the right is bit number 0.) For the purpose of a station mask, custom privileges are numbered in the same way, even though the number shown in the Privileges dialog will be 16 greater than the matching bit number.

The first four bits of the station privilege template are defined as:

Bit #	Meaning
0	Configure privilege
1	Account view privilege
2	Account modify privilege
3	Accounts manager privilege

For further information, refer to the two station mask properties: [StationMaskApp](#) and [StationMaskSys](#)

Administrative Settings

Accounts that possess the Administrator privilege and that are not assigned to a realm can control options such as minimum password length, automatic log-off time and more. You can find these controls within the Administrative Settings dialog, which opens in response to "Options" in the security menu.

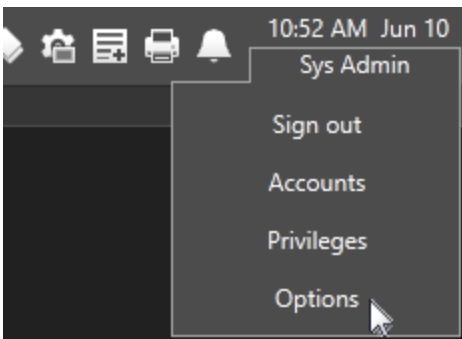


Figure 7-12 Opening the Administrative Settings dialog

This panel is used to control security configuration options that apply to all accounts.

Note: Refer to the documentation menu system, subtopics of this chapter, for details of each group of settings.

VT Administrative Settings

✕

▸ **Automatic Sign Out**

☐ No Automatic Sign Out

15 Minutes of Inactivity (0 - 720)

▸ **Password Options**

1 Minimum Length (0 - 255)

0 Minimum Alphabetic Chars (0 - 85)

0 Minimum Numeric Chars (0 - 85)

0 Minimum Special Chars (0 - 85)

☒ Password Never Expires

0 Days until expiration

☒ No Expiration Warning

0 Days before expiration to warn

▸ **Advanced**

Security Provider:

New Application

User Realm Delimiter

☐ Separate Realm Entry During Sign In

☐ Enable OAuth 2.0

☐ Enable Windows Security Integration

☒ Automatically Add AD Users

☐ Enable Smart Card Authentication

☐ Enable Signed Out VIC Sessions

☐ Permit remote sign-in using VTScada account credentials

▸ **OpenID Connect**

OK Cancel

Tip: It's easier to work with one section of this dialog if you minimize the other sections.

Not all security configuration options are available in this menu. For example, the automatic lock-out feature, where users are locked out of their accounts for a period of time after N failed password attempts, is not configured in the user interface. For this and others, refer to the Security Properties chapter in the reference section.

Notes:

- If it is your intention to share security between applications by selecting an OEM layer to provide the security configuration, do that before spending time configuring accounts and other security options in the current application.
- The Enable Logged Off VIC Sessions option is used for VTScada Internet Client (VIC) connections. When selected, a remote user can sign out, but retain a connection to the VTScada Thin Client Server. This ensures that the user can log back in, but at the cost of a client connection that is not freed for use by others while the client window remains open.

Automatic Sign Out Time Period

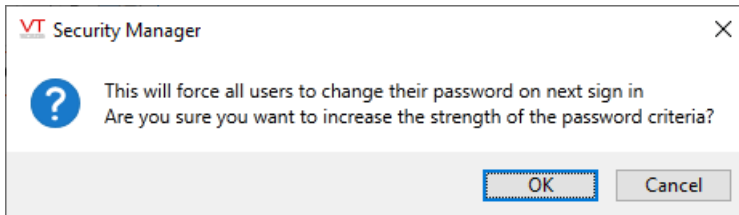
VTScada can sign users out when the application is left idle for a defined period of time. The setting in the Administrative Options dialog will apply to all users, but you may also set a unique sign out time for each user in the Accounts dialog.

The default time period is 15 minutes. The valid range is from 0 (no automatic sign out) to 720 minutes (12 hours). You may also disable this feature by selecting the No Automatic Sign Out option (0 minutes).

Password Options

The four spin boxes define the required strength of passwords. Password strength is a measure of how difficult it is to guess the word. In general, words from a dictionary are easily guessed by a hacker. Words that include a mix of letters, numbers and symbols are more difficult. The longer the password, the more difficult to hack.

Note: When you change any of these values, all existing users will be required to change their password on their next log-on.



Minimum Length - Sets the overall minimum number of characters. It ranges from 1 character to 255 characters. The minimum password length is at least the sum of the other 3 minimum values.

If anyone attempts to set a password shorter than this length, a warning dialog will be displayed.

Minimum Alphabetic Characters - Sets how many letters must be included in the password.

Minimum Numeric Characters - Set to a number greater than 0 if you want to require passwords to contain numbers.

Minimum Special Characters - Set to a value greater than 0 if you want to require passwords that contain symbols such as @#\$%, etc.

If an attempt is made to set a password that does not have the required number of any type of character, a warning dialog is displayed. This will also tell the user how many of each required type of character must be used.

VTScada also limits the number of attempts an operator can make before being locked out and the length of time for which they will be locked out. Settings for both those numbers and much more can be found in [the documentation](#).

Password Expiration sets the number of days between forced updates to every user's password. When the expiration date is reached or passed, existing passwords will work one more time, but users must provide a new password before continuing. Choosing not to set a new password simply returns you to the Sign in stage, allowing time to consider a new password before continuing.

Advanced

Caution: Shared Security (Security Provider) is a deprecated feature and will be removed in a future version of VTScada.

Do not use.

Windows Security Integration is suggested as an alternative.

Advanced

Security Provider:
 Bedford Scada

User Realm Delimiter

☐ Enable Windows Security Integration
☒ Automatically Add AD Users
☐ Enable Smart Card Authentication
☐ Enable Signed Out VIC Sessions
☐ Permit remote sign-in using VTScada account credentials

☐ Separate Realm Entry During Sign In
☐ Use OAuth 2.0

Enable Windows Security Integration

Requires coordination with your IT department. Refer to notes in [Windows Security Integration](#).

User Realm Delimiter

Must be defined before enabling Separate Realm Entry During Sign In. Refer to notes in [Security Realms](#).

Enable Smart Card Authentication

Available only in combination with Windows Security Integration. Refer to notes in [Smart Card Support](#)

Enable Signed Out VIC Sessions

If Enable Signed Out VIC Sessions is selected, behavior on sign-out will be similar to a desktop session, where the application remains visible waiting for the next sign-in. Applies only to VIC clients, not Mobile or Anywhere.

By default, signing out of a VIC session causes the connection to close, freeing the license. When selected, the client license is not released until the connection is closed.

Permit remote sign-in using VTScada account credentials

Applies to thin client connections, remote data access including REST queries, and SOAP service access.

When enabled, authorized users can sign in using their VTScada account credentials, as the label states. If using OpenID Connect to enforce two-factor authentication, this option should be disabled so that remote users do not have the option of using their account credentials instead of OpenID Connect.

See: [OpenID Connect Authentication](#)

Enable OAuth2

Requires coordination with your IT department. Refer to notes in [OAuth 2.0 Configuration](#)

OpenID Connect Authentication

Note: Before using OpenID Connect, your site must implement or have access to an OpenID Connect Provider (<https://openid.net>). Trihedral does not endorse or recommend any provider over another. The VTScada documentation does not, and can not, provide instructions for the configuration of an OpenID Connect Provider.

Note: OpenID Connect logons are supported only on the VTScada Anywhere Client.

OpenID Connect is a standard for user-authentication. If enabled in VTScada, operators can connect to the VTScada Anywhere client using their OpenID Connect credentials. This may be useful at sites where a user must logon once, then access several business applications throughout their day without needing to logon again. It can also provide enhanced security through the use of two-factor or multi-factor authentication. Refer to the [OpenID Foundation](#) website for further information.

OpenID Connect accounts are mapped to each user's VTScada account by matching the content of a **claim**¹ field from the provider to the VTScada account names. Every user must have both an OpenID Connect account and a VTScada account.

If you have implemented Windows Security Integration (WSI), the claim can be "email", assuming that the email addresses match the Active Directory user principal names. Because VTScada account names can include @ symbols only when WSI is enabled, some other claim that specifies the VTScada account name must be used if WSI is not enabled. It may be necessary to configure your provider to add an appropriate claim field and populate it with your user's VTScada account names.

You will continue to manage privileges and roles within VTScada. OpenID Connect is used to authenticate users when signing in. It does not specify what they can or cannot do within an application.

As with any Internet and Mobile device connection, you must first enable security, create at least one account with the Thin Client Access privilege, and you must configure a VTScada Thin Client Server with your application made available through a named realm. A TLS/SSL certificate is strongly recommended.

If a company portal is being used to direct users to the VTScada Anywhere Client then Anywhere Client URLs can be converted into URLs that will automatically initiate an OpenID Connect login. For example `https://example.vtscada.com/realm/app/ANYWHERE/Page` would become `https://example.vtscada.com/realm/app/ANYWHERE-OIDC/Page`

The redirect URIs sent by VTScada to the OpenID Connect provider take the form "`https://hostname/vtscada/oidc/return`" where hostname matches the address of the VTScada Thin Client Server. This must be a secure (https:) connection.

All of the possible hosts used to access your Anywhere Client servers must be accepted or registered with the OpenID Connect Provider.

Sign in using OpenID Connect

OpenID Connect is available only for the VTScada Anywhere Client. The following steps assume that you have finished configuration and now signing in as an authorized user.

1. Use your preferred browser to navigate to your Anywhere Client's sign in page.
2. After sign-in, you should see two options, the regular username / password route to sign in, and below it the Sign In with OpenID Connect button. If not, your system is not yet configured.

¹In OpenID Connect, each field of information describing a user's account is referred to as a "claim".

3. Select the OpenID Connect button, which should redirect you to your configured OpenID Connect provider.
4. Sign in with the username and password assigned for your enterprise's OpenID applications.
(Not required if you are already signed in to another of your enterprise's applications. You may need to provide further authentication, for example if two-factor authentication has been configured.)
5. Upon successful authentication, the Anywhere Client should start and you will be signed in.

OpenID Configuration

Note: All of the URLs configured in the OpenID Connect settings dialogue must be HTTPS, using a certificate trusted by the system running VTScada.

OpenID Connect is enabled and configured within the Administrative Settings dialog:

The screenshot shows the 'VT Administrative Settings' dialog box with the 'OpenID Connect' tab selected. The 'Enable OpenID Connect' checkbox is checked. Under 'OpenID Connect Provider', the 'OpenID Connect Discovery URI' radio button is selected, with a text field and a 'Discover' button. Other provider options include 'Issuer' with a text field, and 'Authorization Endpoint', 'Token Endpoint', and 'UserInfo Endpoint' each with text fields. To the right, the 'OpenID Connect Relying Party' section contains fields for 'Client ID', 'Shared Secret', 'Account Name Field', 'Token Age Limit (seconds)' (set to 3600), 'Requested OAuth 2 Scopes' (set to 'openid profile'), and 'Additional Trusted Audiences (optional)' with a list box and '+'/'-' buttons. 'OK' and 'Cancel' buttons are at the bottom right.

Enable OpenID Connect

You can choose to allow connections using OpenID Connect, standard VTScada username and password connections, or both.

Caution: If you disable both of OpenID Connect and remote sign-in using VTScada account credentials, users cannot connect through Internet or Mobile clients, remote data access (REST) or SOAP services.

Ensure that Enable OpenID Connect is selected before proceeding to the other options in this dialog.

OpenID Connect Provider

Enter the URI (Uniform Resource Indicator) to the metadata endpoint of your OpenID Connect provider. This will have a form similar to: <https://your.provider.com/adfs/.well-known/openid-configuration> (*)

After entering the URI, select the Discover button. The Issuer and Endpoint fields will be populated automatically from the information found there.

Alternatively, you may enter this information yourself, if known.

(*) "adfs" refers to Active Director Federation Services, provided by Microsoft. Your URI may vary.

Client ID

The Client ID that you have registered with your OpenID Connect provider for use with VTScada.

Shared Secret

The Shared Secret provided by the OpenID Connect provider for use with VTScada. Note that client certificates are not supported.

Account Name Field

Do not enter a VTScada Account Name.

This field specifies the claim that contains the VTScada account name and it will vary depending on how your OpenID Connect provider is configured. This may be "email", or "upn" etc.

Refer to the notes earlier in this discussion for more information about mapping OpenID Connect accounts to VTScada accounts.

Token Age Limit

Sets the maximum accepted age of tokens issued by the OpenID Connect provider.

Requested OAuth 2.0 Scopes

Scope is used to limit the authorization granted to the client by VTScada.

Additional Trusted Audiences

As part of the OpenID Connect process, you must check several things. One of these is that the token was issued for VTScada and is not one issued for another application and substituted in. This is done by the OpenID Connect Provider sending a list of audiences the token is intended for, and in some situations this can be a list of several audiences. VTScada must have a list of any other audiences that may be included in the ID token and any ID tokens received by VTScada that contain audiences not in this list will be rejected.

The entries added to the list are other Client IDs besides VTScada that are considered trustworthy.

This is unlikely to be required for most installations and may be left blank.

8 Filtering Tags, Alarms and Realms

You can restrict certain information to authorized users or authorized workstations by using a filtering protocol defined in configuration files. These features are complementary to the VTScada security system, rather than duplicating it.

You can apply several types of filtering to tags:

Global Tag & Area Filtering

The filtering dialog is visible only in a secured application, and then only to those who have been granted the associated privilege. Allows operators to create custom filters as needed, restricting their view of tags and alarms to only those they need to see right now. Global Tag & Area filtering is not linked to any workstation.

Realm Filtering:

Limit the display of tags belonging to one or more specified areas or tag hierarchies to operators who belong to a given security realm. Operators will not see alarms outside their assigned realm. Note that they will still see all tags drawn on any page they are permitted to view, but will not be able to see or edit properties of tags outside their realm and will not be able to issue control actions using those tags.

Unlike tag area filtering and alarm area filtering, realm filtering is not linked to any workstation.

Tag Area Filtering:

Note: Tag area filtering is an older feature and works only with areas that are a single word. Do not use spaces and wildcards in this type of filter.

Prevent tags belonging to one or more specified areas from loading on a given workstation when your application runs. This type of filtering is generally used in applications where memory is limited, and certain workstations do not require access to all of your application's tags (e.g. in a large plant where a certain plant section is beyond the responsibility of operators in other sections).

Alarm Area Filtering:

Display only the alarm tags belonging to one or more specified areas on a given workstation. This type of filtering is generally used in applications where certain users should not see certain types of alarms on their workstation (e.g. in a large plant where alarms belonging to equipment in a certain plant section is beyond the responsibility of operators in other sections).

Using Multiple Filter Types

In general, more filters equates to a slower application. Consider your goal and options carefully before defining more than one kind of filter. If using multiple filter types, they are applied in the following order:

1. Tag area filtering
2. Realm filtering (tag and area)
3. Global filtering (tag and area)

4. Alarm area filtering
5. Local filters applied by dialog boxes or pages.

Realm Filtering

Tip: Customers who are familiar with legacy versions of VTS/VTScada may be looking for "Realm Area Filtering".

Filtering has been extended to allow filters that specify tag hierarchies instead of, or in addition to, area properties and is now referred to as "Realm Filtering".

Note: A new section name notation, [-REALMFILTER] has been defined. Legacy applications may continue to use the older notation, [-REALMAREAS], but that keyword will not appear elsewhere in these notes.

Realm filtering is based on a combination of security realms (as defined using the Accounts dialog) and either a set of selected tags (usually upper-level tags in a hierarchy), a set of area properties, or both. It will affect your application in the following ways:

- Members of a security realm can see and acknowledge only the alarms from tag hierarchies or areas matching their designated realm filtering list.
- When working with reports, members of a security realm can select tags only from tag hierarchies or areas matching their designated realm filtering list.
- When working with the Historical Data Viewer, members of a security realm can select tags only from tag hierarchies or areas matching their designated realm filtering list.
- The tag browser is affected, such that users in a given security realm will see only the tags that match their designated realm filtering list.
- For VIC connections, you must create a realm with the same name as each security-realm. (See: [Internet Realms](#)) Users can sign in to only the realm that matches their security realm.
- Users who are not part of a realm may only sign in to the realm designated by the [RootNamespace](#) property¹.
- Security managers who are members of a realm can see only those accounts and roles that are also members of the same realm.

Caution: Ensure that you maintain one super-user account, which is an account having the manager privilege and which is not a member of any realm. Failing to do so will make it impossible for you to manage accounts outside your realm.

- Realm filtering does not change what is visible on a page. All tags are visible on all pages to all users unless that page is protected by a security privilege. Any tag's trend window may be viewed.
- While an operators may be able to see a control widget linked to a tag outside their realm, they cannot write to hardware using that tag.

¹The value of RootNamespace must not match any defined security realm.

- Similar to the last point, operators with the Tag Modify privilege will not be able view or edit the properties of tags that are outside their realm
- Alarm notification rosters must be configured with each contact user name including the full realm qualifier.
- ODBC queries cannot see tags outside the matching realm list.

Why Should I Use Realm Filtering?

Realm filtering is most often used for larger applications where there users should be restricted to access tags or alarms from one part of the application but not another. Use Realm tag filtering to specify:

- What alarms should be visible to a user, based on their security realm.
- What tag hierarchies or areas should be shown in the tag browser, reports screen and historical data viewer when a user who is part of a defined security realm is signed in to the application.
- What tag areas should be shown in the tag browser (if any) when no user is signed in to the application.

While realm filtering can prevent users from seeing and therefore acknowledging alarms in tags or areas they are not authorized for and can also prevent them from selecting those tags in reports and trends, it does not affect any page displays other than the alarm list. It does restrict access to controls on pages.

Filtering is only one part of security. You should also use application-specific security privileges to [Protect Pages and Output Tags](#). Some benefits of realm filtering can also be achieved by using [Subordinate Applications](#).

How Does Realm Filtering Differ From Tag Area Filtering and Alarm Area Filtering?

Realm filtering affects lists of tags and alarms configured for tags in specific hierarchies or having specific area properties, hiding them from users according to their security-realm. It is not limited to any one workstation. The user may sign in to any workstation and they will still only have access to the alarms permitted by the filter.

Tag area filtering prevents tags that have been configured with specific areas from loading on a given workstation.

Alarm area filtering hides alarms associated with specific area properties from loading on a given workstation.

Where is Realm Filtering Configured?

- [Security Realms](#) are configured for each user in the Accounts dialog.
- Filters are configured in your application's Settings.Dynamic file using the Edit Properties page of the Application Configuration dialog.

How do I Configure Realm Filtering?

The following elements are involved in realm tag filtering:

- Security realms must be configured and accounts assigned to realms.
See Security Realms and follow the steps there to enable realm sign-ins and add

users to realms before proceeding with the remaining steps.

- Define one or more [\[RealmName-REALMFILTER\] Sections](#). While these are stored in Settings.Dynamic, you are advised to use the advanced mode of the Application Properties dialog.
 - Name = properties in the above section.
 - Area = properties in the above section.
- (Optional) Define a [RealmFilter] section containing names or areas whose alarms should be visible when no user is signed in.
- (Optional) Define a [*-REALMFILTER] section containing the names and areas that should be visible with a super-user is signed in.

How should I format my filters?

For both Name and Area filters, an exact text match can be used. Additionally, the question mark (?) and asterisk (*) wildcard characters are permitted. Filters are not case-sensitive.

From an efficiency standpoint, the best name filters formats are:

- An exact tag name with no wildcards.
For example `A\B\C\D` for just one tag.
- An exact tag name followed by "\"*".
For example, `A\B\C\D*` for `A\B\C\D` and all of its children.

In the case where you want to see the parents of `A\B\C\D`, but not all of their children, the pattern set would be:

`A\ , A\B\ , A\B\C\ , A\B\C\D*`

Other patterns are possible, but may be much slower.

If working with Master & Subordinate Applications and attempting to filter for tags in a subordinate application, you must use the name of the subordinate tag as seen the master, rather than wildcards. The only exception is to use a single leading wildcard, which will filter for tags in both the master and subordinate application. Note the use of backslashes in all the following examples.

Good examples:

```
MasterFolder\SubordinateAppName\MySites\*
```

```
*MySites\*
```

Bad examples:

```
Master*Folder?\SubordinateAppName\MySites\*
```

```
MySites\*
```

Security Realms and the VTScada Thin Client Server

Security realms also affect user's access to an application through the VTScada Thin Clients. On the VTScada Thin Client Server, add one realm for each security realm. Each realm must be given the same name as the security realm, and must include a reference to this application.

An operator can then connect to the application using a URL that includes the name of the Realm he is connecting to. For example, members of the Western realm would use the address: <http://www.yourdomain/Western>.

You must also define the RootNamespace property to designate an Internet realm for users who do not belong to any security realm.

[RealmName-REALMFILTER] Section

The Settings.Dynamic [RealmName-REALMFILTER] section holds the list of tag areas or tag names (and their alarms) that should be visible when a user in a given security-realm is signed in.

To specify the visible tags:

1. Open your Application Configuration dialog.
2. Select the Edit Properties page.
3. Select the Advanced Mode.
4. Select Add.
5. Complete the dialog as shown.
(Area filter shown on the left, tag filter shown on the right.)

6. Select OK
7. Repeat for as many tag names as you require.
You may use the asterisk (*) wildcard character as any part of the Value, but for best results you are advised to restrict wildcards to the beginning or end of the tag names.

Note: For each subsequent Name or Area property in the same section, you will be warned that the property already exists. Proceed anyway.

[REALMFILTER] Section

The [REALMFILTER] section of Settings.Dynamic is used only if you want to define the alarm areas that should be visible when no user is signed in. This is typically used if there is an Alarm List drawn on the default page because the Alarms page will not be accessible when no user is signed in. (See also, [RealmAreasExcludeInvalid](#) which controls whether tags that do not have any area defined will be included or excluded from view.)

To create a REALMFILTER section:

1. Open your Application Configuration dialog.
2. Select the Edit Properties page.
3. Select the Advanced Mode.
4. Select Add.
5. Complete the dialog as shown on the left to filter on tag Area properties.
Complete the dialog as shown on the right to filter on named tags. Note the * in the value, which includes child tags of Station 2.

The image shows two side-by-side screenshots of the 'VT Add Property' dialog box. Both dialogs have a title bar with 'VT Add Property' and a close button. They contain the following fields:

- Property Name:** A text input field. In the left dialog, it contains 'Area'. In the right dialog, it contains 'Name'.
- Section:** A dropdown menu. In both, it is set to 'REALMFILTER'.
- Value:** A text input field. In the left dialog, it contains 'Northern'. In the right dialog, it contains 'Station 2*'.
- Workstation:** A dropdown menu. In both, it is set to '-- default --'.
- Comment:** A text input field. In the left dialog, it contains 'Northern alarms visible without sign-in'. In the right dialog, it contains 'Station 2 alarms visible without sign-in'.
- Buttons:** 'OK' and 'Cancel' buttons at the bottom.

6. Select OK
7. Repeat for as many areas and tags as you require.
For each subsequent area, you will be warned that the property already exists. Proceed anyway.

[*-REALMFILTER] Section

This optional section is used to restrict the tag areas that should be visible when a super user is signed in. (A super user is one who does not belong to any realm.) If you do *not* provide a [*-REALMFILTER] section, the super user can see all areas. If you *do* provide the section, then they can see only the areas specified in that section.

Note: (The [*-REALMFILTER] section has no relation to the RootNamespace property. There is no realm named "*" for your super users to be a member of.)

(The following describes a procedure. It is not an exercise.)

To specify the tag areas that should be visible when a super user is signed in:

- Open your Application Configuration dialog.
- Select the Edit Properties page.
- Select the Advanced Mode.
- Select Add.
- Complete the dialog as shown on the left to filter on tag Area properties.
Complete the dialog as shown on the right to filter on named tags.

- Select OK
- Repeat for as many areas or tag hierarchies as you require.
You may use the asterisk (*) wildcard character as any part of the Value.
For each subsequent property, you will be warned that the property already exists.
Proceed anyway.

While realm filtering can prevent users from acknowledging alarms in areas they are not authorized for and can also prevent them from drawing tags having those areas, it does not affect any page displays other than the alarm list, and does not restrict access to controls.

If you wish to restrict user access to pages or to control tags, use application-specific security privileges. Some benefits of realm filtering can also be achieved by using Master Applications.

Note: Restricting realm access to areas does not mean that the operators cannot see the tags belonging to areas outside of their realm's defined set of areas. It does mean that they cannot see or acknowledge alarms resulting from those tags. It also restricts their ability to select tags for use in reports and the Historical Data Viewer.

Exercise 8-1 Configure realm filtering

Preparation:

1. Open the properties of the tag, Station 1, and set its area to `North` if that is not already set.
2. Set the area of Station 2 to `West` if that is not already set.

The next steps activate realm sign ins for the application.

1. Select your name in the title bar to open the security menu.
2. Select Options.
3. Expand the Advanced section.
4. Enter a colon `:` in the User Realm Delimiter field.
5. Press the tab or enter key.
6. Select the option, Separate Realm Entry During Sign In.
7. Select the OK button.

The following steps are for the sake of seeing the result of what you've done:

8. Sign out, then sign back in.
The Sign in dialog will now have a Realm field. Because you have not yet defined any realms, you can ignore this field for now. Press the tab key or select the arrow.
9. Sign in with your own account.

Exercise 8-2 Create and assign security realms

A security realm is created at the same time that it is first assigned to an account. Because there will be no list to select from, it is important that you are careful with your spelling each time that you create a new realm.

1. Open the Security Accounts dialog.
2. Select the NorthOperator account.
3. In the Realm field, type `Northern`
Alpha is the name of the realm you are creating and assigning.
4. Give the Super User role to NorthOperator.
5. Select the WestOperator account.
6. In the Realm field, type `Western`
Leave your own account unchanged - this will be the admin account, which must be free of any realm filters.
7. Apply your changes.

Exercise 8-3 Define Realm-Areas

In the final exercise of this series, you will bring all the pieces together to create realm-area filters. Recall that Station 1 was given the area "North" and Station 2 was given the area, "West". Take a moment to open the tag browser and confirm that this is the case, updating the area property of the two Stations if required.

1. Reopen the advanced mode of the Application Properties page in the Application Configuration dialog.
2. Add the following property:

Figure 8-1 Adding the property, Area = * to the new section, *-RealmFilter

3. Repeat to add the following properties. Note all the details including spaces. After the first property in each section, you will be warned that the next duplicates an existing area. Continue anyway.

Property Name: Area
 Section: Northern-RealmFilter
 Value: North

Property Name: Area
 Section: Northern-RealmFilter
 Value: System

Property Name: Area
 Section: Western-RealmFilter
 Value: West

Property Name: Area
 Section: Western-RealmFilter
 Value: System

Property Name: RootNamespace

Section: System

Value: General

4. Apply the changes and close the Application Configuration dialog.
5. Sign out from your account.
6. Sign in as NorthOperator, using the realm name, Northern
7. Open the Historical Data Viewer page, then the Tag Selector. Note that West area tags are not available.

A related property is `RealmAreasExcludeInvalid`. This variable controls whether tags that do not have any area defined will be included or excluded from view.

If `RealmAreasExcludeInvalid` is set to 1, then no user will be able to view tags that do not have an associated area.

Global Tag & Area Filtering

Operators who have been granted the Global Tag & Area Filtering privilege can use a tool, provided in the title bar, to restrict their view of tags and alarms to only those they need to see right now. This can be valuable in a larger application where operators need to focus on one area of the system at a time, which may change from day to day.

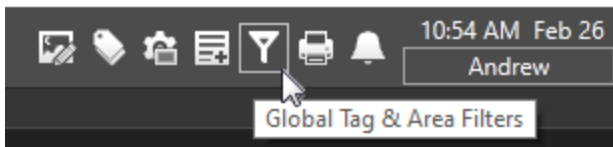


Figure 8-2 The filter tool is visible only to operators with the Global Tag and Area Filtering Privilege



Figure 8-3 When a filter is in effect, the tool will be shown in gold with a check mark

Caution: These filters affect which tags and alarms the user can see and hear in any tag or alarm list. Every user is warned of this fact when first opening the filter dialog. Improper use can mean that alarms are missed. Notifications sent by the VTScada Alarm Notification System are *not* affected by this filter.

Caution: While a name-based filter is in effect, all entries from the System Events database are hidden. This includes operator actions, security events, etc. To see system events in alarm lists, ensure that all tags are selected in the Global Tag & Area Filters dialog.

Caution: When filtering on area, Alarm Databases (and therefore their alarms) are excluded if their area does not match the filter.

Tip: You can enable/disable the coloring, and even make the filter flash when in effect using application properties. See: `UserFilterIconColorize` and `UserFilterIconFlash` in the documentation.

Global Tag and Area Filtering does not affect your ability to see pages, the widgets that you can see on a user-created page, or alarm notifications via Rosters.

This tool does limit the tags and alarms you see when viewing any of:

- Site lists
- Alarm page, alarm lists, and the title bar alarm icon
- Tag Browser
- Tag List widget
- Tag selection in the HDV (dialog must be reopened after changing the filter)
- Tag selection in the Reports page

Filters have all the following characteristics:

- User-defined.
- Dynamic and flexible. If you have access to the filtering tool, you can change the filtering rules at any time.
- Persistent. The last filter you set will remain in effect across page changes, networked workstations, and application restarts.
- Global. Your filter affects all tag and alarm lists.
- Filters cannot be sent in ChangeSets.

Tip: Global Tag and Area Filtering is designed for use by operators who need to focus on smaller parts of the application for a limited period of time. *Do not attempt to do development work while a filter is in effect.*

Create and Apply Filters

Select the Global Tag & Area Filters button to open the filtering dialog.

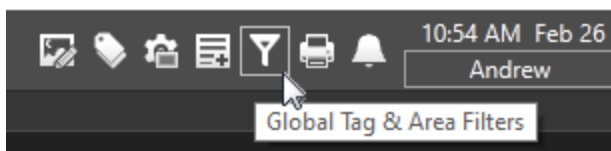


Figure 8-4 The filter tool is located on the title bar of every page that has a title bar.

Selecting this tool, as shown in the previous figure, will open the Global Tag & Area Filters dialog. You must have the Global Tag & Area Filter Privilege to see the icon and to open this dialog. Note that Global Tag and Area filtering can be disabled for all by setting `UserFilterMode` to 0.

You will see a warning the first time you open this tool in any new application. The tag list shows only container tags such as Contexts, Stations, Polling Drivers, and user-defined types. Existing restrictions are shown in this dialog.

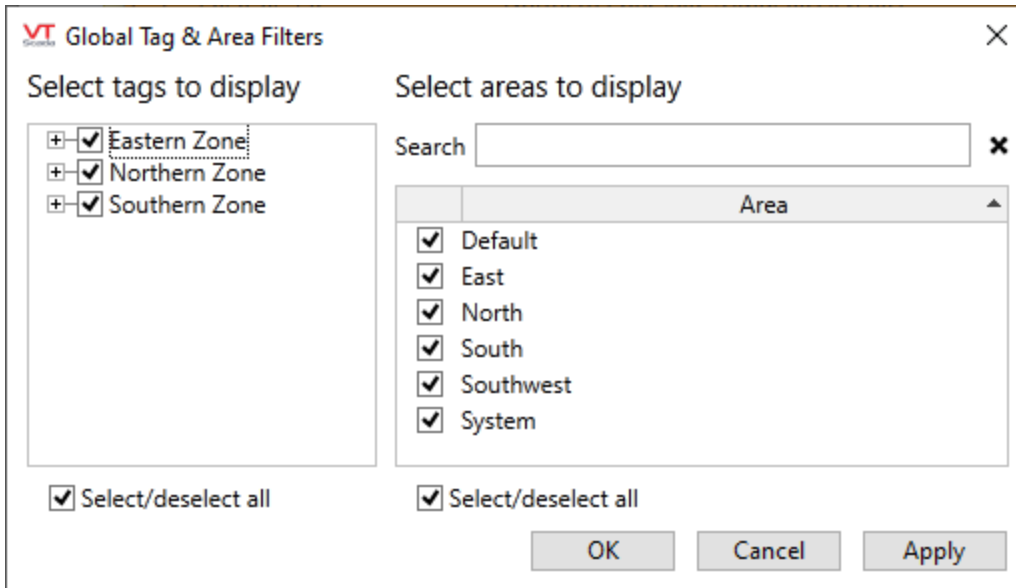


Figure 8-5 The Global Tag & Area Filters dialog with no filters applied
(all are selected for viewing)

You can control which parts of this dialog are shown by changing the value of the application property, [UserFilterMode](#). Advanced users can write code to set or check filters. See: [Custom Filters for Tags and Areas](#)

Caution: At least one tag or one area must be selected in order to apply a filter. The dialog will not allow you to create a filter that excludes all tags and all areas.

Global Tag & Area Filtering Examples:

Example 1: Selecting tags for a report.

The first of the following two images shows the tag selection for a report with no filter in effect. The second shows the same list filtered for the Northern Zone tags. Note that this zone includes tags with three area properties: North, Northeast, Northwest.

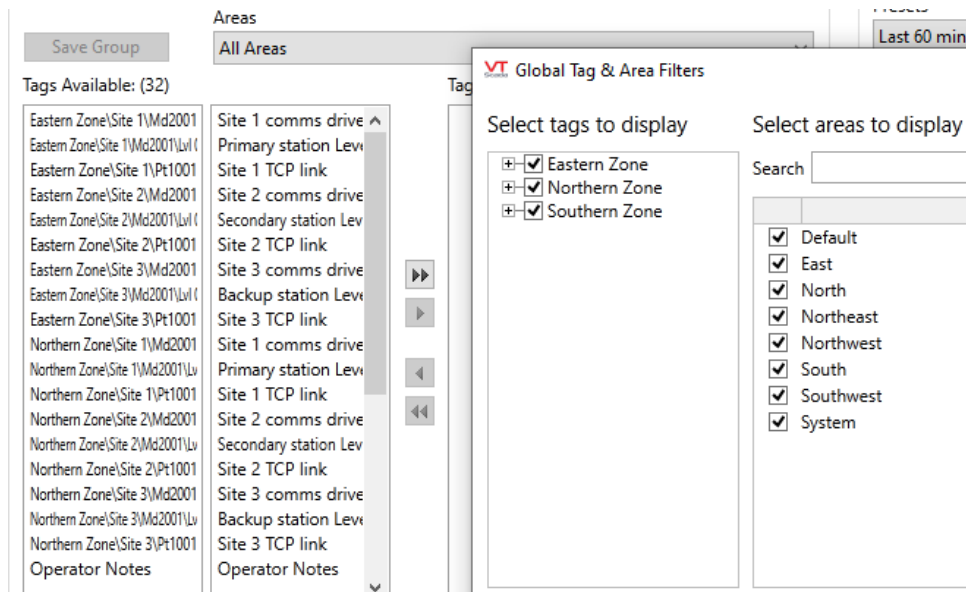


Figure 8-6 Before filtering for the Northern Zone

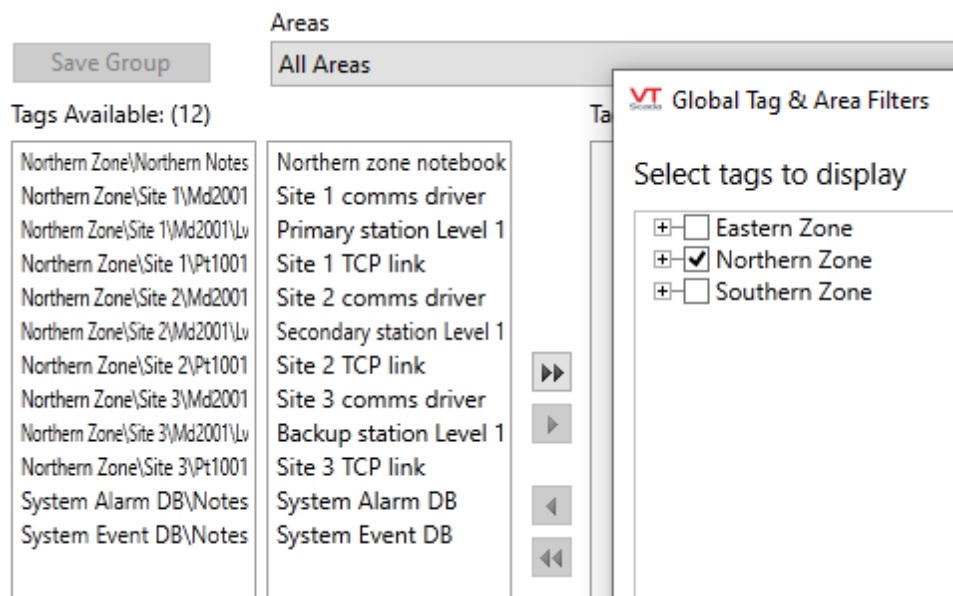


Figure 8-7 The same list after filtering out Eastern Zone and Southern Zone tags

Example 2: Alarm Lists

Filtering for alarms in either the East or the Northwest areas. Note that the System area is included in the filter. If it were not, the System Alarm DB would be excluded and no alarms would be shown.

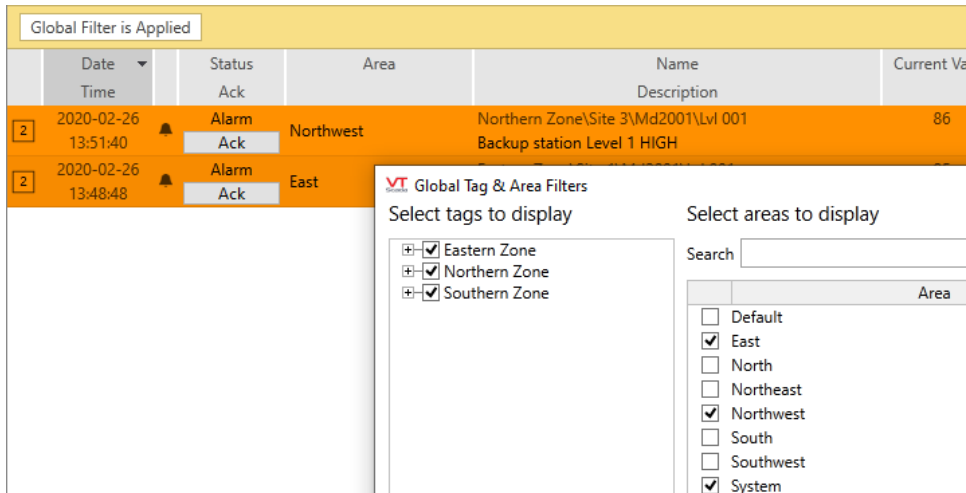


Figure 8-8 Filtering for alarms in either the East or the Northwest areas.

Tag Area Filtering

Tag area filtering will prevent tags that have been configured with a specified area from loading on a given workstation when the application runs.

Note: Tag Area Filtering is an older feature and can use only areas that are a single word. You cannot use spaces and wildcards in this type of filter. Consider using a master application with subordinate tags instead.

Why Use Tag Area Filtering?

Use tag area filtering to reduce the demand for memory on a given computer. It is particularly useful on older PCs with limited memory, or on PCs where tag licensing is restricted.

How Does Tag Area Filtering Differ From Alarm Area Filtering and Realm Filtering?

Tag area filtering prevents tags configured with specific areas from loading on a given workstation. Alarm area filtering hides alarms that have been configured with specific areas on the Alarm page on a given workstation. Realm filtering restricts the display of alarms based on the tag name or area and the security-realm membership of the operator.

Note: Tag area filtering does not affect the data presented on the Alarm page. Users can view alarm data corresponding to the unloaded tags in the History and Configured lists, and the filtered areas are visible in the Area Filtering drop-down list. If you wish to hide these areas from view on the Alarm page, you must use alarm area filtering. See: Alarm Area Filtering.

Where is Tag Area Filtering Configured?

Tag area filtering is stored in Workstation.Startup files, where "Workstation" is the name of the computer on which the properties will apply.

Caution: The Application Configuration dialog will add properties only to the .Dynamic file. Any property that belongs in .Startup must be added using a text editor and then the changes imported. ([Import File Changes Tool](#)) Existing properties in either file can be changed using the Application Configuration dialog.

How is Tag Area Filtering Configured?

The following elements are involved in tag area filtering:

- A Workstation.Startup file named for the PC to which it should apply (e.g. MyPC.Startup),
- Add the AreaFilter property, set to 1.
- Add the AreaExclude property, set to 1 or 0 to define default behavior for areas not explicitly listed.
- Add the [Areas] section and subsequent area declarations to define which areas to include and which to exclude.

The properties in each section will be Area names, with values set to 1 or 0 to include or exclude them. Areas not listed will be included or excluded according to the value of the property AreaExclude.

Note: You cannot add declarations for areas that contain a space. You cannot use wildcards in area names.

AreaFilter Property

The AreaFilter property enables (or disables) tag area filtering. you must create a copy of the AreaFilter property for the workstation, and set its value to 1 to enable tag area filtering at that station.

AreaExclude Property

The AreaExclude property defines how to handle areas not explicitly referenced in your Workstation.Startup file.

- If set to 1, then any area not explicitly included, is excluded.
- If set to 0, then areas not explicitly excluded are loaded at startup.

Caution: If using AreaExclude, be careful that system areas are included in the list of tag areas to load, otherwise you may find that ports, drivers, fonts, etc. are not loaded, thereby preventing the application from running properly on the workstation.

[AREAS] Section

The [AREAS] section is the heading under which you may specify which tag areas to include or exclude from loading at start up. You will need to add each area as a property.

- To load tags configured with a specific area when the application runs, set the name of the property to the area, and the value to 1.
- To prevent tags configured with a specific area from loading when the application runs, set the name of the property to the area, and the value to 0.
- Tags in areas not specifically listed are loaded or not depending on the AreaExclude setting.

Note: Wildcard characters and spaces are not permitted in area names when applying tag area filtering.

After you have completed the configuration, import file changes and restart your application. (Workstation.Startup files are only read when the application initially runs.)

Property Name	Section	Value	Workstation ▾	Restart	OEM
AreaA	Areas	1	StationA	<input checked="" type="checkbox"/>	
System	Areas	1	StationA	<input checked="" type="checkbox"/>	
AreaFilter	System	1	StationA	<input checked="" type="checkbox"/>	
AreaExclude	System	1	StationA	<input checked="" type="checkbox"/>	

Figure 8-10 Areas that load on StationA

The AreaFilter property in the section, System, is set to 1 so that tag filtering is enabled this workstation.

The AreaExclude property is set to 1 so that any areas not specified will not load.

The two properties in section, "Areas", have names matching the areas that will load (System and AreaA), and values of 1 to enable loading.

StationB Configuration

The workstation named StationB requires the following configuration. Note that these properties are added using the Application Configuration dialog, just as the properties for StationA were. You can save time by copying existing properties, changing values of the fields as required.

What matters most is the value in the Workstation field.

Property Name	Section	Value	Workstation ▾	Restart	OEM
System	Areas	1	StationB	<input checked="" type="checkbox"/>	
AreaFilter	System	1	StationB	<input checked="" type="checkbox"/>	
AreaExclude	System	1	StationB	<input checked="" type="checkbox"/>	
AreaB	Areas	1	StationB	<input checked="" type="checkbox"/>	
System	Areas	1	StationA	<input checked="" type="checkbox"/>	
AreaFilter	System	1	StationA	<input checked="" type="checkbox"/>	
AreaExclude	System	1	StationA	<input checked="" type="checkbox"/>	
AreaA	Areas	1	StationA	<input checked="" type="checkbox"/>	

Figure 8-11 Areas that load on StationA or StationB

The result of the above configuration is that the workstation named StationB will only load tags whose Area property was configured as "AreaB" and the "System" area tags .

9 Reusable Application Layers

While VTScada's development tools are useful for a wide variety of industries, you are likely to want to create additional reusable objects. These might be device drivers, reports, tag types, widgets, or other software modules.

With the VTScada layering system, you can create these tools in one application then re-use them in many others. This feature is especially useful for system integrators who develop applications for several clients in a particular industry. It is also recommended for use with subordinate applications ([Subordinate Applications](#)).

Perhaps the best reason to use an OEM layer is that this is the easiest way to distribute and then update custom tags, etc. for use in other applications.

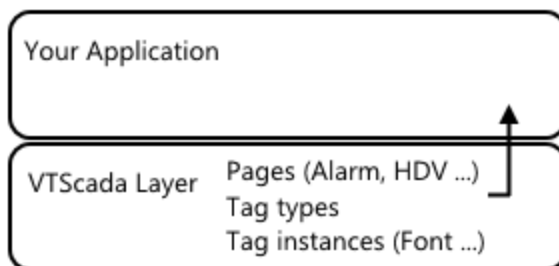
OEM¹ layers are used for any, and sometimes all of the following reasons:

- You have a set of industry-specific tools that you want to re-use for a number of client applications.
- You want to ensure consistency of certain things across applications.
- You don't want to distribute your source code.
- You want to ensure that other developers, using your application, cannot modify the structure of certain tags or other objects. Sometimes, this is to protect those objects, and sometimes the reason is demarcation of responsibilities.

A certain amount of care and planning is required for the design of your OEM layer to ensure that everything works smoothly. Certain items can or cannot be overridden at the local level, depending on how you configure things.

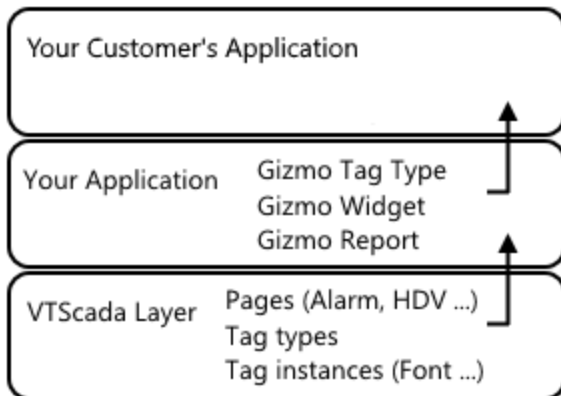
How it works:

Every standard application is built on top of the VTScada System Layer application. All the pages you see in a new application, all of the default tags including Alarm Priorities, Fonts, and Menu Items, all the built-in reports, the Idea Studio with all of its widgets, and much more are defined in the VTScada layer and inherited by your application. You can change some of those tools, such as choosing a different font for one of the Font tags. Other things, such as the Alarm page and Operator Notes page, can be used but not changed.



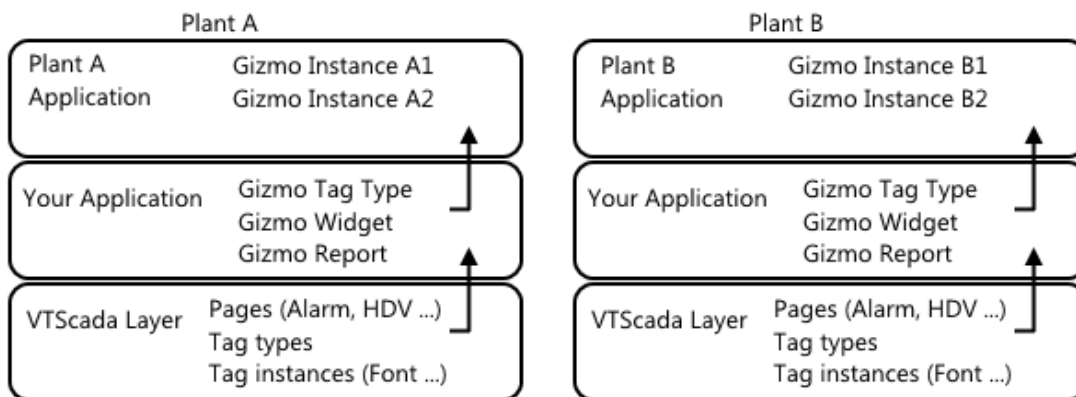
¹Technically: "Original Equipment Manufacturer". In practice, any application can be an OEM layer.

Perhaps in your application, you build a new tag type called The Gizmo for equipment unique to your industry. You also create a widget that represents all the I/O and alarms for The Gizmo. Further, you have created a report so that operators can view a daily summary of how The Gizmo performed yesterday.



You are using tools inherited from the VTScada layer to build new features in your application.

Two plants use The Gizmo brand hardware for their operations. Your application will be distributed to both plants to serve as the OEM layer for their applications. The applications at Plant A and Plant B also inherit all the features of the underlying VTScada layer, so they also have all the standard tags, widgets, reports, etc.



If the Gizmo manufacturing company adds a new feature, you will modify your application to incorporate that into your tag type, widget and report. You then send a Snapshot ChangeSet to both Plant A and Plant B, updating the OEM layer at each location. They can now install and use the latest Gizmo, without needing to do a thing to their applications. The changes were incorporated when they applied the ChangeSet to the OEM layer.

Any VTScada application can be used as the base layer ("type" or "OEM layer") for another. When creating every new VTScada application, a required step is to select its type. (The "Quick Add" option in the wizard simply sets the type to Standard Application, meaning the VTScada layer.) If, in the Add Application wizard, you select the Advanced option, and then New, you are given an opportunity to choose the layer that the new application will be built upon and inherit features from. It is possible to create a chain of many such layers, each inheriting the features of all the layers below it.

Note: There is an exception to the rule: Script applications that you create cannot be used as OEM layers. Script applications are not built on top of other layers.

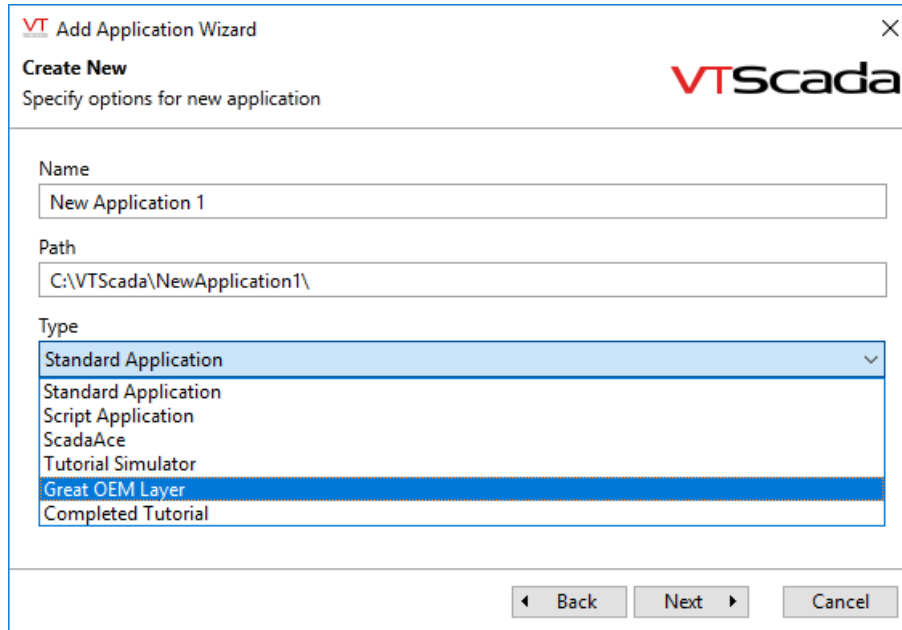


Figure 9-1 Select an OEM layer for a new app using the advanced option

The primary benefits that you gain from the VTScada layering system are re-usability and extensibility:

Re-usability:

Inheritance from lower application layers to upper layers means that the custom tools that you create for an OEM layer can be used in other applications. You do not need to recreate those tools or features for every application.

Extensibility:

New functions can be added to existing objects such as tags and drivers without changing the structure of applications built on the OEM layer. For example, if you extend the feature set of a custom tag in an OEM layer, then applications built on that OEM layer will immediately have access to the new feature set in your custom tag. Note that certain changes may require a restart before taking effect.

Other benefits from the use of OEM layers include:

- Provides a clear distinction of who is responsible to maintain what portion of an application.

- Ensures consistency across applications.
- Separates the fundamental tools (tags, widgets and other code modules) from the application using those tools. Where application-specific customizations are required, it may be possible to apply a local override to the OEM definition, depending on what you are changing and whether source code is provided with the OEM layer.

OEM layers should be used by:

- A system integrator who deploys SCADA/HMI applications with similar general functionality such as water wells or gas plants.
- A organization that uses a standard set of SCADA/HMI functionality across two or more facilities
- An original equipment manufacturer that deploys a common user interface with their equipment, where the user interface needs minimal or no unique configuration for each equipment installation.
- An organization that needs a clear line between the parts of an application that can be edited and the parts that are under more strict change-control.

Best Practices:

- Whenever the application design calls for two or more of the same complex tag type, widget, or other object, consider creating its template in the OEM layer.
- Choose the type of ChangeSet according to your intended purpose.
 - For distributed development, standard ChangeSets are best because updates are merged.
 - For customers, Snapshot ChangeSets are best because they cannot see your development history.
 - To protect your proprietary work and to prevent edits of the resulting application, choose not to include source code with your Snapshot ChangeSet. This works best when used as an OEM layer.

Note: If you intend to send updates, it is important to use "Create from ChangeSet" and never "Clone from ChangeSet". A clone will have a different GUID, and you will never be able to distribute updates to it.

- If distributing a Snapshot ChangeSet without source code, and if you want to leave the default Overview page intact for applications to edit, then delete the OEM layer's copy of Overview. The default VTScada version will take its place and be available to the dependent applications. After deleting your copy of Overview, you will need to edit the page menu to put the VTScada version back in.
- Consider enabling security in your OEM layer.
- Consider setting the flag, DoNotStart, in your OEM layer, just before creating the ChangeSet for distribution.
- Clean the OEM layer application before distribution.
You will inevitably create instances of tags, widgets and pages in the OEM layer when designing and testing new objects. Before distributing the OEM layer, delete

everything that you do not want to be part of applications built on that layer. For example, a new VTScada application has a TCP/IP Port tag type, but it does not come with instances of TCP/IP Port tags.

- In an installation with primary and backup servers, ensure that Application Configuration option, "Synchronize the configuration of OEM layers via derived applications" is selected in the top-level application (found in the "Other" tab of the Edit Properties page of the Application Configuration dialog) so that updates to the OEM layer will be distributed using the same server list.

Inheritance Across Layers

New applications inherit object definitions and application properties from the layers on which they are based. They normally do not inherit instances of objects other than tags. For example, if your OEM layer has a custom report template, your new, derived application will be able to generate reports with that template, but you will not have a local copy of the code. The code remains in the OEM layer. (Tags are treated differently; the derived application will always be given its own copy of every tag in the OEM layer application.)

The same rule is generally true for pages, image files, device drivers, wizards and any other custom code. The new application remains dependent upon its OEM layers to supply various objects rather than becoming a complete, self-contained unit.

For an example, look at the list of application properties in a dependent application: properties with values set in this application are shown in black and can be changed. Properties whose values were set in the OEM layer are visible, and affect the dependent application, but they are not defined locally. Those properties can be changed, but only by creating a local copy, which will override the value from the OEM layer.

Modify the properties of your application

To add an application property, click the "Insert" button. To delete an application property, select the property to delete and click the "Delete" button. To copy a property (for example, to override a setting for a particular workstation), select the property to copy and click the "Copy" button. To modify an application property, select the property and modify the property fields. You can sort by clicking on the column headings.

The changes you make are not applied until you click the "Apply" button.

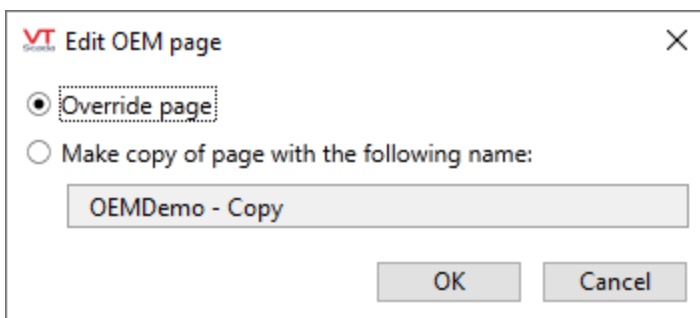
* <input type="checkbox"/> Hide OEM Properties						
Property Name	Section	Value	Workstation	Restart	OEM/	Comment
OperatorNotesConverted	Application	1	-- default --			TRUE indicates that the Operator Notes
Page	System	PageMenuPage(Ir	-- default --			Initial page to show when application st
RepeatMenuTime	System	5	-- default --			The interval between speech utterances
RosterDelay	System	0	-- default --			
SQLQueryHideLegacyTables	System	1	-- default --			If TRUE prevents v11.0- style tag tables f
#DataLines	System	100	-- default --	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Number of communications display line:
5PointStarLabel	Labels	5-Point Star	-- default --		<input checked="" type="checkbox"/>	
ABCommMessagesLabel	Labels	AB Communicatio	-- default --		<input checked="" type="checkbox"/>	
ABDriverDisableCommStats	System	0	-- default --	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
ABDriverStatsLabel	Labels	AB Driver Statistic	-- default --		<input checked="" type="checkbox"/>	

Pages and Widgets have different rules for inheritance

Use Care: Pages and widgets are handled differently from reports and wizard modules. If the OEM layer includes the source code of the page or widget, then upon opening the page or widget in the Idea Studio of the derived application, the default behavior is that the source code will be copied to the derived application. This creates a local override, which means that no further changes in the OEM layer will be seen in the derived application for that object.

For pages, you can control this behavior using the application property, [AutoOverrideOEMPages](#).

When set to 0 (FALSE), developers working in derived applications will see the following prompt upon opening the Idea Studio for any page that exists (and for which there is source code) in the OEM application.



If overriding the page (default behavior) the source code is copied to the derived application.

If making a copy, the source code is copied and also renamed. The original page remains visible and usable, but its source remains in the OEM layer and updates there will be shown in the derived application. The copy is yours edit as you like. You are strongly advised against making the copy's name match the original. Doing so will only lead to confusion as you try to keep track of which page is the original and which is the copy.

Tip: Do not include source code files with your OEM layer unless you intend to make those objects editable in applications built on your layer.

Propagation of Changes Through Layers

You will need to deploy changes in the OEM layer if the AutoDeploy option has been disabled.

If the OEM layer has been installed at a site other than where development is being done, then updates to that layer should be made using Snapshot ChangeSets. If the dependent application (henceforth, "application") is modified using new features from the OEM layer, then it is essential that at the client site, the OEM layer be updated before the dependent application.

A restart of the application will be required if changes have been made any of the *.Startup file files or to source code files, excluding tags and pages. Any change that requires a restart of the OEM layer, including all properties marked "Restart required", will also require a restart of the application dependent on that layer.

Updates from the OEM layer to the application will be blocked by overrides in the application.

Tag instance parameters

Every tag instance in the OEM layer will be visible in the application and can be used there.

No restart required for updates: Changes made to tag parameters in the OEM layer are reflected immediately in the application excepting those that have been overridden.

Application overrides: Any property except the name can be overridden in the application.

Tag structure (child tags of defined types).

Every user-defined type in the OEM layer can be used to create new tag instances in the application.

No restart required for updates. Changes to the structure in the OEM layer will be reflected immediately in the application.

Application overrides. Possible, but discouraged. It is better to define a new type using the OEM structure as a component.

To redefine an OEM type you must have the Manage Types security privilege and the application property, CanRedefineOEMType, must be set to 1. (Review the warnings related to that property before proceeding.) New child tags can then be added and the Redefine Type command can be run in the application.

Updates to the type that are made in the OEM layer will continue to be seen in the application with one exception: Adding a child tag to the type definition in the application, then adding a child tag with the same name to that type in the OEM layer will cause an error. The child tag must be removed or renamed in the application.

Properties list of custom tag types

No restart required for updates. Changes to the structure in the OEM layer will be reflected immediately in the application.

Application overrides. Not possible under any circumstance.

Menus (page navigation, palettes)

The navigational page menu and the contents of all three palettes are stored as instances of Menu Item tags. The same rule applies as for other tag instances.

Pages and Widgets

Every page and widget in the OEM layer is visible and interactive in the application. No restart is required for updates. (Restart may be required in older versions of VTScada.)

Refer to notes earlier in this topic for the effect of AutoOverrideOEMPAGES.

Application properties

Every property in the OEM layer is visible in and applies to the application.

No restart required for updates, unless the property is marked "restart required". Changes to the property in the OEM layer will be reflected immediately in the application.

Application overrides. A copy can be made of any property. The properties list will still show the OEM version of the property in gray, including its current value, but the application's copy of the property will take precedence.

Code modules including reports

Every report and other module defined in the OEM layer can be used in the application.

Restart required for updates. (May vary.) Most module changes in the OEM layer will require a restart, therefore the application will also require a restart.

Application overrides. Possible by writing code. You must copy the module file (or create a new one of the same name) and declare it in the application's AppRoot.SRC file.

Note: If the dependent application contains its own copy of a source code file or an application property, then it will ignore any changes made to those items in the OEM layer. If the OEM layer is distributed without source code as a Snapshot ChangeSet, then the dependent application cannot obtain a copy of any of the source code. It can still set its own values for application properties. Changes to the installed OEM layer at the production site can be made only by applying a ChangeSet obtained from the development workstation.

Create an OEM Layer

As you design your OEM layer, it may be helpful to remember that as inheritance works for the VTScada layer, so it will work for yours.

Consider a new, standard application:

- It starts with a set of tags including fonts, alarm priorities, etc. Those are copied to your new application because instances exist in the VTScada layer. If you create tags in your OEM layer, they will be copied to every new application based on your layer.
- The new standard application does not contain any I/O tags. But, you can create new ones. I/O tags are defined in the VTScada layer, but there aren't any instances.
- No standard application that you ever create will have an Alarm page. At least, it will never have its own instance of an Alarm page. That's because in the VTScada layer, we include only the compiled code for the Alarm page, not the source code. The next time you upgrade to a new version of VTScada, you might see a completely reconfigured Alarms page.
- Every standard application you create will have its own copy of the Overview page as soon as you open that page in the Idea Studio. The next time you upgrade to a new version of VTScada, we might replace that with a completely new page, but you will continue to use your local copy.

Step 1

The first step is to decide whether an OEM layer would be useful for you. If any of the following use-cases apply, then yes.

- You are a system integrator who deploys SCADA/HMI applications with similar general functionality such as wells, or gas plants.
- Your organization uses a standard set of SCADA/HMI functionality across two or more facilities.

- You work for an original equipment manufacturer that deploys a common user interface with their equipment, where the user interface needs minimal or no unique configuration for each equipment installation.
- You have developers who can modify parts of your application, but you want to ensure that certain core features are out of their hands.

Step 2

Next, decide what will go into that layer. If you are thinking of including user-interface elements, consider grouping those into a widget rather than distributing the page. If you supply widgets, then at the application level, developers can add your widget to their pages, and can extra information around it. If you supply pages, then what you create is what they get.

Step 3

Prepare the OEM layer for distribution. Everything you created while designing that application will show up in the applications built on your layer. Delete every tag instance that you created unless it is supposed to exist in every new application. Disabling these is not sufficient. Delete your test pages. If you changed display settings to suit yourself during development, change those back to VTScada defaults.

The Overview page is a special case. If you opened the Idea Studio while building the OEM layer, then a local copy was made of the Overview page's source code. If you distribute a Snapshot ChangeSet file without source code, then applications built on your OEM layer will treat Overview like any other page that you distribute with your OEM layer, meaning that no application built using your OEM layer will be able to edit its Overview page. If this is not desirable...

Step 3a

Delete the Overview page. (Unless you have customized it to be something that you do want to distribute as part of your OEM layer.) A warning message will inform you that the Overview page from the VTScada layer is still available.

Customize the top level of your menu system, adding the (VTScada version) of the Overview page back.

Do not open the Overview page in the Idea Studio again until after Step 4. If you do, repeat this step.

Note: When installing the Snapshot, it is important to use "Create from ChangeSet" and never "Clone from ChangeSet". A clone will have a different GUID, and you will never be able to distribute updates to it.

Step 4

Create a ChangeSet for distribution. If you do not intend to distribute your source code and do not want the people receiving the OEM layer to be able to edit it, then create a Snapshot ChangeSet, and choose not to include the source code files.

If you are sending the application to a colleague who is helping with development, then do send a standard ChangeSet.

Step 5

Create and distribute updates. Things change, and new ideas come along. Your OEM layer will evolve. Ensure that you clean up the development tags, pages, etc. as before, then create a new Snapshot. At the destination, use the Apply ChangeSet command in the Application Configuration dialog.

If you plan to use Snapshot ChangeSets without code for your OEM layer, you will need a separate workstation (or, perhaps a virtual workstation) where you can keep the source code for the layer. What you cannot do is have both the full version with code and the snapshot version without code in the same VTScada folder.

Exercise 9-1 Use and modify an OEM layer

The application you have been using so far can be considered an OEM layer. It has only a few custom tags and widgets, so call it a good starting point. Let's put it to work as an OEM layer...

Tip: Working with OEM layers means working with multiple applications, and this exercise will include a few variations. Pay close attention to the steps, to the file names, and to which application you are working within.

1. Ensure that Auto Deploy is switched ON.
2. Create a standard ChangeSet of the Level 2 application.
Save the ChangeSet to your desktop as "OEM Standard".
3. Stop the Level 2 application.
Do Not Proceed until Level 2 is safely stopped. For the rest of this exercise, do no work in Level 2.
4. Use the Add Application Wizard to *Clone* and then start OEM Standard.
When given the chance to change the name, change it from "Level 2 1" to OEM Standard.

In the next few steps you will clean up most instances of tags and pages but leave the definitions intact.

5. Open the Idea Studio and delete the Overview page and any extra pages you may have created *except* for the Station Status page and the Pump 1 Control page.
Do not delete any of your widgets.
6. Close the Idea Studio.
7. Navigate to the Page Menu and remove any folders and pages related to the pumping stations.
8. Add a menu link to the Overview page.
Wait - didn't you just delete that page? Yes, but the original is still available from the VTScada library, even after the local copy was deleted from your application.
9. Open the Tag Browser.
10. Delete both of your stations and all their child tags. Also delete any extra tags that you created in earlier exercises.

11. Create a fresh Standard ChangeSet of `OEM Standard` on the Desktop, over-writing the original.
Note that this will have a new GUID because you are working in a clone.
12. Create a Snapshot ChangeSet, ensuring that the option to include source code files is NOT selected.
Save the Snapshot to the desktop as `OEM Snapshot`.

<< Pause #1>>

You now have an application that contains custom tags and widgets, but does not show any of those. It's sort of like the VTScada Library application, except more. If you sent either ChangeSet to a customer, they would get a clean OEM layer, with some differences based on Snapshot versus ChangeSet.

For the next set of steps, you are going to make a small change using the Idea Studio so that later on you can see the effect of applying an update. But opening the Idea Studio now would mean creating a new local copy of Overview, which you would then need to remove before making new ChangeSets. Wondering why? Let's see what happens if you don't...

1. Open the Version Log in the Application Configuration dialog.
2. Take note of the current version number, then close the Application Configuration dialog.
(You'll need this in step 10.)
3. Open the Idea Studio.
If the Overview page is the one that opens (as it usually is), VTScada will automatically give you a local copy of Overview, newly copied from the library layer.
4. Open the pump control widget for editing. (The widget, not the page.)
5. Draw a shape (perhaps a circle) on the widget.
The point is only to make a change. Details don't matter, just make it obvious.
6. Open the Overview page in the Idea Studio and make an obvious change.
7. Close the Idea Studio.
8. Create a new Snapshot ChangeSet (without source code) named "Snapshot Update".
9. Reopen the Version Log.
10. Switch to the version number you made note of in step 2.

<< Pause #2>>

For the next set of steps, take the point of view that you are the customer, and you were given OEM Standard as a ChangeSet to use as an OEM layer. By reversing the last set of steps, you're already at that point. So...

1. In the VAM, create a new application...
Select the Advanced option, then Create New.
Name it `Customer App` and ensure that you select `OEM Standard` as the Type.
2. Run the new application.
It should appear to be a standard VTScada application.
Do not open the Idea Studio until directed.
OEM Complete should continue to run.

3. Create a new Station 1 tag.
You should find a StationType tag in the Tag Browser.
4. In the Page Menu, add a new page, selecting the Station Status page (DO NOT open the Idea Studio yet!)
5. Navigate to the page Station Status.
You'll note that you have an unlinked widget, but it should otherwise look as expected.
6. Switch your screen focus from the customer application to the OEM application.
7. *Working in the OEM application*, open the Station Status page in the Idea Studio.
8. Make a change to the page - perhaps draw a green circle. The point is to do something that's an obvious change.
9. Close the Idea Studio.
10. Switch back to working in the customer application. You should see your dot (or whatever) on the Station Status page. Everything done in the OEM layer is automatically a part of applications built on top. Applying a fresh ChangeSet will do the same thing.
11. *Working in the Customer application*, open the Station Status page in the Idea Studio.
You will be prompted to either override or make a copy.
12. Choose to override.
You now have your own copy of that page. You can do this because your OEM layer is a full ChangeSet with source code.
13. Change the page in some obvious way and close the Idea Studio.

The above is demonstrating layer inheritance when the source code is available in the OEM layer. Make another change to Station Status in the OEM layer; you'll find that the change does not show up in the customer application. The override that you created broke inheritance of that page.

Now let's see the difference when using a Snapshot without source code.

1. Stop both the OEM and the customer applications.
2. Remove both from the VAM, and do NOT keep the application's directories or files.
3. Create a new application using Get from ChangeSet (not Clone...) and selecting `OEM Snapshot.snapshot`.
Note that you do not get an opportunity to change the name this time.
After the application starts, note that you do not get the Idea Studio.
4. Create a new customer application using the Advanced mode and setting the type to be the OEM application.
5. Make a note of the version number in customer application.
Write that down for use later.
6. Experiment to discover what you can (and cannot) do with the tags, pages, and widgets from the OEM application. You should find that you can build the new

application freely using features from the OEM layer but you cannot change those features.

7. Be sure to open the Overview page in the Idea Studio of the customer application and make a change.

That last step seems perfectly normal, and it is. But it was possible only because, at the time you created OEM Snapshot, the OEM application did not have a local copy of the Overview page. (You deleted it.) If it did, then Overview would be the same as Station Status in the customer application: a page you can view and use but not edit.

Before the next exercise, use the Version Log to switch back to the noted version. By doing so, you ensure that the customer application no longer has its own local copy of Overview.

Distribute OEM Layer Updates

If you have modified your OEM layer, then the process to distribute that change depends on how the OEM layer was originally distributed. You may also need to restart the application, depending on the type of change made to the OEM layer.

Networked applications

If your application runs on multiple servers, and is based on an OEM layer, then that OEM layer must also be present on each server. It can be difficult to maintain separate server lists for both the primary application and its OEM layer application, especially if the OEM layer is used by more than one application. Fortunately, one server list can work for both the application and the OEM layer at any site, so long as the [SyncOEMLayers](#) property is set TRUE, as it is by default. To verify, open the Application Configuration dialog of your application (the top layer application, not the OEM layer) and look for the check beside "Synchronize the configuration of OEM layers via derived applications" in the OEM Layer section of the Other properties tab.

With SyncOEMLayers set to TRUE (1), updates in the OEM layer's configuration will be distributed automatically using the application's server list as soon as those changes are deployed in the OEM layer.

Note: Certain layers supplied by Trihedral will not synchronize automatically under any circumstance. Instructions for deployment are provided with all updates to these layers.

In rare cases, it is possible that you might not want this OEM layer to synchronize automatically. For example, you want to apply ChangeSets individually on each server, perhaps to test changes as you roll them out. Another possibility is that the update will require a full restart, and you want to control when that happens on each server. For these cases, you should add the property [DoNotSyncLayer](#) to the OEM layer application. When set, DoNotSyncLayer takes priority over SyncOEMLayers.

Note: DoNotSyncLayer requires a full restart of VTScada, not just the application.

Snapshot Distributions

Snapshot ChangeSets without source code are run-only applications. This is the most common way that OEM layers are distributed from system integrators to clients. Updates to the OEM layer must be made at a workstation where the OEM layer's source code resides, then distributed to clients via a ChangeSet. If you are also distributing updates to an application and those updates depend on new features in the OEM layer, then it is essential that the OEM layer be distributed first, followed by the dependent application.

Remember that ChangeSet updates can be applied only to applications having the same globally unique identifier.

Note: After an application has been created, its name is only a display property. You can take advantage of this fact to add a version number to the OEM layer's name, and update that number each time you are ready to distribute a new set of features. Some integrators do this to help keep track of which release is installed at a site. The name property must be updated manually.

Exercise 9-2 Apply an update.

Continuing from the last exercise...

1. Working in the OEM application, open the Application Properties dialog.
2. Select the Apply ChangeSet File page within that dialog.
3. Select the file `Snapshot Update.snapshot`.
4. Acknowledge the warning and finish the operation.
5. Close the Application Properties dialog.
6. Switch to the customer application.
You should see the updated Overview page from the OEM application.
7. Continuing to work in the customer application, open the Idea Studio and attempt to edit the Overview page.
You should find that it is not available. Because the updated OEM application includes that page and because it is distributed as a Snapshot ChangeSet without source code, you have no access to edit that page in your customer application.

Troubleshooting:

- Changes made to a page or widget in the OEM layer are not showing up in the dependent application.

The OEM layer was distributed with source code. The page or widget was opened in the Idea Studio of the dependent application, causing a local copy to be made.

Use the Idea Studio to delete the local copy in the dependent application.

If it is necessary to have local variations to a page, then the OEM page should be made into a widget that can be added to a local page.

- The OEM layer has no link to the Idea Studio.

This is by design. The OEM layer was distributed to this workstation (or server in your network) as a Snapshot ChangeSet without code. Pages and widgets within the OEM layer can be edited only on a workstation where the full version, including source code, is kept.

10 Master & Subordinate Applications

A master application is one that is able to monitor and control one or more subordinate applications. Master applications provide the following features:

- Monitor the status (including alarms) of tags in one or more subordinate applications.
 - View, trend and report on logged data from tags in the subordinate applications.
 - View all pages in the subordinate applications for which you have security privileges.
- Pages are automatically added to the master application's menu, contained within a folder named for the subordinate application.
- View tag configuration in the subordinate applications.
 - Acknowledge alarms in the subordinate application from the master application. (Alarm from subordinate applications do not sound in the master application, but are visible.)
 - Draw and use subordinate application tag instances in master application pages.
 - View page notes from subordinate applications.
 - Add page notes to subordinate applications.
 - View alarm database groups in subordinate applications.

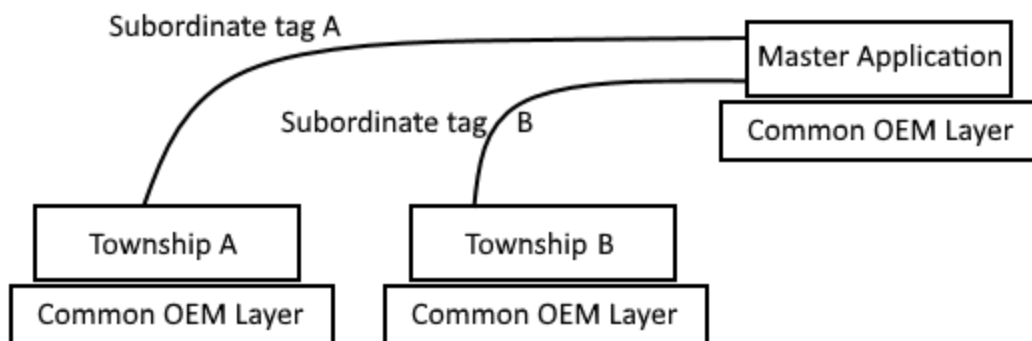


Figure 10-1 Township A and Township B are completely separate applications. Master Application can monitor both.

Restrictions and Notes

- A master application can see only the immediate subordinate applications. If one of those is also a master application, its subordinates are not visible to its master application.
- There is no requirement that you layer the master application on top of the subordinate application. The concept of master and subordinate applications is distinctly different from VTSceda's system of layered applications.
- You must install the subordinate application in the same VTSceda Application Manager as the master application. The subordinate application need not run when you create the Subordinate Application tag, but no tag values are available unless it is running.

- *User-defined tag types defined in any subordinate application must also be defined in the master application.*

It is significantly easier to manage master and subordinate applications when they are all built on a common layer. In particular, this will eliminate the need to copy tag type definitions (and updates to same) from subordinate applications to master applications. It also reduces the likelihood of conflicts if subordinate applications contain custom tag types with the same name but differing definitions.

- Master applications do not have access to draw custom widgets from subordinate applications unless that widget is also copied to the master application. Likewise, master applications cannot edit pages from subordinate applications. (Subordinate pages and their custom widgets are visible to the master application.) Like custom tag types, custom widgets should be stored in a common OEM layer that both the subordinate and master application are built upon.
- Idea Studio palettes from subordinate applications are not available in master applications.
- System resource limitations may restrict the number of subordinate applications a master can monitor, especially if those applications are "large" with many thousands of tags. Numbers are not available because any hypothetical limit will depend on hardware and on the design of the subordinate applications.
- Master applications cannot log data to subordinate applications. Nothing will stop you from selecting a subordinate application's Historian while working with logger tags in the master application, but no logging will occur. Likewise, do not link SQLLogger tags from a subordinate application to SQLLoggerGroup tags in the master application.
- If working in a master application and configuring any of Polling Drivers, DriverMux or Transaction tags, do not select child tags from the subordinate applications. Behavior in this situation is undefined.
While master applications can write to tags in subordinate applications, you are advised to keep process control of an application within that application.
- Subordinate alarms are labeled for their application, as viewed from the master application. They are not sent out by the alarm notification system in the master application.
- Tag naming rules continue to apply. The overall length of a tag name, including components from both the master and the subordinate application cannot exceed 255 characters.
- If using realm filtering, refer to notes in [\[RealmName-REALMFILTER\] Section](#)
- On root session (Server):

- Master App plays all alarm sounds
- When a Master App is running the Subordinate App:
 - Plays no sounds
 - Disables Mute and Silence buttons

- On VIC/Anywhere connection:

- All Apps play all sounds

Process

An application becomes a master application when you add a Subordinate Application tag, which is the parent for all tags in the subordinate application. Before creating a Subordinate Application tag, ensure that the following are in place:

1. Configure security.
 - Enable security in what will become the master application.
This is not a technical requirement, but it is strongly recommended.
 - Secure the subordinate application.
 - Ensure that you have access to an account with the Manager privilege in the subordinate application.
After authentication, the Subordinate Application tag will continue to function regardless of any changes made to the account used for authentication.
 - If output tags in the subordinate application are protected by custom privileges and if you intend to write to those tags from the master application, then your security account in the master application must possess custom privileges that match the privilege *index numbers* (not the names) of those in the subordinate application.
 - By definition, master applications are meant to have extensive abilities to monitor subordinate applications. If your intention is to allow an employee to monitor application A and B, but not C, then it is better to have them sign in to applications A and B directly rather than a master application that monitors all three.
2. Ensure that tag definitions and widgets are available to all applications.
 - All user-defined tag types in the subordinate application must also exist in the master application. *This includes any defined in underlying OEM layers of subordinate application.* The type and its child tags are not visible otherwise.
 - All user-defined widgets in the subordinate application must also exist in the master application.
 - The recommended approach is to ensure that the master application and subordinate applications are based on a common OEM layer where those tag types and widgets are defined. This is by far the easiest method to use in terms of long-term maintenance.
 - As an advanced option, steps to copy types are provided in [Copy Types to Other Applications](#). This practice is not recommended, but may be necessary in rare circumstances.
3. Create one Subordinate Application tag in the master application for each subordinate application.

Security Management for Subordinate Applications

It may be convenient for a user to have a single sign-in account that can be used across applications. The only practical way to bundle security access to multiple applications within a single account is to use [Windows Security Integration](#) with all the relevant applications.

Active Directory accounts are independent of your applications. They gain access to your applications through AD Security Groups, which map to roles within the applications for which you have enabled Windows Security Integration (WSI). By associating a Windows AD account with the AD Security Groups for several applications, you achieve your goal of creating a single account with access to multiple applications.

For example, consider three applications, A, B and C. They all have a common role Operator and each has a unique role, Role-A, Role-B and Role-C respectively. Assume that all three apps use a common ADGroupPrefix of "VTScada". The AD account will be a member of four AD Groups; VTScada-Operator, VTScada-Role-A, VTScada-Role-B and VTScada-Role-C. The AD account, when used in each app will be assigned two roles, the common Operator and the appropriate Role-X.

You may wish to differentiate the Operator role between the three applications. Do so either by setting a different ADGroupPrefix for each application (the preferred solution) or by creating unique names for roles within each application.

For more information, refer to the Windows Security Integration topics, starting with [Running Multiple Applications with WSI](#).

Caution: Do not use the "Shared Security" feature of the Administrative dialog. Windows Security Integration is a better solution. If you attempt to select a subordinate application that uses shared security, a warning message will be displayed.

Subordinate Application Tags

Not counted towards your tag license limit.

Provides monitoring and access to all tags in a selected subordinate application. The application holding the Subordinate Application tag is referred to as a Master application.

All the tags in the selected subordinate application will be visible as children of the Subordinate Application tag, *so long as all user-defined types defined in the subordinate application are also defined in the master application*. If the master application and subordinate application are not built on the same layer where that custom type is defined, then you must copy the definition from the subordinate application to the master application.

A master application can have several subordinate application tags, but each must link to a separate application. If two link to the same application, only one will function at a time.

You must have a security account with the Manager privilege in the subordinate application before you can configure a subordinate application tag to link to that application. You will be prompted for a username and password.

Security Management for Subordinate Applications

It may be convenient for a user to have a single sign-in account that can be used across applications. The only practical way to bundle security access to multiple applications within a single account is to use [Windows Security Integration](#) with all the relevant applications.

Active Directory accounts are independent of your applications. They gain access to your applications through AD Security Groups, which map to roles within the applications for which you have enabled Windows Security Integration (WSI). By associating a Windows AD account with the AD Security Groups for several applications, you achieve your goal of creating a single account with access to multiple applications.

For example, consider three applications, A, B and C. They all have a common role Operator and each has a unique role, Role-A, Role-B and Role-C respectively. Assume that all three apps use a common ADGroupPrefix of "VTScada". The AD account will be a member of four AD Groups; VTScada-Operator, VTScada-Role-A, VTScada-Role-B and VTScada-Role-C. The AD account, when used in each app will be assigned two roles, the common Operator and the appropriate Role-X.

You may wish to differentiate the Operator role between the three applications. Do so either by setting a different ADGroupPrefix for each application (the preferred solution) or by creating unique names for roles within each application.

For more information, refer to the Windows Security Integration topics, starting with [Running Multiple Applications with WSI](#).

Caution: Do not use the "Shared Security" feature of the Administrative dialog. Windows Security Integration is a better solution. If you attempt to select a subordinate application that uses shared security, a warning message will be displayed.

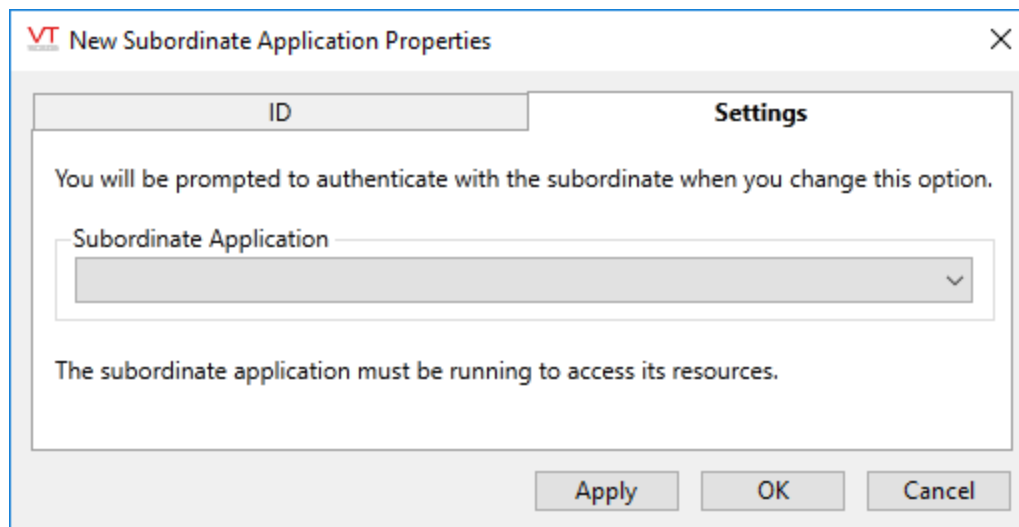
Note to advanced developers: If writing an expression that involves a subordinate application, the value of a SubordinateApplication tag is TRUE when the subordinate application is running and synchronized with the master application, and FALSE when it is not.

Note: Subordinate Application tags are excluded from tag exports.

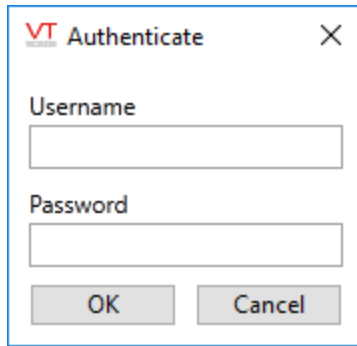
Subordinate Application tag properties: Settings Tab

The Settings tab holds a drop-down list, from which you can select the subordinate application.

Authentication to an account possessing the Manager privilege is required if the subordinate application is secured. Trihedral strongly recommends that all applications be secured.



Immediately upon selecting the subordinate application, you will be prompted to authenticate with that application.



Enter the user name and password for an account in the subordinate application that possesses the Manager privilege.

Exercise 10-1 Practice with subordinate applications

Note: This exercise uses more tags than are available under a VTScadaLIGHT license.

Before creating a master application that can monitor subordinate applications, you will need at least two subordinate applications. Both subordinate applications have custom tag-types, therefore the master application must have those as well. The easiest way to achieve this is to ensure that the custom tag types are defined in an application layer that is common to all.

As a result of all this, there is a bit of preparation to do before starting the exercise. Note that the username and password for both of the subordinate applications is "student" and "student". Security has not been enabled in the common layer, SubSim.

Preparation:

1. At the beginning of the course, you were given a set of ChangeSet files, either on a memory stick or downloaded from Trihedral's website. Ensure that you have access to those now.
2. Browse to the handout folder, Subordinate Apps.
This is with the handout files, not part of the VTScada installation folder.
3. Double-click on `Base Layer.ChangeSet`.
When prompted, add it to VTScada.
4. Double-click on `EasternZone.changeset`.
When prompted, add it to VTScada
5. Double-click on `WesternZone.changeset`.
When prompted, add it to VTScada
6. Run both Western Zone and Eastern Zone.
Signing in is optional. The username/password combination is Student/student

The exercise:

1. Create a new application as follows:
2. Select Add New Application
3. In the Add Application Wizard, select the Advanced option.
4. Select next
5. select Create New, then Next.
6. Name the application `Control Room`

7. For the Type, select `Base Layer`.
8. Select Next
9. Ensure that Start application now is selected.
10. Select Finish.
The new application opens.

We strongly recommend that you secure your master applications. But, for the sake of shortening the exercise, you will skip that process here. In the next set of steps you will link the subordinate applications to this one.

1. Working in the Control Room application, open the Tag Browser.
2. Select Add New.
3. Under the category, All Tag Types, find and select Subordinate Application.
The New Subordinate Application Properties dialog opens.
4. Name the tag `Eastern`
5. Open the Settings tab.
6. In the Subordinate Application drop-down, select the eastern Zone application.
The Authenticate dialog opens.
7. Sign in as `Student`, password `student`
The Authenticate dialog closes.
8. Select OK.
On the screen behind the Tag Browser, a folder appears: Eastern Zone. This contains all the pages from that application.
9. Add another Subordinate Application tag.
10. Name this one `Western` and select the Western Zone application.
11. A folder is added to the Page Menu for Western Zone.
12. Explore what you can do with the two applications from within the master application. Be sure to trigger and acknowledge at least one alarm.

11 Sites & Maps

Sites and maps are typically used together but refer to separate features. It is not uncommon to find applications that use maps but not site lists, while other applications will use site lists without ever displaying a map. Both features have their own relative advantages.

Site:

A tag that is designed to contain other tags.

A site list is a useful way to view I/O, grouped together in meaningful units. While the term "site" implies location, equipment such as pumps and valves can also be considered as sites, with or without location information.

Sites include:

- Polling drivers and Data Flow Station tags. The concept of site tags began with these two drivers.
- Context tags and user-defined types derived from context tags. These are defined as sites whether you add site configuration parameters to those tags.
- Station tags including MultiSmart, MPE Duplexer and MPE SC types.
- Analog Statistics and Digital Statistics tags.
- Alarm Database tags.

Only the listed types can be considered "sites". Adding child tags to other types, for example adding a Logger as a child of an Analog Input, does not make that tag a site.

Map:

A display screen of map tiles, which may display pins or other icons marking the location of site tags that have latitude and longitude coordinates.

Map tiles are downloaded on demand from a host site of your choice, or they may be cached locally on servers that do not have an Internet connection.

If permitted under your map provider's terms of service, then you may use the bulk download tool to retrieve and cache tiles for a given area and given resolution.

Site List Display

You have control over whether and how site tags are shown in the site list display. Refer to Site List Display Options for more information.

Disable / Enable Maps

The VTScada installation wizard gives you an option of not using maps, but this refers only to Open Street maps. Carto® maps are always enabled.

To enable Open Street maps later, if you choose not to during installation, edit your Setup.INI file to change the value assigned to SlippyMapOSMDisable from 1 to 0, then restart VTScada.

To disable all maps, open your Setup.INI file and locate the [SlippyMapRemoteTileSourceN] sections (where N is a number from 1 to 4). Comment-out every property in each of the sections by adding a semicolon at the beginning of each line. DO NOT DELETE PROPERTIES. If you do, and then decide to re-enable maps, you will need to reinstall VTScada. Better to simply disable the properties by adding semicolons. Save the edited file and restart VTScada.

Sites List Display Options

The Sites page (or Site List page), found in the menu of every new application, will show a hierarchical list of all tags that are considered **sites**¹. Site tags include the Polling Driver, DataFlow RTU, MultiSmart, MPE Duplexer, MPE SC, Context tags and user-defined types derived from Context tags, Analog Statistics and Digital Statistics.

You have extensive control over the Sites page including which sites are displayed, how they should be displayed, and whether and how a map is included. Control is exercised through tools in the page and through configuration of the site tags.

Sites Page Controls:

Four controls are provided across the top of the page:

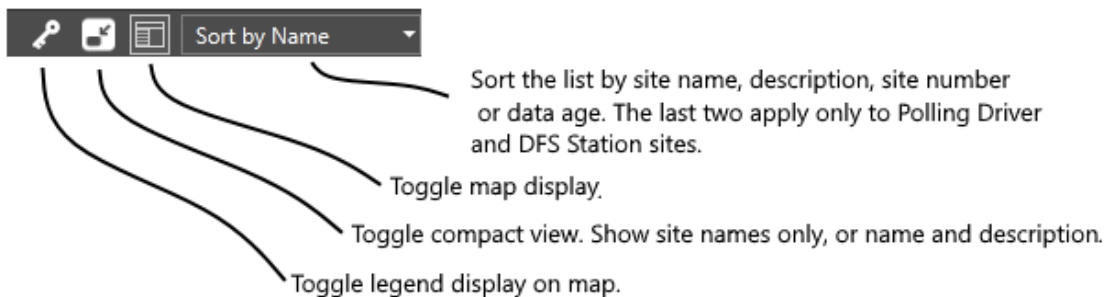


Figure 11-1 Sites Page display options

You can change the background color of the page by adding the application property, ThemedPageBGColor.

¹Tags designed to contain other tags. Includes Polling drivers and DFS stations, Context tags and user-defined types derived from Context tags, Stations, Analog Statistics and Digital Statistics.

Tip: There are three situations where it is helpful to set Exclude from Site List on your Context tags:

- When requesting a Go To Page on any Context tag (or other container type) from the Tag Browser or Alarm Page, the Sites page will be included in the list of possible pages and might be the default.
- The above is also true for child tags of the Context tag.
- Context tags (and new types created from them) are always included in the list of a Sites List Display Options. This may not be appropriate if the Context represents equipment rather than a site.

To prevent these actions, do the following:

1. Select the Add Site Properties button in the Settings tab of the Context tag.
(See: Change the Parameter List if you have already turned the Context into a new type.)
2. Delete all site properties except for SiteListDisplay.
3. In the Display tab, select the option **Exclude from the site list**
(If any child tags should be included in a site list then select Display as a Folder instead.)

Note: Customers upgrading from a version of VTScada released prior to 12.0 might expect the following behavior: Clicking the bottom-most node in a sites hierarchy would open the Site Details page as a pop-up. If an intermediate node in the sites hierarchy contained I/O as well as child sites, the I/O values were displayed below the child sites in the Sites List portion of the Sites Page.

The Site Details page now opens within the same page, occupying the space where the map was displayed. You can override this behavior by adding the property OpenSitesAsPopup, setting its value to 1.

The in-page Site Details page also includes an Open as Popup button in its title bar, providing a way for those who prefer to have the page open as a pop-up to continue opening it as such.

Site List Display Options

For all site tags in your application, you have control over whether and how it will be displayed in the list. Use the tag's configuration panel to adjust settings. Note that for Context Tags and user-defined types based on Context tags, you must add site properties to the tag before you can choose any option other than the default.

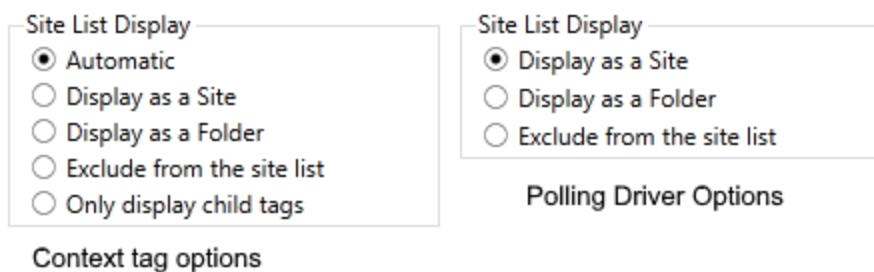


Figure 11-2 Variation between options by tag type

- Automatic: If this tag contains child sites and there are other tags like this one, folder display will be used. Otherwise, site display will be used.
- Display as Site: A click will open the Site Details page.
- Display as Folder: Indicated by an arrow. A click will change the list to show the child sites. At the top of the new list a back arrow will be added to allow operators to navigate back out of the site.
- Exclude: This tag and its children will not be shown in the Sites page list.
- Only display child tags. This tag is effectively transparent and its child tags will be shown instead of this site.

Example: The Volcano Monitoring Application

Figure 11-3 (This example is intended only as an illustration of Site List Display options. It shows an example of how and why you might choose each of the options.)

Volcano monitoring tag structure:

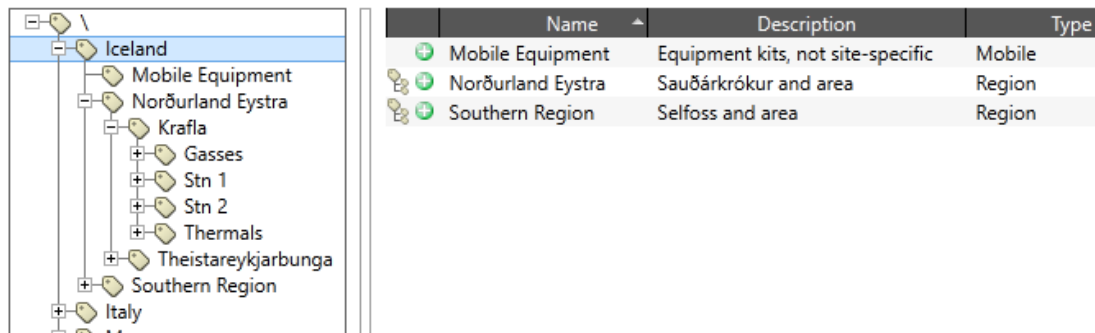


Figure 11-4 Key parts of the hierarchy: Countries -> Regions -> Volcanoes -> Stations -> Equipment

Countries --> Display as a Folder

These are not drawn on maps (monitoring stations are.) Countries are treated as "Folders" within the site list and are used only to organize the monitoring stations.

Note that if there is only a single site, VTScada will automatically navigate into that site. There must be at least two sites configured as folders.

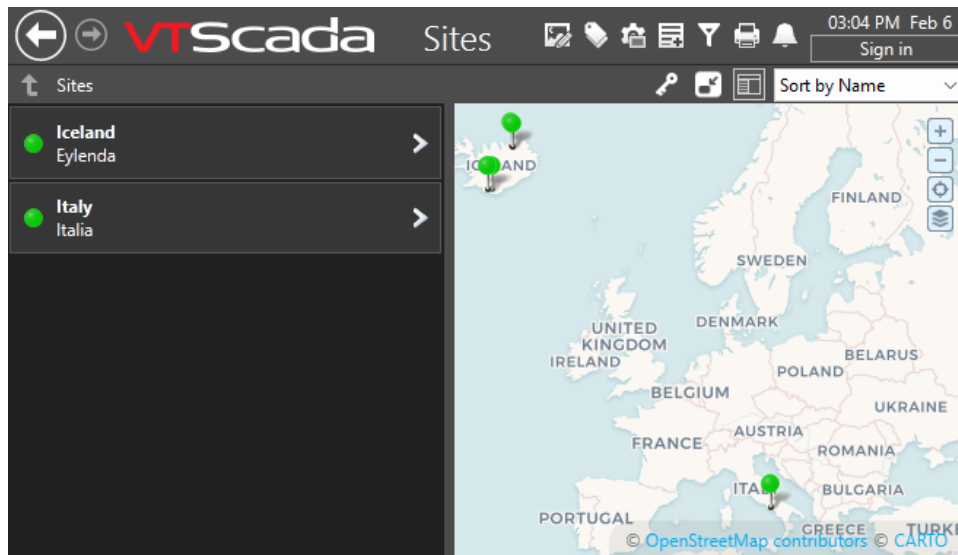


Figure 11-5 Countries, displayed as folders

Regions --> Only display child tags

If a country has many volcanoes, they are grouped into regions for administrative purposes. But when looking at a list of a country's volcanoes, we have decided not to force people to click into each region; instead all volcanoes are shown. Regions therefore, are configured as "Only Display Child Tags", effectively making the region itself transparent when viewing the site list.

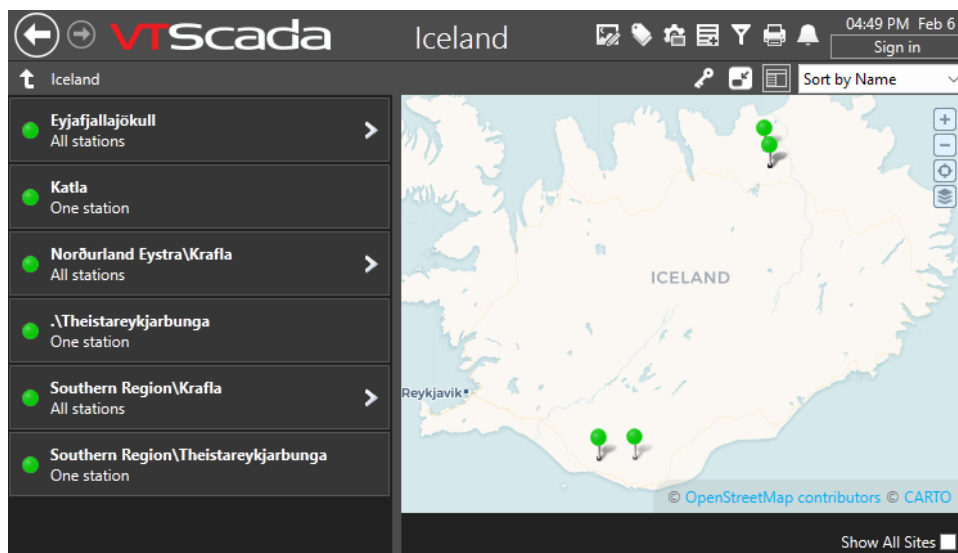


Figure 11-6 Navigating into Iceland. Regions are not shown (set to only display child tags).

Mobile Equipment --> Exclude from the site list

At the same level as the regions, each country has a collection of mobile and miscellaneous equipment. Some of this collects data and therefore has tags, but none of this equipment is relevant to a Site List or a map.

The Mobile Equipment tag is configured as "Exclude from site list display" and has no other site properties. Mobile equipment might be drawn on other pages and can be included in reports but you won't see anything from that part of the hierarchy in a Sites display list.

<< No image / nothing to be seen >>

Volcanoes --> Automatic

<< See previous image showing all volcanoes >>

Some volcanoes have multiple monitoring stations (each pinned on the map) and some have just one set of monitoring equipment. Volcanoes are configured as "Automatic" meaning that if one has monitoring stations it will be treated as a folder, and if it directly contains the monitoring equipment it will be treated as a site. Click the site (at the top of the list) to toggle between the two display options:

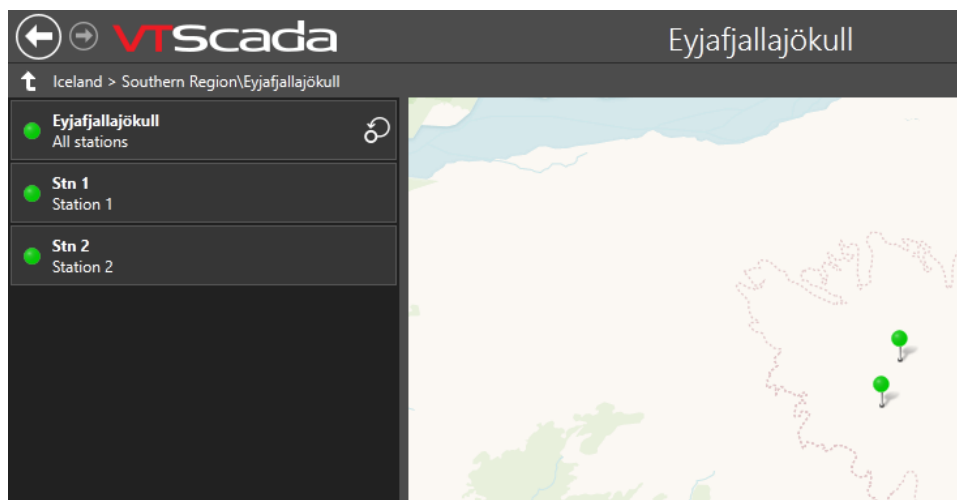


Figure 11-7 Eyjafjallajökull as a folder

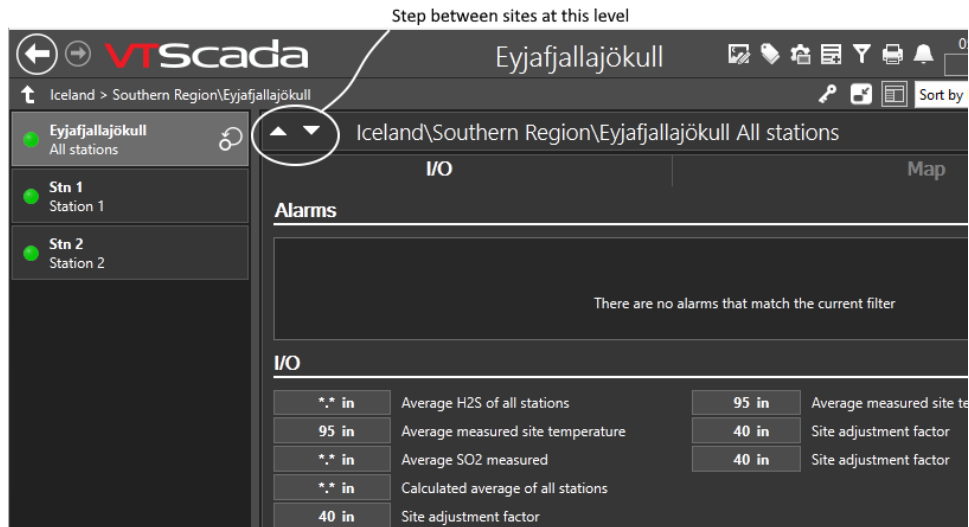


Figure 11-8 Eyjafjallajökull as a site



The "self" button indicates that this is the parent site. The self button is shown only for sites that contain both I/O and child sites, and then only when you have navigated into that site.

Stations --> Display as a Site

These are configured to be sites, always and only. They need latitude and longitude parameters to be placed on a map and they can be configured to (when clicked) open either the built-in Site Details page or a parameterized page of your choice.

Site Details Page

The Site Details page shows communication, alarm and I/O information for site and station tags. These include the Polling Driver, DataFlow RTU, MultiSmart, MPE Duplexer, and MPE SC tag types, as well as all custom types that you create, based on Context tags.

The Site Details page is automatically generated for you by VTScada. It can be opened within the Sites page or as a pop-up page. You may choose to use pages of your own creation instead, on a site-by-site basis.

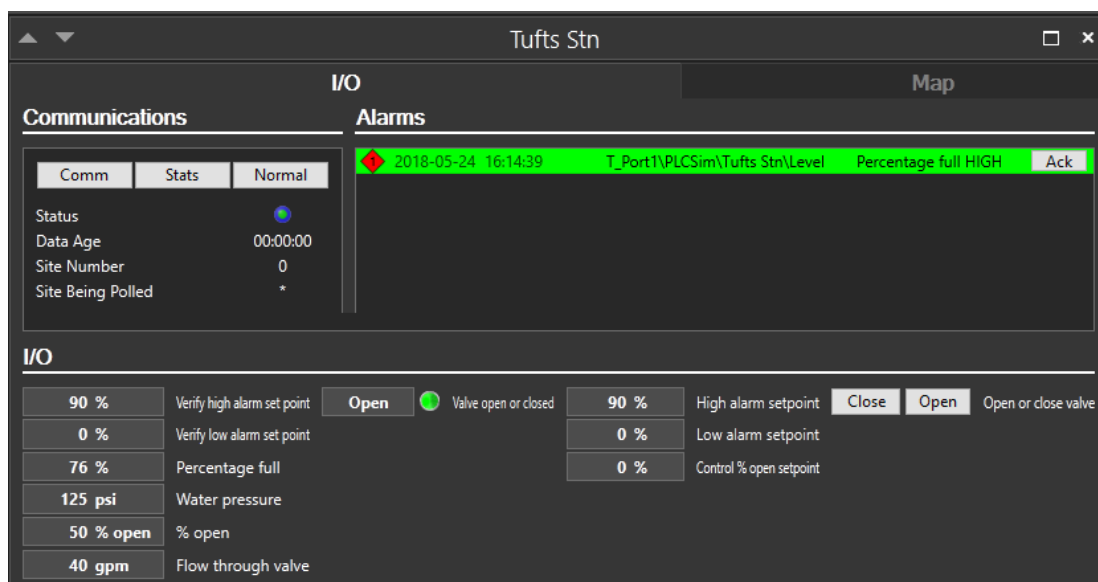


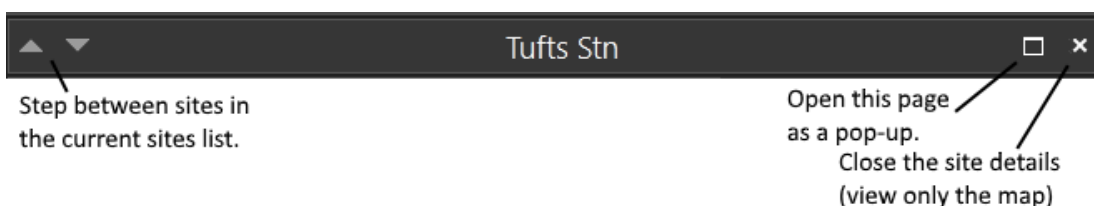
Figure 11-9 Example for a Polling Driver site, as viewed within the Sites page

There are four methods that operators can use to open this page:

- Click on a site that is included in the list of the Sites page.
- If you use the Site Draw widget to draw a Polling tag or DataFlow RTU, then operators can click on the Site Draw to open this page.
- If you have drawn any of the available site types as a Site Icon or Site Summary, (this includes a pin, drawn on a Site Map), then operators can open the Site Details page by clicking on the icon or pin.
- You can draw the page within any other application page. In general, this method is discouraged in favor of the previous three. You are likely to encounter scaling problems when the Site Details is drawn inside a standard VTSkada page.

Components of the Site Details page:

Title Bar Controls



Visible only when the Site Details is viewed as a component of the Sites page, not as a pop-up.

Tabs:

Every Site Details page has two tabs: I/O and Map. The map is simply a Site Map, showing the location of the station.

Driver Statistics Section:

Applies only to sites that are Polling drivers or DataFlow Station drivers. Visible when the I/O tab is selected. Shows statistics associated with the driver. Also included are buttons that open the Comm Messages dialog or the Comm Statistics dialog for the driver.

Alarms Section:

The Alarms list will be populated with all current alarms associated with the driver. Operators may use the Ack button here to acknowledge any given alarm.

Note: Only alarms whose area matches the area of the site tag will be displayed. If two site tags share the same area, it is possible to see alarms from both. In general, each site tag should have its own area.

I/O Section:

All of the I/O tags associated with the site are shown at the bottom of the page. These are grouped into separate columns according to the type of I/O: analog or digital, input or output. Input tags that allow writing are still considered input tags. Selector switch tags are included with the digital outputs.

Any unacknowledged alarm on an I/O tag will be indicated by the color red. Red text for analog tags and a red dot for digitals. Controls are provided for all output tags associated with the station.

Note: The application property, SiteToolsConfirmOutput, controls whether a confirmation prompt will be shown before new output values are sent.

I/O Section Filtering:

If the site tag has a parameter named CustomSiteListFilterType, then the I/O list will be filtered to show only the type, or types in the group, specified for that parameter.

You can change the background color of the page by adding the application property, ThemedPageBGColor.

Site Details Configuration

The Site Details page normally opens within the Sites page, unless an operator navigates to the site by clicking a Site Icon. You can configure it to open in the main display window in response to the Site Icon as well by setting the application property SiteDetailsWindowed to false (0).

You can control the colors used for digital values (Digital Input, Digital Status and Pump Status). The application properties are included in the following table, which also shows the default color for each state.

State	Color	Property
Invalid	black	DigitalIndicatorInvalidColor
0	light gray	DigitalIndicator0Color
1	green	DigitalIndicator1Color

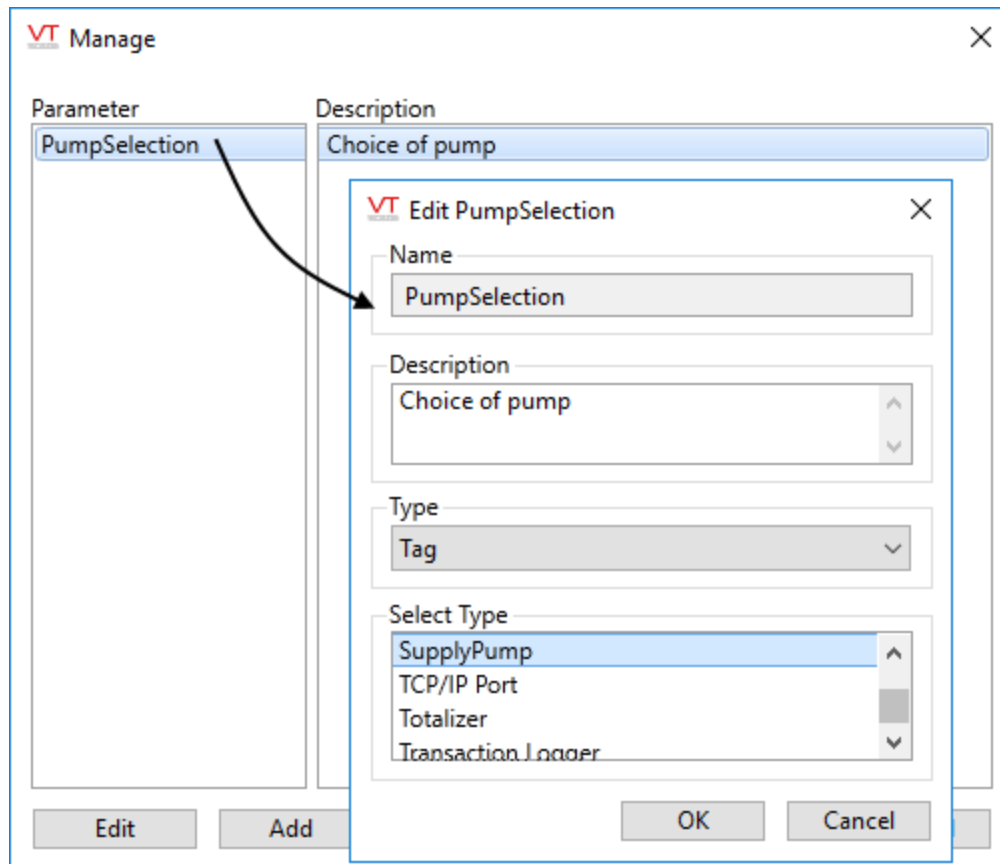
2	blue	DigitalIndicator2Color
3	purple	DigitalIndicator3Color
alarm active	red (blinking if unacknowledged)	DigitalIndicatorAlarmColor
bad quality	orange	BadQualityColor

Create a Custom Site Details Page

You may prefer to create your own site details page in place of the built-in version. Any page can be used as a Site Details page, but in general these tend to be parameterized pages (Parameterized Pages) designed to show the I/O values, controls, and other information associated with a site. Therefore, custom site pages should be configured to have a parameter of type tag, by which you will pass in the site that should be displayed.

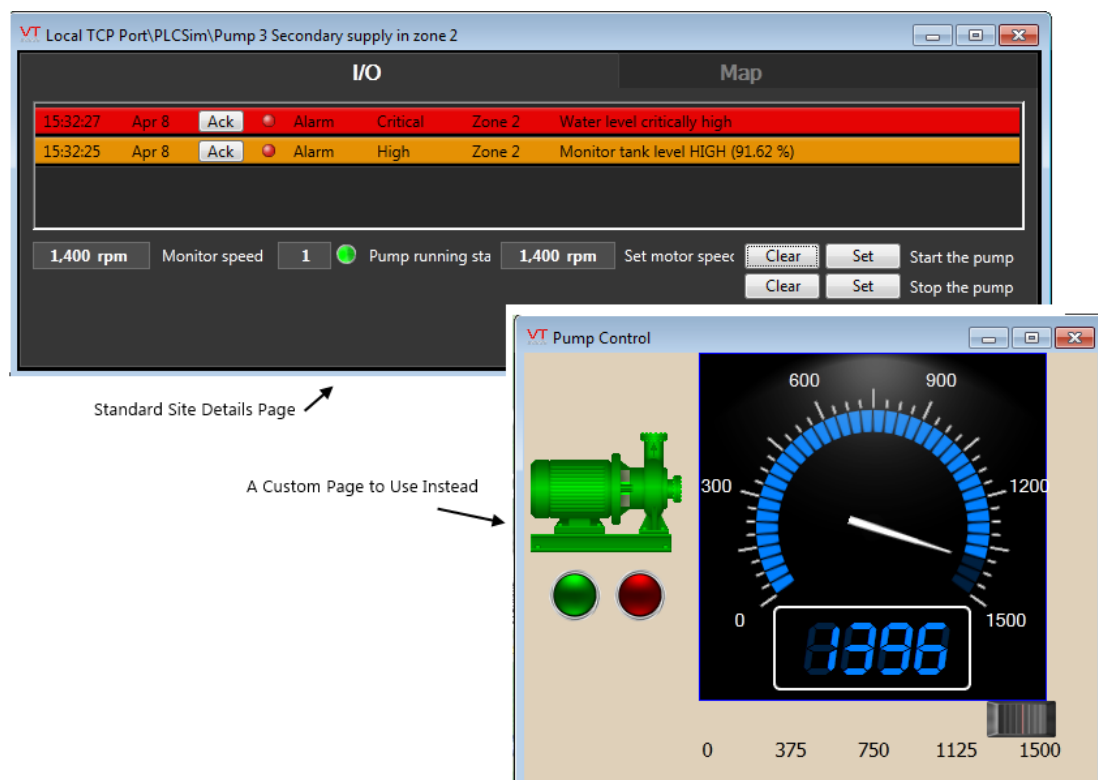
Caution: If your intent is for operators to open your custom page from the Site Details list, your page must have only one parameter, which is of type Tag and which can be linked to the site. Everything shown on that page should be held in a single Tag Widget, linked to the parameter.
If opened from a Site Icon, there are more options for parameter configuration.

Tip: While you can certainly create custom site details pages for Context tags, it's far easier to keep track of which page is for which type of site if you first turn your Context tags into custom types. (Design Your Own Tags).



The Manage Parameters dialog for custom site page.

In this example, SupplyPump is a custom type (site) with several child I/O tags.



In this example, everything that you see on the Pump Control page is drawn within a Tag Widget, designed to show child tags of the user-defined tag, SupplyPump.

There are two methods to specify a custom site details page:

Method 1: Configure your site tag:

1. Configure your site tag so that it has the parameter, CustomDetailsPage. This parameter is part of every station tag, and can be added to your Context tags with the button Add Site Properties in the Display tab. If you have already turned your context tag into a custom type, use the same button in the Manage Types page of the Application Configuration dialog.

New Context Properties

Settings

Property Name	Value	Comment
Latitude		Latitude (decimal degrees)
Longitude		Longitude (decimal degrees)
InitZoom		Map Zoom Level
CustomDetailsPage		Custom Details Page
CustomMapIconParam		Custom Map Icon
SiteListDisplay	0	Site List Display

Figure 11-10 If the Context has been turned into a new type, use the Manage Types dialog (Modify a Custom Type).

2. For each instance of your site tag, select the page to use.

Figure 11-11 Selection of a custom details page. The page must take a parameter matching this tag type.

Method 2: Configure the Site Summary or Site Icon widget that displays your site tag.

1. If your site is linked to a Site Summary widget or Site Icon widget, then configure that widget to use the custom site details page.

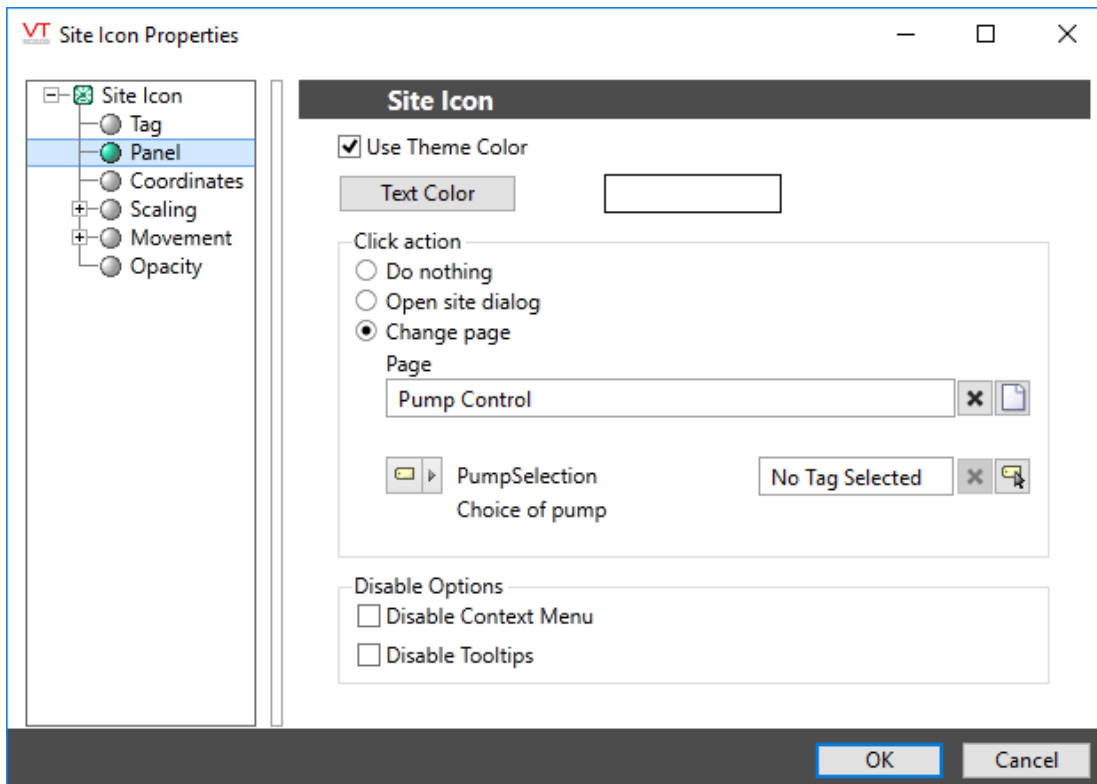


Figure 11-12 This option gives more control over matching tags to parameters.

Advanced Configuration:

The default page selection dialog has a filter for "all pages" or "site pages", meaning those that take a site tag as the first parameter.

You can add either of the following two properties in the Edit Properties page of the Application Configuration dialog to restrict this filter:

LimitPageListToSitesPages

System section. Set to 1 (TRUE) to prevent access to the "all pages" option in the page selection dialog for all users.

LimitPageListToSitesPagesIfRealmUser

System section. Set to 1 (TRUE) to prevent access to the "all pages" option in the page selection dialog if the user belongs to a security group.

Site Map

A Site Map is a dynamically-loaded map, primarily used to show the location of any of the site-related tags: Polling drivers, DataFlow RTU drivers, MultiSmart™ stations, MPE™ Duplexer stations, or MPE™ SC stations. You can also link a Context tag (or a user-created type derived from a Context tag) to a map by adding the properties "Latitude" and "Longitude" to the parameter list. A Site Map is a standard part of the menu, and is built into every automatically-generated lift station details page and into the Site Details page. While you can add extra Site Maps to pages within your application using the Site Tools palette or the Menu, it is usually not necessary or beneficial to do so.

In addition to the Site Map page, maps are also an optional part of the Sites page and the Site details page. If viewed in a Sites page, you can choose to add a legend to the map. This is the Site Legend widget, which can be drawn on any of your application pages.

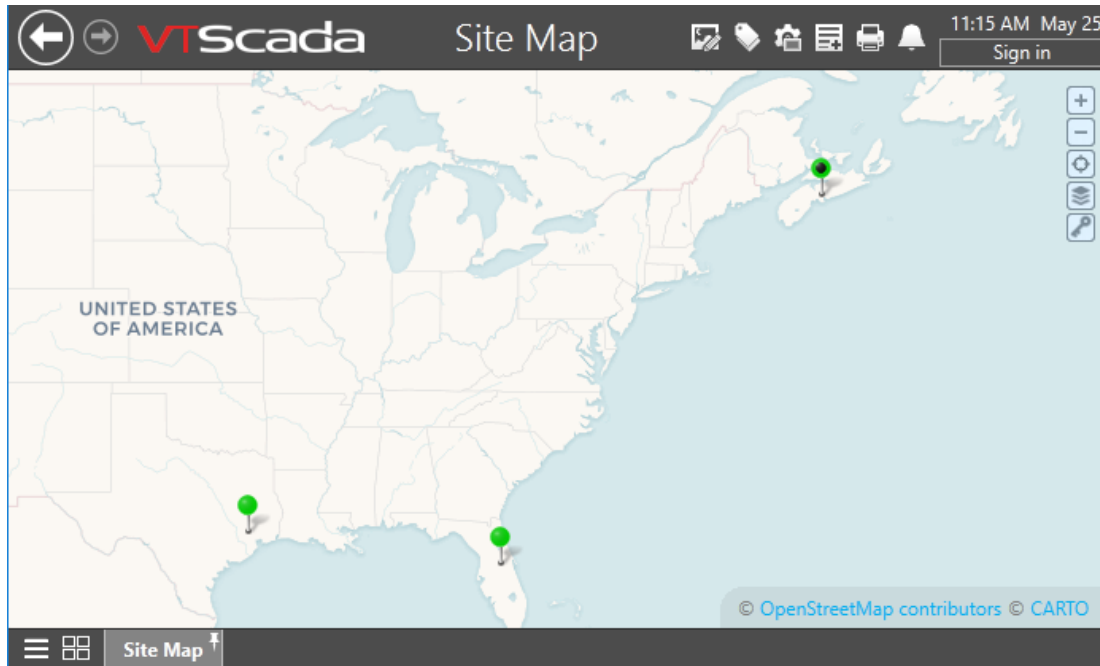
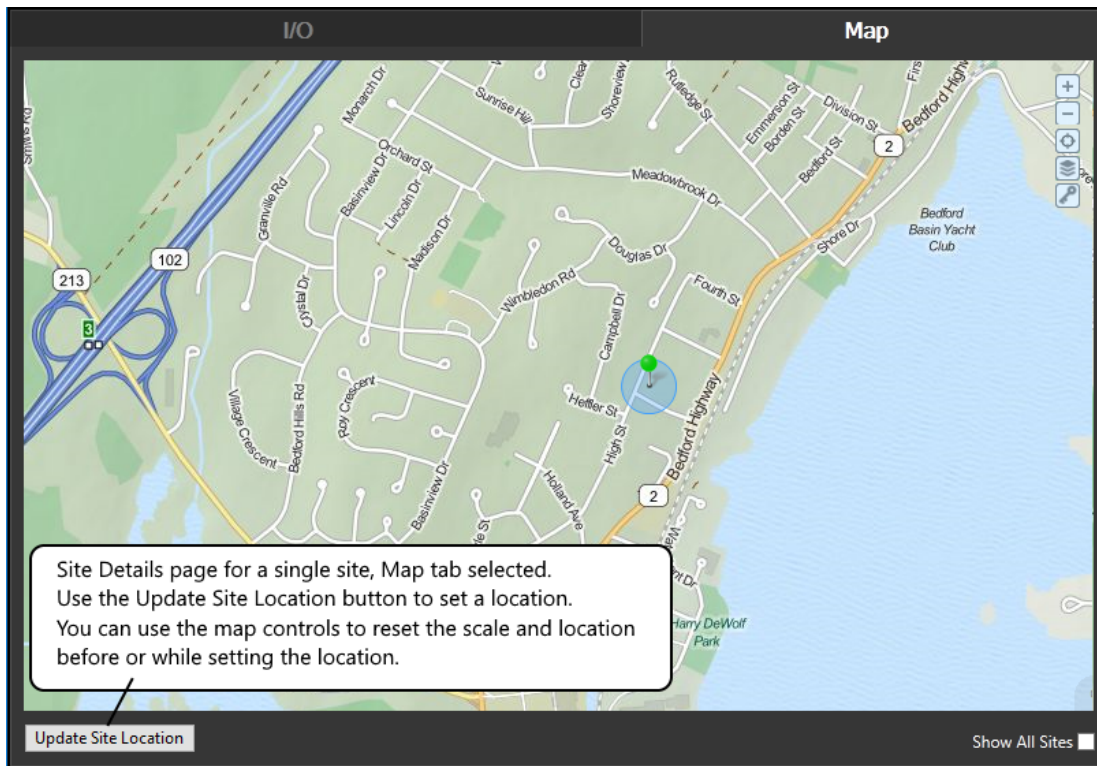


Figure 11-13 Use the controls at the upper right to zoom or to select the map style.

Set the location of a site:

To set the location of a site based on a Context tag, that tag must have Latitude and Longitude properties. See: Add Site Properties in the description of the Context Tag.

While you could open a tag's configuration panel and type in Latitude and Longitude values, it is usually easier to click a location on the map. In order to do so, you must be viewing a single site using the Site Details page, or by drilling down to the site in the Sites page. When doing so, an Update Site Location button will be provided at the bottom left corner of the map.

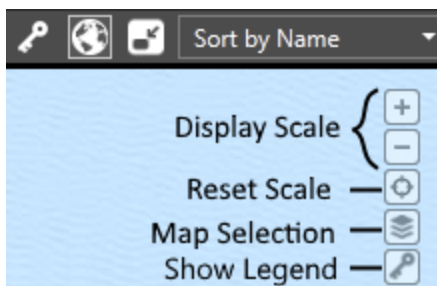


The currently selected site is indicated by a pulsing beacon. This is useful when the Show All Sites option is enabled, allowing more than the currently selected site to be seen.

Display Control

Use the scroll wheel of your mouse or the plus and minus buttons at the upper right corner of the map to change the scale. For each magnification level, a new set of map tiles must be downloaded to your workstation. A reset button is provided below the scale, allowing you to return quickly to the original display.

Note: You can set a default scale factor for each site tag in the Site Display tab of its configuration panel.



Sites:

Sites are displayed using markers, either the default pin or a shape of your own creation. Open the Site Map page for a specific tag to access the Update Location button to place or move the site.

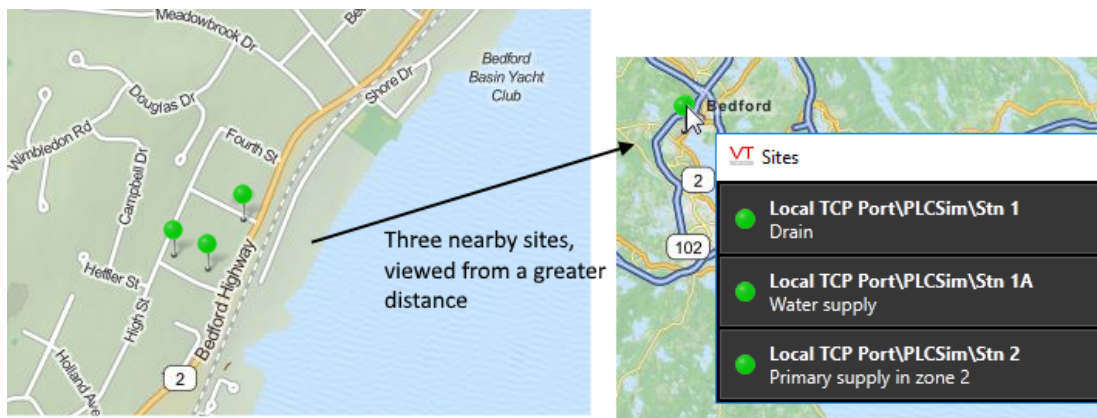
Any of the following tags can be a site when configured to have Latitude and Longitude values, and to be shown on a map:

- Context tags and user-defined types based on Context tags, if they have site properties.
- Polling drivers.
- DataFlow RTU drivers.
- MultiSmart™ stations.
- MPE™ Duplexer stations.
- MPE™ SC stations.

Overlapping Sites:

It is typical for sites to be near together. At a close scale, it is easy to tell one from another, but when viewing a larger area, only one pin can be seen.

If sites appear to overlap, a click on the pin icon will open a menu of sites near that location. Choose the site that you want to view in more detail.



Note: The setting SitesPageIconLoadTime is the time interval VTScada waits to see if a particular site icon has finished loading on a map. The default is 0.1. If this is too short, the following symptoms may occur. On a map with many site icons, if any icons overlap, clicking on the overlapping area might cause both of the site pages to open in addition to a site selection dialog. Increasing this timeout to 1 second can alleviate this problem.

Connectors:

Connectors are displayed as a line or pipe between two Sites. The color and style of the connector is configurable, but not the width. Connectors are displayed only when two sites with latitude and longitude values have been selected, and when the connector is configured to be shown. Only Context tags or user defined types derived from a Context tag can be connectors. A connector will remain visible if you zoom in to a map tile between two sites.

Because a connector is based on a Context tag, it can have user-defined properties and child I/O tags.

Slippy Map

The map is sometimes referred to as a "slippy map" because of the way it can be moved within its frame. Operators can pan and zoom to change the area displayed.

The map display is built of "tiles" - static images. At each higher magnification level, four times the number of tiles are used for a given area. As the operator moves the display or changes magnification level, new tiles must be downloaded. After being downloaded, they will be cached locally on your computer. These tiles are not distributed with VTScada, therefore your server must have an active Internet connection to be able to download tiles that have not been cached.

It is possible to switch from map view to satellite view, or to a different map provider. See the task list at the end of this topic.

Any instance of the tag types listed above can be located on a Site Map as a pin. Operators can click on the pin to open the associated Site Details Page. Instructions for create your own pin shapes are provided elsewhere. Refer to the list of tasks at the end of this topic.

The primary site associated with a map will be marked by an animated bubble. The head of the standard marker pin uses the same color-indication system as the Site Draw Drawing Method and described by the Site Legend widget.

(c) OpenStreetMap Contributors © [CARTO]

The copyright notices at the bottom corner of every map are each click-able links, allowing you to learn more about the source of the tiles you are viewing.

Map tiles are downloaded only once and cached on your hard drive. In a networked application, they are downloaded only the primary server. All other servers and workstations copy from there. If you believe that newer images are available, delete the files from sub-folders of C:\VTScada\Data\SlippyMapTiles. New tiles will be downloaded the next time you view the map.

Available controls for the map include: (*)

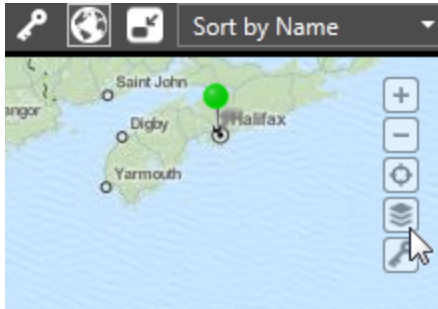
Control	Action
Show All Sites	A check box to enable or disable pin display for sites other than the root.
Update Site Location	After clicking this button, the next click within the map area will re-locate the site.
Magnification Control	(+) and (-) buttons. Provides a way to increase or decrease the magnification (zoom level) one step at a time.
Reset	A button below the magnification control slider. Resets the scale and location to the original view, as seen when the map was opened.
Map Selection	Opens a list of maps configured in your Setup.INI file.
Legend Display	Toggles the display of a legend describing the colors in a site icon (pin)
Click on any pin	Opens the Site Details page for that tag, as appropriate for the type.
Click and drag	Relocate the focus of the map.
Mouse-wheel	Zoom in or out.
Double-click	Zoom in, one step, at the location clicked.

(*) Some controls will not be available, depending on how the map is viewed.

Change the Map Source

The default configuration of the Site Map page is a basic map showing roads and towns and using tiles from Carto™ / Open Street Maps™.

Click the Map Selection tool, available on any map, to choose an alternate tile source. A pop-up will open, showing a preview of each. Your choice of map is stored with your user account and will be the map you see regardless of which workstation you use. Other users can make their own selection.



As an advanced feature, it is possible to change the list of available maps as you see fit. You can change to another source, including:

- Any other third-party provider of map tile images.
- Satellite image tiles rather than map tiles.

You are free to edit your Setup.INI file to change map sources, add new ones, or remove options. Note that changes to Setup.INI do not take effect until VTScada is restarted. Map sources are stored in sections headed [SlippyMapRemoteTileSourceN] where N must be a number starting at 1 and increasing consecutively for each source.

Caution: Do not edit tile sources from CARTO, unless you are removing those sources. CARTO maps are supplied by licensing agreement and include an API key that is unique to each VTScada installation. API keys cannot be shared.

Note: To disable all maps, edit your Setup.INI file to add a comment character (a semi-colon) to the beginning of every line in every SlippyMapRemoteTileSourceX section.

In all cases, the images that make up the map (or satellite view) are square tiles, downloaded to the folder C:\VTScada\Data\SlippyMapTiles\MapTypeN where N is a number starting at 1 and matching the number for the SlippyMapRemoteTileSourceN property.

Within each folder, sub-folders are used for each zoom level, with the numeric folder name corresponding to the zoom level. Map tiles should use the .PNG file format.

If you change the URL for a map source, you must delete all existing tiles from the previous source. New map tiles are downloaded only when not found on your workstation.

If a URL is not going to be used because you have a static tile set on a server, a URL must still be provided. Use a placeholder such as "Localhost".

A copyright attribution should be included with the tile source. This will become a click-able link on the map.

Procedure:

1. Stop VTScada.
2. Using a text editor, open the file, Setup.INI
3. Search for [SlippyMapRemoteTileSource1]
4. Copy the entire section changing the 1 to the next unused number and editing properties are required.
(At time of writing this will be [SlippyMapRemoteTileSource5], but that may vary.)
5. Save the file.
6. Restart VTScada and run your application.

Default values for the second tile source:

```
[SlippyMapRemoteTileSource2]
URL1 = https://enterprise-a.basemaps.cartocdn.com/rastertiles/voyager/
URL2 = https://enterprise-b.basemaps.cartocdn.com/rastertiles/voyager/
URL3 = https://enterprise-c.basemaps.cartocdn.com/rastertiles/voyager/
URL4 = https://enterprise-d.basemaps.cartocdn.com/rastertiles/voyager/
Label = MapVoyagerLabel
APIKeyName = api_key
APIKeyValue = SASfaFFKUUVfARJM2LzyE7d1LkENJ6udok5vmwejZWSOe7glZ7zYSqJPLI3P3UP/
APIKeyEncrypted = 1
BulkDownload = 1
CopyrightURL1 = www.openstreetmap.org/copyright
CopyrightLabel1 = © [OpenStreetMap contributors]
CopyrightURL2 = https://carto.com/attribution
CopyrightLabel2 = © [CARTO]
Default = 1
BGColor = <FFC0C0C0>
Extension = .png
```

Note: Do not skip numbers when defining tile sources. If you have [SlippyMapRemoteTileSource1], [SlippyMapRemoteTileSource2] and [SlippyMapRemoteTileSource4], then only the first two will be available for users to select.

Troubleshooting:

- No map or satellite image is shown.

Check for typos in the setting of SlippyMapRemoteTileSourceN, where N is a number matching the selected map.

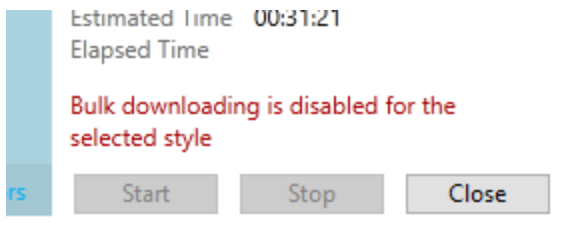
Ensure that the computer can connect to the Internet.

Bulk Downloads of Map Tiles

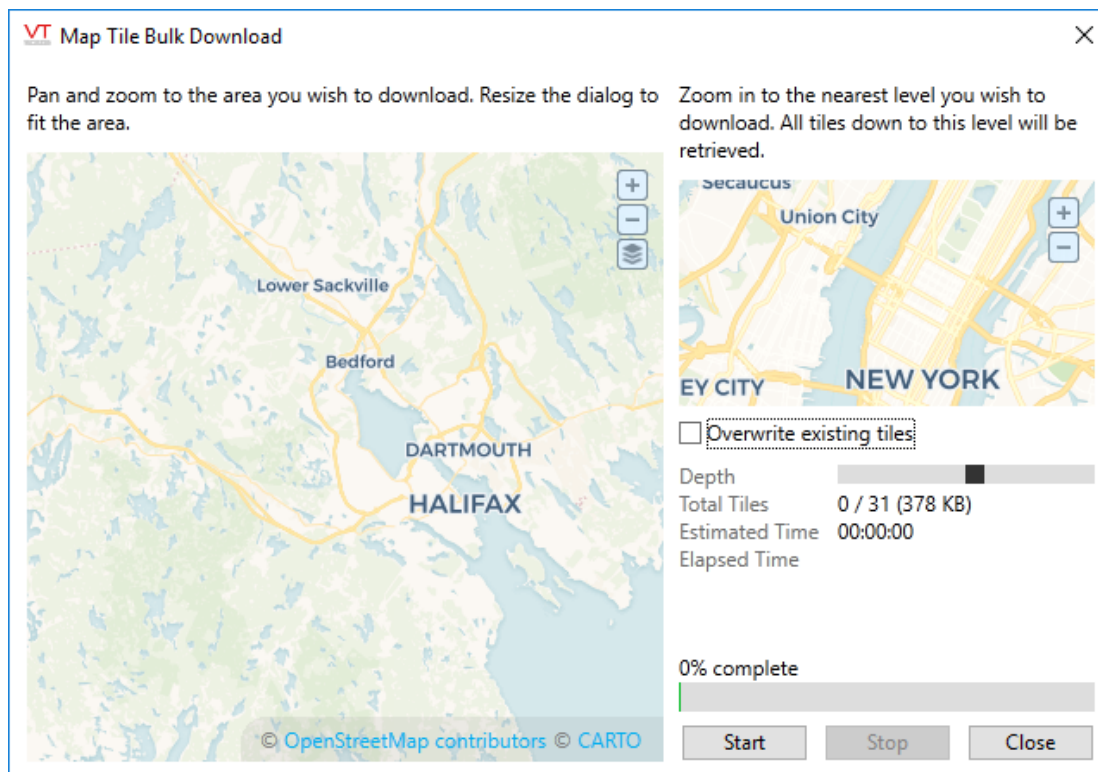
The tool to begin a bulk download of map tiles can be found in the Application Configuration dialog, Edit Properties page, under the Display tab (using the basic mode of the Edit Properties page).

Notes:

- The button is enabled only while your application is running.
- If the provider of your currently selected map style does not allow bulk downloads under their terms of service, you will be warned upon opening the tool. The start button will be disabled.



Caution: Depending on the area and the zoom level that you select, a bulk download can include nearly 450Mb of map tiles. The tool limits the area or zoom level or both so that no more can be downloaded in a single session.

Controls within the Bulk Download dialog

- The larger map (to the left) defines the area for which tiles will be downloaded.
- The smaller map (to the right) defines the zoom level.
While not displayed numerically, zoom levels range from 2 (far) to 18 (near). The zoom level of this map must always be greater than or equal to the zoom level of the area selection map.
(New York was chosen for the focus of this map because it contains detail down to

the nearest zoom level, which may not be the case in your selected area. This does not affect the area for which you will be downloading tiles.)

- Tiles will be downloaded for the entire area in the larger window, for all zoom levels from that shown in the area selection map down to that shown in the zoom level map.
- The depth bar indicates the range of zoom levels that will be included, where the furthest zoom level (2) is to the left and the nearest (18) is to the right.
- You can resize the Bulk Download dialog as needed to match the area selection window to the area of the map you are interested in.
- The display of Total Tiles tells you the number and disk space required. The estimated time will update during the download.

To download map tiles in bulk:

1. Ensure that the application is running.
2. Open the Application Configuration dialog.
3. Select the Edit Properties Page.
4. Ensure that the Basic Mode is selected, and that you are viewing the Display tab.
5. Scroll to find the Mapping section, then click the Bulk Download button.
The Bulk Download dialog opens.
6. Use the larger map (left window) to define the area for which tiles will be downloaded.
Resize the overall dialog box as required so that the larger map shows only the area you need.
7. Use the smaller map (right window) to define the zoom level to which tiles will be downloaded.
8. Review the information displayed beside Total Tiles.
9. Chose whether to overwrite existing tiles (refreshing the view if newer tiles are available) or not (reducing the time required for the download).
10. Click the Start button.
You may click Stop at any time to cancel the download.

Use Maps Without an Internet Connection

In many instances, VTScada will be running on a computer where internet access is forbidden or severely restricted. You can still use slippery maps by copying the map tiles that you need from a computer that has internet access to the computer that does not. The following method works because each time you view the same map again, VTScada will use the tiles that are in the cache, rather than pulling new copies from the Internet.

Note: Due to licensing restrictions, Trihedral cannot supply map tiles. The files must be downloaded.

1. On a computer that has Internet access and a copy of VTScada (possibly a trial version or VTScadaLight), view all map areas that will be required for your application. Ensure that you view all areas at all zoom factors.

2. Copy the folder and sub-folders C:\VTScada\Data\SlippyMapTiles[N] (where N matches the map style you are using) to the station that does not have Internet access.

For networked applications, ensure that the map tiles are copied to the primary server. All other workstations and backup servers will copy their tiles from the current primary server rather than downloading the tiles from the Internet. If using the advanced mode of the Edit Server Lists dialog, look for SlippyMapService.

The names of the folders and the map tiles match the area and zoom factor viewed and are not unique to your computer or VTScada in any way.

VTScada will expect the map tiles to use the .PNG file format.

Troubleshooting:

- The steps were followed, but the map is not displayed.

Ensure that the tile images are in the folder "Data\SlippyMapTiles" under your VTScada installation.

Ensure that you are attempting to view an area that matches what was saved in the map tiles.

Custom Map Icons

You can create your own custom map icons (pins). For example, you may wish to provide visual clues to help operators identify sites when more than one is displayed on a map.

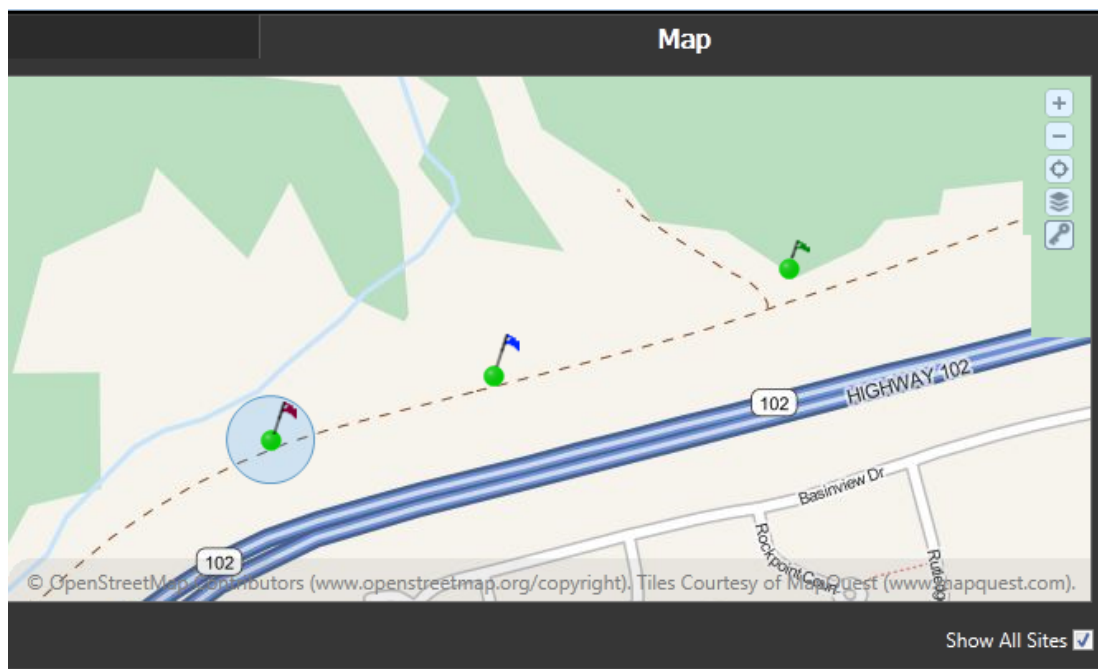


Figure 11-14 Custom map icons showing flags rather than pins.

Each icon is a Tag Widget that you create. To function as a map icon, various parameters will need to be set and mapping-related VTScada widgets will need to be included. A pulse beacon will be added automatically for you. When an operator clicks your icon, it will open a site page (or your custom page). You do not need to add a hotbox.

Note: VTScada will not add animation to the colors within your icon. If you want the functions of a Site Icon Widget, add one to your custom map icon. Be sure to edit the site icon widget's properties to use Linked Tag Properties. Alternatively, you could add an Alarm Priority Icon, or you could write expressions that will control the color of shapes within your icon.

General Steps to Create a Custom Map Icon:

1. Create a new Tag Widget.
2. When prompted to select tag types, choose only the types that this icon will be used with.
For example, Context, Polling Driver or your user-defined types.
3. When prompted for a name, choose one that will help you remember what the widget is for.
4. Choose whether to keep the options of including operator interaction features and the Tag Icon Marker.
5. Add the graphics that you want for your custom icon.
See the following list of key details.
6. Instruct your site tag to use that widget instead of the built-in pin using one of the following methods.

Use Your Custom Map Icon

There are four ways to select the custom map icon:

Note: After changing the selected icon, you may need to refresh the map or update a site's location before the new icon will load in place of the old.

To use the icon for a specific site that has display / site properties.

- a. Open the tag's configuration folder.
- b. Within the Display tab, choose the custom map icon widget.

Note: this assumes that, when creating the widget, you configured it to be a tag-widget for the type of site you are now trying to configure.

If the site is a custom type, ensure that the type definition includes site parameters, including Custom Map Icon.

To use the icon for a group of sites that are the children or grandchildren of a specific Context tag:

- Ensure that the parent tag has been assigned the site display parameters.
- Ensure that the icon widget can be linked to all the child tag types.
- Set the Custom Map icon property to the name of your widget.

To use the icon for all instances of a given type of site

****** For the following, note that the type name for a tag may differ from the name displayed in the Tag Browser. See:

Tag Names in Code

Similarly, widget names in code will match the file name of the widget, and may differ from the displayed name.)

- Use the Basic Mode of the Application Configuration dialog to Add a Property to the [System] section of your Settings.Dynamic file.
- Name the property after the *type* of tag you are configuring by putting the type name in front of the keyword, "MapIconName".
For example, `PollingMapIconName` or `ContextTagMapIconName`.

- c. Set the value of the property to the widget name
For example `MyMapIconsModuleName`

To use the icon for all sites of all types:

- a. The widget must have been named and therefore stored in a file name, "CustomMapIcon".
Note that this is the *name* of the widget, matching the file it is stored in. This is not its *title*.

Key Details:

Image or Shape:

You may wish to keep the following details of the default pin shape in mind while creating your new icon:

The default icon is approximately 30x30 pixels in size.

It uses a pin image that has a transparent background.

Pin Center:

The center of the new site icon's bounding box will be used as the pin location when shown on a map. The center's location is calculated using the bounding box of all graphics that make up the whole.

Pulse Beacon:

Automatically a part of map icons.

Site Icon:

If the widget is to indicate an alarm, or if the operator is to be able to click your custom widget to open a site map, then include a Site Icon. Within the properties dialog of the Site Icon, change "Tag" to "Linked Tag Property".

Parameters

Give your widget the following parameters to take advantage of all the available features. Parameter names need not match those shown here, but you are advised to use these names for clarity and simplicity. What does matter is the order.

Not all parameters need be created, but none can be skipped from the middle of the list. If your widget is to have a ZoomLevel parameter, it must also have the three preceding parameters.

DisableTrend

(Status) Should exist and be selected using the Values tool in the Widget Properties ribbon. This will prevent the HDV window opening when an operator clicks the icon to open a map.

DisableNavigation

(Status) Should exist if you want to allow developers to choose whether operators can right-click the widget to open the pop-up tools dialog.

DisableTooltip

(Status) Should exist if you want to allow developers to choose whether operators will see a tooltip with the tag's name and description when they hover the pointer over the widget.

ZoomLevel

(Short with allowed values ranging from 2-18) With this parameter, your widget can know what the current zoom level of the map is because VTScada will automatically set its value. The value, "2", corresponds to the furthest out that an operator can zoom on any map and "18" is the closest in.

A possible use for this information is to set the opacity property of component parts to use ZoomLevel in an expression that will become zero, thereby hiding the icon, or portions of it, when the operator has zoomed past a given value.

ConnectorAngle

(Double with allowed values ranging from 0-360) If your site is configured as a Connector, this is the angle at which the connector is drawn, from the start site to the end site.

ConnectorColor

(Color) If your site is configured as a Connector, this the color of the connector.

Example:

1. Open the Idea Studio
2. Click the File button, then New
3. Select Tag Widget.
4. Provide a name for the widget.
For simplicity of use, avoid spaces.
5. Make note of the name.
This is the file name, and therefore the module name that you must provide when using the icon. You may change the title of this widget later, but the file name will remain the same and will continue to be the module name.

Note: Note: If you have previously created a page or widget of the same name, then the new file name will have a "0" or "1" appended to the end.

6. Select all the types of site for which you might use the map icon (Polling driver, Station tag, custom types...).

The new widget will be created, and will open in the Idea Studio.

7. Delete the Tag Icon marker (yellow diamond).
8. Ensure that the Widget Properties ribbon is open.
9. Click, Manage Parameters in the Widget Properties ribbon.
10. In the Parameters tab, click Add.
11. Set the name to "ZoomLevel," and the type to "Short".
Any of the numeric options would work, but short is the most appropriate.
A description is recommended.
12. Click OK to save the new parameter and close the properties dialog.

13. Add the image(s) that you wish to use for the pin.
 - For reference, the Site Icon marker is 14x14 pixels. A Pulse Beacon will extend to 82x82 pixels. Your pin should be somewhere within this range of dimensions.
 - You may wish to add a Site Icon to the custom icon pin. This provides a visual indication to operators, showing the status of the site and also allows them to click on the pin to open the related Site Page. Note that the Site Icon's data source must be linked to the Drawn Tag.
14. Adjust the position of all objects so that the center of the overall bounding box of all objects coincides with the location that you wish to use as the pin-point. The location of the objects within the Idea Studio does not matter for custom map icons.

Put the new icon to use as described earlier in this topic.

Site Map Widget

VTScada has built-in links to site maps, both within the menu, and within Site Pages and Site Details pages. These are usually sufficient for most applications, but if you wish, you may add a Site Map to any page.

To add a site map to a page:

1. Open the Idea Studio and the page to which you intend to add a map.
2. Open the Site Tools palette (found within the Widgets palette section).
3. Locate the Site Map widget and drag it onto the page.
4. Provide parameters as appropriate. Parameters are described later in this topic.

Note that the map is drawn within its own window, which will have a higher z-order than the page it is being placed on. The result can be loss of part of the VTScada header or footer if the map is larger than the page.

Site Map Properties:

Parent

The station tag instance (and any child tags that also have valid latitude and longitude values) that is to be the primary target of this site map. All sites shown will be marked by a pin, but the root site will be further indicated by an animated dot.

Show All Sites

Choose whether this map instance should show only the root site, or if all sites visible in the map area should be shown. In either case, the pin marking the root site for the map is the only one highlighted by an animated bubble.



Use Theme

Choose whether the borders of the map should use the VTScada color theme

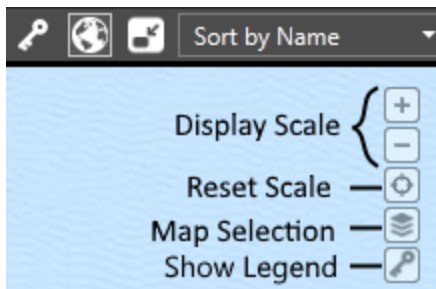
Text Color

Specify the color to be used for the labels below the map.

Disable Zoom Buttons, Disable Reset Button, Disable Style Button, Disable Legend Button

Use the listed controls to hide any of the tools labeled in the following image, which would otherwise be available on your map.

"Zoom Buttons" refers to the controls for the display scale. "Style" refers to the tool for map selection.



12 Log, Note, and Report

All reports are based on three fundamentals:

A selection of data, collected for the purpose of a report.

This includes raw data from any I/O tag, processed data from VTScada analytic tags, driver quality statistics, alarm history, and operator notes. It can also include data pulled from outside sources.

Data is logged using Historian tags, which offer several configurable options.

A process that calculates, filters and summarizes the raw data into meaningful information.

Sometimes you will want to see raw data as it was stored, but most often you will want to filter and summarize the information to see various statistics.

VTScada's Historian Manager provides several tools to process information as it is retrieved. Analytic tags will do that processing before storage.

The presentation of the information in a way that is appropriate to inform a target audience.

Options include the Historical Data Viewer (trend graph), the Reports Page where you can build reports, and a variety of tools to extract VTScada's data for processing and display in third-party programs.

Reports are not only those things created by report generators. This discussion takes a much broader view, where report generators are only one tool of many available to help you turn raw data into useful knowledge. Anything that achieves these three fundamental components can be considered a report. If your goal in creating a report is to take information that's on the screen and put it onto paper, why use a report generator when you can just click the print button?

Choose Your Reporting Solution

Your choice of reporting solution will be influenced by the trade-off between several factors.

- Whether the data for the report is found entirely within VTScada or if it is a mixture of data from several sources.
- The amount of processing required to turn the raw data into meaningful information.
- The amount of formatting required to present the information properly in your reports.
- The skills and knowledge at your disposal. Any given solution may require knowledge of VTScada, VTScada's scripting language, VBA, SQL, and third-party programs.
- The expense you are willing to make for third-party programs that provide either extra features or greater ease-of-use for reporting.
- The time at your disposal for creating customized reports and learning new skills that may be required for a given reporting solution.

Integration With Third-Party Reporting Tools

You have several options for using third-party reporting tools with VTScada:

- VTScada provides a data query add-in for Excel. Use this to connect to your application data and build custom queries that can be further refined using Excel's calculation and formatting tools.
- If your license includes the Remote Data Access option, you can use any ODBC-compatible program to query VTScada using SQL selection statements.
- VTScada is compatible with XLReporter® from SyTech Incorporated. A datasheet is available from the Trihedral website: https://www.vtscada.com/wp-content/uploads/2014/10/Tech_VTScada.pdf
- VTScada is compatible with DreamReport® from Ocean Data Systems®. A datasheet is available from the Trihedral website: <https://www.vtscada.com/wp-content/uploads/2014/10/Dream-Report-Product-Overview-for-VTScada.pdf>

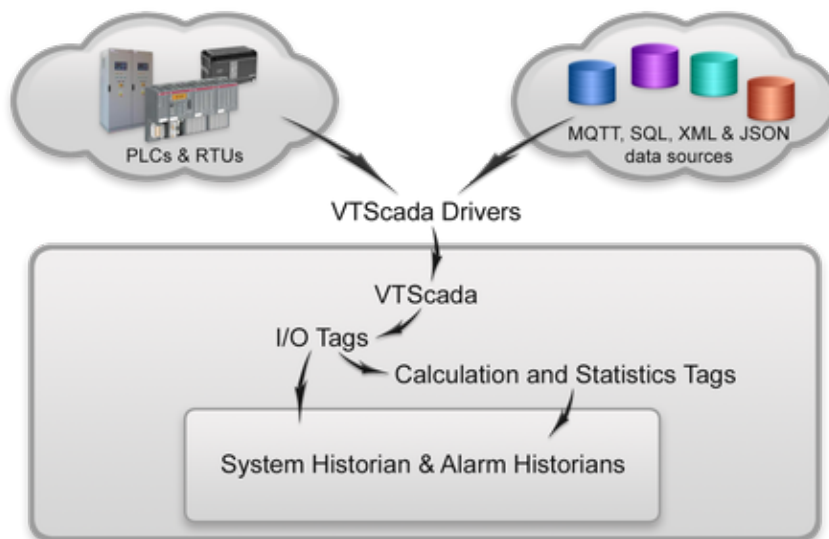
Collecting Data

In VTScada, data comes from tag values. At a minimum, you can count on the following three pieces of information being available:

- The name of the tag from which data is being collected.
- A UTC timestamp recording when each value was collected.
- The value of the tag.

Alarm history is similar:

- The identity of the tag triggering the alarm.
- A status describing the alarm transaction: triggered, normalized, acknowledged, shelved, etc.
- The UTC time stamp recording when the transaction occurred.
- The setpoint of the alarm and the value of the trigger at the time of the transaction.
- Further details about the transaction...



Values from status tags (Analog, Digital, Pump) are logged automatically. For certain other tags, you must add a logger. If a tag is not logged, then historical information will not be collected for any report.

Note: Logging isn't free. It costs CPU time and disk space.

Log everything that you need but think about your choices, especially the frequency at which tags poll and the deadband you set on analog values.

The frequency of data collection depends on the tag configuration. Status tags with a built-in Historian link write to the log when the value changes by an amount set in the tag configuration (deadband setting). Logger tags can write at a set interval or they can log on change (without the benefit of a deadband.)

The value being recorded depends on the tag to which the logger is attached. In most cases, this will be a simple measurement taken directly from a tag such as an Analog Status. If the data source for the logger is a calculation tag, then anything that can be calculated from the system (the sum of several other tags, an average over time, etc.) can be logged.

The logged data for each tag is stored in directories named after that tag. There will be one set of these directories for each Historian in the application, all of which are collected under a folder named DATA. (For example, C:\VTScada\MyAPP_AH\Data\History\[SystemHistorian]\.)

The data is stored using a proprietary format that is not published. You are advised to use the tools provided with VTScada to access the information.

As an alternative, you can configure the Historian tag(s) to save data to a 3rd party database such as Oracle or MySQL. If doing so, it is your responsibility to install and maintain the database program and data. Whether you use the VTScada data storage system or a 3rd party database, the primary tools for reviewing historical data are the History Data Viewer and the VTScada Reports Page.

Tip: Dealing with Daylight Savings Time:

VTScada records all data using UTC timestamps. In UTC, time advances at the rate of one second per second (barring leap-seconds) and if you were to plot your data against UTC time, you would see an uninterrupted line, assuming that your system is up and running, logging data.

Complications arise in places where daylight savings time (DST) is used. When DST starts in the spring, clocks jump forward from 2:00:00am to 3:00:00am. As far as UTC is concerned, data continues to be logged at the same rate, but if viewed on a graph that shows the local time, there will be a gap or a straight line showing no data for that hour. This is not a problem with VTScada. The gap exists because no time passed when the clock jumped forward, and therefore there is no data for that hour. Nothing can be shown.

Also, that day is 23 hours long, not 24, which may affect your reports. Again, this is not a problem with VTScada. According to your local clock, the DST-transition day had only 23 hours, therefore a daily report can include only 23 hours worth of data.

In the autumn, when daylight savings time ends, you can expect the same in reverse. It will appear that an hour's worth of data is over-written during the transition as the local clocks repeat that hour. That day is 25 hours long.

Rest assured that VTScada is faithfully and steadily recording your data, regardless of what your local clocks are doing.

Edit Data

Note: It is not possible to change or delete values logged by VTScada. View the original value by clicking the note symbol displayed beside the edited value.

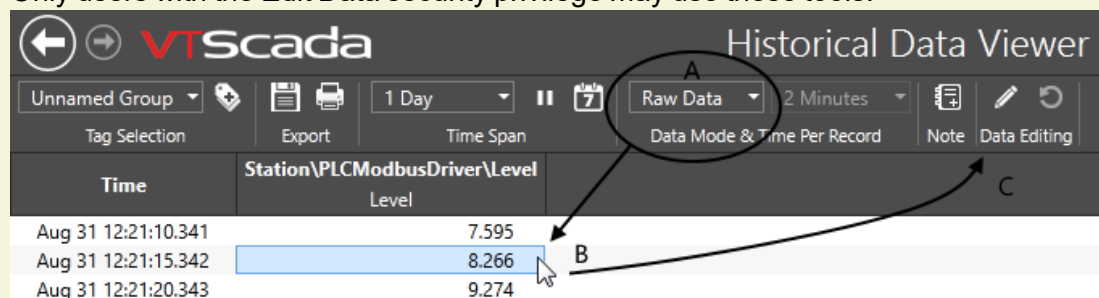
Authorized users can add or amend logged values by creating overrides. This may be useful for a number of situations:

- It may be necessary to adjust a single data point to replace an incorrect value.
- A broken instrument reports a series of bad values during a time range. This can happen when a sensor breaks. When a worker takes a manual measurement they need a way to replace the range of bad values with a single manual reading.
- In a situation where measurements are taken manually, a value might be obtained at 2pm but not entered until 5pm.
- After editing a historical value, a correction may need to be entered at a later time.
- After editing a historical value, the edit may have to be removed because the wrong value was changed.
- There may be a need to annotate a single value or a range of values.

How to Edit Data:

Override values are created or removed using the Grid View of the Historical Data Viewer.

Note: Data editing tools are enabled only when viewing the Raw Data mode, and when one or more records in the grid view are selected. Only users with the Edit Data security privilege may use these tools.



1. Open the Historical Data Viewer, with one or more tags (pens) selected.
2. Switch to the grid view.
3. Ensure that the data mode is set to Raw Data. (A)
4. To change an existing entry or to add a new entry, select one row. (B)
To change a range of entries to a single new value, use the shift key to select several rows.
(Is it not possible to use the Ctrl key to select a random set of rows.)

- Click the pen tool in the Data Editing tools. (C)
The Edit Historical Data dialog opens.

- Provide a new value and enter a note for auditing purposes.
See following discussion relating to the use of Start Time and End Time.
- Click OK.

The overridden value will be indicated by an exclamation mark (!) and a notebook symbol.

Aug 31 12:33:55.524	5.750 !	←
Aug 31 12:34:00.525	6.221	

If there is a notebook symbol but no exclamation mark, then either the original value was restored or a note was added without changing the value.

To create a new record, change the Start Time in the Edit Historical Data dialog after opening. If Start Time matches an existing value, that value will be overridden. If there is no existing record for the specified Start Time, then a new record will be created.

As an alternative to selecting a range of rows in the grid, you can set both a Start Time and an End Time in the Edit Historical Data dialog. The result will be the same: all selected values / all values between the start and the end, will be overridden with the new value.

Restore a Value

Original values are never lost as a result of data editing. They can be restored as follows:

- Do steps 1 through 3 in the instructions for editing to enable the tools.
(Previous set of instructions in this topic.)
- Select the value or range of values whose values are to be restored.
- Click the Remove Selected Data Override(s) tool.
The Revert to Original Value dialog will open.

4. Enter a note for auditing purposes.
5. Click OK.

The original values are restored. All notes are maintained, as is the history of edits made. As with Operator notes, there is no means to remove or edit any note after creation.

If the value at a selected timestamp exists only because it was entered manually using the Edit Historical Data dialog, then there is no logged VTSkada value to which it can revert. In this case, the most recent value (logged or overridden) prior to the selected timestamp will be assigned. The manually-entered row cannot be deleted.

Symbols for Edited Values

The value column in the grid view, or the value in the legend of the plot view may include any of an exclamation mark (!), a question mark (?) or a notebook icon. Here's an example from the grid view showing all three:

12:05:14.253	11.380 ?
12:05:19.254	11.000 !📓
12:05:24.254	11.441 ?

The question mark indicates that the Questionable Data flag was set at the time the value was recorded. This does not mean that there is anything wrong with the value, only that the flag was set.

An exclamation mark beside a notebook symbol indicates that someone used the Data Editing feature to set this value manually. A note is always created when a value is edited to record when it was changed, what it was changed from, and who changed it. The ability to edit values to correct errors or to add values that were measured manually is an advanced feature, protected by a security privilege.

It is also possible to add notes without changing the value, in which case there will be a notebook symbol without an exclamation mark. You must have the Edit Data privilege to create these notes, but you do not need any extra privileges to click the notebook symbol and read those notes. Operators who do not have the Edit Data privilege can still create notes.

An exclamation mark without a notebook symbol indicates that the Manual Data property of this tag is set.

Historian and Logger Configuration

VTSkada uses its own storage system for saving data. It includes all the tools you need to extract data when and how you wish.

The task of collecting and saving data is managed by Historian tags. There are at least two in every application (SystemHistorian and SystemAlarmHistorian) and you can add more if required.

By default, information is written to a proprietary storage format, located in the Data folder of your application. You can choose to store your data in a 3rd party database such as Oracle®, PostgreSQL®, SQLite®, MS SQL Server® or MySQL®, but this is not a common configuration. Note that VTSkada will use its own data storage system within these databases, therefore you should plan to query and retrieve your data using the VTSkada tools, regardless of how and where it is stored.

Tip: Dealing with Daylight Savings Time:

VTScada records all data using UTC timestamps. In UTC, time advances at the rate of one second per second (barring leap-seconds) and if you were to plot your data against UTC time, you would see an uninterrupted line, assuming that your system is up and running, logging data.

Complications arise in places where daylight savings time (DST) is used. When DST starts in the spring, clocks jump forward from 2:00:00am to 3:00:00am. As far as UTC is concerned, data continues to be logged at the same rate, but if viewed on a graph that shows the local time, there will be a gap or a straight line showing no data for that hour. This is not a problem with VTScada. The gap exists because no time passed when the clock jumped forward, and therefore there is no data for that hour. Nothing can be shown.

Also, that day is 23 hours long, not 24, which may affect your reports. Again, this is not a problem with VTScada. According to your local clock, the DST-transition day had only 23 hours, therefore a daily report can include only 23 hours worth of data.

In the autumn, when daylight savings time ends, you can expect the same in reverse. It will appear that an hour's worth of data is over-written during the transition as the local clocks repeat that hour. That day is 25 hours long.

Rest assured that VTScada is faithfully and steadily recording your data, regardless of what your local clocks are doing.

Note: When the connection to an Historian is built into a tag, that tag's values are written with every change. The use of a deadband on analog-value source tags is recommended to avoid logging system noise, and in fact is present by default in the tags, I/O and Calculations & Analog Status. See: [Optimize Your Configuration](#) for related steps that you should take to avoid problems when logging data.

If using tag types that do not have a built-in connection to the Historian, you must add a Logger tag to serve as a go-between, passing data from the collection tag to an Historian.

Note: The simulation runs at a speed that provides feedback quickly as you build your application. But it won't look very realistic when plotted. Before proceeding, switch to the Training Simulator application, open the Sample Pages folder, open the Demo Pumping Station page, and change the speed factor to 15. Switch back to the Training App for the exercises.

Historian tags have been designed to provide dependable, efficient service with minimal configuration. Refer to [Historian Tags](#) for configuration options.

Two additional configuration options for your Historians are done using the Edit Server Lists panel of the Application Configuration dialog:

- Redundant storage locations. In a multi-server application, you can (and should) configure backup servers. Data will be copied automatically, and backup servers can take control when the primary is offline.
- Load distribution between multiple Historian tags. By adding more Historian tags, each running on a separate server and saving to separate databases, you can ensure that the data logging requirements of extremely large systems does not exceed any one server's I/O capacity.

System Historian versus System Alarm Historian

Both are Historian tags and both write data to long-term storage. In general, it is best to keep process I/O data separate from the alarm history for the following reasons:

- Retrieval speed.
If alarm data were mixed with process I/O data, it would take longer to retrieve. The stored history of alarms and events tends to be very small compared to I/O process history.
- Local storage.
For applications that run on multiple workstations, every workstation will have its own copy of the alarm database so that alarm information remains available in the event that the workstation becomes separated from the rest of the network. This is not a good idea for your stored process I/O data. Therefore, the System Alarm Historian is configured to maintain a local copy of its database on all workstations, while the System Historian is not.
(Caution: do not confuse this concept with that of backup Historians.)

Historian Data Storage

Unless otherwise configured, Historian tags will save data to a VTScada proprietary database system within the application's data folder. These files are designed for use solely by VTScada. Values can be read only by VTScada and cannot be modified by any means without damaging system integrity.

Caution: Do not use the same storage location for multiple applications.
Do not attempt to use an external USB drive for your Historian data.
Do not configure your Historian to write to a storage device located elsewhere on your network.
For all the above, you run the risk of data loss or corruption. Your safest course is always to do initial data storage on the machine that is the current server, using the proprietary VTScada format. Rely on backup servers for remote storage, including the use of a third-party database.

Note: While the properties described in this topic can be set in the Edit Properties dialog, you are strongly advised to use the configuration fields of the Historian tag instead.

Storage Requirements

As a rough guideline, 1000 data changes per second will require 1500 GB per year. It is wise to use deadbands on I/O tags and set polling frequency as large as safely possible to minimize storage requirements. Also, do not save information that you will never need to retrieve in any report. Logging is enabled by default on every I/O tag, but may not be required in every case.

Storage Location

You can configure an Historian tag to send its data to an alternate location, or to a third-party database.

Compatible databases are as follows:

- SQLite, using ODBC driver version 0.86 or later
- Oracle version 10g or later. Requires the following Oracle privileges:
Create any index, Create any sequence, Create any table, Create any trigger, Create session, Create user, Drop any table, Drop user, Insert any table, Select any table, Update any table
- SQLServer version 2000 or later
- MySQL using the MySQL ODBC Connector versions between 5.1.6 and 8.00.11.00¹. Requires the following MySQL privileges:
Select, Insert, Update, Delete, Create, Drop, Index
- PostgreSQL version 9.4 or later

If using a third-party database, you do not need to create tables, as this will be done by VTScaDa. It is strongly recommended that a specific database and user be reserved for use only by the VTScaDa Historian.

VTScaDa stores data in a form that has been optimized for performance, reliability and synchronization. Informal testing shows that the native VTScaDa Historian can store data at a rate that is roughly an order of magnitude faster than SQL. You are advised to use an SQL database only on a backup server. (See notes later in this topic under Configure Alternate Data Stores.) If you are using a third-party SQL data store, then when creating queries, you will find that it is easier to use the VTScaDa ODBC Server for direct SQL access because it has been designed to handle the complexities of the data structures for you.

Note: If using an Oracle database as a target for historian data, the database must have a "tablespace" named "Users", which is not present in a new Oracle DB. This is used as the default tablespace for the user created by a historian tag, and is where all the historian tables are stored.

All tables are created with the "owner" (or schema) named after the tag name. So, given a historian tag of "SystemHistorian", then all tables are created with the owner SYSTEMHISTORIAN and have a name such as "SYSTEMHISTORIAN.FRIENDLYTAGNAMELOOKUP".

If you are using the VTScaDa database system for data storage, then there is no need to set any value for the Historian tag's Storage Type property. You might set the Storage Location property if you want to save the data somewhere other than the default Data\History folder of your application. Writing across a network to another machine is strongly discouraged.

If writing to a third-party DBMS, the storage location should be either a connection string in the form:

```
Driver=ServerBrand;Server=ServerName;Database=DBName;Uid=user;Pwd=pwd
```

Alternatively, it may be a reference to a DSN, as follows:

```
DSN=MyDSNName
```

¹ Versions of MySQL following 8.00.11.00 appear to have two bugs that affect its compatibility with VTScaDa (Bug #108434 and Bug #108452) For the time being, you are cautioned to avoid MySQL versions later than 8.00.11.00.

Use of a DSN is recommended to avoid having the user ID and password for your database visible in the Historian tag's configuration.

Prerequisites:

If modifying the location to a new folder on your computer, then the only prerequisite is that the folder exist on all Historian servers. This should be configured before commissioning the application.

- In general, do not specify a network location other than the hard drive of the Historian server(s).
- To configure for redundancy see Client / Server Configuration.
- To configure for backups see Backups

If configuring VTScada to use a third-party database, all the following must be in place:

- The database program must be installed and running.
- An ODBC driver for your program must be installed. VTScada can work with either 32 or 64 bit ODBC drivers unless you are running the 32-bit version of VTScada.
- You may use either a connection string or you may use a DSN, created using the Microsoft ODBC Administrator tool.
- A database must be created for use by VTScada.
- A user account must be created in your database for use by VTScada. This account must have CREATE and WRITE privileges in the database, so that VTScada can create tables and write data to those tables.
- Choose whether to set your Historians' storage type to ODBC or ODBCSingleTable.

This choice affects how your data is stored, and therefore how you can query it. See notes at the end of this topic and also Query a 3rd-Party DBMS

- After completing all configuration steps, ensure that the database is running and then restart your application.

VTScada will create the tables required to store data if you do not restart, but may not create support tables such as 'friendlytagnamelookup'.

It may take up to 30 seconds for all tables to be created.

Example 1:

For example, to direct the System Historian to use a SQL Server database, identified by a connection string, the Historian tag's data store properties would be set as follows:

StorageName: SystemHistorian

Type: ODBC

Storage Location: Driver=SQL Server-

;Server=ServerName;Database=DBName;Uid=user;Pwd=pwd

Example 2:

Connecting to a MySQL database.

A database named 'vtscada' has been created.

A user named 'VTS' has been created.

'VTS' has been granted CREATE and WRITE privileges on 'vtscada'.

The ODBC connection is created as shown. The user and password are for the MySQL database, not a VTScada account.

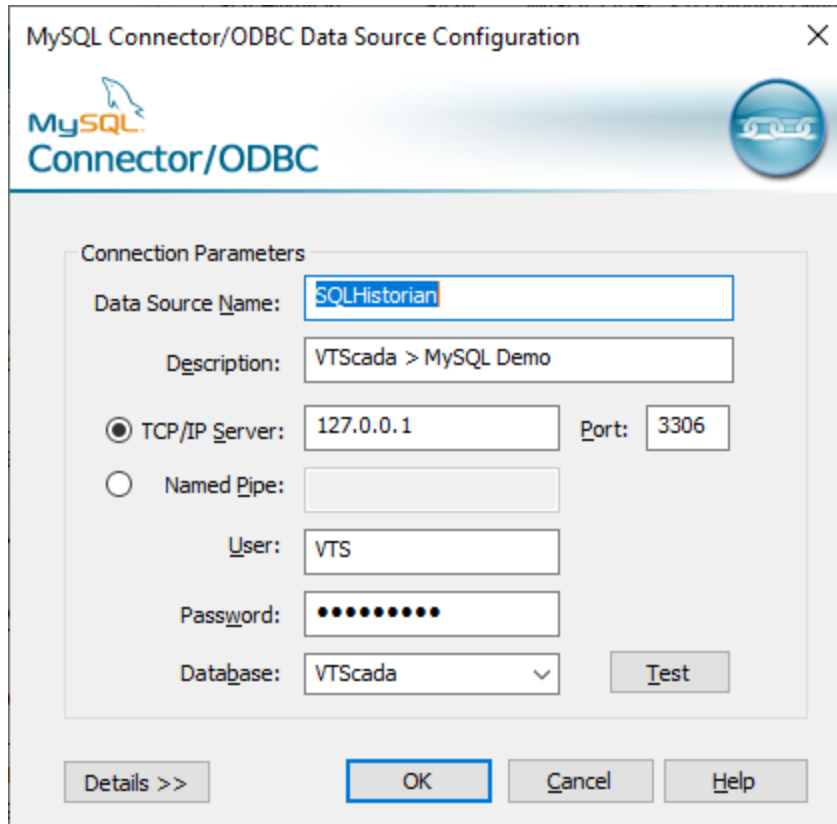


Figure 12-1 Sample DSN creation, using the MySQL connector within the Microsoft ODBC Administrator tool.

The VTScada System Historian is configured as follows:

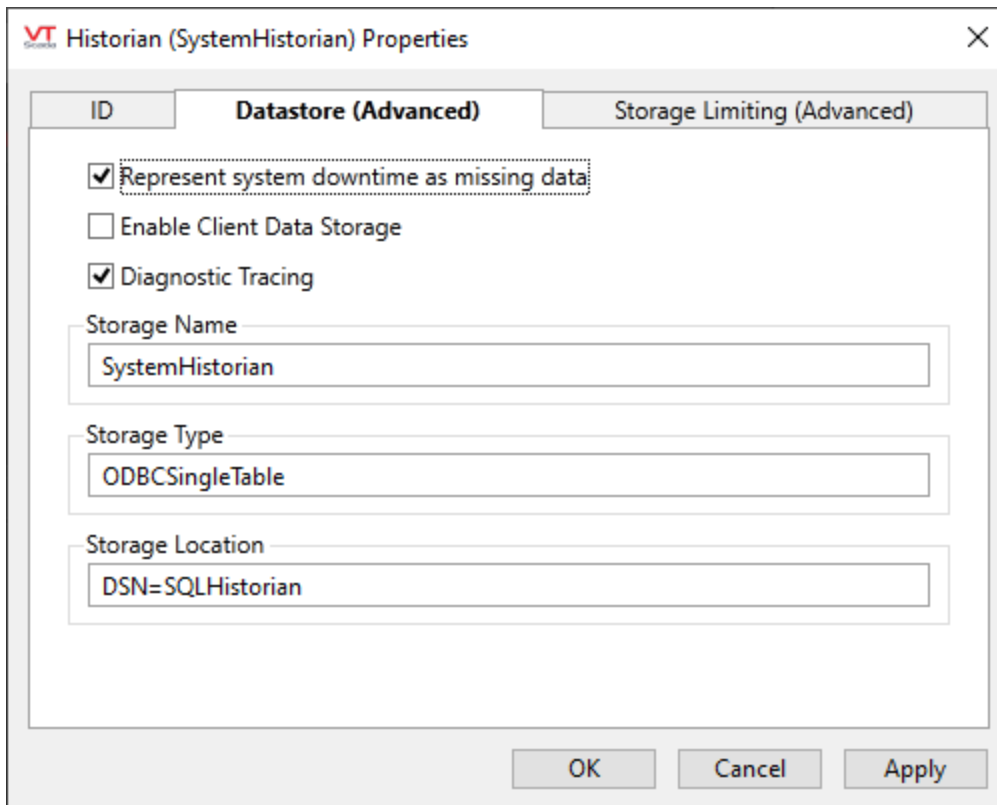


Figure 12-2 The storage type is ODBCSingleTable for ease of data access. The DSN matches the previous example image.

Tip: When using ODBC Historians, it is suggested that you specify the Storage Name parameter. This will be used as the schema name in the database. If left blank, the unique id of the Historian tag will be used as the schema name rather than its short name. Note that the unique id's length and complexity may cause difficult for some database programs. The Storage Name must be unique for each Historian.

Configure Alternate Data Stores

One Historian in an application that runs on a primary and backup server. Each server has its own distinct database.

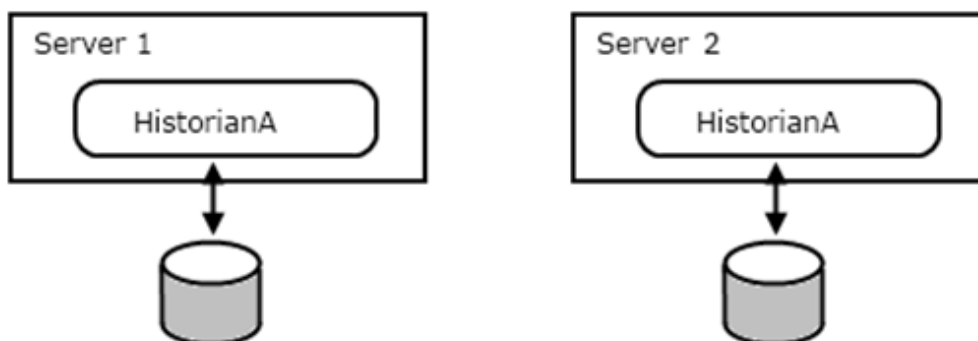


Figure 12-3 One Historian with two separate databases

To record to both the VTScada data store and a third party ODBC database, or to save to the C: drive on one machine and the D: drive on another, there must be at least two VTScada workstations in your system. You can then configure the Historian tags using parameter expressions that use a different configuration on each workstation. In the following examples, "SQLBackup" is the name of the computer where the ODBC database is installed. Parameter Expressions are added to the configuration fields of the Datastore tab of the System Historian as follows. These expressions check the workstation name using "WkStaInfo(0)", and configure for the ODBC database if the workstation is SQLBackup, or use default settings otherwise, by returning "Invalid". Note that the test for workstation name is case-sensitive.

StorageName:

```
WkStaInfo(0) == "SQLBackup" ? "SchemaNameForDB" : Invalid
```

Type:

```
WkStaInfo(0) == "SQLBackup" ? "ODBC" : Invalid
```

Storage Location:

```
WkStaInfo(0) == "SQLBackup" ? "Driver=SQL Server-  
;Server=ServerName;Database=DBName;Uid=user;Pwd=pwd" : Invalid
```

Caution: Data store configuration should be done before the application begins to collect information. Upon changing any Historian's storage type or location, that Historian will lose access to previously collected values.

ODBCSingleTable - Single Table Schema Structure

When your Historians' storage type is set to ODBCSingleTable, tag values are stored using one of the schemas described here. Note that while the table name is the value shown in the column **Table name**, in your database that name may be prepended with other information including (but not limited to) the schema name for each Historian tag.

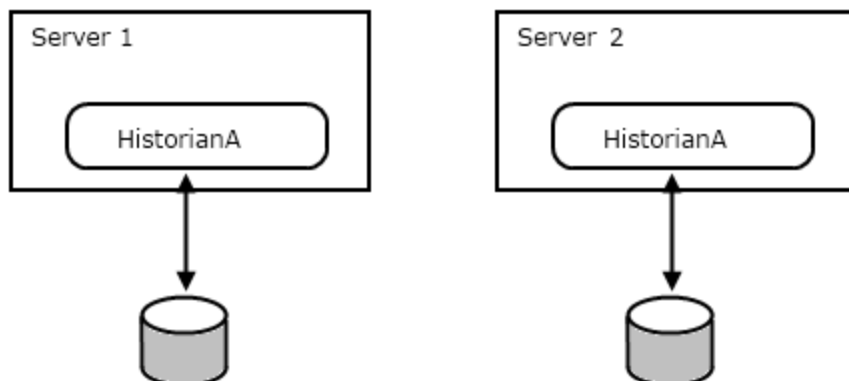
Historians and Multiple Servers

In most smaller systems (ranging up to a few thousand tags) you may not need to have more than a single Historian tag.

Reasons why you might consider adding one or more additional Historians include:

- Load distribution in larger systems - each Historian can be configured to save its data to a separate server, thus reducing load on any individual server.
- Alternate storage locations for selected tags - you may configure each historian to save to a different directory, or to a different storage format.
- Alternate configurations for data limiting - you may choose to limit the amount of data stored for some tags, but not limit data collected from others. Each Historian can have its own configuration.

One Historian in an application that runs on a primary and backup server.
Each server has its own distinct database.



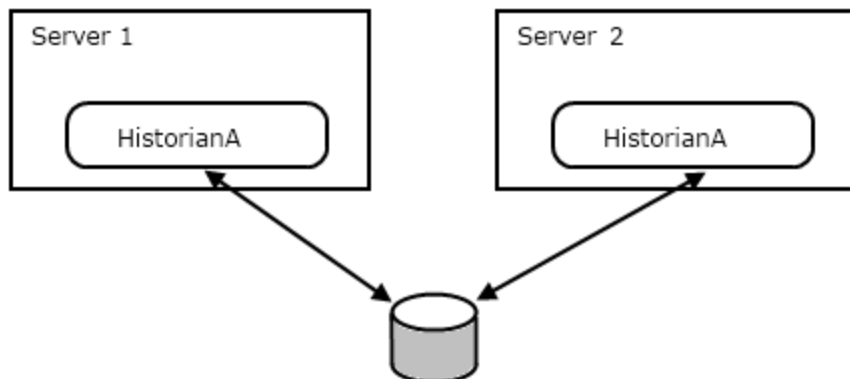
This is the default configuration of a multi-server application running on a primary and backup server. HistorianA is the System Historian, saving data on the primary server, then copying it to the backup. The advantage of this configuration is that it creates a fully redundant copy of all data. If Server 1 is the primary, it will direct Server 2 to also write all data as it arrives. If one or the other server goes offline for a period of time, data will be synchronized when access to that server is restored.

A variation of the above is to configure the Historian to save data to specific database on one server. Typically, this is done only by sites that use a 3rd-party database. On one server, data is stored using the default VTScada system but on the other, data is stored using the commercial database. Users will see no difference.

Note: To prevent data loss, ODBC Historians should always have a backup server using a distinct, alternate data store. The VTScada database format is always to be preferred for reliability.

Using only a 3rd-party database

One Historian in an application that runs on a primary and backup server.
Both Historians log to the same datastore.

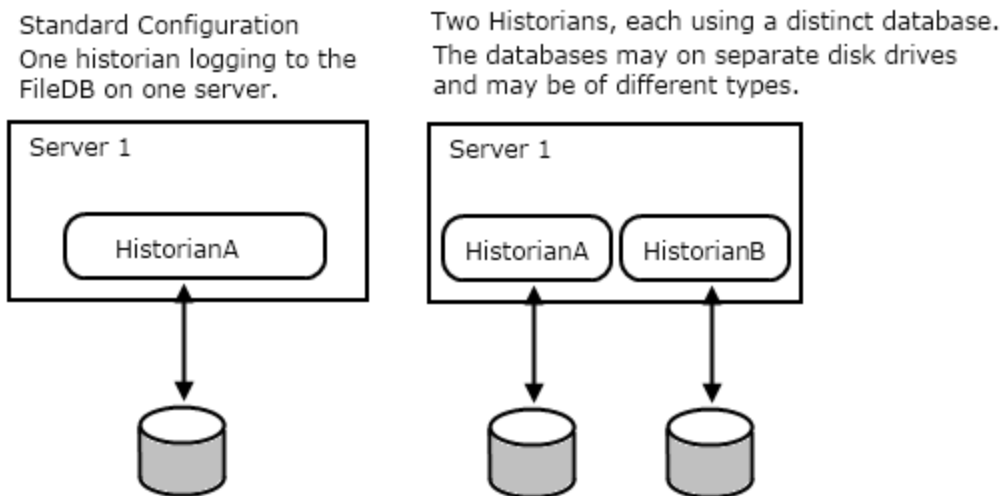


For this configuration, while both Historians will write to a single database, they will be writing to separate schemas on that database. This configuration would occur if the following configuration is one on the primary VTScaDa server and left unchanged when the application is installed on the remote server.

```
SystemHistorianStorageType      = ODBC
SystemHistorianStorageLocation  = Driver-
r=SQLite;Server=DBServer;Database=DBName;Uid=user;Pwd=pwd
```

There is no particular advantage to having the System Historian write to separate schemas on the same server. Therefore, if using an ODBC storage type you should consider using a workstation configuration file to set a unique StorageLocation value on each server.

Two Historians



This figure compares the difference between having one versus two Historians on a single server. There is no advantage to this configuration - it merely illustrates the difference.

Historian Load Distribution

This example is designed to handle an extremely high data-logging load by providing three Historians, each with one redundant server. It is assumed that each Historian is logging data from 1/3 of the tags.

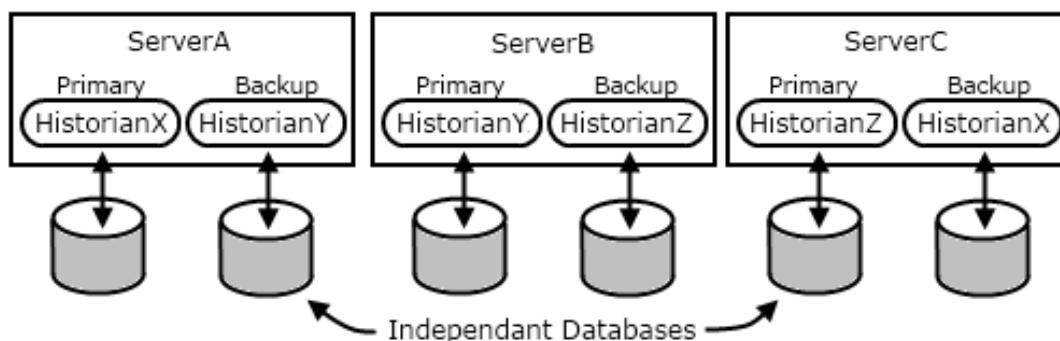


Figure 12-4 Configured for load distribution and backups

In this configuration, each of the machines will carry 2/3 of the disk activity load.

Server detail shown from the Advanced Interface of the Edit Server Lists page of the Application Configuration dialog.

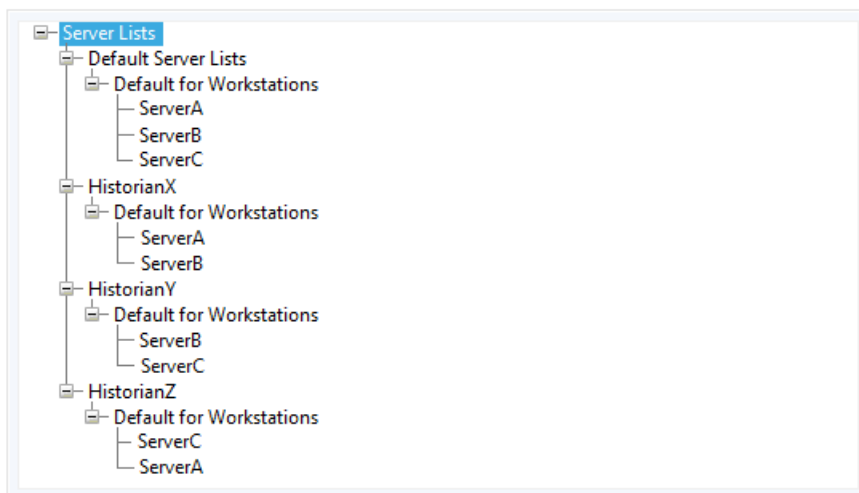


Figure 12-5 Server list matches previous figure

HDV Grid as a Reporting Tool

Values that make up the Historical Data Viewer (HDV) graph can be viewed in a table by clicking the Grid tab at the bottom of the page.

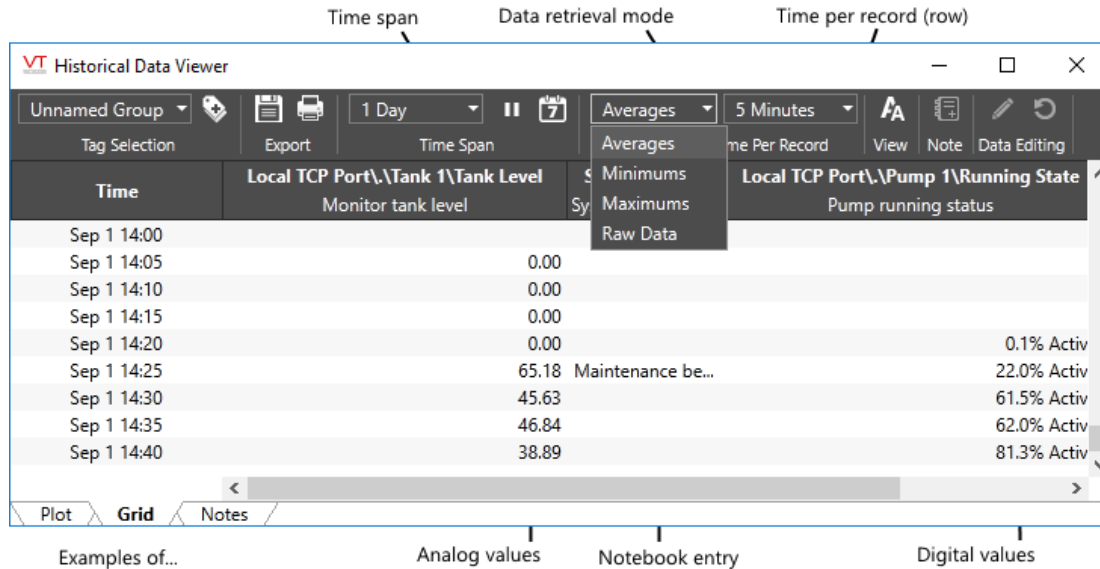
13:01:14
13:01:16
13:01:18
13:01:20

Plot
Grid
Notes

Figure 12-6 Selecting the Grid tab.

The grid is a powerful tool for viewing a report of process values. You may choose to view raw values as logged or you can select averages, minimums or maximums over a given amount of time per record (row). Note that the list of available Time Per Record options will not include a value larger than the selected time span of the grid. Also, values that are significantly smaller than the displayed range will not be available.

As an example, you might configure the grid to display maximum values recorded in each fifteen-minute time span for the past day. For a finely-detailed report, you might configure the grid to show average values for each one-minute time span (or less) over the past week. Any combination that you configure may then be exported to a file for use in a report. A significant advantage to using averages over small time spans is that the time stamps will match for each tag's value, which will not be the case when exporting raw data.



There is one column for every tag that was plotted in the HDV. Columns can be re-sized by clicking and dragging on the edges of their titles. Whenever the window is re-sized, all the columns will also re-size automatically in an attempt to display as much of each tag name as possible. A horizontal scroll bar may be added if any portion of the columns does not fit within the window area.

To re-order the columns, change their order in the tag selector.

Raw Values Data Retrieval Mode

If viewing raw values for more than one tag, you might notice that some entries are gray and some are black. Values tend to be logged a few milliseconds apart and sorting of the rows is done by timestamp. For each row, there will be at least one entry in black, showing the value logged at that timestamp. Rather than leave other columns blank, their last logged value is carried forward, but shown in gray.

When exporting raw data, timestamps are truncated to the nearest second. If greater accuracy is required, use a standard report. [Reports Page](#)

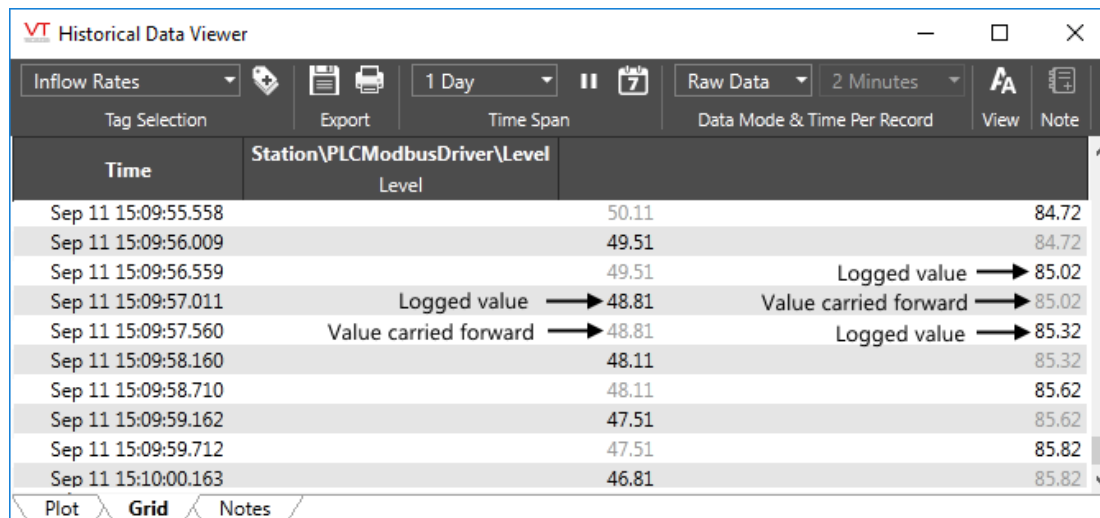


Figure 12-7 Raw data example showing millisecond differences between logging events.

Average, Minimum or Maximum Data Retrieval Mode

For each tag, all logged values within the chosen Time per Record will be used for the selected calculation. The result will be evenly-spaced time intervals with a calculated value for every tag at each time interval.

Averaged Values and Digital Tags

When viewing averaged values, Digital tags will periodically display a message such as "99.8% Active". This is the percentage of the non-zero, logged values for the time period covered by that row.

Reports Page

Note: The Reports Page is only one of many options for retrieving data from VTScada. Refer also to the other chapters under [Log, Note, and Report](#)

Use the Reports page to run existing reports on demand. This is also where you will find the tool to open the The Report Studio, where authorized users can build custom reports. In the default VTScada menu system, you will find the Reports page within the Alarms Reports and Diagnostics folder.

Two types of report exist: legacy reports and studio reports. Operators need not care about the difference other than that there are more steps to run a legacy report than a studio report, and that only studio reports can be saved to PDF and HTML formats. For developers, legacy reports include those provided with VTScada and any that you might write using the scripting language. Studio reports are those that you create using The Report Studio. In the Reports page, the only visible difference between the two types is in which options are enabled.

Tip: To configure a report to run automatically according to a schedule, create Report Tags.

Report Page Features

Example 1: A legacy report

1. Report Type
Standard Report

2. Tag List

Load Group
Save Group

Types
Trenders

Areas
All Areas

Tags Available: (29)

ms1\DN3 Driver
ms1\Faults\3 Phase Power Fault
ms1\Faults\DC Power Fault
ms1\Notebook
ms1\Well\Faults\Level Alarm
ms1\Well\Faults\Level Sensor Alarm
ms1\Well\Outflow
ms1\Well\Septic Time
My Computer
My Computer\IODataBytesPerSec
Operator Notes
Station 1\PLC1
Station 1\PLC1\High Alarm Set
Station 1\PLC1\High Level Set
Station 1\PLC1\Level
Station 1\PLC1\Low Alarm Set
Station 1\PLC1\Low Level Set
Station 1\PLC1\PLC1_Port
Station 1\PLC1\Pump 1\Amps
Station 1\PLC1\Pump 1\Flow
Station 1\PLC1\Pump 1\HOA Control
Station 1\PLC1\Pump 1\Speed
3 Phase Power Fault
DC Power Fault
Notebook
Level Alarm
Level Sensor Alarm
Outflow
Septic Time
Operator Notes
PLC Simulator
High level alarm setpoint
Setpoint to switch pumps
Holding tank level
Low level alarm setpoint
Setpoint to switch pumps
Network connection
Current drawn
Flow rate
Hand-Off-Auto Control
Motor speed

Tags in Report: (2)

Station 1\PLC1\Inflow
Station 1\PLC1\Pump 1\Running
Fluid flow into the system
Pump running status

Sort Alphabetically

3. Reporting Period

Start Time
Oct 22, 2024 10:53 AM

End Time
Oct 22, 2024 11:53 AM

Presets
Last 60 minutes before trigger time

4. Number of Previous Periods

5. Report Destination

Output Type
Printer

☐ Open Report when Complete
☐ Email Report
☐ Email Report as Attachment

Email Settings

Printer
Browse

6. Report Options

☒ Use Excel to display screen reports
☐ Use separate sheets/tables
☐ Rename sheets/tables

Figure 12-8 Report page showing a legacy report(click to expand)

Example 2: A studio report

1. Report Type
Daily Summary

2. Tag List

Load Group

Save Group

Types
All

Areas
All Areas

Tags Available: (78)

AlarmPriority0	Event
AlarmPriority1	Critical Alarm
AlarmPriority2	High Alarm
AlarmPriority3	Warning Alarm
AlarmPriority4	Notice
AnalogFont	Analog value default font
BiggerFont	22 Pt Arial
Default Call-Out Off	Disables Call Out For Default Ro
LabelFont	Font for normal label text
MediumFont	12 Pt Arial
MeterFont	Font for Meter Legends
ms1	Auxiliary IO
ms1\Auxiliary	
ms1\DNF3 Driver	
ms1\Faults	Station Faults
ms1\Faults\3 Phase Power Fault	3 Phase Power Fault
ms1\Faults\DC Power Fault	DC Power Fault
ms1\General Faults	General Faults
ms1\Notebook	Notebook
ms1\Pump1	Pump 1
ms1\Pump1\Energy	Pump 1 Energy
ms1\Pump1\Fault Stats	Pump 1 Fault Stats

Tags in Report: (0)

Sort Alphabetically

3. Reporting Period

Start Time
Oct 21, 2024 12:00 AM

End Time
Oct 22, 2024 12:00 AM

Presets
Previous calendar day

4. Number of Previous Periods

5. Report Destination

Output Type
PDF File

☐ Open Report when Complete

☐ Email Report

☐ Email Report as Attachment

Email Settings

File

Browse Options

6. Report Options

☒ Use Excel to display screen reports

☐ Use separate sheets/tables

☐ Rename sheets/tables

Figure 12-9 Report page showing a studio report(click to expand)

7) [Studio reports only] Report parameters

If the report is parameterized you will be prompted to give values for the report parameters

Note: If generating a file using the VTScada Anywhere Client, it will be transferred to your browser automatically. Check your browser's download history. (Will vary by browser.)
Only text files can be generated by a thin client, not spreadsheet or database files. Similarly, Excel templates cannot be used for legacy reports via a thin client.

This applies to legacy report generation (Prior to Report Studio), HDV exports, note exports, ChangeSet creation, and tag exports.

You cannot view a generated file (such as a report) within the Anywhere Client. The client shows only the VTScada user interface.

Caution: When running VTScada as a Windows service, DO NOT select Excel as an output destination or option from a Report Tag. As noted in the [MSDN forums](#), office applications assume they are being run under an interactive desktop. If Excel attempts to open a **modal**¹ dialog from a non-interactive service, the result is an invisible dialog that cannot be dismissed, stopping the thread.

Besides the technical problems, you must also consider licensing issues. Microsoft's licensing guidelines prevent Office applications from being used on a server to service client requests, unless those clients themselves have licensed copies of Office. Using server-side Automation to provide Office functionality to unlicensed workstations is not covered by Microsoft's End User License Agreement (EULA).

Exercise 12-1 Run a legacy report

1. Open the Reports page.
2. Set the type of report to Standard Report.
3. Select the tag monitoring fluid level in Station 1.
4. Set the reporting time to the last two hours.
5. Set the Output Type to Screen Display.
6. If you do not have Excel installed, deselect the option, Use Excel to display screen reports.
7. Run the report.
8. Experiment with other report types.

Report Tags

Not counted towards your tag license limit.

The Report tag provides most of the features of the Reports Page, plus two extras:

- It will trigger automatically on the interval you specify (perhaps every morning at 8:00).
- It adds an offset to the time periods. For example, you could set the time period to be the Last Hour but offset such that data is for the hour ending 15 minutes before the report runs.

Report tags do not allow for multiple iterations on reports.

A record is added to the events history each time a report is generated using the Report Tag. Use this to verify that automated reports were created on schedule.

The Report tag need not be drawn to function. If drawn, you can show it as a time display, showing when the report was last run. You can also draw it as a button, allowing operators to run the report at a time of their choosing, or to repeat a missed report (a configuration option of the report button widget).

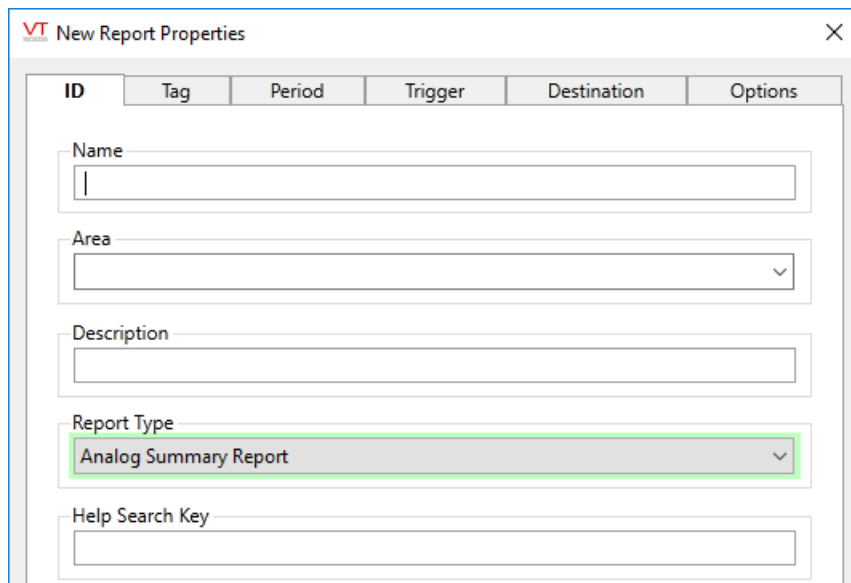
¹A modal dialog is always displayed on top of the calling window and prevents further interaction with that window.

Worth noting: The Report Tools Library provides a number of widgets you can place on a page to allow an operator to adjust the report, without needing to grant tag configuration privileges.

The value of the Report tag will be 1 when writing a report and 0 otherwise. When viewed in the Tag Browser, the address of the report will be the configured report type. ("Analog Summary Report", "Hourly Snapshot Report", etc.)

Before configuring an email destination for a report, you must configure VTScada so that it can access your email server.

Report properties ID tab



The screenshot shows a window titled "VT New Report Properties" with a close button (X) in the top right corner. Below the title bar is a tabbed interface with six tabs: "ID", "Tag", "Period", "Trigger", "Destination", and "Options". The "ID" tab is selected. Inside the "ID" tab, there are five input fields: "Name" (a text box), "Area" (a dropdown menu), "Description" (a text box), "Report Type" (a dropdown menu with "Analog Summary Report" selected and highlighted by a green rectangle), and "Help Search Key" (a text box).

Figure 12-10 The Report tag's ID tab has an extra field: Report Type

The Report Type drop-down list should be used to select the type of report that you want to generate using this tag. This may be one of:

- Daily Snapshot Report
- Daily Total Report
- Driver Communication Error Detail Report
- Driver Communication Summary Report
- Hourly Snapshot Report
- Hourly Total Report
- Standard Report

Report properties Tag tab

Use tools within this tab to select tags to include in the report. Use the Type and Area filters to limit the number shown in the "available" list. Note that, if using either of the pump-related reports, the type filter will automatically choose the Pump Status type. You may switch to the Digital Status type for these reports.

Tag selection sets must be saved as a named group before you may proceed to the remaining configuration tabs.

Tip: In an application with many tags, it can be challenging to select those tags in this dialog. Instead, use the Report Page, where there is much more space, to select your tags and save named groups. Then, you need only use the Load Group button when configuring reports with this tag.

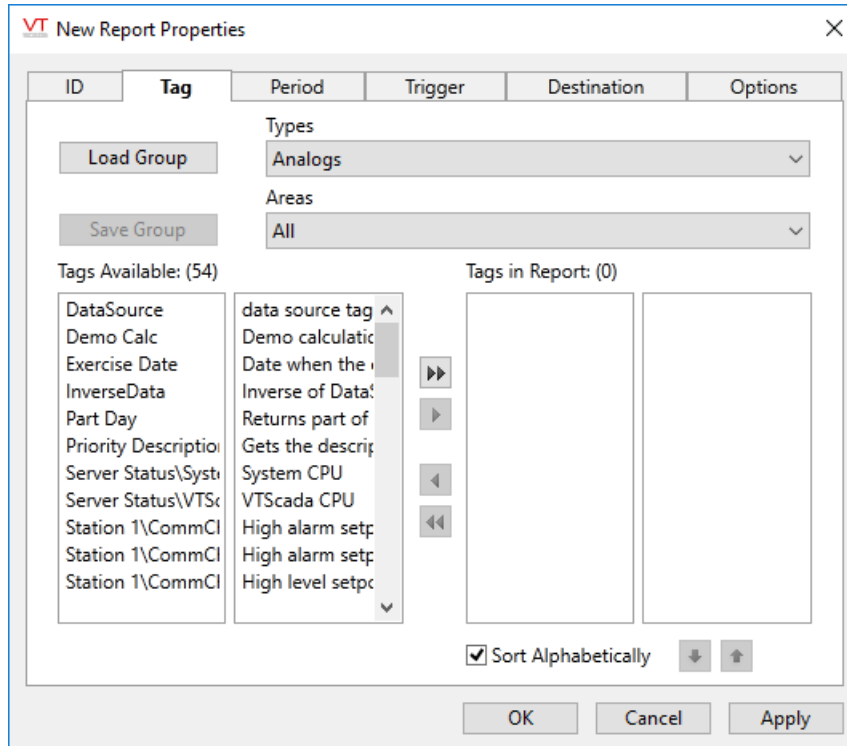


Figure 12-11 Options for tag selection. Sets must be saved as a group.

Report properties Period tab

Use the Period tab of the Report Tag properties folder to select a preset time period or configure a custom time period for the report to be generated by this tag. Note that the Report Button widget provides an option for an operator to re-run the last scheduled report in the event that a system interruption prevented that report from being generated.

Last versus Previous:

The terms "Last" and "Previous" mean different things in the context of a report. "Last" refers to a time period ending at present (or at a defined end time if defining a custom period). "Previous" refers to the most recent full period where weeks end on Sunday night, days end at one second before midnight, and hours end one second before the top of the hour.

For example, suppose that the current time is 3:25 p.m. on a Tuesday. "Last 24 hours before trigger time" refers to the period from 3:25 p.m. Monday until 3:25 p.m. Tuesday.

"Previous calendar day" refers to the period from 12:00 a.m. Monday until 11:59 p.m. Monday.

"Current" refers to the time frames such as "so far today" or "so far this week".

The screenshot shows the 'New Report Properties' dialog box with the 'Period' tab selected. The dialog has tabs for ID, Tags/Parms, Period, Trigger, Destination, and Options. The 'Period' tab contains the following settings:

- Preset: Previous calendar day (dropdown)
- Period Type: Previous Time Period (dropdown)
- Report Duration: 0 (input) months (dropdown)
- Report End Time: 0 (input) hours (dropdown) prior to trigger event
- Report Offset: 0 (input) hours (dropdown)

Report properties Trigger tab

Select a trigger that will generate the report. A variety of timing options are available, or you can link to any tag that will change in value from zero to non-zero as the trigger. (For example, an Alarm tag or a Trigger tag.)

You may also leave the selection as "No Trigger" and draw this tag as a button that operators can press to generate reports on demand.

The screenshot shows the 'New Report Properties' dialog box with the 'Trigger' tab selected. The dialog has tabs for ID, Tag, Period, Trigger, Destination, and Options. The 'Trigger' tab contains the following settings:

- Trigger: ☒ No Trigger
- ☐ Hourly at 12:00 AM and every 1 Hours
- ☐ Daily 12:00 AM
- ☐ Weekly on: Mon 12:00 AM
- ☐ Monthly on: 1 12:00 AM
- ☐ By Tag: No Tag Selected

Below the trigger options is a 'Workstation' field.

Figure 12-12 Schedule an automated report run

Report properties Destination tab

Use the Destination tab to configure an output format and destination for the report. All options from the Reports page can be found here. Available options will change according to the selected output type.

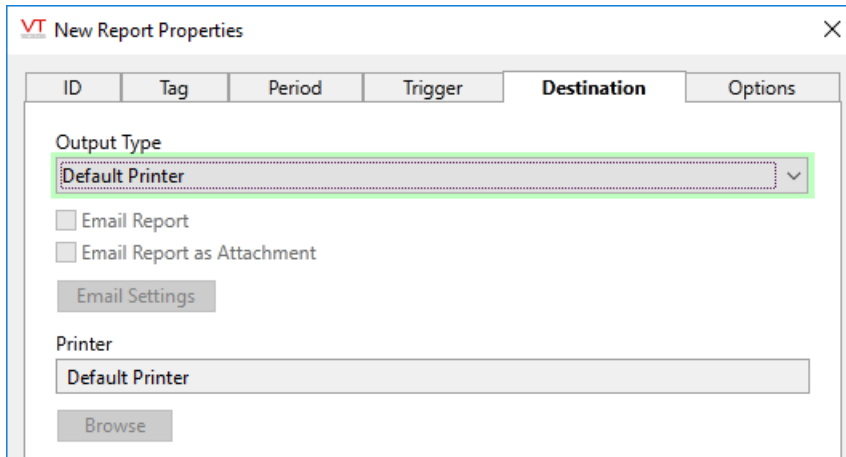


Figure 12-13 Email must be configured in the Alarm Properties dialog

Report properties Options tab

This tab includes two options that are not available in the Reports page. You may choose to record an event when the report triggers, and if doing so, log that event using "Report" as the area name. This is recommended as it will give you a way to verify that automatically-generated reports did in fact run.

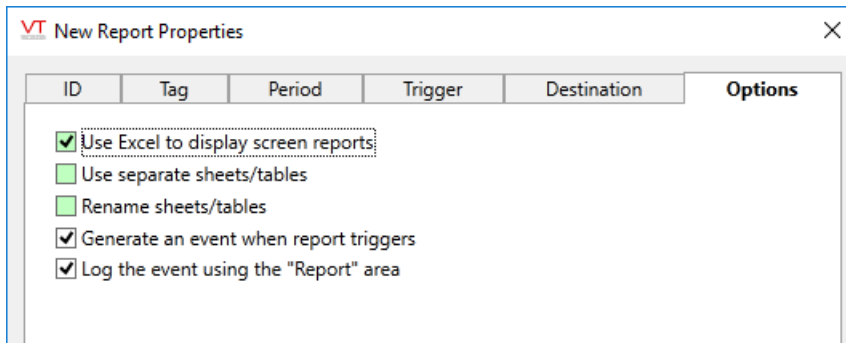


Figure 12-14 Final display and record-keeping options

Exercise 12-2 Report Tag

1. Create a report tag that will run an Hourly Snapshot report on the fluid level.
If you completed the bonus exercise, building extra tags to monitor the pump, feel free to add flow rate and current to your report.
2. Set the trigger time for the report to a minute or two ahead of the current time.

Run-Time Configuration of Report Tags

Create and draw Report Tags to have reports run automatically on schedule or to provide operators with a way to run a pre-defined report with a single click. For the second, you can provide the users with options to make selected changes to reports before running.

The Report Tools Folder

The components of the Report Tools folder of the Widgets palette can be used to add flexibility to Report tags. Report tags are extremely useful in that they hold all the configuration required to generate a report. This saves operators time by avoiding the need to work through the steps of the Reports page.

However, it may be useful to provide certain configuration options for a report tag, such as allowing an operator to choose the output destination, or the time span covered in the report. This can be achieved, without granting tag configuration privileges to the operator, by using the elements in the Report Tools folder.

Note: This library duplicates the widgets available by drawing any Report tag. The exception is the Report Iterations widget because Report tags cannot include iterations as part of their configuration. The Report Iterations widget is included for the sake of completeness as one of the report page tools, and might be used if you were creating a custom report page.

The Report Tools folder can be found in the Widgets palette of the Idea Studio. All the components that make up the Reports page are available here. Note that, it is not expected that you would use these to duplicate the Reports page. The usefulness of these tools lies in the ability to provide selected configuration options for a given Report tag.

After dragging each tool from the palette to a page, link it to the Report tag instance that it should modify.

Excel Add-in for Data Retrieval

The VTScada data retrieval add-in for Microsoft Excel® puts VTScada query tools directly into your worksheets. Create multiple queries that you can update on demand, then use Excel's features to analyze and present the information.

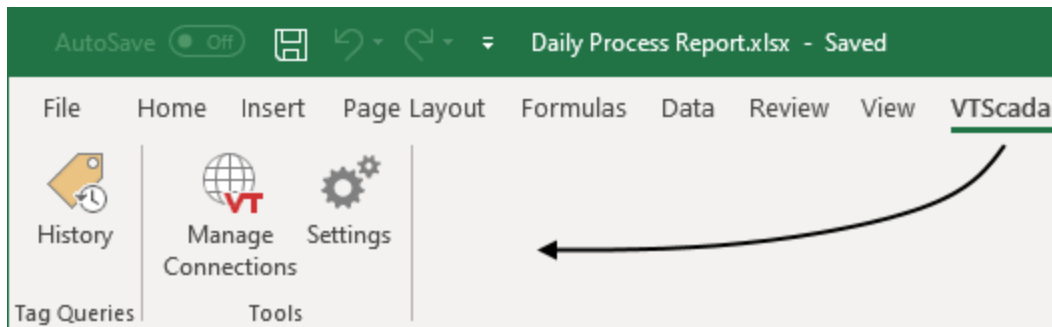


Figure 12-15 The VTScada toolbar within Excel

Preparation

Note: The use of a secured connection (X.509 certificate) is mandatory. The VTScada Excel add-in will not connect to an unsecured server.

You can use a self-signed certificate, provided that you add a local loop-back exemption if both Excel and VTScada are running on the same workstation. Refer to [Microsoft's trouble-shooting instructions](#). (Link opens in a new tab or window.)

The instructions for creating a self-signed certificate are well outside the scope of this documentation. There are many references available on the Internet, as well as many guides to the risks and benefits. *Ensure that you take all necessary security precautions.*

The Excel Add-in does not support connection through OpenID Connect. Ensure that the setting "Permit remote sign-in using VTScada account credentials" in the application's Security options is enabled.

The connection uses the VTScada Internet Server. Follow the instructions within Configure a VTScada Internet Server and Configure a Realm.

Your VTScada license must include the Remote Data Access feature.

Tip: For greater security, consider using one or more of the [SQL View Tag](#), each of which can be assigned a custom privilege.

The Remote Tag Value / History Retrieve privilege is necessary in order to run custom SQL queries. Accounts with only the Remote Data Access privilege can see only their assigned SQL views or AlarmHistory, and cannot run their own SQL queries or browse the tag parameter tables.

The add-in will work best with the 1.4 version or later of the Excel API. It is compatible with the 1.1 API version, but may not perform as well.

In terms of Excel version, the add-in is compatible with Office 2016 build 16.0.4390.1000 or later and any version of Office Online.

An uncommon configuration requirement is to allow for Domain Aliases (CORS).

Installation of the Add-in

Your goal is to navigate to Microsoft's store for adding Office Add-ins. The route will vary depending on your version of Excel. In older versions, you would open the Insert ribbon and select Get Add-ins. In the version that was current as these notes were written, the Add-ins tool was found in the Home ribbon. If neither works for you, consult your Excel documentation.

When you arrive at the Office Add-ins dialog, it should be similar to the following. Type "VTScada" into the search field and press [enter].

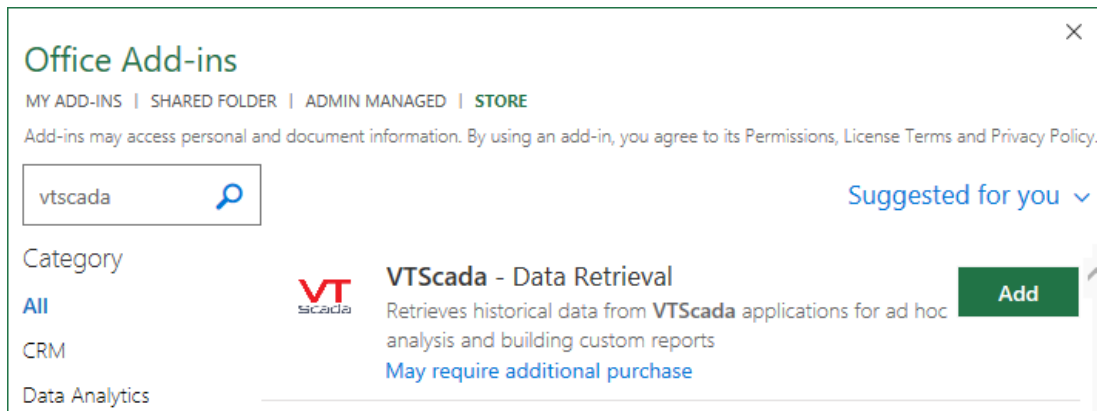


Figure 12-16 Getting the add-in from the Office store by searching for "vtscada".

Settings for the Excel Add-in

In addition to the following three options, the Settings menu is also where you will find a link to the About menu, where you can check the version number, the privacy policy and review a list of third-party software used by the add-in.

Select data range when editing a query.

When opening the edit query panel, automatically select the cells where the results of the query are located.

Deselect if you have typically have other cells selected and do not wish to have the selection area change.

Select data range when running a query.

When running a query by using the refresh button, this selects the cells where the results query are located.

Show welcome screen.

The welcome screen is shown automatically when the history query panel is first opened.

Create and Manage Connections

Note: The use of a secured connection (X.509 certificate) is mandatory. The VTScada Excel add-in will not connect to an unsecured server. You can use a self-signed certificate, provided that you add a local loop-back exemption if both Excel and VTScada are running on the same workstation. Refer to [Microsoft's trouble-shooting instructions](#). (Link opens in a new tab or window.)

The instructions for creating a self-signed certificate are well outside the scope of this documentation. There are many references available on the Internet, as well as many guides to the risks and benefits. *Ensure that you take all necessary security precautions.*

The Excel Add-in does not support connection through OpenID Connect. Ensure that the setting "Permit remote sign-in using VTScada account credentials" in the application's Security options is enabled.

The connection uses the VTScada Internet Server. Follow the instructions within Configure a VTScada Internet Server and Configure a Realm.

Your VTScada license must include the Remote Data Access feature.

Tip: For greater security, consider using one or more of the [SQL View Tag](#), each of which can be assigned a custom privilege.

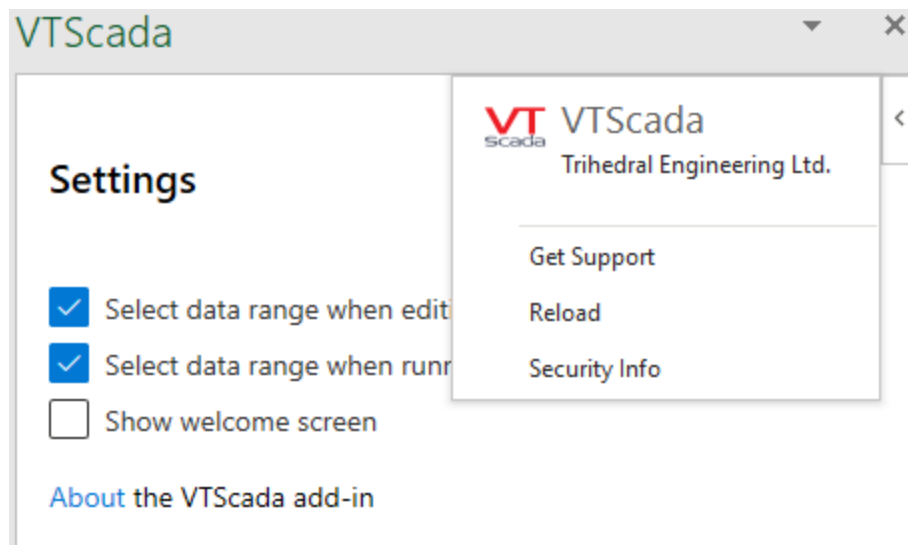
The Remote Tag Value / History Retrieve privilege is necessary in order to run custom SQL queries. Accounts with only the Remote Data Access privilege can see only their assigned SQL views or AlarmHistory, and cannot run their own SQL queries or browse the tag parameter tables.

The add-in will work best with the 1.4 version or later of the Excel API. It is compatible with the 1.1 API version, but may not perform as well.

In terms of Excel version, the add-in is compatible with Office 2016 build 16.0.4390.1000 or later and any version of Office Online.

An uncommon configuration requirement is to allow for Domain Aliases (CORS).

When each tool of the VTScada Add-in is opened, there will be a fly-out menu at the upper right. Use this to learn more about the tool.



Connections are saved in the current workbook. Note that the History panel will guide you through the creation of your first connection. You need only use the Manage Connections panel if you intend to set up multiple connections, edit, or delete an existing connection.

The procedure to create your first connection is the same whether you do so using the Manage Connections panel or the History panel.

You must connect to your VTScada server and sign in before creating or running queries. To do so:

1. Open the VTScada toolbar in Excel and select either the Manage Connections tool or the History tool.
2. The Connections panel will open beside your worksheet.
If you have not already created a connection in this workbook the Connection sub-

panel opens automatically when you select the History tool. Otherwise, select New Connection.

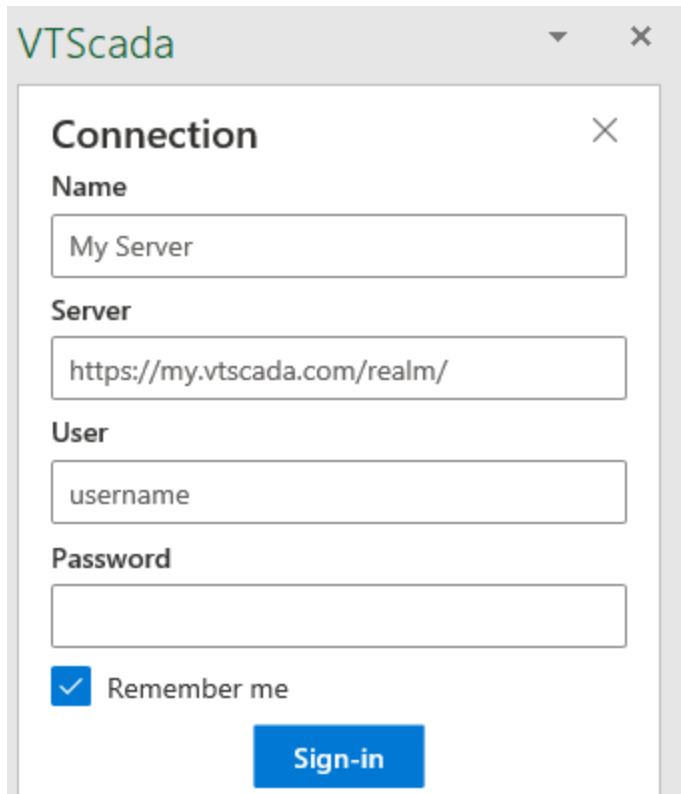


Figure 12-17 The Connection panel.

Note the X in the upper right corner, which will close this sub-panel.

3. Provide the following information:

- A name for the connection.
Any name may be used, but it may be helpful to create a connection name that matches the server name.
- The URL for the server. This will typically include the realm name, where the application can be found.
This must use a secured connection. If you are using the standard port of 443, it need not be specified.
- An account name for the VTScada application.
The account must have the Remote Data Access privilege.
- The password for that account.

4. Choose whether or not to remember the user name and password.

5. Select Sign-in .

If successful(*), the connection name will be listed in the Connections panel. If you began the process by selecting the History tool, the Tag History Query panel will open immediately.

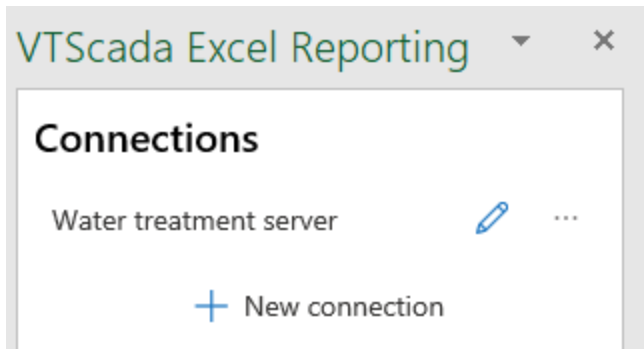

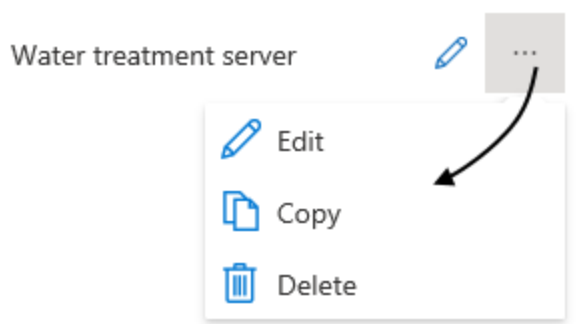


Figure 12-18 The Connections panel is available only after successfully creating a connection. Use the Manage Connections tool to open.

Select the editing tool  to change the name, change the server address or to sign-out.

Within the expandable menu, you can also edit, as well as copy or delete the saved connection.



You may choose to create additional connections for any of the following reasons:

- You have more than one server.
- You connect to more than one realm.
- You have more than one application.

(*)Troubleshooting

An attempt to connect might fail for any of the following reasons:

A bad server address

Make sure the address can be reached from the local computer. If the address works for the VIC or Anywhere Client, it should work for the add-in.

Unsecured server

Make sure that the server is secured with a certificate.

Wrong or missing realm name

Provide the necessary realm as part of the server address.

The application is not running on the VTScada server.

Ensure that the application is running.

The VTScada server does not support the Excel add-in

Ensure that the server is running VTScada version 12.0 or later.

The VTScada server is not licensed to allow remote data retrievals

Purchase the required option for your VTScada license.

Create Queries

You can build several queries within the same worksheet. Query results are returned in a tabular format, starting in the cell that is selected when you create that query. Before creating a new query, ensure that you have selected the cell where you the results to be sent. See notes later in this topic for using cut and paste within Excel to move the query result to a new location.

Queries are saved in the current workbook.

Caution: Be sure to leave space for the full query result. Every selected tag and every calculation you make will require a column(*). If you are querying for a day's data with hourly summaries, you will need 24 rows for the data and one for the title. The first time a query runs, it will use every cell that it needs, whether empty or not.

On subsequent queries, cells that contain other information will be moved as needed in order to accommodate a block that changes in size. Formulas within those cells may not adjust correctly. You are advised to find the space that will be needed by your query results and leave room.

(*) Or row rather than column if you choose to flip the result. These notes will refer to the default orientation where tags are listed across the columns and data fills the rows below each tag name.

Tip: If you need a presentation-quality report, you might choose to send the query results to a second sheet in the workbook. The first sheet can be used for the presentation layout, with links to data that is sent to the second sheet.

Note: A query is built of several parts: Tag Selection, Time Ranges, Query Attributes (including calculations) and a choice of More Settings. For each part, there are many options. Therefore, these notes do not attempt to provide a single set of steps to build a query. Instead, each part is discussed on its own, the options presented, and instructions or notes are provided to help you use each option.

Tag Selection

Ensure that you have selected an appropriate target cell in the worksheet before beginning. See the notes at the beginning of this topic. The initial stages of query building can be presented as a series of steps:

Note: There is no "show children" option when browsing tags. Only one level at a time is shown.

1. Begin the process of building a query by selecting the History tool in the VTScada toolbar to open the Tag History Queries panel.

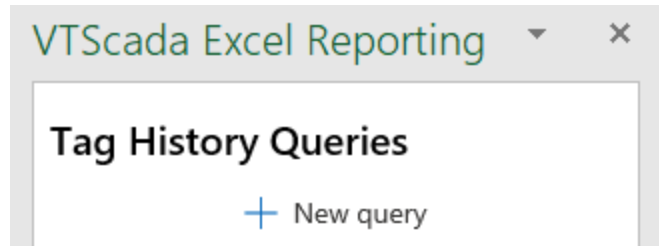


Figure 12-19 The Tag History Queries panel with no saved queries

2. Select the New Query tool.
The Tag History Query panel opens. (see following image)
3. Give the query a name that describes its purpose.

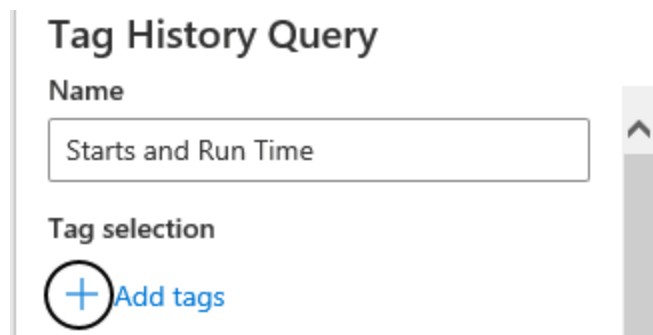


Figure 12-20 Creating a query named "Starts and Run Time"

4. Select the Add tags tool (circled in previous figure).
The VTSkada Tag Browser opens:

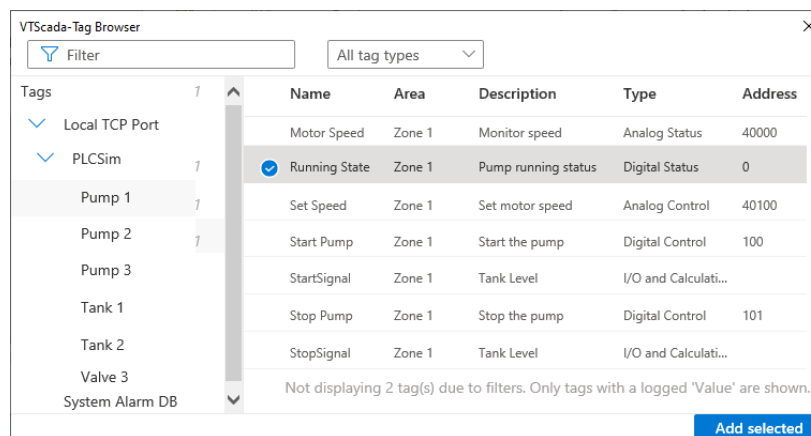


Figure 12-21 The Excel Reporting Tag Browser showing one tag selected for addition to the set.

Tip: Note the two filters. The name filter (left) assumes wildcards. In the example shown, "Start" would find both "Start Pump" and "StartSignal". Do not add your own wildcard characters.

The type filter (right) will limit the view by group such as "Analog", "Digitals" etc. Only types that are present in your application are available in the filter.

Filters apply only to the current level in the tag hierarchy; there is no "Show Children" option.

The digit beside each parent tag in the left window indicates the number of tags selected below that parent tag.

5. Use the Add selected tool to finish creating the tag selection.

You can select several tags at once. There is no need to use "Add selected" until you have finished selecting tags. Selected tags will remain selected as you navigate through the tag hierarchy.

After using the Add selected tool, you will be given a chance to rename the selection set, and to review and edit the list (adding more tags if you clicked "Add selected" before you were finished.)

6. Select "Save" when you have finished selecting tags.

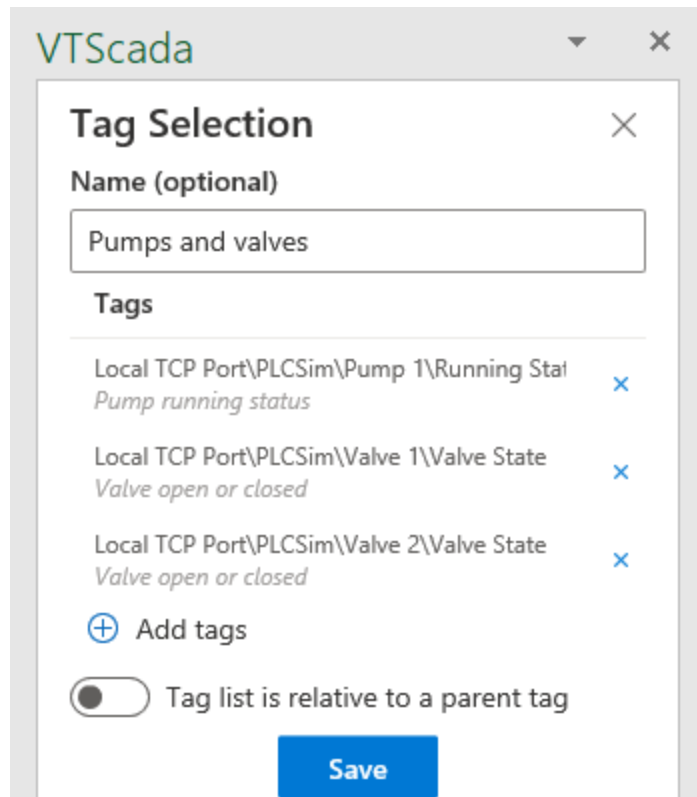


Figure 12-22 Saving your tag selection

The option, "Tag list is relative to a parent tag" is described later in this topic.

Tip: The default name for a tag selection set is the list of tags within the set. You may find it easier to manage sets if you provide your own descriptive names instead.

You are returned to the Tag History Query panel.

Tag History Query

Name

Starts and Run Time

Tag selection

Pumps and valves



Figure 12-23 A named selection set.

Tip: You can modify the tag selection at any time. (The edit tool is circled.) You can also switch to another saved tag selection. But, only one named tag selection can be used in any query.

Variation: Tag list is relative to a parent tag

If your tags were built according to the advice in Plan your tag structures carefully, you will have a number of pumps, stations, generators, etc. that each contain a set of tags with the same names. Rather than select the same tags in every instance, you can select the tags from one instance, then create a list of stations or equipment instances containing those tags.

1. Start by selecting the tags in one instance using the instructions from the preceding section. In the following example, one tag is chosen from Pump 1: Running State.
 2. Use the "Add selected" tool to return to the Tag History Query panel.
 3. Select the option, Tag list is relative to a parent tag.
 4. Select the closest parent to the set of tags. In this example, that is the Pump 1 context.
 5. Save the tag selection.
- You will have a chance in a later step to select Pump 2, Pump 3, etc.

Tag Selection

Name (optional)

Pumps and valves

Tags

Local TCP Port\PLCSim\Pump 1\Running State
Pump running status



⊕ Add tags

☒ Tag list is relative to a parent tag

Parent tag

Local TCP Port\PLCSim\Pump 1



This parent portion of the tag name will be substituted at run time. This allows the tag list to be mapped onto different parent tags.

Save

Figure 12-24 The tag list consists of one tag: Running State. It's parent is identified as Pump 1.

6. After selecting Save, you are returned to the Tag History Query panel. Note the new option, circled in the following figure.

Tag selection

Pump Running Status

Parent tag selection

+ Add tags

Figure 12-25 Parent tag selection is available only when the tag list is flagged as relative to a parent tag

7. Select the Add tags tool below Parent tag selection.
The Tag Browser reopens.
8. Select the parent tags that include the tag(s) you chose in the previous step.
Note that you must select the parent as indicated earlier. In this example, those are Pumps. If you select a grandparent instead, VTScada will not search it for all the matching parents that it might contain.

	Name	Area	Description
<input checked="" type="checkbox"/>	Pump 1	Zone 1	Water supply
<input checked="" type="checkbox"/>	Pump 2	Zone 2	Primary supply in zone 2
<input checked="" type="checkbox"/>	Pump 3	Zone 2	Secondary supply in zone 2

Figure 12-26 A set of selected parent tags.

9. Finish by clicking the Add selected tool, then Save.

You are returned to the Query panel, which will display the named Tag Selection and the set of selected parents that contain tags in that named selection.

Name

Tag selection

Parent tag selection

Figure 12-27 A tag selection that is relative to a set of parent tags.

Time Ranges

All times are specified using the current time zone as set for your workstation. You do not need to calculate for UTC time values.

You can specify either a fixed time range with a defined beginning and end:

Time range



Use fixed time range

Start

End


  

Figure 12-28 A sample fixed time range.

Or, you can specify a time range that is in some way relative to when the report is run. This option is more likely for reports that you save in order to run on a regular basis.

Three groups of preset relative time ranges are provided (Relative to now. Relative to the start of the day or week. Previous day or week.), with each group holding several options. You can select any of these to use as-in, modify one for your own purposes, or create your own.

The same tools are used both to edit and to create time ranges, therefore these notes will describe only the process of creating a new range.

1. Select the "Create a new time range" tool.
The Time Range panel opens

Time Range

Name

Time Range 1

Relative time range

Last ▾

1

days ▾

Offset (seconds)

First day of week

0

Sunday ▾

Save

Figure 12-29 The time range creation tools


2. Give the range a descriptive name.
 3. Using the three tools in the Relative time range section:
 4. Choose between Last, Current, or Previous
 5. Set the number of time units
 6. Set the size of the time units, ranging from minutes to years.
 7. Optionally, set an offset measured in seconds, into the range.
 8. If you chose "weeks" as the size of the time units, then you have the option of specifying the first day of the week. Sunday is the default.
 9. Select Save.
- The time range panel will close, returning you to the Tag History Query panel.


Query Attributes

Choose between calculated data and raw data (value property). If you select raw data, there are no other choices to make in this section. Note that raw data, even for a short time span, can require many rows.

There are more options for calculated data:

Query Attributes


Calculated data


Raw data

☒ Single value only ⓘ

Calculation type(s)

▼

Time per record

1

hour ▼

Figure 12-30 Default query attributes section of the Tag History Query panel

Use "Single value only" to run the selected calculations for all values in the time frame as a whole. For example, the average, minimum and maximum temperature values recorded yesterday. Or, you can repeat those calculations every X time units over the time frame. For example, hourly averages yesterday.

Tip: To display both, you could create a second query, located in a cell below the first, or you could use Excel tools to calculate a grand total.

The Calculation types selection tool allows you to choose as many as you like of:

- Average
- Minimum
- Maximum
- Total
- Change in value
- Value at start (snapshot)
- Time of minimum
- Time of maximum
- Number of starts (zero to non-zero transitions)
- Runtime (non-zero time)

Each selected calculation adds another column for each tag in the selection set. Columns are ordered first by tags, then by calculation for each tag.

Calculations are performed for data collected over a time span (the time per record). For example the average of an hour's data, or the maximum of a day's data. For calculated data, timestamps always indicate the start of each time span.

More Settings

The following options are available to refine your query:

- By default, queries are limited to 10,000 records. You may adjust this number as required.
- You may choose to include a timestamp column (not included when calculating a single value for the entire time range).
- You may choose to sort by timestamp in descending order rather than the default of ascending (older later).
- You may choose to include title row above the columns (default)
- Format the cell data type according to the result returned rather than allowing Excel to choose. The add-in performs the following formatting:
 - converts timestamps to yyyy-mm-dd HH:mm:ss
 - converts values to integer or double depending on the tag type
 - converts Runtimes to HHH:mm:ss
- To change this after running a report once, you may need to format the result cells as "general" before running the report again.
- Adjust column widths to fit. Ensures that each tag title is fully displayed.
- Flip rows and columns. Given cell A1 as the location for the query result, the default is to display titles in row 1 with data filling downward through the rows. Select this to place titles in Column A with data filling to the right through the columns.

Run (and refresh) your query

When ready, select the Run button. Output will go to the indicated row and column. After running once, the space required by the output is displayed and you have the option run the query again or to save.

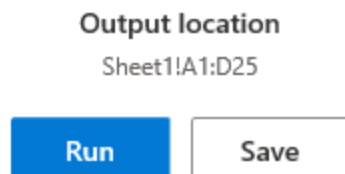


Figure 12-31 Options after the first run of the query.

Your saved queries for the current workbook are available in the Tag History Queries panel. (The default panel opened by the History tool in the VTScada toolbar, and when none of the sub-panels used to generate a query are open.)

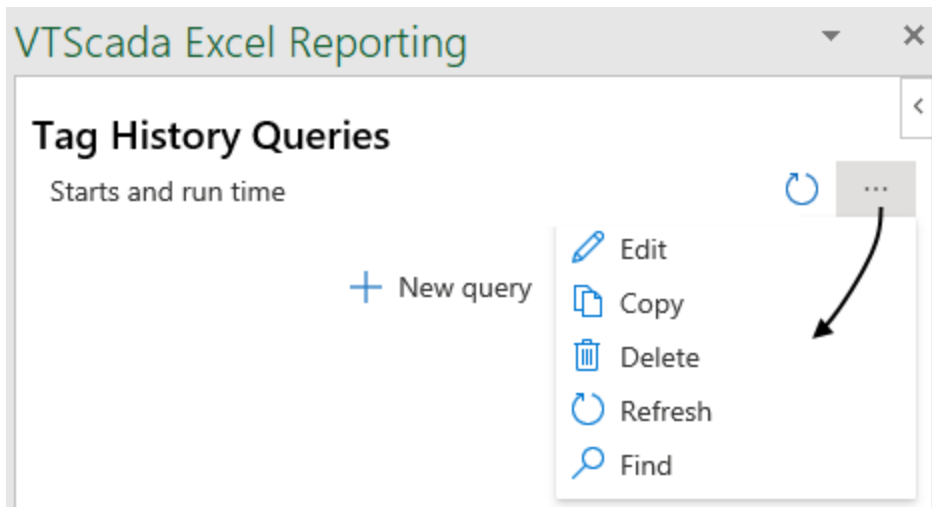


Figure 12-32 The panel with one query: "Starts and run time"

The default tool for each query is Refresh, as indicated by the circular arrow. If your workbook has more than one query, there will be a Refresh All tool.

Expand the menu to access other tools as shown. Of particular note is Find. This tool will select all the cells in the worksheet that are the result of a query.

Move a query result to a new location

Queries are tied to the cell and worksheet that was selected when you began the process of creating the query. To move the result to a new location:

1. Refresh the query.
2. Select the Find tool from the menu.
The cells holding the query result will be selected on the worksheet.
3. Cut those cells.
4. Paste them to a new location on the worksheet.

Example Queries with Excel

If you have not yet used the VTScada Thin Client Server to allow secured (TLS/SSL) access to your application, you can use Trihedral's online demo application to explore the Excel add-in for data retrieval.

Preparation:

Before connecting in the Add-in, use your favorite Internet browser to open <https://vts-demo.trihedral.com/exceldemo>. Sign in with the username and password, "demo" and "demo". This will allow you to browse the tags in the context of the application.

Note that this is a demo application, accessible to all who visit Trihedral's website. Set-points and controls may be at unexpected levels. When running in automatic mode, the simulator uses a set of simple increasing & decreasing values.

Explore the Add-in

1. Start your copy of Microsoft Excel and create a blank workbook.
2. Install the Add-In as described in Excel Add-in for Data Retrieval.

3. Select the VTScada toolbar.
4. Select the Manage Connections tool in the toolbar.
The VTScada panel opens on the right
5. Select the New connection tool in the panel.
6. Type a name for the connection such as "Water Demo".
7. Specify the following as the server:
`https://vtsdemo.trihedral.com/exceldemo`
8. Username: `demo`
9. Password: `demo`
10. Select Sign-in.
Your connection name, `Water Demo`, is now listed.
11. Select History in the VTScada toolbar.
The Tag History Queries panel opens.
12. Select a cell in the sheet other than A1. Perhaps B3.
This is the starting cell, for where the data will be returned.
13. Select New query in the panel.
14. Give the query a name such as `Water Levels`
15. Select the "Create new tag selection" plus-sign tool, beside the Tag Selection field.
16. In the Tags list, navigate to: `WaterTreatmentPlantDemo >> TCPIP Port >> Tanks >> Tank0`
17. Select the tag, `TankLevel`.
Do not click the Add Selected button yet.
18. Navigate to `Tank1`, `Tank2` and `Tank3` in turn, selecting `TankLevel` in each.
19. Click on the Add Selected button after selecting the `TankLevel` tag from each tank.
You are returned to the Tag Selection panel, showing four selected tags.
20. Change the name to "`Water Level Tags`".
21. Select the Save button.
You are returned to the Tag History Query panel.
22. In the Time Range drop-down, select `Last 12 hours`
23. Expand the Calculation Types drop-down.
24. Deselect Average.
25. Select Maximum and Time of maximum.
26. Click anywhere on the panel outside the drop-down to close it.
27. Ensure that the Time per record is set to 1 hour.
28. Expand the More Settings option.
29. Deselect the Include Timestamp Column
30. Verify that the output location is still B3 (or wherever you chose other than A1).
31. Select the Run button.

The data should be returned to your sheet. As noted, the simulator follows a predictable pattern. If no-one has adjusted the setpoints, the maximums will be close to uniform.

Note that the tag names make the columns rather wide. Let's improve that report...

1. In the cell about each column, create a new, short title for the data in that column.
For example, in B2 you might type `Tank 0 Hourly Max`
2. In C2 you might type, `Tank 0, Time of Max`

3. After adding your new titles, deselect the Show Title Row option in the VTScada panel.
4. Ensure that the Output Location is still B3 (or your chosen cell). You may need to re-select this.
5. Save the report, then select Run again.

Your report is configured to return hourly values for the last 12 hours. Therefore, you can expect exactly 12 rows of data, every time you run the report. Use your Excel skills to add a function below each column, such as calculating the

maximum overall level and time. (You will probably need to format the cell showing the time, otherwise you will see the timestamp number instead of the human-friendly version.)

SQL Queries

You can generate reports of VTScada data using Structured Query Language (SQL) queries in third-party programs such as XLReporter™, Dream Report™, a REST or JODBC interface, Microsoft Access or Excel™. These queries are handled by VTScada's Remote Data Access feature, an optional component that must be purchased with your license agreement. An ODBC driver is provided free of charge, for installation on any workstation that is to send queries to your VTScada server.

A significant advantage of using ODBC is the ability to query VTScada from computers that do not have VTScada installed. Managers, engineers, operators and others can view up-to-date reports from any location that has network access to your VTScada server.

Using this interface, you can treat a VTScada application as if it were a relational database containing logged tag values, aggregate tag data, and alarm data. After the connection is configured, your reporting program can send SQL queries to VTScada to retrieve tag values that are being logged. (Tags that are not being logged have no stored history to query.)

Note: VTScada stores history using only its own proprietary format, which is not a relational database.
The SQL interface allows a limited subset of the SQL language to be used **as if** querying SQL tables.

Pros:

- Able to query VTScada process and alarm history in ways that are not available using the built-in reports.
- Third-party tools often have extensive options for creating well-formatted reports.
- Does not require a VTScada license on the workstations where the reports are generated, so long as there is local network access to VTScada.

Cons:

- Requires a third-party reporting program.
- May have an additional cost depending on the selected program.
- Requires a VTScada license that includes the Remote Data Access option.
- May require knowledge of the SQL language, depending on the availability of query-building tools in your chosen reporting program.

- Will require knowledge of the selected third-party program.
- VTScada supports only a sub-set of the data aggregation and calculation options commonly found in SQL. It may be necessary to run several queries then do further calculations with the combined result sets.

Use When:

- Complex report formats are required
- Additional cost and development time are not prohibitive
- Developer has knowledge of a third-party reporting program or has time to learn
- Multiple users are required
- Advanced data calculations are required

Requirements:

- If querying from another program, a license key that includes the Remote Data Access option is required. Check by clicking the License Management button in the VAM.
VTScada modules that use the SQL functions need no special licensing.
- Install the VTScada ODBC Driver on any computer that is to generate the queries. To do so, run the program VTScadaODBCDriverInstall.exe.
- Secure your application, and grant the required data access privileges to at least one account. We recommend that a dedicated account be created for remote data access, possessing no other privileges.
The privileges required depend on the level of access to be granted. Refer to the table later in this topic.
- If you plan to query tag parameters, ensure that the account has the Tag Parameter View privilege.
- If using [Realm Filtering](#), A realm, in which the application has been selected.
- A system DSN, configured using the Microsoft Windows™ ODBC Administrator program, or knowledge of the configuration so that you can build your own connection string.

The results of an SQL query are sorted by ASCII value as opposed to an alphabetical or lexicographical order.

Realm filtering (if configured) will limit the tags that can be queried within a realm. This may be useful to limit access due to security concerns or to avoid overwhelming your query viewer with an excessive number of tags.

Security and Remote Data Access

At a minimum, you will need access to a VTScada user account that has the Remote Data Access privilege. A common configuration is to create one or more accounts with only that privilege and one or more of the following privileges as required. Such a user will not have any access to the system other what is required to run reports.

Tip: For greater security, consider using one or more of the [SQL View Tag](#), each of which can be assigned a custom privilege.

The Remote Tag Value / History Retrieve privilege is necessary in order to run custom SQL queries. Accounts with only the Remote Data Access privilege can see only their assigned SQL views or AlarmHistory, and cannot run their own SQL queries or browse the tag parameter tables.

Access to...	Requires the additional privilege...
The History table and time-based summary tables	Remote Tag Value / History Retrieve
The Alarm history and time-based summary tables of the same.	None
SQL View tables	Any custom privilege assigned to the view tag.
Tag parameter tables	Parameter View
Other custom tables	Custom privileges specified when registering the table. (This feature is for advanced programmers.)

Configure an ODBC Server

To create an ODBC connection to your application (including REST and Java ODBC), at least one server must be configured as an ODBC server.¹

The steps are much the same as those to set up VTScada Thin Client Server and Realm, with a few small exceptions. For example, the Max Clients setting has no effect in this context. That setting is relevant to thin client connections, rather than ODBC clients.

While this topic provides an overview of the steps required to configure your ODBC server, you are urged to refer to [Thin Clients: Mobile and Internet](#), where the process is described in much more detail.

Caution: A security certificate is optional but strongly recommended. [Internet Security \(TLS, X.509, SSL\)](#)

1. Ensure that your application is secured and that the account you will use for queries has the Remote Data Access privilege.
(While testing the connection, it may be helpful to have the Thin Client privilege. Once configured, we recommend the use of an account that has few or no privileges other than Remote Data Access.
The privilege, Remote Tag Value / History Retrieve, may be granted if users are allowed to browse the virtual tables.
The Tag Parameter View privilege may be useful in some instances.)
2. In the VTScada Application Manager (VAM), select the tool labeled Internet Setup. The VTScada Thin Client/Server Setup dialog will open.

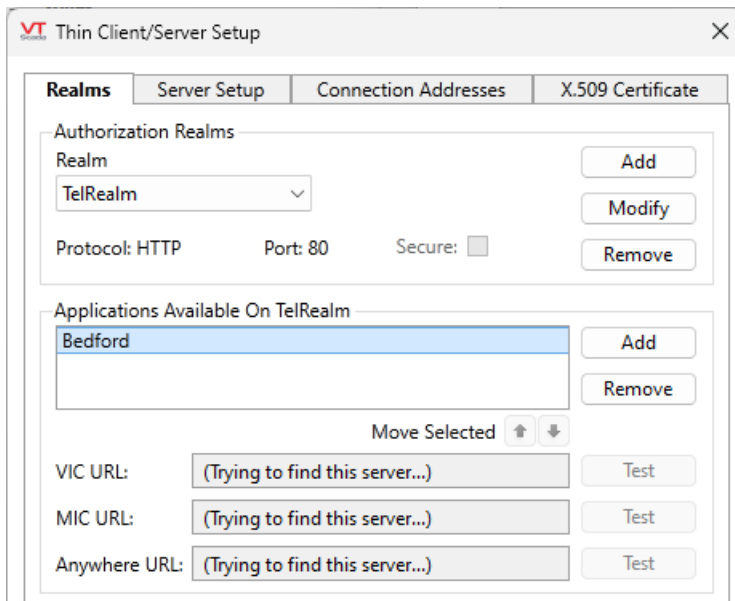


Figure 12-33 Adding a realm in the Thin Client/Server Setup dialog

3. (This step is optional for intranet-only configurations. For Internet-connections, it is optional but strongly recommended).
Open the tab, X.509 and configure a security certificate according to the instructions in [Internet Security \(TLS, X.509, SSL\)](#).
4. Open the Server Setup tab and add the name of your primary thin client server. ([Define the Server Setup](#))
If using a security certificate, this will match the name of the server listed there. Otherwise, it is likely the name of server where you are working.
5. Open the Connection Addresses tab and define a default address list. ([Define the Connection Addresses](#)).
6. Open the Realms tab and create a unique name for your ODBC connection, then add your application to that realm. [Configure a Realm](#)

The name should clearly indicate what the server is. As a suggestion: by including the letters "ODBC" you will make the name easier to recognize later when configuring the ODBC connection. If using [Security Realms](#), your choice will be limited to realms created for that feature.

Caution: Do not name any realm, "Rest" or "SQLQuery". Doing so will interfere with remote access to VTScada data.

7. Save the new Realm

Tip: To check your configuration, give your account the Internet Client Access privilege and then test the connection using the Anywhere client. If you can connect that way, the steps you've just completed are correct.

Note: The property, SoapServicesRealmName, is no longer used. If your technology relies on SOAP services, request a copy of the VTScada version 11.3 ODBC driver. As of version 12, we now use REST.

Optional Step:

(This is not required for queries. It is used only for convenience when using an external program that presents these summary tables as if they existed as real tables. You may query any TPP time frame without defining any value for SQLQueryTableTPPs)

You can take advantage of VTScada's ability to group data records by time interval. By adding the application property, SQLQueryTableTPPs, you can retrieve tag data from specific time intervals. The format of the statement is:

```
SQLQueryTableTPPs = <time frame specifier>
```

where the time frame specifier takes the form of a digit and a letter. The letter indicates the units in which time interval is measured (hours: h , weeks: w, etc.) and the digit specifies the number of units of each interval that should pass between each record.

Note that you can have only one TPP statement in a settings file, but that statement can have multiple values, separated by semi-colons and no spaces.

Two examples follow:

```
; two hours...
SQLQueryTableTPPs = 2H
```

...or, using the Add Property dialog and configuring two TPP values:

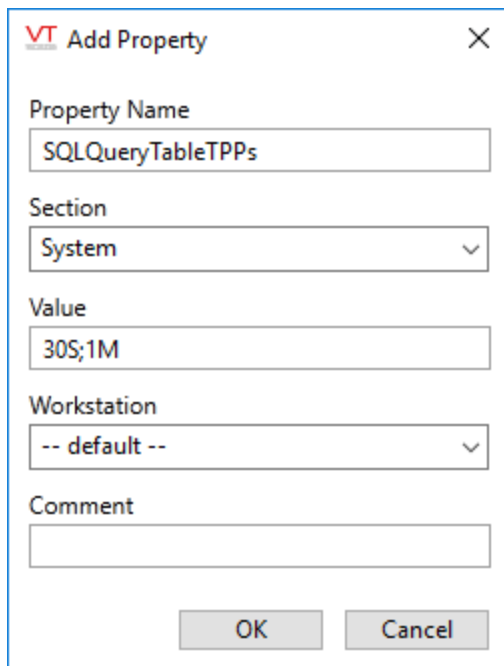


Figure 12-34 Adding TPP values for aggregated reports

The available time interval units are:

- MS - milliseconds
- S - seconds (this is the default; the S may be omitted)
- M - minutes
- H - hours
- D - days

- W - weeks
- Y - years

Connect to the ODBC Server

Any computer that has a network connection to the ODBC Server may connect. There is no need for VTScada to be installed on that computer, but the Trihedral-supplied ODBC Driver must be. On each computer that is to connect to the server, run VTScadaODBCDriver-Install.exe. This will install the ODBC driver needed to connect to the server.

Connection parameters must be supplied to the program that will run the query. This can be done using a data source name (DSN) configured using the Microsoft ODBC Data Source Administrator, or you may use an ODBC connection string.

If using an ODBC connection string, it will take the following generalized form. Note that none of the parameters are encrypted or hidden.

```
DRIVER=VTScada ODBC Driver;SERVER=Your_Server_Address;REALM=Your_Realm;UID=D=Account_Name;PWD=Password
```

If using a DSN, open the Microsoft ODBC Administrator utility on your computer. There will be two such utilities: one for 32-bit connections and one for 64-bit connections. A search for "ODBC" from the Windows Start button should find both. Choose based on the program you will be using to generate the queries. VTScada can work with either.

Open the System DSN tab of the Administrator utility and create a new connection, selecting the VTScada ODBC Driver.

Examples:

- A configured DSN. The password cannot be viewed.

The screenshot shows the 'VTScada ODBC Setup' dialog box. It contains the following fields and controls:

- DSN:** WaterServerDSN
- Description:** ODBC Interface to VTScada History
- VTScada Server:** << your server name goes here >>
- Port:** 80
- SSL:** ☐
- Use backup servers:** ☐
- VTScada Realm:** WaterTreatment
- User ID:** Chief_Engineer
- Password:** Masked with 10 dots
- Connection timeout (sec):** [Empty text box]
- Query timeout (sec):** [Empty text box]
- Max. concurrent connections:** [Empty text box]
- Buttons:** Test Connection, OK, Cancel

Figure 12-35 Example of a new DSN in the ODBC Admin tool

In the following examples, square brackets indicate text for you to replace. They are not part of the connection string.

- Connection string for a secured application:

```
DRIVER=VTSkada ODBC Driver;SERVER=123.456.123.456;REALM=[RealmName];UID=[User-name];PWD=[Password]
```

- Connection string for a realm configured to use a port other than the default 80, or to use a security certificate:

```
DRIVER=VTSkada ODBC Driver;SERVER=[ServerAddress];REALM=[RealmName];UID=[User-name];PWD=[Password];PORT=443;SSL=YES
```

Connection string that includes a five-minute timeout:

```
DRIVER=VTSkada ODBC Driver;SERVER=[ServerAddress];REALM=[RealmName];QUERYTIMEOUT-T=300;UID=[UserName];PWD=[Password]
```

Notes:

- The DSN name you enter should be as descriptive as possible. Be aware that long names might not save properly.
- For the VTScada Server field, enter the fully qualified domain name of the machine running the VTScada application.
- The port should match what was configured in VTScada for the Realm.
- The SSL box is selected only if the VTScada Thin Client Server has been configured to use a security certificate. (Strongly recommended.)
- Selecting 'Use Backup Servers' causes a list of VTScada Thin Client Servers to be retrieved from the VTScada ODBC server and stored, any time a successful connection is made. If future attempts to connect to this server fail, the list of VTScada Thin Client Servers will be tried one-by-one for a connection
- Connection Timeout. Initially blank, defaults to 10 seconds. The connection timeout is how long the driver waits for a response from the target VTScada server. If the user application is making many concurrent queries then if the actual number of attempted concurrent connections through the driver instance exceeds the configured amount, then a new connection will be stalled until a connection "slot" becomes available. In such a case the user might need to increase the timeout or max. number of concurrent connections.
- Queries that take longer than one minute to complete will time-out. If you have queries that need more time, you can set a value larger than 60 here.
- Use the Max. concurrent connections option to adjust the limit of concurrent connections. If the application is running multiple long-running queries then increasing this number might reduce timeouts at a cost of increased resource usage on both the driver host and the target VTScada server. Simply increasing this number because a client is making lots of expensive queries will cause performance issues on the target VTScada server.

Note: If security realms are enabled (NamespaceDelimiter is set), then the username for ODBC queries should be configured for the realm. For example, if you have a security realm named "ODBC" and a NamespaceDelimiter of ":", then you need a VTScada Thin Client Server realm named "ODBC" and a user account within the matching security realm, "ODBC:username".

If querying tag parameters, the account must have the Tag Parameter privilege.

- The Test Connection button should succeed if the application is running.

Troubleshooting:

If there is no connection, confirm that all the following are in place:

- Server and Realm configuration completed on the VTScada server.
- Security is active and the user account has the VTScada Thin Client privilege.
- The querying workstation is able to connect to the server.
- If using a DSN, you are using 32-bit or 64-bit as required by your querying program.
- The connection string or DSN uses the correct names and addresses as configured on the Server.

¹Note to VTScada programmers: Configuration of an ODBC server is necessary for those making remote data connections to VTScada from third-party programs. It is not required for those writing SQLQuery statements in their VTScada modules.

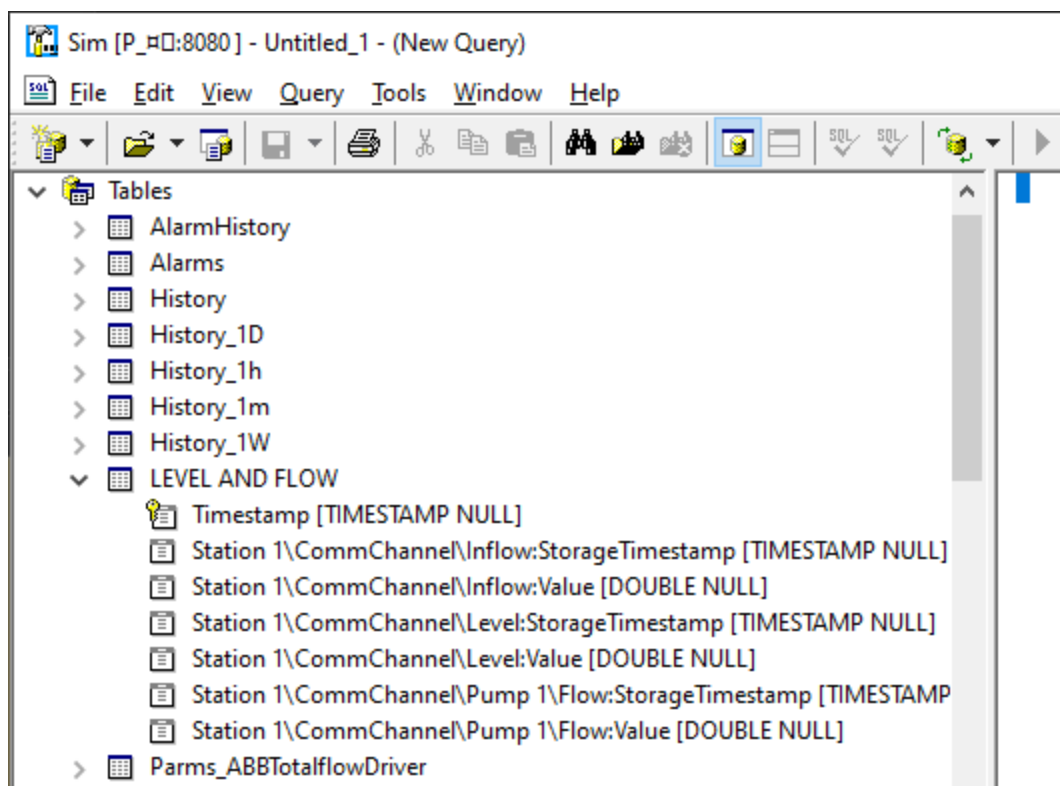
SQL View

This is a tag intended for customers who are using VTScada's Remote Data Access and ODBC features to query the application's stored history. Use it to create views of data within your application, for use by SQL querying tools.

Without this tag, 3rd party query tools must connect to the virtual History table (or one of the time-aggregation modes of that table), which makes all information from all tags available. Typically, this is far more information than most 3rd party programs can work with. As an example of how many columns there can be in the virtual History table, consider just the data available from driver tags: each driver records seven values tracking the driver's health (quality, error, etc.). If you are querying a History_TPP table (data aggregation) then there are seven columns for every aggregation time period, for every driver.

Note also that VTScada will limit the number of columns returned in any query to 64.

When you create an SQL View tag, you specify what data from which tags you want to see, then run your queries against that. For example, in the following image, the table, LEVEL AND FLOW, as seen in a 3rd party ODBC query tool, is the result of an SQL View tag. It contains only data from tags selected for the view.



Every view must have a name, which is configured on the View tab.

By default, views include only the historical values of tags selected in the Tag Filters tab. Use the Value Attributes and TPP Modes to extend the view to include selected flags that may be relevant to the data returned, or value aggregations such as averages or maximums over a defined time period.

If you are interested only in tag values and not in data aggregation over defined time periods, there is no need to select a TPP mode.

New SQL View Properties

View Name

Value Attributes

☐ Value!

☐ ManualData ☐ Questionable

☐ ManualEntry ☐ StaleData

☐ EditedData ☐ Imported

TPP Modes

☐ Average ☐ ZToNZCount

☐ Minimum ☐ NonZeroTime

☐ Maximum ☐ Total

☐ Delta ☐ Interpolated

☐ ValueAtStart ☐ RolloverTotal

☐ TimeOfMin ☐ BitwiseOR

☐ TimeOfMax ☐ BitwiseAND

Privilege

No Security

OK Cancel Apply

Figure 12-36 Optional attributes and calculations of an SQL View

The Tag Filters tab does what it says. The selection tool is very similar to the Tag Selector used by the Historical Data Viewer. You can choose specific tags, but a better option is to use a query. A query is simply the filter that you create using the tools at the top of the Tag Selector dialog. For example, setting the name filter to `Station 1*` results in a query that includes only the tags within that station. Adding filters for area, type, etc. further restricts the set that will be included.

REST Queries - Notes & Examples

REST stands for REpresentational State Transfer. It allows web applications to query data using either HTTP GET or POST requests, and a subset of SQL. Results are returned in the JSON format of name and value pairs. Result sets are limited in size and pagination codes are used to allow you to retrieve subsequent portions of data returned from the original query. This greatly reduces the time required if querying a large data set.

Note: Information is returned as a (potentially large) set of name / value pairs, suitable for an application that can process JSON data. It is up to you to create that application in order to organize and format the results. Part of this should include a convenient user-interface to build the queries and to request the next page of a large data set.

Note: The pagination is supported only when retrieving historical data from the History table, the custom table name set by `\SQLQueryHistoryNoOverridesTableName`, and the History TPP tables.

Retrieving alarm history doesn't support pagination.

Given a VTScada server configured for ODBC and a user account that has the Remote Data privilege plus other privileges as described in Security and Remote Data Access, you can query data using a URL in the following form:

```
https://Your.ServerName.com/RealmName/REST/SQLQuery?query=Select Something...
```

(Example assumes that you have configured security Use "http://" otherwise.)

Caution: Do not name any realm, "Rest" or "SQLQuery". Doing so will interfere with remote access to VTScada data.

Tip: Examples of several common queries are provided in the file `REST Interface Demo.html`, distributed with your copy of VTScada. Assuming that you installed to `C:\VTScada`, you will find the file within the folder: `C:\VTScada\Examples\REST Interface Example`.

To use the examples file, you must have an application configured for Remote Data Access and a user account in that application with the Remote Data privilege. Full instructions are provided within the HTML file.

Tip: For greater security, consider using one or more of the [SQL View Tag](#), each of which can be assigned a custom privilege. The Remote Tag Value / History Retrieve privilege is necessary in order to run custom SQL queries. Accounts with only the Remote Data Access privilege can see only their assigned SQL views or AlarmHistory, and cannot run their own SQL queries or browse the tag parameter tables.

For GET requests, the query parameter should always be fully URL encoded. For example, '%' must be encoded as '%25', and spaces should be escaped as '+' or '%20'. This applies to all special characters. (Further information is available at https://en.wikipedia.org/wiki/URL_encoding and <https://developer.mozilla.org/en-US/docs/Glossary/percent-encoding>)

If using JavaScript you might do this with [encodeURIComponent](#):

```
const encodedQuery = encodeURIComponent("GET Tables");
```

If using .NET, you might do this with `URLEncode`:

```
System.Web.HttpUtility.UrlEncode
```

Notes for using POST requests

For POST requests, the query parameter should never be URL encoded.

If using POST requests, the body of the request cannot be empty and must be a JSON encoded string that contains the query and an optional pageToken parameter. The pageToken parameter may be empty if not using pagination. For example:

```
{
  "parameters":{
    "query": "...",
    "pageToken": "..."
  }
}
```

```
}
}
```

The only accepted "Content-Type" header value is "application/json".

Samples of queries are provided in [SQL: Reference and Examples](#).

Limitations

The default limit for the POST request body size is 32768 bytes (32KB), which is same as the limit for GET requests. You can send POST requests longer than 32KB by adjusting the setting [SQLQueryMaxRequestLength](#) in SETUP.INI.

By default, a maximum of 64 columns and 10,000 rows will be returned, and a pagination token added to the return structure if there are more rows available. You can disable pagination by setting the property [SQLPaginationEnabled](#) to 0 (FALSE).

If your query uses the wildcard "Select * from...", it is likely that the interface will return an error message due to the result containing more than 64 columns, unless the application has very few tags.

Note: URL encoding will be required for some characters. An example follows.

When making queries with GET requests, if your query includes a LIKE clause you must encode the % wildcard character as %25. For example:

```
...WHERE Area LIKE 'East%'
```

Will fail because the browser will not encode the % properly. Instead, use:

```
...WHERE Area LIKE 'East%25'
```

Returned data

Examples in this discussion use data returned from the following query:

```
select timestamp,'Station 1\PLC1\Level:value' from history
```

would be placed in the URL and encoded as follows:

```
.../REST/SQLQuery?query-  
y=select%20timestamp%2C'Station%201%5CPLC1%5CLevel%3Avalue'%20from%20history
```

Header

The header describes the names and types of the returned data. Values follow immediately after. For the query shown, this will look like the following. Note that timestamps are always returned as UTC values.

```
{"results":{"fieldNames":["timestamp","Station 1\\PLC1\\Level:value"],  
"fieldTypeNames":["TIMESTAMP","DOUBLE"],"values":  
[[1562084981.054,12.603042138914],  
[1562085057.977,12.503052503053], ...
```

Body

The body of the returned data consists of the comma-separated values of the returned data where each record is enclosed in square brackets. Two records are shown in the preceding example.


```
Select Timestamp, 'WaterTreatmentPlantDemo\TCPIP Port\Water-
TreatPlantSim\Tanks\Tank0\TankLevel:value' from history order by timestamp desc
limit 10
```

URL:

```
https://vt-
s.tri-
hed-
ral.-
com/wa-
ter-
demo/REST/SQLQuery?query-
=Select%20Timestamp%2C%20'WaterTreatmentPlantDemo%5CTCPIP%20Port%5CWaterTreatPl-
antSim%5CTank-
s%5CTank0%5CTankLevel%3A-
value'%20from%20history%20order%20by%20timestamp%20desc%20limit%2010
```

2) For this example, you can use either Command Prompt or a Windows PowerShell window. It will not work from a web browser. This is GET request using basic authentication:

```
curl.exe -u demo:demo https://vt-
s.trihedral.com/waterdemo/REST/SQLQuery?query=GET+Tables
```

3) This next example works only in PowerShell due to quoting being PowerShell-specific. This is a POST request using basic authentication.

```
curl.exe "https://vts.trihedral.com/waterdemo/REST/SQLQuery" -X POST -u demo:demo
-H "Content-Type: application/json" --data-raw ""{"parameters":{
{"query":"GET Tables"}}""
```

4) The final POST example stores the query in a JSON file, "query.json". To run this, create a file named query.json and within it store the following:

```
{
  "parameters": {
    "query": "GET Tables"
  }
}
```

You can now run the following query:

```
curl.exe "https://vts.trihedral.com/waterdemo/REST/SQLQuery" -X POST -u demo:demo
-H "Content-Type: application/json" --data "@query.json"
```

The Report Studio

Use the VTScada Report Studio to create your customized reports. Open this tool from the [Reports Page](#), which is typically part of the Alarms, Reports & Diagnostics folder.

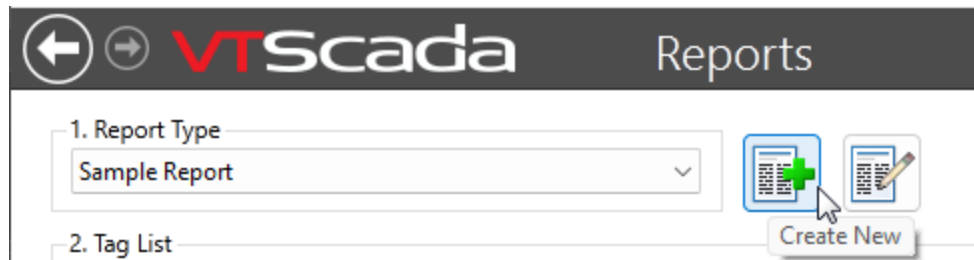


Figure 12-37 Creating a new report from within the Reports page.

Reports are tables of information where you select which tags to include, what calculations to perform, and what summary information to calculate. Any report can contain multiple tables, allowing you to create comprehensive overviews of your system.

In addition to the tables, you can add images, spacers, and text to provide clarity and context.

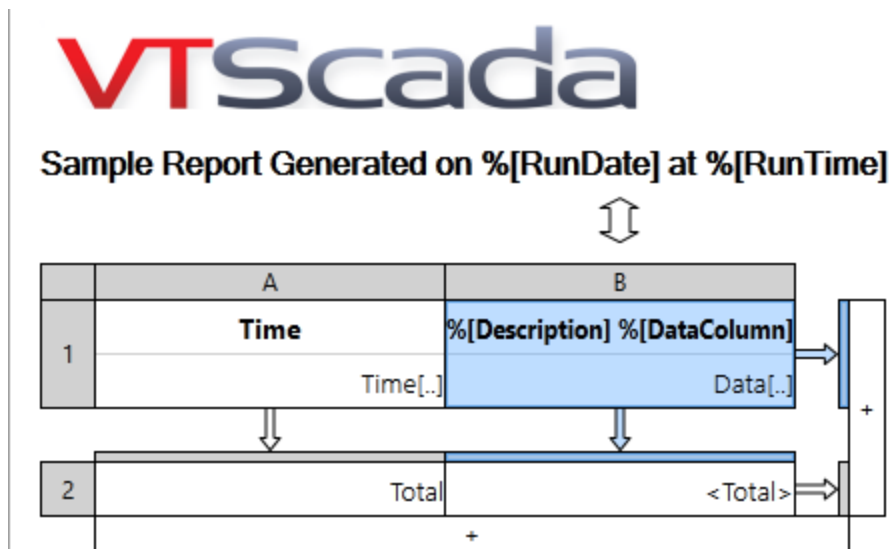


Figure 12-38 Report elements in the studio

Sample Report Generated on 2024-03-07 at 11:22 AM

Time	P1 Starts	P1 Running Time	P2 Starts	P2 Running Time
2024-03-07 10:22:00 AM	1	130	1	130
2024-03-07 10:37:00 AM	2	264	2	264
2024-03-07 10:52:00 AM	3	316	3	316
2024-03-07 11:07:00 AM	1	219	1	219
Total	7	929	7	929

Figure 12-39 The resulting report

Preview

A preview button in the Report Studio gives you a way to check your work. After you click Preview, a pop-up prompt will ask you for a time span. Note that what matters is the time and date shown rather than the current preset, the list of which serve only to give you a quick way to set time and date.

The destination for a preview is always the screen.

Tips:

Column A is typically used for Time but can be used for data retrieval. If you add a summary row to your report, column A must display a Time column so that room is left for the summary labels.

Column B will expand into as many columns as needed to show the results of your query.

Column C, etc., can be used to add additional tag queries to a table. It does not display summary data for a row.

All data columns other than Raw Timestamp and Raw Data are retrieved with a common time interval per row (Time per Record), which is set for the table as a whole. *Click in the upper right corner of a table to set this value.* You can set the time per record as large or small as needed, but cannot set it to 0 (zero).

If reporting on Raw Timestamp and Raw Data values for more than one tag, note that the column for each tag is generated independently. Timestamps are unlikely to match across any row.

If the available list of data columns does not include the calculation you're looking for, don't forget that you can create tags from the Analytics group including I/O and *Calculation* tags to generate additional statistics. (Referring here to the selection of tags for a query, not the summary information compiled at the bottom of a column.)

Add rows showing summary statistics by clicking the plus at the bottom of the table and then configuring the summary row. You can have multiple summary rows as needed, up to the number of possible summary actions.

If your table has multiple tag query columns, not all summary statistics will apply to all queries. Suppress statistics that do not apply on a column-by-column basis (referring to configuration columns B, C, etc., not output columns).

Under the current configuration, there is a limit of 30 columns and 10,000 rows per report.

All elements in a report can be duplicated by doing a copy and paste (Ctrl-C / Ctrl-V). Copied elements are always added to the bottom of the report.

The table and parts within it are configured by clicking on a region and then using the panel on the right edge of the Report Studio to customize the selected region. Note the blue highlight, indicating which aspect of the table you have selected to customize.

Tip: You might decide to create a series of very similar reports. A quick way to do this is to re-open the first and then use the Save As button to save it to a new report. Edit the new report as needed. By saving first, you eliminate the risk of accidentally replacing the new report with the old.

The definition of each report that you create is stored as a JSON file in the folder named "ReportDefinitions" within your application. You are advised not to edit those files directly.

The VTScada installer offers WebView2 as an option during installation. Ensure that WebView2 is installed on your VTScada workstation in order to use the Report Studio, and to generate PDFs of reports built using the Report Studio. For the VTScada Internet Clients (VIC), WebView2 should be installed on the machine running on the VIC in order to display report previews in the Report Studio. For workstations without internet access, WebView2 can be downloaded directly from Microsoft. Refer to notes on this topic within the description of the [Web Browser Widget](#)

Caution: As noted elsewhere, VTScada should not be run under an account that has Windows Administrator privileges. This remains true when running as a service. One of the problems you will encounter if attempting to do so is that WebView2, required in order to generate a PDF report, will not run when VTScada is running as a service and either of the following is true:

- VTScada is running as the "Local System" account.
- VTScada is running under an account that is part of the Administrators group.

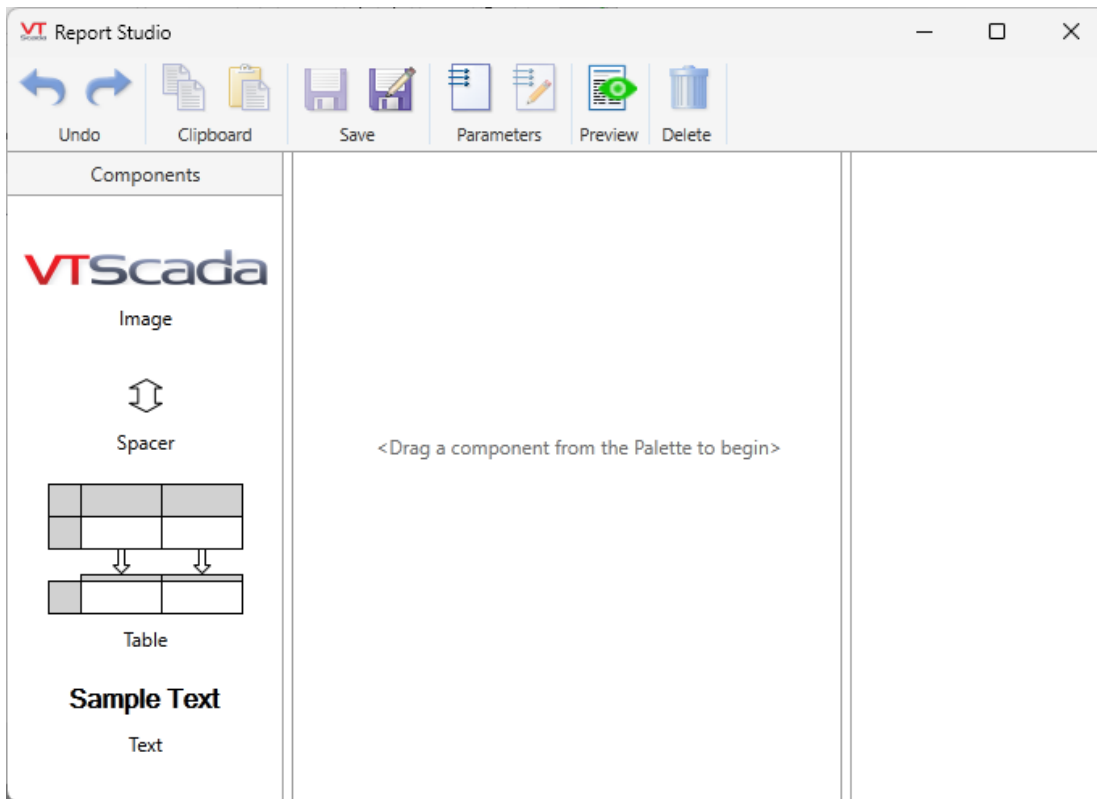
Exercise 12-4 Create a New Report

1. Navigate to the Reports page.
(Usually found in the Alarms Reports and Diagnostics folder.)
2. Click the Create New button.
3. Drag the VTScada image from the Components palette to the canvas.
4. If you have a company logo or other image available on your computer, feel free to use the Import feature to change the image from VTScada to your own. The steps are left as an exercise for the student, but reference notes are available if needed: [Import Images](#)
5. Change the image width to 500 (pixels).
6. Use the Save tool to save the report as `All Flow Rates`.

You will continue developing this report in further exercises.

Report Studio Tools

Tools in the report studio are listed here. Descriptions and instructions can be found in the following topics.



Tools

Undo / Redo

The result of a Report Studio editing session is stored in the VTScada Version Control system. Steps within an editing session are not. Use these buttons to back up while editing a report.

Clipboard - Copy / Paste

Reuse elements within a report. Pasted copies are always added to the end of the report and can be moved as required from there.

Save / Save As

As labeled.

Parameters - Manage / Values

Use "Manage" to create and edit parameters that allow you to (for example) use the same report for any selected station. Use "Values" to select (again, for example) the station to use while building and previewing the report.

Preview

Run a sample report to check your work so far.

Delete

Remove the report.

Components Palette

Drag components from the palette to the report body. When a component is selected in the canvas area its properties are available for editing in the right-most panel. Tables are always centered within the report body. Decorative components can be left-, right-, or center-justified relative to tables.

Image

Add a logo or other image to your report. See: Add Decorative Elements to Reports

Spacer

No space is left between report elements other than what you define using spacers. See: Add Decorative Elements to Reports

Table

Tables define the body of the report. See: Configure Table Appearance and Data Aggregation

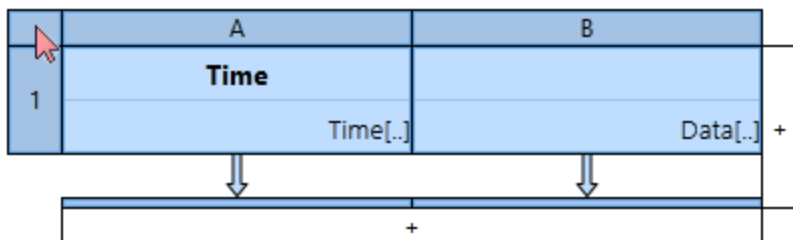
Text

Add a title, explanation, or other notes to the report. See: Add Decorative Elements to Reports

Configure Table Appearance and Data Aggregation

Tables hold the information that you want to include in your report. The columns in those tables are populated by queries that you create. You can adjust table properties to control how the result is presented.

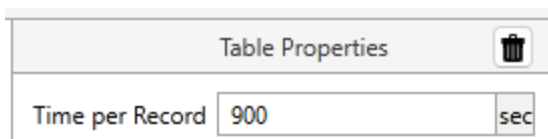
Add tables to your report by dragging them from the palette in the Report Studio. The Table Properties panel will open whenever the table as a whole is selected, usually by clicking in the upper left corner of a table as shown:



With the Table Properties panel, you can do the following:

Delete the table

Click the trash can icon in the header of the Table Properties panel:



Set the Time per Record.

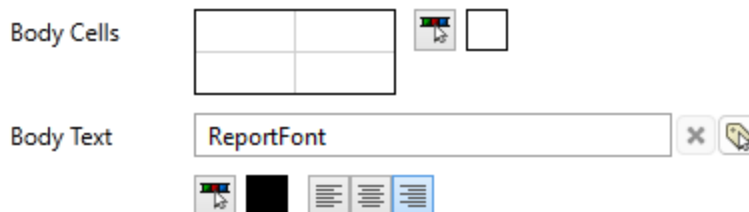
Measured in seconds, this is used for consistent data aggregation from row to row for all queries in the table. Neither 0, nor "Raw data" are an option here. If raw data is needed, select that option when defining your tag queries.

The following table of typical values may be helpful:

Time per Record (sec)	Equals
900	15 minutes
3600	1 hour
86400	1 day
604800	1 week

Set the format of header, body, and summary cells.

Each part of the table can be configured independently. The following image shows the formatting options for the body cells, but the same applies equally to the header and the summary portions of the table. (Except that the header is always a single row.)



For the body cells, click on a line in the sample image to add or remove that line from display in the table. The color selector sets the background color for all cells in that section. Lines are always single width and colored black.

The Body Text (also, Header Text and Summary Text) shows the font selected for use when displaying that text. It is not a space to write your own text. Use the Tag Browser button to choose or create an alternate font. Buttons below the font selection are used to select the default color for the text and whether it should be left-aligned, center-aligned or right-aligned in the cells.

Configure Time Columns

Tip: All tables start with a single column. Click the [+] bar to the right to add more. Note that a tag query that includes multiple tags or calculations will generate as many columns as needed to display the result of the query.

Time values (where the data set type is "Time") are typically displayed in column A of a table. This is recommended because summary labels (if summaries are used) are always placed in the first column and therefore work best below time values, which are never summarized. You are free to choose any column for time and might choose to have several time columns if the format differs or if you want to repeat the timestamp in a report with many columns.

The Time data cell type is recommended for reports with aggregated data. A separate Time column may not be needed if creating an Alarm & Events report.

Whether configuring time or a query, start by clicking in the title row as shown following to open the Data Cell Properties panel.

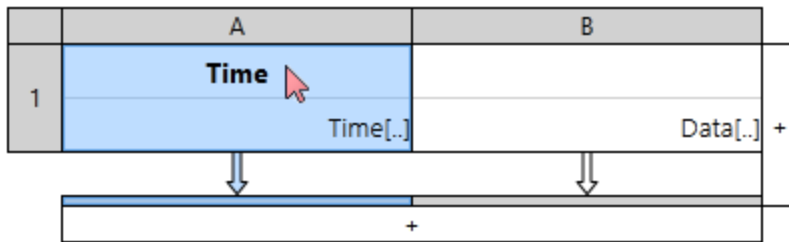


Figure 12-40 Select column A to define a time-based data set

Actions for a Time data set type:

Delete the column

Click the trash can icon in the header of the Data Cell Properties panel. This deletes the column from the table but does not delete the data set definition.

Select or create a Data Set

Every "time" column definition, every tag query, and every alarm and event query is stored as a named "Data Set". You are advised to give each a descriptive name because it is likely that you will eventually have many.

A predefined data set named "Timestamps" is provided. You are free to create more if different reports should display date and time values in different ways.

- Click the pencil button to allow editing of the data set's properties.
- Click the plus button [+] to create a new data set with a name of your choosing.
- Click the copy button to clone a data set.

Select a Type

Must be set to "Time" for this to be a time / date column.

Tip: You are advised not to change your "Timestamps" data set to become a tag query.

Define a Header

Text displayed at the top of the column. Defaults to "Date". Replace with any other wording that is appropriate.

Select a Date Format and Time Format

The format for time and date can be edited by making a parameterized phrase.

Click on button to the right of the field to open the [Make Parameterized Phrase](#) dialog. Multiple formats are available for selection, or you can enter a format of your own using the available [Date Formatting Strings](#) and [Time Formatting Strings](#).

Exercise 12-5 Initial Configuration of All Flow Rates

1. Drag a spacer to the report, placing it below the image you added earlier.
2. Leave the height of the spacer set to 20.
3. Drag a table to the report page, placing it below the spacer.
4. If the Table Properties panel is not open, click once on the square in the upper-most left corner. This should highlight the whole table and open the Table Properties panel.
5. In the Table Properties panel, set the Time per Record value to 900. (15 minutes)
6. Note the other table properties.
This exercise does not ask you to change any of those properties but feel free to experiment.
7. Click once in Column A to select the Timestamps data set. (This is the default data set, as shown in the Data Cell Properties panel.)
8. Click the pencil icon to the right of the data set name to enable editing.
9. Change the title to Time Period.
10. This exercise does not ask you to change the time or date formats but feel free to experiment. Advanced users might experiment with date or time formatting strings as described in the notes above this exercise.
11. Click the Save tool to save your work so far. The exercise will continue.

Create Tag Queries

Tip: All tables start with a single column. Click the [+] bar to the right to add more. Note that a tag query that includes multiple tags or calculations will generate as many columns as needed to display the result of the query.

Tag queries are typically placed in column B of a table. Recommended practice is place a Time-type data set in Column A because summary titles (if summary lines are configured) are always placed in the first column, therefore any data that you intend to summarize should be placed in other columns. (Timestamps are not summarized.)

Note: If your query involves more than one tag or more than one calculation, additional columns will be created as required when the report runs, to a limit of 30 columns. You do not need to create a separate column for every tag and every calculation.

Whether configuring a Time column or a query, start by clicking in the title row as shown to open the Data Cell Properties panel.

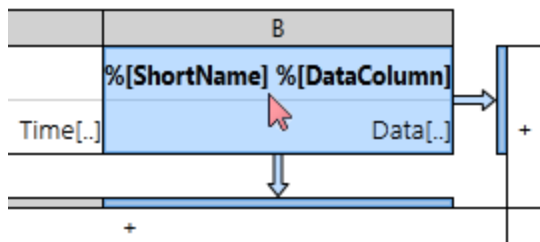


Figure 12-41 Select column B to define a tag query data set

To Create or Edit a Tag Query

In the Data Cell Properties panel, click the pencil icon to edit an existing query or click the [+] button to create a new one..

Name	Area	Description	Field	Data Source Type	Type

Tag Selector

Data Columns

Average

Group Data Columns By Tag ☒

Summary Suppression

☐ Total
 ☐ Average

☐ Minimum
 ☐ Standard Deviation

☐ Maximum
 ☐ Median

☐ Range
 ☐ Count

Figure 12-42 A new Tag Query being created.

With the Data Cell Properties panel, you can do the following for Tag Query data sets:

Delete the column

Click the trash can icon in the header of the Data Cell Properties panel. Note that deleting the column removes it from the table but does not remove the data set definition.

Note: There is currently no way to delete or rename a data set definition.

Create a Data Set

Every "time" column definition, every tag query, and every alarm and event query is stored as a named "Data Set". You are advised to give each a descriptive name because it is likely that you will eventually have many.

No predefined data sets exist for tag queries. You must create one using the tags in your application.

- Click the plus button [+] to create a new data set with a name of your choosing.
- Click the pencil button to allow editing of the following properties.
- Click the copy button to clone a data set.

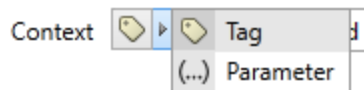
Set the Type to Tag Query

The type property must be set to "Tag Query" for this column to include tag data from your application. This also applies to the Raw Timestamp column.

Set a Tag Hierarchy Context [Optional]

Context can be used in two ways:

- As a filter, limiting tag selection to those below a parent tag such as a station.
- As a parameter, allowing the same report to be run for multiple contexts or stations.



To use Context as a filter, select a parent tag. The Tag Selector will show only child tags of that parent.

To use Context as a parameter, you must first create a parameter. Refer to Parameterized Reports. Note that a value must be assigned to the parameter for use in the Report Studio before you can continue with further steps. Proceed to tag selection as usual. When operators run the report, they will be prompted to select a station or other parent tag to use for that instance of the report. Use [Report Tags](#) to predefine a choice for the parameter.

Open the Tag Selector

Click on the Tag Selector button to open the Tag Selector Dialog, where you can choose the tags to include in the report. If a Context was set, only child tags of that parent will be shown. The order of the tags in the Tag Selector dialog will affect the order of the columns in the report.

Note that it is often helpful to define a query rather than select specific tags.

Add Data Columns

Data columns are the calculations applied to tag values. Examples include Average, Minimum, Run Time and also "Raw Value".

Add as many data columns as needed for each tag in the data set by clicking the <<Click to Add>> tool at the bottom of the list. There must always be at least one data column for every data set. Values in a report can be averages, snapshots, minimums, etc. A list is provided within this topic.

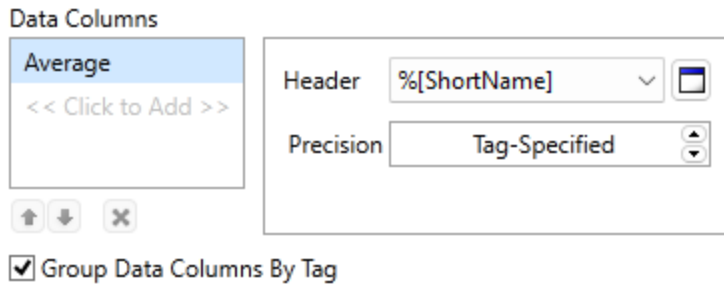


Figure 12-43 Select each column to adjust header and precision.

If the data column displays a numeric value (e.g. Average, Minimum, ...) you can specify the header text for the column and the precision with which the value is displayed. The default is to use the precision specified with the tag's display configuration. Otherwise, you may specify a pre-

cision between 0 and 9.

If the data column displays a time value (e.g. Time of Minimum, Raw Timestamp, ...) you can open the [Make Parameterized Phrase](#) dialog to display time and date in the format of your choice. This is usually required for Raw Timestamps, which are measured to the millisecond.

If reporting on Raw Timestamp and Raw Data values for more than one tag, note that the column for each tag is generated independently. Timestamps are unlikely to match across any row.

To remove the default function, add a second data column using the function you want, then delete the first. Arrows to the side of the Data Columns list allow you to change the order of the functions.


Group Data Columns By Tag

Given a Data Set with multiple tags and multiple data columns, the default is to group the output by tag. You could choose instead to group by the data column function. For example, a set of columns showing all of the averages, then a set of columns showing all of the minimums, etc.

For complex groupings, you might use multiple tag-query columns. (C, D, etc...)

Define Header Text

Sets the title for each column. Because columns are generated automatically according to your tag query, it is typical to use parameters to define titles dynamically.

Select from the drop-down list of predefined titles or click the provided button  to open the Make Parameterized Phrase dialog. Refer to Make Parameterized Phrase for examples.

Summary Suppression

Typically used when *both* of the following conditions are true:

- You have created one or more summary rows
- Your report includes two or more data sets.

If either of those conditions is *not* true, there is no need to suppress summary information.

An example of where you might use this is a pump report that includes a data column for flow characteristics, and a column for pump run time. The first data set might be summarized with a Range. The second might be summarized with a total. For each column you would suppress all summaries that do not apply.

Data Column Reference for Tag Queries

Except for the raw timestamp and raw data options, all calculations are performed for time span defined by the table's time per record value. Refer to Set the Time per Record.

- *Average* - Average of all values within the set time per record (time span).
- *Minimum* - Smallest of all values within the set time per record.
- *Maximum* - Largest of all values within the set time per record.
- *Delta* - Difference between the largest and smallest of all values within the set time per record.
- *Value at Start* - Use for "snapshot" reports. The value at the beginning of each time per record interval.
- *Minimum Time* - Time within the span when the smallest value was recorded.
- *Maximum Time* - Time within the span when the largest value was recorded.
- *Starts* - Number of zero to non-zero transitions.
- *Running Time* - Length of time when the value was non-zero.
- *Totalize* - Adds all the recorded values within each time per record.
- *Start and End Difference* - The difference between the last value recorded and the first value recorded for a time per record.
- *Bitwise AND* - Should be used only with digital tags (those having a value of 1 or 0) Compares the current row to the previous and returns 1 if both are 1.
- *Bitwise OR* - Should be used only with digital tags (those having a value of 1 or 0) Compares the current row to the previous and returns 1 if either are 1.
- *Raw Timestamp* - Use with Raw Data. The time when the value was recorded. A display format that includes milliseconds is recommended.
- *Raw Data* - Returns all values stored rather than a calculation over the time per record interval. Should not be used in combination with other calculations.
- *Runtime Data* - A constant value, calculated at runtime. Use this to display any of the following from a selected tag: ShortName, Relative name, Description, Area, Units. The value will be shown in every row.

Exercise 12-6 Create a Data Set for All Flow Rates

1. Click the [+] box to the *right* of column A. (Not the box below column A!)
2. Click once in Column B to activate that column in the Data Cell Properties panel.
3. In the Data Cell Properties panel, click the [+] button to the right of the data set name.
4. When prompted, name the data set, `Flow Rates`.
The type should be set to Tag Query automatically.

5. For now, you will not set a Context. Leave that field blank.
6. Click the Tag Selector button to open the Tag Selector dialog.
7. In the field, Filter by Name, type `*flow*` and then press <Enter>.
8. Click the Add Query button at the bottom of the Tag Selector dialog.
9. Click OK to save your work.
10. In the list of tag columns, click where it says "Click to Add"
11. Select "Minimum" from the drop down list that you just enabled.
12. Repeat to add "Maximum".
13. For the title, open the Make Parameterized Phrase dialog.
14. Create a parameterized title that looks like the following:
`%[..\ShortName] %[ShortName] %[DataColumn]`
15. Save your work then click the Preview tool.
16. When prompted for the reporting period, adjust the start and end times to cover the last full hour (from X:00 to Y:00).
 (This assumes that you have been running the application through that time period.)

The exercise will continue.

Create Alarm & Event Queries

Note: If your query involves more than one tag or more than one calculation, additional columns will be created as required when the report runs, to a limit of 30 columns. You do not need to create a separate column for every tag and every calculation.

To report on alarms and events, start by clicking in the header row of column A as shown in the following image to open the Data Cell Properties panel. (Timestamps are available within the alarm and event query, and need not be specified in a separate column.)

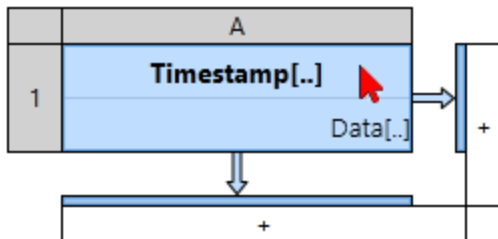


Figure 12-44 Select column A to define an Alarm and Event query data set

With the Data Cell Properties panel, you can do the following for Alarm and Event data sets:

Delete the column

Click the trash can icon in the header of the Data Cell Properties panel. Note that deleting the column removes it from the table but does not remove the data set definition. There is currently no way to delete or rename a data set definition.

Create a Data Set

Every "time" column definition, every tag query, and every alarm and event query is stored as a named "Data Set". You are advised to give each a descriptive name because it is likely that you will eventually have many.

No predefined data sets exist for alarm and event queries. You must create your own.

- Click the plus button [+] to create a new data set with a name of your choosing.
- Click the pencil button to allow editing of the following properties.
- Click the copy button to clone a data set.

Set the Type to Alarm and Event

Must be set to "Alarm and Event" for this column to include information from any of the alarm and event databases.

Select a Database

By default, all alarm and event databases will be queried. Select one or more to restrict the query. For example, you should choose the System Event DB if you want a report that shows only operator actions.

Add Data Columns

Data columns are the alarm and event properties that will be included in the report. Add as many data columns as needed by clicking the <<Click to Add>> tool at the bottom of the list. There must always be at least one data column for every data set. Columns can include any of alarm event information and alarm configuration information but note that columns that are useful for events might not be useful for alarms and vice versa.

- *Timestamp* - The date and time of the alarm or event.
- *Action* - Describes what kind of alarm or event record this is. Also known as "Event". Examples include "Active", "Normal", "Acknowledge".
- *Area* - The area property of the triggering tag.
- *Deadband* - The deadband configured on the associated alarm.
- *Description* - The description property of the triggering tag.
- *Device* - Thin client device, if one was used to work with the alarm.
- *FriendlyName* - The full name of the triggering tag.
- *OnDelay* - The on-delay configured on the associated alarm (if any).
- *OffDelay* - The off-delay configured on the associated alarm (if any).
- *PriorityText* - The priority of the alarm.
- *RearmDelay* - The rearm delay configured on the associated alarm (if any).
- *Setpoint* - The setpoint to trigger the alarm.
- *SetpointLabel* - Text associated with the value used for a setpoint (i.e. for discrete tags).
- *Units* - The engineering units configured in the triggering tag.
- *Username* - The name of the user associated with the alarm action or event.
- *Value* - The actual value of the tag when the alarm was triggered.
- *ValueLabel* - text associated with a value (i.e. for discrete tags).
- *Workstation* - The name of the computer where the event occurred.

Define Filters

Use filters to define the alarm or event report that you want to generate. For example, to create a report of all alarm acknowledgments you should ensure that the System Alarm DB (or your own alarm database) is selected and add an Event filter of "Ack". Columns might include Timestamp, Action, User, Description.

Filters match those available within the Alarm Page except that here you can add multiple criteria to each category to be ORed together. e.g. User of "a" or user of "b". Different categories are ANDed together. (See: Sort and Filter the Alarm List).

Description

Filter using the text of the alarm's description field. For the description field only, the filter will include all alarms that include the text you provide, not just those that match the text exactly.

This filter can be made case sensitive.

Name

Filter using the name (or unique id) of the alarm tag. This will find only tags with an exact match for the name. Use leading or trailing wildcards to expand the filter to names that include the provided text.

Select the option, Include children, to filter for a parent tag name and populate the list with all alarms in all child tags of that parent.

Area

Use the drop-down list to select one area property for the field. You may type an area value if you would like to expand the filter using wildcards. For example: Zone* will filter for alarms in Zone 1, Zone 2 and Zone 3.

Priority

The priority option creates a filter matching alarm priority values. Use this to filter the list to show only Critical or High alarms. By selecting the "Or higher priority" option, you can filter for High and also include Critical alarms in the list.

User

Enter the account name of an operator to filter for alarms and events attributed to that operator. It is not possible to filter for more than one operator at a time.

All operator actions including setting output values and shelving alarms, are attributed to the operator signed in at the workstation where the action occurred. Each alarm is attributed to the operator who acknowledges it. Unacknowledged alarms will not be included in this filter.

Event

Select one of the predefined event types from the list to filter for all alarms or events matching that type. Event types are described in [Alarm Event Reference](#)

Workstation

Enter the name of a workstation to restrict the report to the events associated with that computer. It is not possible to filter for more than one workstation at a time. (WorkstationA OR WorkstationB is not an option.) No event is associated with multiple workstations (WorkstationA AND WorkstationB would always return an empty report.)

Notes

By default, retrieval is restricted to a maximum of 10,000 rows. You can modify this as required.

Summary information is typically suppressed for alarm and event data sets.

Add Additional Columns [Optional]

Note: If your query involves more than one tag or more than one calculation, additional columns will be created as required when the report runs, to a limit of 30 columns. You do not need to create a separate column for every tag and every calculation.

One example of why you might want a column C is if you are creating a report of flow rates and also need to track the matching pump run times. In column B you would create a tag query that includes the tag measuring the flow rate. In column C you would create a tag query that includes the matching pump status tag.

Another reason to create an additional column is that your data might require different summary operations. Perhaps one column should be summed and the next should be averaged. By creating separate tag queries in separate columns you can suppress summary operations for each to show only those that are appropriate.

Add Additional Columns

Begin by clicking the [+] box to the right of column B to create column C.

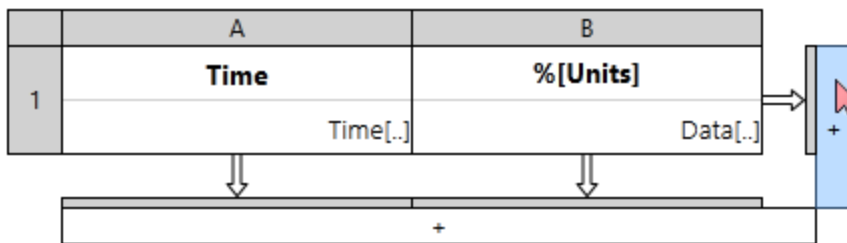


Figure 12-45 Adding a new data set column.

You will now have a new column, in which you can select or create another new data set. Proceed as described in Tag Query Columns for the Report Studio.

Add Summary Rows

Use summary rows to display totals, averages, and other statistics for the values in each column other than the first, which will contain the label for each summary.

Create a Summary Row

Click in the [+] bar below the main part of the table. You can add as many summary rows as needed, up to the number of possible operations.

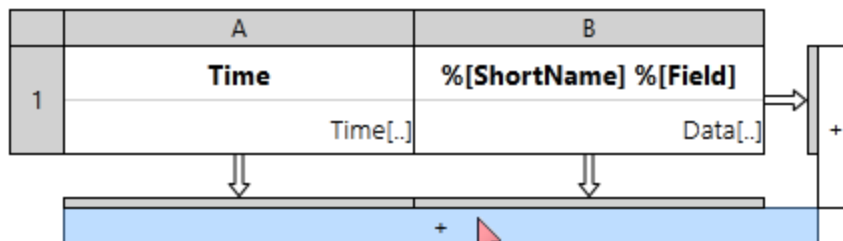


Figure 12-46 Adding a summary row

The row is added and can be configured using the Operation selector and Title, shown in the right-hand panel.

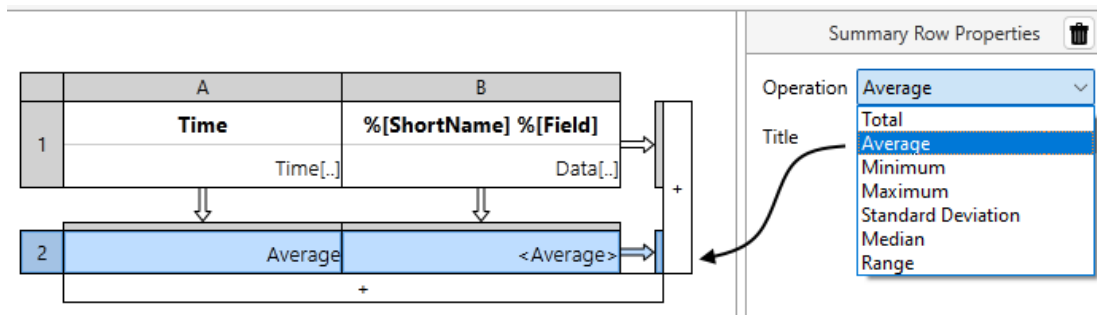


Figure 12-47 Selecting the operation for a summary row

By default, labels match operations but you are free to replace these with any phrase you prefer. The label will always be displayed in the first column.

When the report runs, the selected operation will be applied using all the values in the column above it.

The operation will apply to all columns for all tag queries. If your table includes multiple data sets it is likely that operations may be relevant to some columns but not others. In this case, update each column's configuration to suppress summary operations that are not useful to the query. Refer to notes for Data Cell Properties -> Summary Suppression in Create Tag Queries.

Available Summary Operations

Multiple operations can be selected. Each will be shown on its own row below the table of reported values.

- Total
- Average
- Minimum
- Maximum
- Standard Deviation
- Median
- Range
- Count

Most operations should be clear based on the name but note that "Count" will return the total number of valid entries in a column, not the total number of non-zero entries. To create a count of non-zero or other specific entries, copy the report output to Excel, where you can do operations such as COUNTIF. Within a VTSceda report, Count is most useful for data that has not been aggregated. For example you might create a count of alarm event messages over a given time period.

Exercise 12-7 Add a Summary for All Flow Rates

1. Click the [+] box at the bottom of the table to add a summary row.
2. Change the operation to Range.
3. Save your work and preview the report.

The exercise will continue.

Add Decorative Elements to Reports

Decorative elements include spacers, images, and text.

Add Spacers

Represented in the Report Studio as a bar with a vertical arrow, these are replaced by empty space of the specified height in pixels when the report runs. The space in the generated report will match what you see on the screen. (Some variation is possible depending on whether the selected destination is HTML, PDF, or other.) Note that adjacent elements will touch if you do not place a spacer between them.

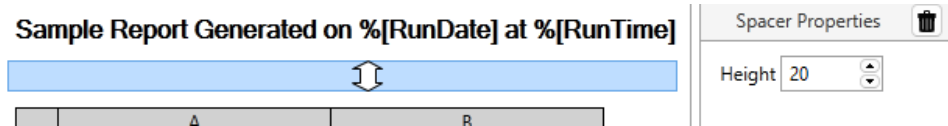


Figure 12-48 A spacer between text and a table

Add Images

The VTScada logo is provided as a sample image. Drag it to the report, then use the Image Properties panel to select and adjust any image you prefer. Click the [...] button to the right of the image name to open the Select Image dialog.

Images must be part of your application before they can be included in the report. To import a new image into your application, click the + button at the bottom left of the Select Image dialog. The process is the much the same as for adding images in the Idea Studio except that images cannot be dragged from the Windows File Explorer to the Report Studio. Refer to [Import Images](#).

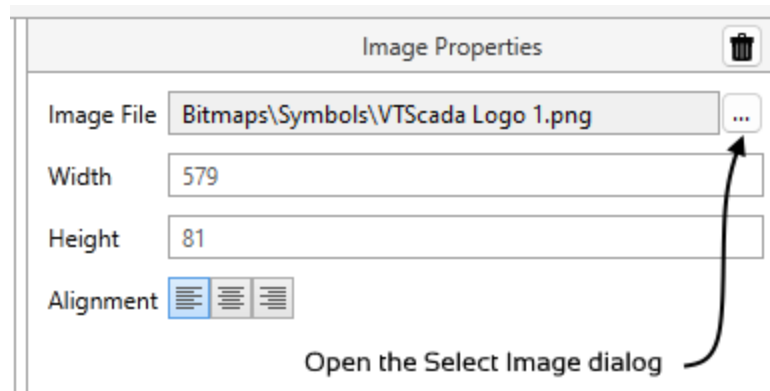


Figure 12-49 The Report Studio Image Properties panel

Add Text

Drag the Sample Text to your report and edit to suit using the Text Properties panel. You cannot include line breaks in your text, but long text will flow over multiple lines as required when the report is built for PDF or HTML.

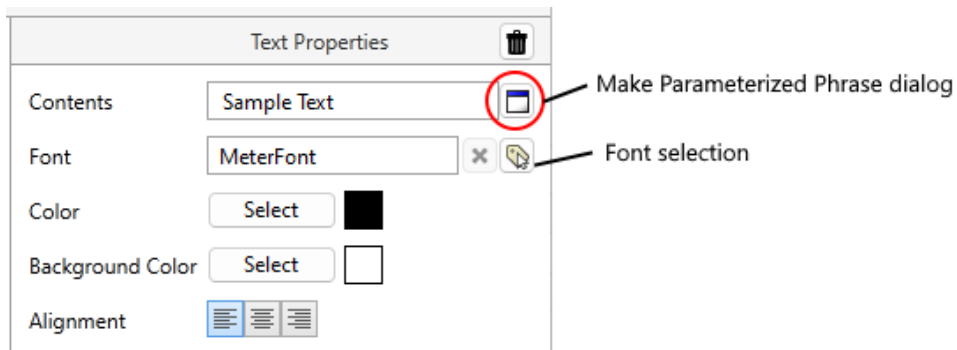


Figure 12-50 The Report Studio Text Properties panel

The Text Properties panel includes a tool for building parameterized phrases. Click the icon to the right of the text entry field (circled in the previous image) to open the Make Parameterized Phrase dialog.

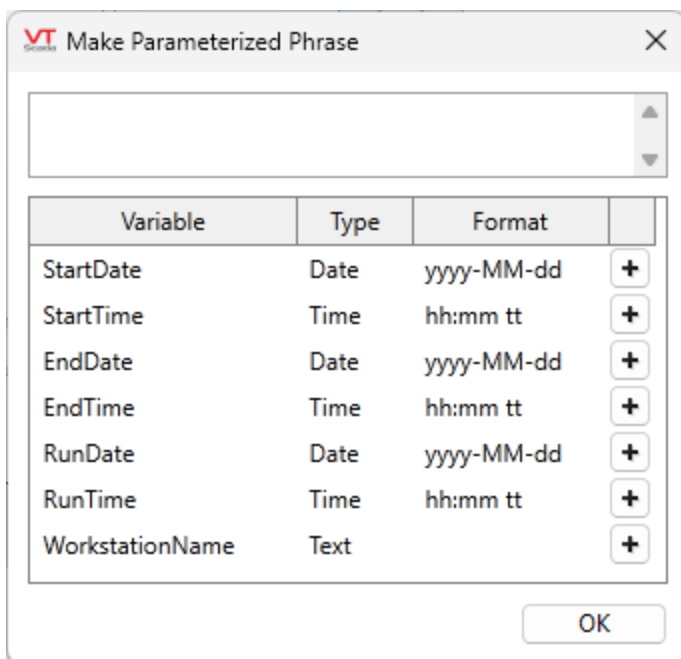


Figure 12-51 The Make Parameterized Phrase dialog

The example shown here includes the standard variables for a report header. Any parameters that you add to the report in general will be also included. When building a parameterized phrase, you can include any text in addition to selected variables. Be sure to leave spaces to ensure that words do not run together unless that is your intent. For example:

Sample Report Generated on %[RunDate] at %[RunTime]

Exercise 12-8 Add a Title to All Flow Rates

1. Drag a text element to the report screen, placing it above all other elements.
2. In the Text Properties panel, open the Make Parameterized Phrase dialog using the button to the right of Contents.

3. Edit your phrase to look like the following:

```
15-Minute Flow Rates - %[StartDate] %[StartTime] to %[
[EndDate] %[EndTime]
```

4. Save your work and create a preview of the report.

Define Parameterized Reports

Parameters can be used select which tags to use for your custom report each time you run it. For example, if your application has several stations and you would like to produce the same report individually for each station, then you can use a parameter.

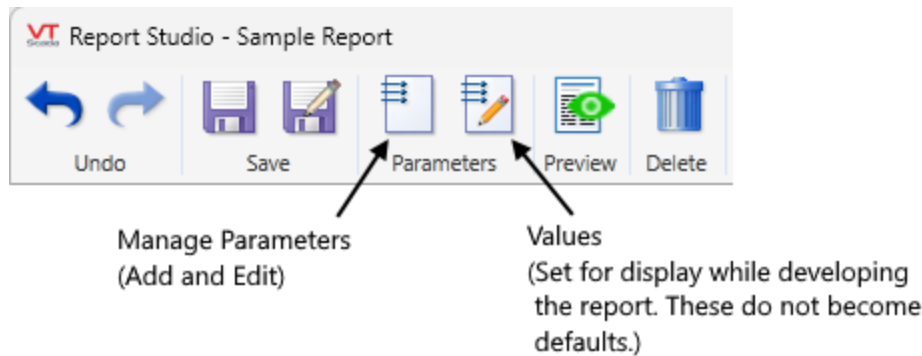
When running a parameterized report from the Reports Page, you will be prompted for the necessary values when the report runs. If running the report with a Report Tag, you can preset the parameter values as part of the Report Tag's configuration.

There are three main tasks:

1. Create one or more parameters.
2. Use those parameters as part of your tag query.
(Optionally, you can also use those parameters as part of any text that you choose to add to the report.)
3. Provide values to the parameters, either when running the report or configured within a Report Tag.

Add and Manage Parameters

Use the tools in the Report Studio ribbon to add parameters:



The Manage Parameters button opens a dialog where you can create new parameters and modify existing ones. An example of adding a new parameter being created follows:

Figure 12-52 Adding a tag parameter.
The type is a user-defined tag in this example

The example shows a new parameter being configured. Recommended practice is to ensure that the parameter name is unique, avoiding any conflict with tags or other named objects. The description field should guide developers or operators when selecting the tag or value for the parameter. If the parameter is of type tag, then you should guide the user by selecting which types can be used. You may use the Ctrl key to select several types.

The same dialog is used when editing existing parameters.

Parameterized pages, plain widgets, and the Report Studio use parameters in the same way and share configuration dialogs.

(For pages and widgets only.) Text parameters can be marked as non-translatable. Select this for text that should never be translated, such as I/O addresses.

Removing Parameters:

By design, there are no tools in the user interface to delete parameters. The easiest way to

remove one is to use the [Version Control](#) system to find the change where the parameter was added to the page and reverse that. Otherwise, if you are certain that a parameter is not needed and that it cannot be edited to use for a new purpose, then experienced developers can remove parameters by editing source code, then importing the changed file.

Values for Parameters

Use the Values tool in the Report Studio to set display values for use while building the report. This allows you to preview the effect of a parameter while continuing development work.

These values are saved as part of the report definition and will be used as defaults if other values are not provided when running the report.

Use Parameters in Tag Queries

After creating a parameter, you can use it in your tag queries and parameterized text. Tag queries assume that the parameter will contain a parent tag and do not work otherwise.

Tip: The following is easier to do if you use the Values tool to select a tag whose child tags should be displayed while you continue configuration.

Working in a Tag Query, set the Context to a parameter that contains a tag as shown in the following example:

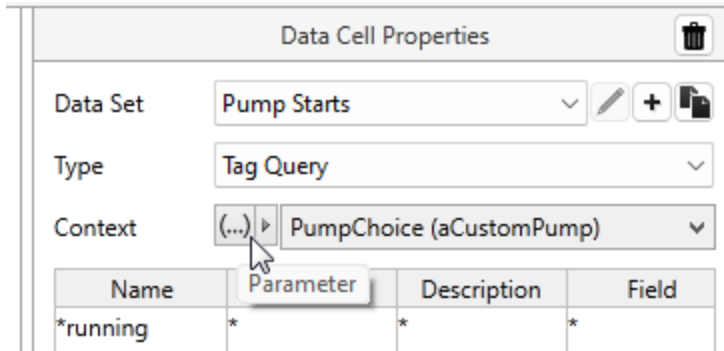


Figure 12-53 A data set with a context used as a parameter

Open the Tag Selector. The available tags are limited to child tags of the one associated with your parameter.

Select the tags for your report or create a query as shown in the preceding example.

([*running | * | * | * |, which is any tag whose name ends with "running".) In a query, "Field" will be "value" for every tag except certain custom-code tags and communication drivers (see [Communication Driver Log-Enabled Variables](#))

Use Your Parameters in Report Text

After adding text to your report, look in the Text Properties panel to find and click the button to open the Make Parameterized Phrase dialog.

Your parameter will be available for inclusion in the phrase. For tag parameters, Name, ShortName, and Description will be included.

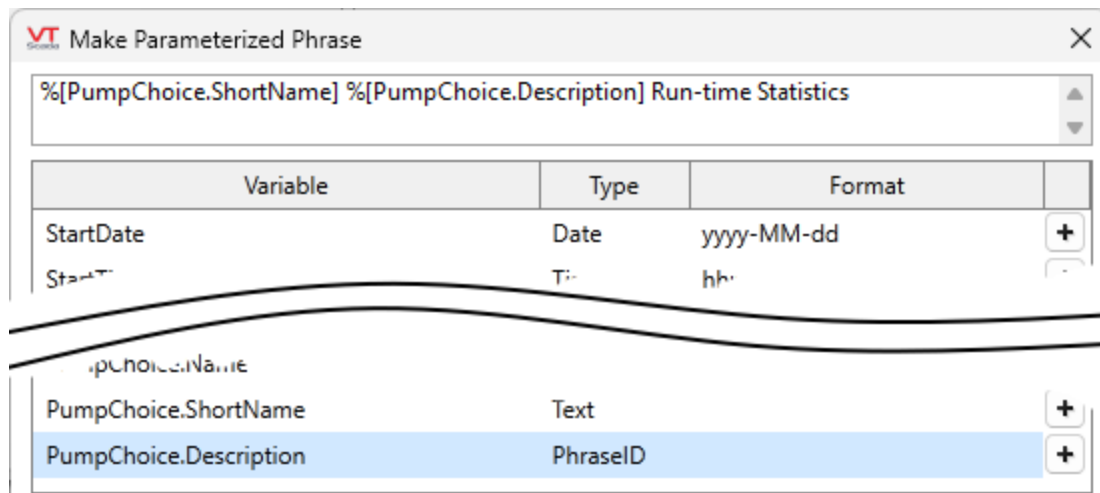


Figure 12-54 Tag properties for a parameter.

Supply Parameter Values when Running the Report

When running your parameterized report from the Reports Page, users will be prompted to provide values for the parameters.

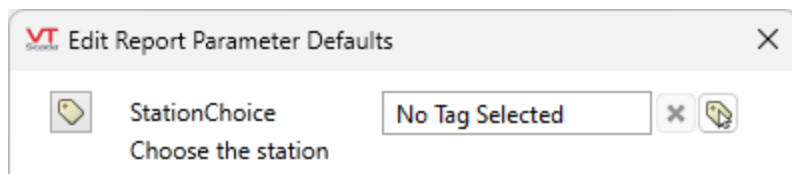


Figure 12-55 Sample prompt for a parameter when running the report

If configuring a Report Tag to run your report, you can set values for your parameters in the tag as shown in the following image. To run the report for multiple stations, create multiple Report tags.

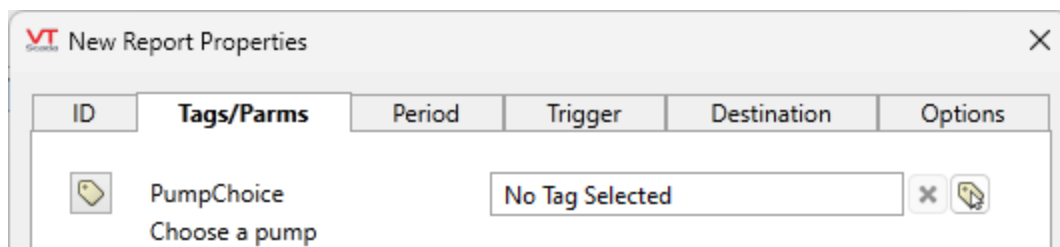


Figure 12-56 Setting a parameter value in a Report tag

Exercise 12-9 A Station Flow Rate Report

1. Click the Save As tool to save your report as `Station Flow Rates`.
You now have a copy of the report that you can use to extend the work you did in First Report.

Before continuing, you're going to create a user-defined type for your Station tag. You could proceed with where stations that are Context tags but in the application you're working with, your pumps are also Context tags. That could become confusing. So...

1. Close the Report Studio.
2. Open the Tag Browser.
3. Open the properties dialog for Station 1.
4. In the Type field, enter `aStationType`
5. Click OK to save and close.
6. Right-click on Station 1 in the Tag Browser.
7. In the menu that opens, select Create New Type
8. When prompted for the widgets to use, leave all selected and click OK.
You now have a user-defined tag type named `aStationType`.
9. Navigate to the top level of the tag tree. (\)
10. Add a new tag of type, `aStationType`.
11. Name the new tag `Station 2` and set the Area to `Western`.

If you navigate to Station 2 you will see a complete second station, identical to the first. (Except that the simulator created a unique instance for it, so each station is independent.) There is much more to user-defined types than what you've just done but at least stations and pumps are no longer the same thing (i.e. Context tags). The next set of steps will return to the Report Studio.

1. Close the Tag Browser.
You should be looking at the Report Page.

2. Ensure that Station Flow Rates report is selected and then click the Edit button to reopen the Report Studio.
3. Click the Manage Parameters button in the toolbar to open the Manage Parameters tool.
4. Click the Add button to begin creating a new parameter.
5. Name the parameter `StationChoice`
6. Set the description to `Select a Station`
7. Set the type to `Tag`
8. Select `aStationType` from the list of tag types.
9. Click OK to save your work and exit the Manage Parameters dialog.
10. Click the Parameters >> Values button in the Report Studio ribbon. The Edit Report Parameter Defaults dialog opens.
11. Select Station 1 as Station Choice.
This will provide values while continuing to work in the studio and will also be the default station when running the report unless another station is chosen.
12. Click OK to save your choice and close the dialog.
13. Returning to the Report Studio, click to select column B.
The Data Cell Properties panel should open, showing the data set Flow Rates, which you created earlier.
14. Click the pen icon to put this data set into editing mode.
15. Change the Context source from "Tag" (displaying "No tag selected") to Parameter.
16. Use the drop-down selection that appears to select the parameter, `StationChoice (aStationType)`.
17. Click once to select the text at the top of the report.
18. Open the Make Parameterized Phrase dialog for the text contents.
19. Add `StationChoice.Shortname` to the parameterized phrase and then move that to the beginning of the phrase. (Additions always go to the end.)
The phrase should look like the following:
`%[StationChoice.ShortName] 15-Minute Flow Rates - %`
`[StartDate] %[StartTime] to %[EndDate] %[EndTime]`
20. Save the report and close the Report Studio.

You now have a parameterized report, good for any station of type `aStationType`. If you return to the Reports page and run Station Flow Rates, you will be prompted for a station. Let's create a Report tag to automate that...

1. Open the Tag Browser and navigate to the top level.
2. Create a new `Report` tag at the top level of the tag hierarchy.
3. Name it `Station 2 Flow Report` and set the description to match.
4. On the Tags/Parms tab, select `Station 2`.
5. On the Period tab, set the Preset to `Last 60 minutes before trigger time`.
6. Set the Trigger to "Daily", with a time two minutes later than right now.
7. Set the output type to PDF File.

8. Set the Path to one you prefer (your desktop, documents folder, or other...).
9. Save the tag.

Within two minutes, your report should be created for Station 2. This is a new station so you can expect that most of the last hour will not have data. In a real application, you might make this tag part of the aStationType hierarchy and use a parameter so that every station automatically has this report.

Make Parameterized Phrase

This dialog is available for any customizable wording within a report, including:

- The title of the report.
- The header text of any column.
- The display of time and date information.

Parameter keywords begin with a % and are enclosed in square brackets as shown in the following example. All other text is passed through as written.

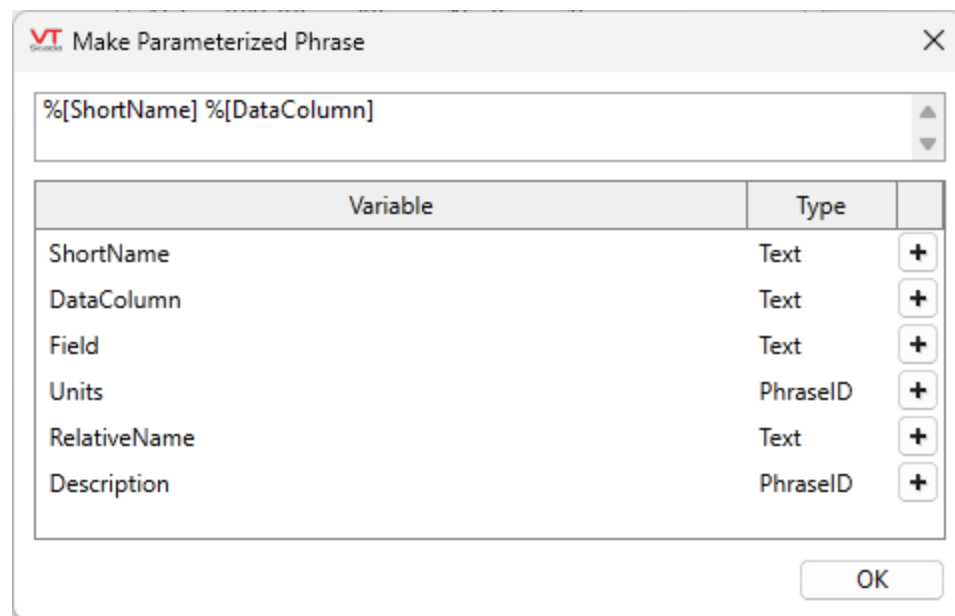


Figure 12-57 Standard parameters available to tag query parameterized phrases

Tip: When building a parameterized phrase, leave a space between each element unless you want words to run together. If building your own time format, it is up to you to include colons between hours, minutes, and seconds.

Tag Query Parameter Variables Include:

ShortName - The name of a tag, excluding all parents.

DataColumn - The function applied to the column.

Field - For any I/O tag (including legacy input / output tags) this is simply "Value".

Field is most useful when displaying driver statistics or values from custom tags that store data other than value.

Units - Engineering units configured for the tag displayed in any column.

RelativeName - The name of a tag including parents to the selected Context or the top of the tag tree.

Description - The configured description of each tag.

Time Parameter Variables Include:

Time - Displayed as hh:mm tt, where tt represents AM/PM.

Date - Displayed as yyyy-MM-dd.

Hours12 - The current hour using a 12-hour clock.

Hours24 - The current hours using a 24-hour clock.

Minutes - Minutes within an hour.

Seconds - Seconds within a minute.

SubSeconds - Three decimal points of accuracy.

AMPM - AM or PM.

DayNum - The numeric day within a month.

MonthNum - The numeric month within a year.

Year - The four-digit representation of the year.

DayString - The name of the day of the week.

MonthString - The name of the month of the year.

Label & Report Title Variables Include:

StartDate - Beginning date of the time span formatted as yyyy-MM-dd

StartTime - Beginning time of the time span formatted as hh:mm tt

EndDate - Date ending the included time span.

EndTime - Time ending the included time span.

RunDate - Date when the report was generated.

RunTime - Time when the report was generated.

WorkstationName - The identity of the workstation where the report was generated.

Custom Parameters

As described in Define Parameterized Reports, parameters that you create for your report can also be used in your parameterized phrases. This is especially useful for report titles and column headers.

13 Advanced Configuration Topics

This chapter contains a diverse range of topics. It's a set of short topics that did not fit elsewhere, examples and ideas that you can use to customize your application to work the way you need.

Alarm on Low Disk Space

VTScada requires available disk space in order to continue logging and continue operating. Rather than checking the disk regularly, you might create an alarm that will warn you when the available space runs low.

The amount of space required for VTScada, and the setpoint at which you want to be notified so that you have time to take action will vary according to your application. For this example, 50Mb is used, but that number is randomly chosen for the sake of the example.

1. Create a Workstation Status Tag
The workstation name should match the machine you want to monitor.
To monitor several workstations, create a Workstation Status tag for each.
2. Create a new I/O and Calculations tag as a child of the Workstation Status tag.
The remaining steps refer to the configuration of this tag.
3. On the ID tab, set the mode to Analog.
4. On the I/O tab, set the Read Address to FreeDiskSpaceC
(Or, FreeDiskSpaceD if monitoring a D: drive. Refer to Workstation Status Driver I/O Addressing)
5. On the Scaling tab, ensure that both Unscaled and Scaled process data values range from 0 to 100.
The numbers chosen do not matter, so long as both the unscaled range and the scaled range match.
If you prefer to work in Mb or Gb rather than bytes, configure the scaling as appropriate.
6. On the logging tab, either disable logging, or if you choose to log this tag, set the deadband to a value of at least 1Mb (1048576) if not 10Mb (10485760).
There is no need to log every one-byte change in the available space.
7. On the Alarms tab, select a Low Alarm priority as appropriate.
8. Configure the setpoint of that Low Alarm to be 52428800
(50Mb in this example. You may choose a setpoint as seems appropriate.)

Alarm Database Groups

(Property name: AlarmDatabaseGroups)

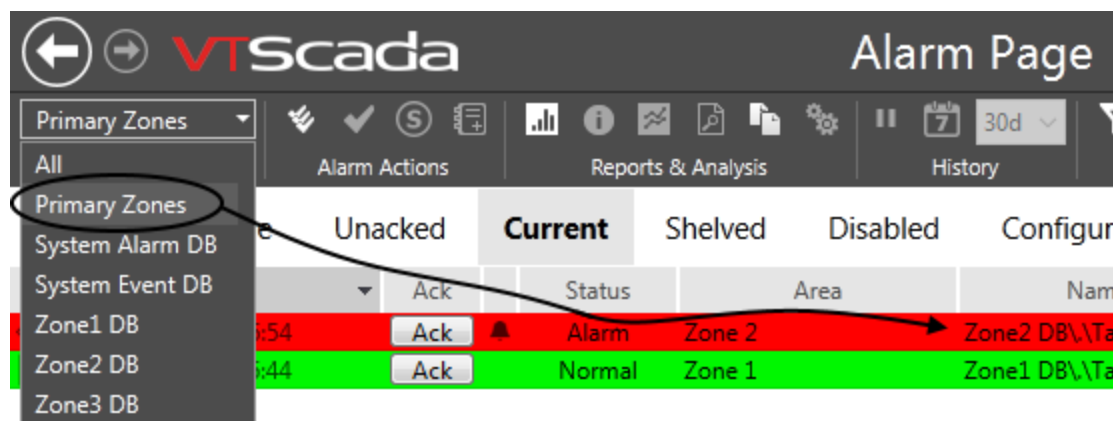
For the majority of applications, there is no need to create extra alarm databases, but locations that maintain large distributed systems might choose to do so. For those that do, it is possible to create display groups so that alarms from several, but not all, databases can be viewed at the same time.

Database groups are defined as an application property, AlarmDatabaseGroups. Multiple groups can be defined within the single property, each with a list of databases. Separate the groups with semi-colons and database names within each group by commas.

Property name: AlarmDatabaseGroups

Section: System

Value: Northeast Group:NorEast,NorWest;Southwest Group:SouWest, SouEast



The group, Primary Zones, shows alarms from both Zone1 DB and Zone2 DB

Example: Using an alarm database group

Add Property

Property Name

AlarmDatabaseGroups

Section

System

Value

Primary Zones:Zone1 DB, Zone2 DB

Workstation

-- default --

Comment

OK

Cancel

The property definition used for this example:

Section: System

Default: AlarmDatabaseGroups =

*Restart Required (Settings.Startup property)

Customize Columns in Alarm Displays

The structure of lists in the Alarm Page and in Alarm List Widgets is controlled by the XML file, C:\VTScada\VTScada\AlarmListFormats.XML. You may decide to customize the structure for any of the following reasons:

- Add or remove columns in a list.
- Set the default width of columns.
- Add customized columns to a list.
- Add a customized list format.

Note: Do not edit C:\VTScada\VTScada\AlarmListFormats.XML. Your changes will be lost with your next VTScada update.

Do not copy this file to your application folder. Local definitions of column formats and list formats that you have not customized will prevent updates from taking effect.

To make the customizations described in this topic, create a file named `AlarmListFormats.XML` in your application folder. The structure must be as described in this topic. You may copy sections of `VTScada\VTScada\AlarmListFormats.XML` to use as a template, but do not save any definition that you do not intend to customize. Your custom definitions will override or be added to those from the VTScada file.

The structure of the file is as follows:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<AlarmList>
  <ColumnFormats>
    <Format name="SingleSet1">
      <Column width="160">AlarmCellTimestamp</Column>
      ... more column definitions ...
    </Format>
    ... more format definitions ...
  </ColumnFormats>

  <ListFormats>
    <Format name="AlarmStandard" label="AL_StandardFormatLabel">
      <list name="History" label="AL_HistoryListLabel">
        <Single>SingleSet1</Single>
        <Double>DoubleSet1</Double>
      </list>
      ... more list definitions...
    </Format>
    ... more format definitions...
  </ListFormats>
</AlarmList>
```

XML Format Hierarchy:

- ListFormat definitions are linked to user-interface tools, and are selected according to rules coded into that tool. Examples follow.
- Within each ListFormat will be one or more lists such as Active, Current, etc.

- Each list definition within each ListFormat will contain two versions, Single and Double. This selection is controlled by the operator by toggling the Row Height option.
- The single version and the double version each specify a ColumnFormat.
- Two ListsFormats may each contain a list with the same name such as "History", but these are separate definitions. Different column formats can be specified for History (and any other list) in different ListFormats.
- ColumnFormats specify the display modules to be shown in the column cells, the order of the columns from left to right and the default width of each column.

For example, an alarm popup uses the "PopupStandard" list format, which contains only one list: "Unacked". In the Alarm Page, the default is to show the AlarmStandard format, containing History, Active, Unacked, etc, but if an operator chooses to view only the System Event DB, then the "EventStandard" list format will be selected automatically limiting the selection to just the History list.

Application properties are used to set the text used for the labels of lists and columns in the user interface. For example:

```
AL_HistoryListLabel = History
```

The column widths specified in the XML file are the initial default widths only. After the alarm list has been viewed by a user, the widths are retained for that user. Any further modifications to the column sizes in the XML file will affect only new users who have never viewed the alarm list.

To test the column sizes, you must either sign in as a new user, or delete the retained files and reload the page. To delete the retained files, follow these instructions:

1. Switch to a different page so the alarm list isn't being displayed
2. Using a file explorer, navigate to the application's \Data\Retained folder
3. Filter this folder by *DrawAlarmList-AlarmList-ColWidths*
4. Delete all the files that match the filter
5. Switch to the page with the alarm list - the column widths should now match what is in the XML file

Column Format Definitions:

The set of lists shown in the Alarm Page is predefined, but you may alter the appearance of any list.

For each list, History, Active, etc. two sets of column formats are defined. Two are required so that the operator may use the Row Height button to switch between a list with one item per column and a list with (in some cases) two.



Row Height selection tool

Standard display

Time		Ack	Status	Area	Name	Description	Value	Setpoint	Units
2015-12-21 10:56:29	2	Ack	Alarm	Zone 1	Local TCP Port\...\Tank Level	Monitor tank level HIGH	90	90	%
2015-12-21 10:55:33		Ack	Normal	Zone 1	Local TCP Port\...\Critical Level	Water level critically high	90	95	%

Columns stacked after using the Row Height button

Date		Status	Area	Name	Value	Setpoint
Time		Ack		Description		
2015-12-21 10:56:29	2	Alarm	Zone 1	Local TCP Port\PLCSim\Tank 1\Tank Level	90	90
		Ack		Monitor tank level HIGH	%	%
2015-12-21 10:55:33		Normal	Zone 1	Local TCP Port\PLCSim\Tank 1\Tank Level\Critical Level	90	95
		Ack		Water level critically high	%	%

In the ColumnFormats section of the XML file, these are given generic names. The display name is set in the second section of the file. An example of the format follows:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<AlarmList>
  <ColumnFormats>
    <Format name="SingleSet1">
      <Column width="160">AlarmCellX</Column>
      <Column width="26">AlarmCellY</Column>
      ...
    </Format>
    <Format name="DoubleSet1">
      <Column width="26">AlarmCellX</Column>
      ...
    </Format>
  </ColumnFormats>

```

Each "<Format" section sets the columns to be included. Columns are displayed in the list from left to right in the order found in the Format section.

The width sets the default size to be used by that column. Changes to column widths by the operator are stored on a per-operator, per-session basis, overriding your defaults. There are three ways to specify the width of a column:

- Column width = "30". Sets the specific number of pixels to be used by the column.
- Column width = 30%. This column will occupy 30% of the area remaining after the columns with specific width settings have been accounted for. The total of percentages should be 100 or less. See next option.
- No width specification. All columns with no width specification will share equally the space remaining after columns with a specific or percentage width have been accounted for.

The example text, "AlarmCellX", "AlarmCellY", etc. must be replaced, either by one of the VTScada modules provided to format and display the contents of each cell in an alarm list, or by a module of your own creation. See link under Related Information.

Example 1:

For the standard alarm page display of unacknowledged alarms, move the name and description to the first column.

This will require a change of the ColumnFormat ordering, but the first step is to discover which ColumnFormat is used by the Unacked list in a standard display. Fortunately, the names in the XML file make this easy to find:

Under <ListFormats> find <Format name="AlarmStandard" ...> You can safely assume that this is the standard format. Within that section, find the list named "Unacked":

```
<list name="Unacked" label="AL_UnackedListLabel">
  <Single>SingleSet2</Single>
  <Double>DoubleSet2</Double>
</list>
```

From the above, it is clear that ColumnFormats SingleSet2 and DoubleSet2 are used. These can be copied from the original file, together with enclosing XML specifiers and reordered. The file you save to your application as "AlarmListFormats.XML" should look like the following. Only the two column formats are being overridden in your application. All others will continue to use the default XML file. Don't forget to import file changes to add your version of the XML file to the application.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<AlarmList>
  <ColumnFormats>
    <Format name="SingleSet2">
      <Column>AlarmCellName</Column>
      <Column>AlarmCellDescription</Column>
      <Column width="26">AlarmCellPriority</Column>
      <Column width="160">AlarmCellTimestamp</Column>
      <Column width="50">AlarmCellAck</Column>
      <Column width="18">AlarmCellIcon</Column>
      <Column width="80">AlarmCellAction</Column>
      <Column width="140">AlarmCellArea</Column>
      <Column width="90" extra="1">AlarmCellValue</Column>
      <Column width="90" extra="1">AlarmCellSetpoint</Column>
      <Column width="70" extra="1">AlarmCellUnits</Column>
      <Column width="18">AlarmCellNote</Column>
    </Format>
    <Format name="DoubleSet2">
      <Column>AlarmCellDoubleNameDescription</Column>
      <Column width="26">AlarmCellDoublePriority</Column>
      <Column width="83">AlarmCellDoubleTimestamp</Column>
      <Column width="18">AlarmCellIcon</Column>
      <Column width="80">AlarmCellDoubleActionAck</Column>
      <Column width="140">AlarmCellArea</Column>
      <Column width="90" extra="1">AlarmCellDoubleValue</Column>
      <Column width="90" extra="1">AlarmCellDoubleSetpoint</Column>
      <Column width="18">AlarmCellNote</Column>
    </Format>
  </ColumnFormats>
</AlarmList>
```

The "extra" attribute, when present and set to 1, enables the visibility of the column to be toggled by the Show/Hide Extra Columns tool.

There is also an "alwaysShowShelved" attribute. For example:

```
<list name="Shelved" label="AL_ShelvedListLabel" alwaysShowShelved="1">
```

This attribute, when true, enables records that are marked as "shelved" to appear in the list even when the "Show Shelved Alarms" tool is not toggled.

Example 2:

Remove a column from display, such as the device history (AlarmCellDevice) column.

1. Create an AlarmListFormat.XML in the application
2. Create empty <AlarmList> and <ColumnFormats> sections.
3. In the <ColumnFormats> section, copy and paste the SingleSet1 and DoubleSet1 formats from the original
4. Modify SingleSet1 and DoubleSet1 by removing the AlarmCellDevice column

There is no way to customize the standard Alarm Page on a per-user basis. Instead, to allow admins to see the device column, they will need to draw the Alarm List widget on a page and configure it to use a custom format. They can secure this page so that only admins can see it. To create the custom format, do the following:

1. Edit the AlarmListFormat.XML in the application
2. In the <ColumnFormats> section, copy and paste the SingleSet1 and DoubleSet1 formats from the original and rename them, e.g. MySingleSet and MyDoubleSet
3. Create a <ListFormats> section below the <ColumnFormats> section
4. In the <ListFormats> section, copy and paste the AlarmStandard format from the original and rename it, e.g. MyAlarmFormat
5. In the MyAlarmFormat format, for the History list, rename SingleSet1 and DoubleSet1 to MySingleSet and MyDoubleSet

The result will be similar to:

```
<AlarmList>
  <ColumnFormats>
    <Format name="SingleSet1">
...copied from original, but with no device...
    </Format>
    <Format name="DoubleSet1">
...copied from original, but with no device...
    </Format>
    <Format name="MySingleSet">
...copied from original and including the device...
    </Format>
    <Format name="MyDoubleSet">
...copied from original and including the device...
    </Format>
  </ColumnFormats>
  <ListFormats>
    <Format name="AlarmStandard" label="AlarmsLabel">
      <list name="History" label="HistoryLabel">
```

```
<Single>MySingleSet</Single>
<Double>MyDoubleSet</Double>
</list>
</Format>
</ListFormats>
</AlarmList>
```

In the Alarm List widget drawn on the protected page, select MyAlarmFormat so admins can see the device column that the standard Alarm Page cannot see.

Tag List Widget for Reports

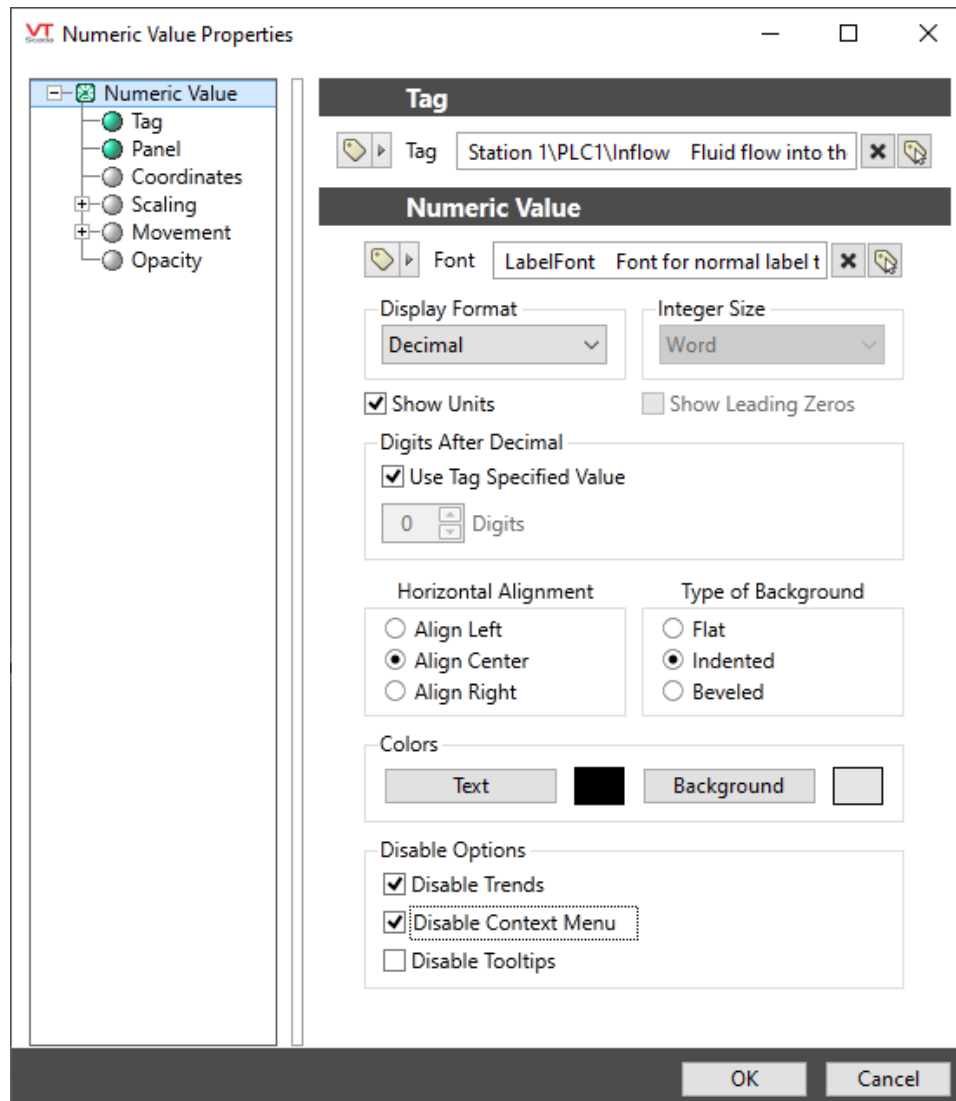
In the chapter, Log, Note and Report, we mentioned that a page could be considered a type of report if it presents summarized information to a user. The [Tag List Widget](#) is a great tool for building that.

The following will be a fairly simple example with live data rather than summarized values. If we had a set of History Statistics tags that had been logging (for example) hourly totals, flows, etc. for the past few days then this example would be much more report-like.

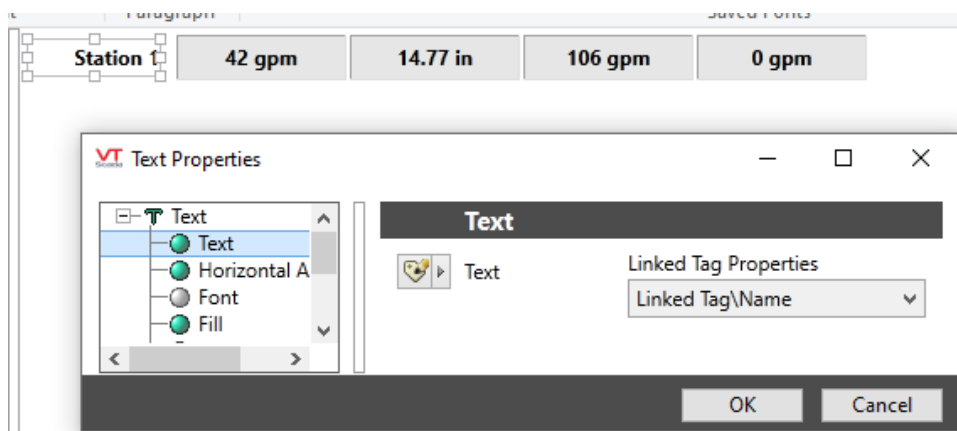
1. Create a new standard page named Station Monitoring.
2. Draw the following tags using Numeric Value widgets, in order as shown.
Inflow, Level, Pump 1\Flow, Pump 2\Flow

38 gpm	36.92 in	0 gpm	0 gpm
--------	----------	-------	-------

(An example of a suggested set of widget properties follows.)



3. Group the four widgets into a new Tag Widget named Station Stats.
(Ensure that the selected tag is the station, not the driver.)
4. Make note of the overall size of the widget (bottom left of the Idea Studio).
You will need to know the height later. When creating this example my widget was 410 x 27 pixels.
5. Open the widget for editing and move the four Numeric Values about 50 pixels to the right.
6. Add text to the left of the Numeric Values. Set the paragraph to be right-justified.
7. Set the data source of the text to be Linked Tag\Name.
(You will need to set the Widget's Drawn Context to one of the stations in order to see a name.)
8. Adjust the size and location to be approximately as shown:



9. Close the widget editing tab to return to the Station Monitoring page.
10. Delete the widget from the page.
11. Drag a Tag List widget to the page (Tools > Standard Library)
12. Open the Properties dialog.
13. Click the Plus (+) button to start.
14. In the Widget Selection dialog, set the Type to your Station tag type and click OK.
When the pop-up palette of widgets appears, your Station Stats widget should be the first in the list of Recent Items.
15. Select the Station Stats widget.
16. Open the Filter / Sort tab.
17. For this example, you want both stations, so there is no filtering to be done.
18. The default sort order is alphabetic, which seems reasonable for now.
19. Open the Table Layout tab.
20. In the Cell Settings, set the minimum height to be the same or a few pixels taller than the Station Stats widget.
(This is why you noted the widget height earlier.)
21. Set the number of rows to Best Fit.
22. Set the row spacing to 1.
23. Set the number of columns to 1.
24. Click OK.
25. Adjust the size of the overall widget so that there are two rows as shown, then add labels above each column:

	Inflow	Level	Pump 1 Flow	Pump 2 Flow
Station 1	29 gpm	28.66 in	106 gpm	0 gpm
Station 2	23 gpm	17.04 in	0 gpm	0 gpm

There are many ways to use a Tag List widget, but you can simplify the configuration (and gain better control) by creating your own widget to fill each row. You are free to use any widget you like for the cells, not just the Numeric Value that was shown here.

Control Locks

Create Control Locks to prevent use and operation of selected equipment via the VTScada screens. Among other uses, locks are typically applied when technicians are physically working on dangerous equipment to prevent remote activation of that equipment.

Tip: If you are looking for a way to lock controls for some users but not others, you should apply a security privilege, not a control lock.

Feature	Effect	Use...	Reference
Security Privileges	Control who is allowed to write to an output.	Use in all applications	Restrict Access to Output Tags
Control Locks	Prevent use and operation of selected equipment via the VTScada screens. If the tag is protected by a privilege, users must also have that security privilege.	Only where there is a need to lock controls for all users, regardless of privilege.	Control Locks
Control Tokens	Ensure that only the current token owner can write to an output tag. Users must also have the required security privilege and the tag must not be locked. Only one user at a time can hold a Control Token.	Only where there is a need to restrict control to one operator at a time.	Control Tokens

Caution: VTScada Control Locks do not override physical switches or alternate controls for the PLC. They apply only to the VTScada user interface.

Control Locks are applied to tags. A locked tag cannot be operated by anyone (regardless of their security privilege) until the lock is removed. Locking a parent tag will automatically lock all of its child tags, making them inoperable as well. This allows you to prevent operation of an entire subsystem with a single lock.

Tip: Control Locks use features of the VTScada Alarm Manager. When working with Control Locks, you may note similarities, including the fact that active locks are shown in a customized view of an Alarm List widget.

Enable the Control Lock System

To use the Control Lock system in your application, the ControlLocksEnabled property must be set to 1. Disabling this property will not remove existing locks but will render all locks useless. (They will not prevent tag operation.)

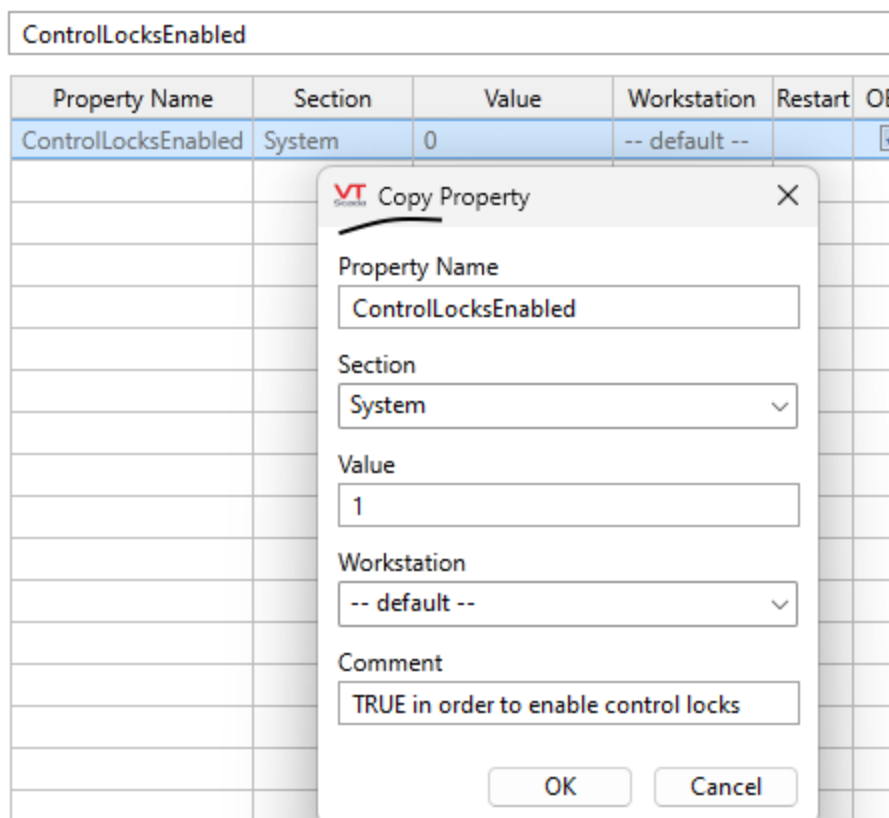


Figure 13-1 Detail from the Edit Properties tab of the Application Configuration dialog.
(Refer to [Configuration Properties](#).)

Lock Management via Widget

Tip: Locks are added, removed, viewed, and managed using widgets that you draw. Do not look to the Tag Browser or tag properties for locks. Lock-related widgets can be found in the palette folder, Tools\Control Locks.

Add Locks

To create a lock on a tag (either a specific I/O tag or a parent of many I/O tags) draw an [Add Lock Button](#) on a page of your choice and link it to an output tag or a parent of several output tags. To create a lock, users must have the security privilege, Lock Add / Remove. (Refer to [Control Lock Security](#) for further notes about security and locks.)

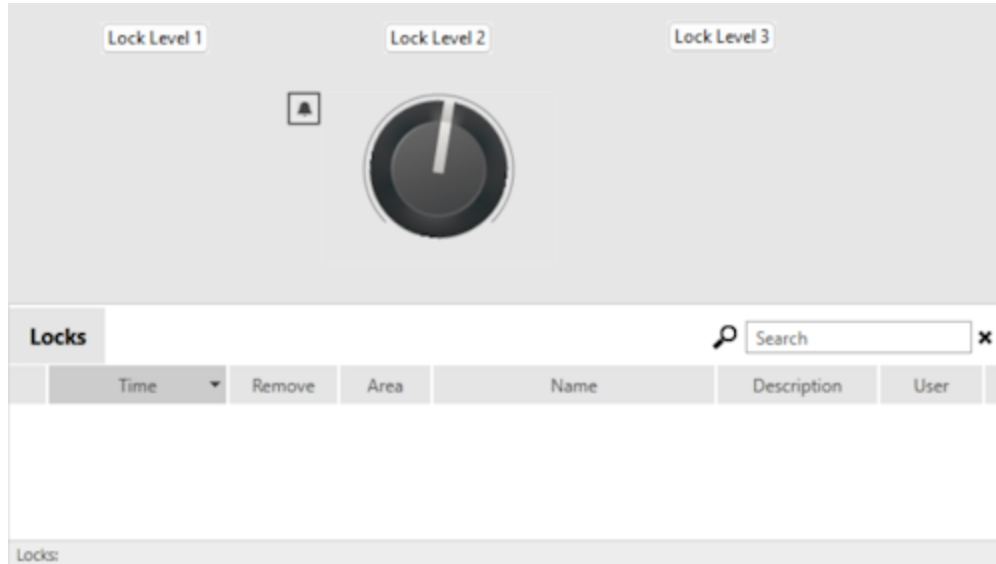
View / Remove Locks

To view and manage locks in your application, draw the [Locks List Widget](#) on a page of your choice. (The Locks List widget is in fact the Alarm List widget, configured to display Locks rather than alarms. The Alarm Page does not have this configuration option.)

To provide confirmation to operators that a tag is locked, draw either a Lock Level Icon or Lock Level Box on any page where that tag is shown, linking the icon or box to the tag for which you want confirmation. These widgets serve both to indicate when the tag is locked and the lock level applied.

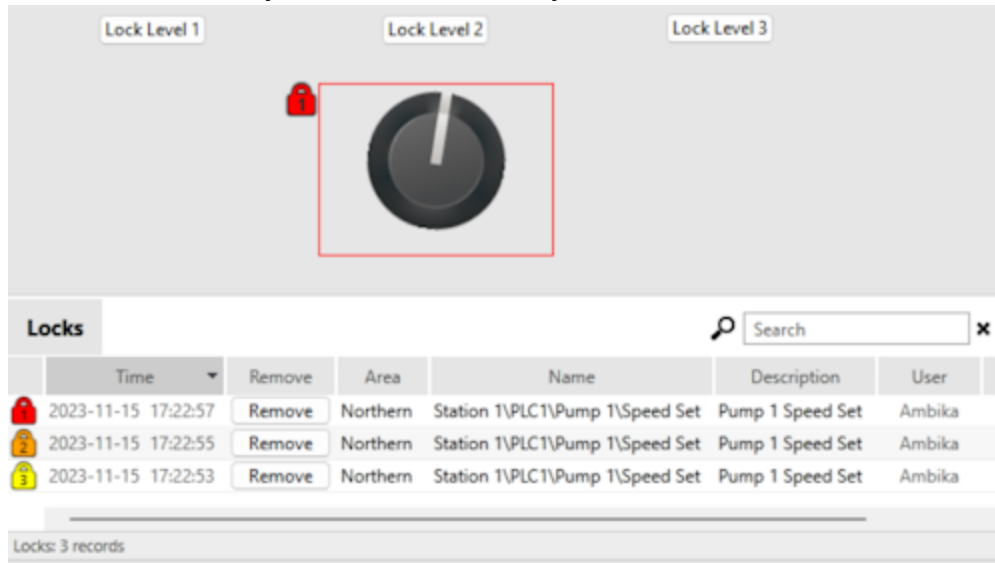
Exercise 13-1 Add Control Locks

1. Make sure to enable Control locks by setting the property `ControlLocksEnabled` to 1 in the Application Configuration dialog.
2. Create a new page in Idea Studio, naming it `Control Locks`.
At the end of the exercise your screen should look like the following:



3. In the widgets palette, open the Analog Controls folder.
4. Drag the Gray knob to the page from the palette.
5. Link the Gray knob widget to the tag, Pump 1 Speed Set.
6. Open the widget Tools folder and then Control Locks.
7. Drag the Add Lock Button to the page.
8. Link it to the Pump 1 Speed Set tag.
9. Open the properties dialog of the Add Lock Button and change the label to `Lock Level 1`.
10. Ensure that Level 1 is selected.
11. Select the Confirmation option and then click OK.
12. Add two more Add Lock Buttons, linking both to the Pump 1 Speed Set tag.
Configure one with Level 2 and label `Lock Level 2`
Configure the other with Level 3 and label `Lock Level 3`.
13. Drag the Lock level box widget to the page and resize it to fit around the Gray knob widget.
14. Link the widget to the Pump 1 Speed Set tag.
15. Drag the Lock List widget to the screen.
Adjust it to fit the space available in the Control Locks page.
16. Switch to the Operator View.
17. Click the Lock Level 1 Button.
When it asks for confirmation, click OK.
You should see the Lock Level box widget indicated in red and the lock level as 1, which you had selected in the Add Lock button widget properties.
18. Use the Lock Level 2 and Lock Level 3 widgets to see the effect of adding more locks to the same control.

19. The Lock List widget shows the details of all locks and provides a Remove button that can be used by the lock owner or by a lock administrator.



20. Try to rotate the Gray Knob to change the speed of the pump, A pop-up message should appear with the message, Write Denied. This tag is locked.

The locks can be removed by clicking the Remove button in the Lock List widget.

Exercise 13-2 Bonus Exercise: Widgets for all controls

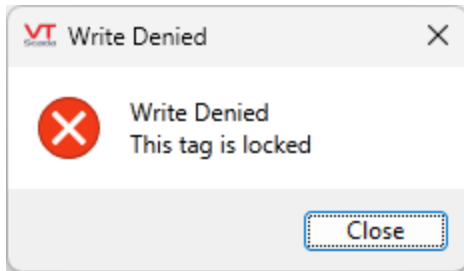
Your task in this exercise is adding a lock widget for all controls under the station. No instructions are provided as you can find any information that you might need in the help files. Start now.

Caution: Before moving to the next lesson, ensure that no locks remain set.

Lock Levels

By default, all locks will prevent control of the tags they apply to. However, there are situations where you might want to use locks to inform the user of a situation but not prevent control. This behavior can be configured by setting a Lock Level and configuring the matching ControlLockLevelXPreventControl property (where X=1,2,3,...) to 0.

To inform the operator that a lock has been set (whether that lock prevents control or not), draw a [Lock Level Icon Widget](#) or [Lock Level Box Widget](#) on a page with the tag. If the Lock Level does not prevent control, no warnings are displayed; the icon or box widgets are the only indications that an information-only lock has been set. If the Lock Level prevents control, a warning message is displayed:



Like Alarm Priorities, the lower the number of a Lock Level, the higher its priority. Three lock levels are defined in every application and you can add more if required by adding new `ControlLockLevelXPreventControl` properties to your application, setting X to 4, 5, 6, etc. .

It is possible for any output tag to have more than one lock applied (for example, one on the tag itself, one on the parent equipment type, one on the parent station type, etc). In this situation, the enabled lock with the lowest Lock Level (highest priority) is the one that takes precedence and is shown by the tag's Lock Level Icon or Lock Level Box widget.

Note: The highest priority (lowest number) lock takes precedence and dictates how the tag is locked. For example, a tag with a Level 1 lock that does not prevent control and a Level 2 lock that does prevent control will still be operable.

Tip: You can disable Control Locks on any tag structure by adding `ControlLocksEnabled` as a property of the parent Context tag (or user-defined tag) and setting that to 0. Locks will then not apply to any child tag within the hierarchy.

The opposite is not true: you cannot selectively enable Control Locks on tag hierarchies without enabling them for the application as a whole.

Control Lock Security

Privileges

Two privileges relate to the use of Control Locks:

Lock Add/Remove

This privilege allows users to add locks without restriction, and to remove locks subject to any restrictions imposed by the application property, `ControlLockCanOnlyRemoveOwnLock`. (See following notes for Lock Ownership Mode.)

Lock Administrator

This privilege allows users to remove any lock. It does *not* give users the ability to add locks. The intent of this privilege is to allow a supervisor to remove locks when the person who would ordinarily do so is not available.

Lock Ownership Mode

You have the option of enabling the Ownership feature of Control Locks by setting `ControlLockCanOnlyRemoveOwnLock` to 1. Doing so restricts which locks users with the Lock Add/Remove privilege can remove, such that they can only remove locks that they 'own'. This does not affect users with the Lock Administrator privilege, who are always able to remove any lock.

Ownership can be defined in several ways and is set using the `ControlLockOwnerMode` property.

Note: Anyone with the Lock Administrator privilege can always remove any lock.

0. Session ID ownership (ControlLockOwnerMode=0)

Users only own locks that they create during a given session. Each time a user signs on, a new session ID is generated. Therefore, signing in and setting a lock, signing off, and then signing in again would result in the same user being in a different session and not be able to remove the locks they set in the first session. This extends to both thin and thick clients. Even if the same user is signed in on several clients, locks added on one will not be removable on other clients.

1. Account ID ownership (ControlLockOwnerMode=1)

This is the most straightforward ownership mode. As the name suggests, ownership is defined by the account ID that added the lock. If the same user is signed into several machines they would be able to remove locks set on other machines / clients.

2. Machine ID ownership (ControlLockOwnerMode=2)

In this mode, ownership is defined by the machine ID. This is useful way to segregate removal permissions based on physical workstations. *When ownership is set to this mode, AutoUnlockUponLogout does not apply.*

3. Custom Hook ownership (ControlLockOwnerMode=3)

Applies only to those who are developing their own code that involves Control Locks. *When ownership is set to this mode, AutoUnlockUponLogout does not apply.*

Ownership is defined by a custom hook provided by the customer within their code: ControlLockIsOwnerHook.SRC.

AutoUnlockUponLogout

You can activate an automated lock release mechanism by setting ControlLockAutoUnlockUponLogout to 1. If lock ownership is any of: disabled, set to Session Id, or set to account ID, then signing out will release all locks set during the session.

Auto lock release is disabled for MachineID ownership or custom hook ownership.

Control Tokens

The Control Token system ensures that only one person (owner) at a time has control over a tag or set of tags. It is complimentary to the Control Locks system. A lock denies control and a token grants exclusive control (providing that the tag is not otherwise locked).

Caution: Upon enabling the Control Token system, all users must request a Control Token for every output.
DO NOT ENABLE THIS FEATURE WITHOUT PROVIDING USERS WITH A WAY TO REQUEST A TOKEN.

It is essential that [Request Token Button Widgets](#) exist or that appropriately configured [Control Token Box](#) widgets be drawn on any page that contains an output widget.

If a token is applied to a tag, only that token's owner has control over the tag. A control token restricts user actions. It does not restrict automated actions. Any tag can have only one token. (This is in contrast to Control Locks where multiple locks can be placed on a tag for different reasons.)

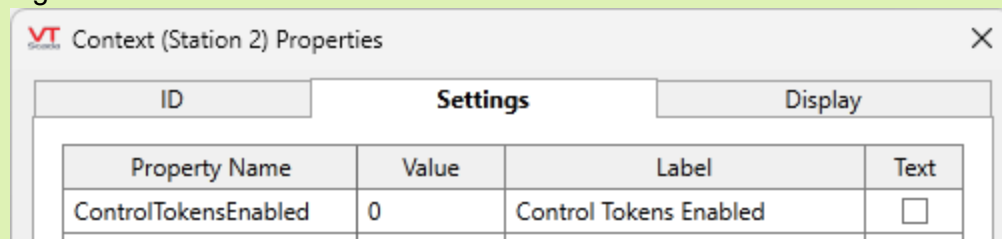
The Control Token system can be enabled globally by setting the application property [ControlTokensEnabled](#) to 1. ([Configuration Properties](#)) Alternatively, it can be enabled for a specific tag tree (or part thereof) by adding the property to a Context tag (or tag type derived from a Context tag) to enable Control Tokens in child tags of that parent. (You can also disable Control Tokens for a section of a tag tree by adding the property to a parent tag and setting the value to 0.) VTScada will check for ControlTokensEnabled starting with the current tag and working up through the scope of the tag tree to the application in general.

Placing a token on a tag grants the token owner control over that tag and all of the children that do not already have another control token. Adding a token to a parent tag does not override any existing token on a child tag.

Tip: Tokens are requested, viewed, and managed using widgets that you draw. *Do not look to the Tag Browser or tag properties for Control Tokens.* Token-related widgets can be found in the palette folder, Tools\Control Tokens.

Caution: VTScada Control Tokens do not override physical switches or alternate controls for the PLC. They apply only to the VTScada user interface.

Tip: You can disable the Control Token feature on a site by site basis through an application. Do so by adding the property ControlTokensEnabled to the parent Context tag(*) of the site and set the value to 0. Tags within that structure can then be operated without first obtaining a Control Token.



(* or Context-based custom type if the parent tag is no longer a Context)

Requesting Control Token Ownership

Authorized users(*) can request a Control Token for that tag either explicitly using a [Request Token Button Widget](#), or implicitly by opening a page that contains an appropriately configured [Control Token Box](#) widget.

(* An "authorized user" is anyone with the Token Request / Release privilege, as well as any privilege required to use a given output tag.)

These requests are made using widgets, and can be made for a single output tag or for all the tags under a common parent. If no-one else has a control token on the tag, the default behavior is for the request to be granted immediately.

It may happen that one or more people want to request control over a tag when someone else already has a control token for that tag. What happens in this case depends on your configuration: by default, later requests must wait until the current token owner releases control. You can configure the system such that incoming requests with a higher priority level will automatically take control from the current token owner.

To keep track of who has control and who is requesting control, two separate lists are maintained: the Token list (who has ownership) and the Token Requests list (who is requesting ownership).

The Token Administrator privilege can be granted to selected users so that they can release any existing tokens and can grant or deny any existing token requests. The Token Administrator privilege by itself does not grant the user the ability to request a Control Token.

Releasing Tokens

At any time, you can choose to release tokens that you hold. Do so with a Release Token Button widget that matches the token you wish to release. (Like the Control Token Request button widget, the release button is linked to a tag and affects only the token applied to that tag. A more convenient option might be to use a Token List, where all tokens are shown and you will have the option to release any that you hold.

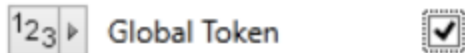
By default, token owners can release only the tokens that they own. You can change this behavior by setting the application property, [ControlTokenCanOnlyRemoveOwnToken](#), to 0. (This does not affect Token Administrators, who can release anyone's tokens.)

Ownership can be defined in several ways and is set using the [ControlTokenOwnerMode](#) property.

Tip: As described in the notes for that property, ownership may be linked to any of the user's sign-in session, the user's account in general, the user's current workstation, or to a custom programming hook that you provide.

Global Tokens

Formally known as Root Tag Tokens, a Global Token will grant control over every output in the tag tree. The Request Token Button, Release Token Button and Control Token Box widgets all have a Global Token check box within their properties configuration.



A Global Token does not force the release of other control tokens. If a Global Token has been requested and is available it will be granted immediately-- but control of any output already being controlled by someone with a control token will stay in their control until they release their token. Once the control token for a specific device has been released, the user holding the Global Token will assume control.

The Global Token works by declaring a fictitious 'Root Tag' in the ControlTokenManager. The syntax to use this would look similar to:

```
\ControlTokenManager.RequestToken(\ControlTokenManager.RootTagID, Level)
```

The Root Tag name and area can be changed by adding ControlTokenRootTagName and ControlTokenRootTagArea properties.

Token Lifetime

Control Tokens will continue to exist after the holder signs out. You can alter this behavior such that the act of signing out will always release all tokens held by a user, automatically. Do so by setting the property [ControlTokenAutoReleaseUponLogout](#) to 1. This also denies all token requests made in that session.

The lifespan of Control Tokens (and Control Token Requests) can be controlled by the properties, [ControlTokenMaxAge](#) and [ControlTokenRequestMaxAge](#), neither of which are set by default, meaning that there is no automatic expiry.

Stale Tokens & Requests

These are tokens or requests left behind when a user shuts down the application without releasing (by any means) the tokens or requests that they hold. This can occur with a workstation crash or a user simply shutting down their computer without signing out.

The owner of every token and request will send a "heartbeat" signal to the token server. If a pulse does not arrive for a set amount of time, VTScada will assume that the token or request is stale. The server will release stale tokens and requests when it checks its dictionary of timestamps every `ControlTokenMonitorTime` seconds or when the list of tokens and requests changes. This behavior is controlled by the following properties:

- `ControlTokenEnableOwnerMonitor`
Defaults to 1 (TRUE) to enable release of stale tokens.
- `ControlTokenMonitorTime`
Defaulting to one hour (3600 seconds), this is the frequency at which the control token manager checks its list of token and request owners to ensure that it is current and release / deny stale tokens / requests
- `ControlTokenStaleTokenTime`
If there are no requests for the token, it will be determined to be stale after not receiving a heartbeat for one day (86400 seconds) (It has been a day since we heard from the owner.)
The interval also applies to all requests.
- `ControlTokenPendingRequestStaleTime`
If there are requests for the token, it will be determined to be stale after not receiving a heartbeat for one minute (60 seconds) (It has been a minute since we heard from the owner.)
This interval does not apply to any requests.
- `ControlTokenOwnerHeartbeat`
Defaulting to 30 seconds, this is the frequency at which a token owner confirms with the server that it is still active.

Token Actions

As noted, if a user requests a token and no-one else has a token on that tag, the request will be granted immediately.

If a user requests a token and someone else currently has a token for that tag or its parent, then a Token Request will be appended to the Token Request list. This pending request can then be either Granted or Denied, either by the current token holder or by a Token Administrator. If the request is Granted, then the current token holder loses control over the tag and a new token with a new owner is placed. If the token request is Denied, the token request will be removed from the Token Request list. A user cannot have multiple identical token requests.

Releasing a token initiates an auto-granting mechanism where the next person in the list of Token Requests will be granted control (and a new token). They will be removed from the list of Token Requests so that when they release their token, the next request can be granted.

If you hold a token and another user requests control over that tag, you will be presented with the Token Requests dialog:

Time	Grant	Deny	Area	Name	Description	User
2023-08-31 12:24:06	<input type="button" value="Grant"/>	<input type="button" value="Deny"/>	Northern	Station 1\PLC1\Low Level Set	Setpoint to switch pumps off	Minion

Your choices are:

- Grant the request. (You will lose control of the tag.)
- Deny the request. (The request will be removed from the Token Request List.)
- Close the dialog without taking action. (The request will remain on the Token Request list and will be granted when you release the Control Token.)

Token Levels vs. Token Actions

Like Control Locks, tokens have levels. The default level for token requests is 1 (where lower level numbers indicate higher priorities). All levels grant control over the tag. (There is no equivalent to ControlLockLevelXPreventControl.) Levels are used in combination with application properties to control the behavior when someone else requests a token that you hold.

The relevant properties that control what will happen when someone requests a token that you hold are as follows:

[ControlTokenAutoStealOption](#)

Defaults to 0. Controls whether to grant an automatic token request for a token on a tag that is currently being controlled by another owner.

[ControlTokenAutoRequestOption](#)

Defaults to 1. Controls whether automated requests for tokens will be added to the request list.

[ControlTokenManualStealOption](#)

Defaults to 0. Controls whether to grant a manual token request for a token on a tag that is currently being controlled by another owner.

[ControlTokenManualRequestOption](#)

Defaults to 1. Controls whether manual requests for tokens will be added to the request list.

Together, the default values mean: "never release someone else's token and add all requests to the token request list". Levels do not apply.

Note that each of the Levels will apply to the actions of requesting / granting / denying / releasing tokens if you choose to set these properties to values other than the default. The rules vary according to the values you set for these properties but the overall effect is to allow automated actions and control whether any action can proceed based on a comparison of the priority of the request.

For example, suppose that a user has a control token with a low priority level (3) and another user creates a request with a higher priority level (1). By default, the level is irrelevant and the second user must wait until the first token is released. But you could configure the system such that the act of creating a higher priority request will automatically release the first token and grant a new token to the person making that high priority request. Other combinations are possible.

Three token levels are provided and more can be added (and simultaneously given an indicator color) by inserting `ControlTokenLevelXColor` properties, where X can be 4, 5, etc. Do not skip values.

Possible values for each of these properties:

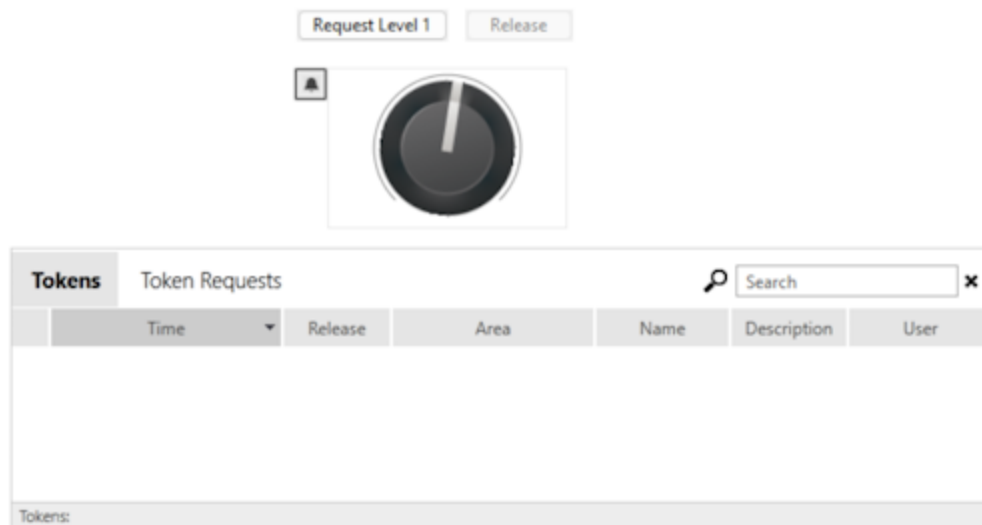
0	Never
1	Always
2	If request level priority > token level priority (lower number / higher priority takes precedence)
3	If request level priority >= token level priority
4	If request level priority < token level priority
5	If request level priority <= token level priority
6	If request level priority == token level priority
7	If request level priority != token level priority

Exercise 13-3 Control Tokens

Preparation: Security should be enabled and you should have an account that has been granted the Super User role. Ensure that you also have the Token Request / Release privilege.

1. If you do not already have a user named, 'NorthOperator', create this account now.
2. Grant NorthOperator the Super User role and Token Request/Release as an additional privilege.
3. Create a new page in the Idea Studio and name it Control Tokens.

At the end of the exercise your screen should look like,



4. In the widgets palette, open the Analog controls folder.
5. Drag the Gray knob to the page from the palette.
6. Link the Gray knob widget to the tag, Pump 1 Speed Set.
7. Open the widget Tools folder >> Control Tokens.
8. Drag the Request Button to the page.
9. Link it to the Pump1, Speed Set tag.
10. Open the properties dialog of Request Button and change the label to `Request Level 1`.
11. Ensure that Level 1 is selected.
12. Select the Confirmation option and then click OK.
13. Drag the Token Level box widget to the page and resize it to fit around the Gray knob widget.
14. Link the widget to the Pump 1 Speed Set tag.
15. Drag the Tokens List widget to the screen. Adjust it to fit the space available in the Control Tokens page.
16. Open the Operator View.
17. Click the Request button. When it asks for the confirmation, click Ok.
No-one else has a token on that tag already, therefore your request will be granted immediately and you will be the owner of the token until you release it.
18. Sign out and sign in as NorthOperator.
19. Request the token by clicking the Request button.
The token request gets added in the Token List widget with the Grant and Deny buttons.
20. Sign out and sign in as yourself.
When you switch users and sign in as yourself, you can see the token request added to the list in the Token Request list widget. You can either grant or deny the request by clicking the Grant or Deny buttons.
If you grant the request, you lose control over the tag and NorthOperator becomes the new owner for the token. On the other hand, if you deny the request, the token request will be removed from the Token Request list and you will remain the owner.
21. Deny the request.

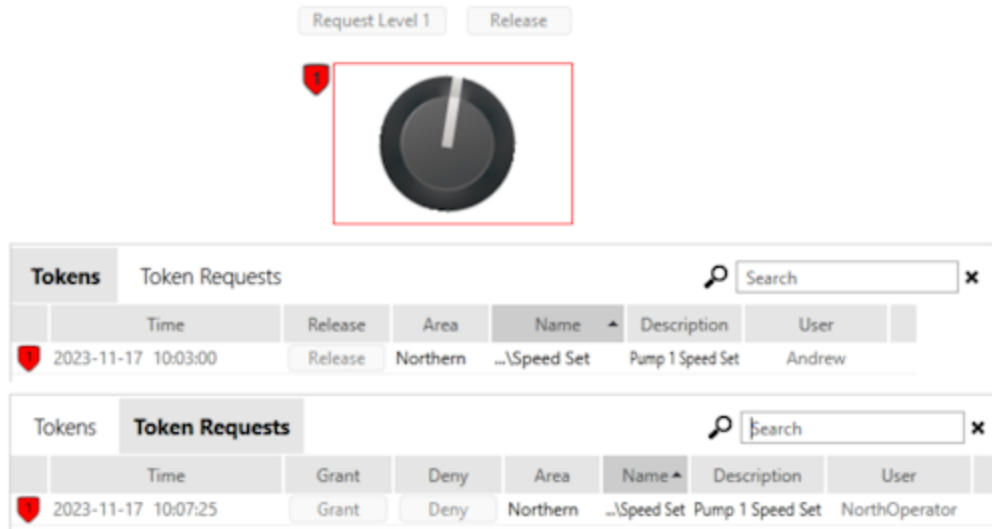


Figure 13-2 Sample Tokens list and Token Request list.

22. Open the Application Configuration dialog / Edit Properties tab and set ControlTokenManualStealOption to 1.
23. Sign out and sign in as NorthOperator.
24. Request the token.
Since ControlTokenManualStealOption property is set to 1, the token request will be granted immediately and NorthOperator will now become the owner of the tag, without the previous owner needing to grant the request or release their token.

Tip: The last few steps illustrate what is possible but not necessarily what is recommended. A value of 3 might be better for ControlTokenManualStealOption. Refer to the table of related property values, presented earlier in this topic.

Exercise 13-4 Bonus Exercise: Different level tokens

Set up two request buttons with different levels of priority and experiment to see the effect of trying to request a token using a higher or lower priority. No instructions are provided as you can find any information that you might need in the help files. Start now.

Caution: Before moving to the next lesson, disable the Control Tokens feature by setting the application property ControlTokensEnabled to 0.

Related Widgets

Recipes and Batch Processing

In version 12.1 of VTScada, we added support for recipes. The 12.1 simulator includes just enough I/O and routines to allow you to see how the feature can work.

Control over running the batch is expected to remain with the PLC. VTScada's role is to provide sets of values for the recipe used in a batch, start the run (if allowed by the PLC), and report when the batch has finished (if that information is provided by the PLC).

Recipes are not stored in databases

Recipes are stored under the VTScada Distributed Version Control System using CSV files, rather than under the Historian. Unlike other systems, where loss of access to the central database could cause downtime, the VTScada system ensures that production can continue because the files are synchronized across all workstations, automatically.

Any changes by an operator on any workstation will immediately be reflected in the running copy (and files) on all connected workstations. If a workstation was offline when an update was made, its copy of the recipes will be updated as soon as it comes back online.

The other benefit of storing recipes in the version control system is that you get a complete revision log of every change, including the ID of the operator and on which workstation that change was made.

Recipes can be imported from and exported to an external database to facilitate the use of other recipe management tools.

General Process

1) In all cases, the first step is to create a [Recipe Book Tag](#). This will hold your recipes, and define the communication link to the PLC. (Refer to [Tag Structure for Recipes](#))

Exercise 13-5 Create a set of recipes

1. Ensure that the V12.1 simulator application is running.
You can do the following steps in that application or in any other.
2. Create a Context tag named `Coffee Processing`.
Configure the area as `Recipe Demo` and description as `Demo of a batch process`. This is only to hold the communications chain and the Recipe Book.
3. Under Coffee Processing, add a TCP/IP Port tag named `Coffee Port`
Set the IP Address to `127.0.0.1` and the Port to `501`.
4. Add a Modbus driver tag named `Coffee Driver`
Ensure that it is linked to the port tag and that the Comm Channel is Open Modbus TCP.
5. Create a Recipe Book tag as a child of Coffee Processing, configured as follows:

ID tab:

Name: `Coffee Recipes`

Area: `Coffee`

Description: `Coffee variations`

I/O tab:

I/O Device: `Coffee Driver`

Start Address: `201`

Done Address: `203`

Batch Number Address: `40207`

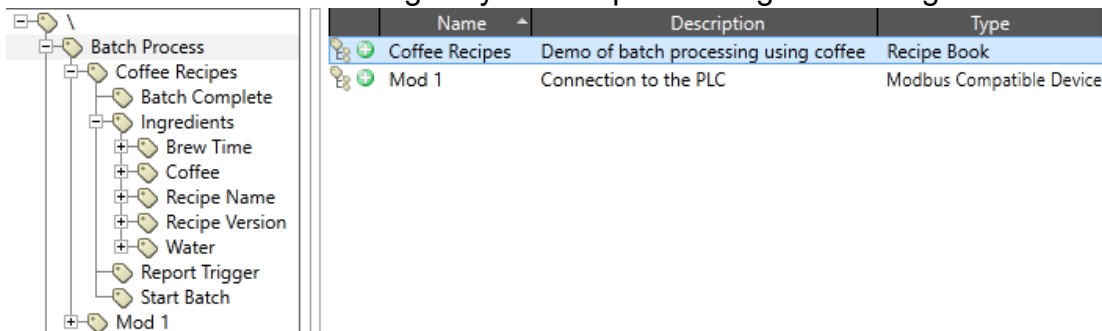
Select: `Allow versioning of recipes`

All other properties remain at the defaults.

6. Click OK to save the tag.

Note that a set of child tags are created automatically. The Recipe Page provides a user interface for configuring all of these; you will seldom, if ever, need to edit these tags within the Tag Browser.

2) A recipe needs ingredients, each of which is stored in a [Recipe Ingredient Tag](#). These should be created as child tags of your RecipeBook\Ingredients tag.



Each Recipe Ingredient is a two-tag structure where the parent *Ingredient* tag holds the overall description of what the ingredient is, and a child I/O tag holds the I/O address and other parameters required to write a value for that ingredient to the PLC.

Optionally, some or all of your Recipe Ingredients can use a [Recipe Proportions Tag](#). Use these when ingredient amounts should be multiplied by a given factor from one batch to another, or when amounts can vary proportionally to each other.

Tip: In each Recipe Book, there is only one Ingredient tag structure for each ingredient, no matter how many recipes or recipe versions call for that ingredient in varying amounts (including the amount of zero).

Exercise 13-6 Add ingredients to your recipe

Typically, the ingredients are available from the PLC or a comma-separated values (CSV) file. Tools to import from either are available in the Recipe Page.

But that's easy. For the sake of seeing the process in more detail, this exercise asks you to create your own ingredient tags. Fortunately, our cafe has a very limited selection of coffees, and therefore very few ingredients.

1. In the Tag Browser, navigate to Coffee Processing > Coffee Recipes > Ingredients. Note that Name and Version are stored as Ingredient tags. Most information about a recipe is stored as one of these.
2. Add a new Recipe Ingredient tag as a child of Ingredients, configured as follows:

ID tab:

Name: Ground Beans

Description: Medium ground, dark roasted arabica beans

I/O tab:

Address: 40213

Engineering Units: g

Min: 0

Max: 2000

Leave all other properties at their default values.

Note: We're brewing coffee, not roasting beans. If we have recipes that specify other roasts or grinds or bean types, we would need an ingredient for each. We do not need an ingredient for each amount.

3. Three more ingredients are required. Create a Recipe Ingredient tag for each of:

Water (addr: 40214, units: ml, max: 2000)

Brew Time (addr: 40215, units: minutes, max: 1440)

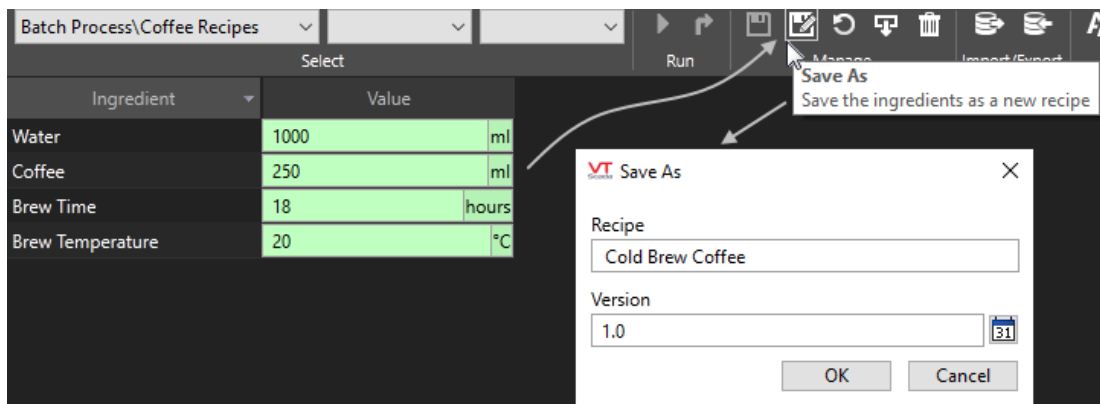
Brew Temp (addr: 40216, units: deg, max 100)

Cream and sugar are added to cups after the coffee is brewed and poured, and therefore as part of a second, consecutive process. You can absolutely string batches together, brewing the coffee in one process and then automatically starting the next process when the PLC signals that the first has completed. But we won't attempt that for our first recipe.

3) After creating the Recipe Book tag and a set of Recipe Ingredient tags, open [The Recipe Page](#) to set, retrieve, edit, and store ingredient amounts for each recipe in the Recipe Book. (It is possible, but rarely necessary, to build your own recipe page using the library of [Recipe Management Widgets](#)

After setting values for the ingredients that are included in a recipe, save them to a new recipe name. The option to set a version number is available only if you selected that feature when you created the Recipe Book tag.

Tip: Any ingredient that is not used in a given recipe should have a blank (or zero) value.



Exercise 13-7 Open the Recipe Page and create recipes

1. Close the Tag Browser.
2. Navigate to the Page Menu and add a menu item link to the Recipe page. If you need instructions for that, refer to the documentation.
3. Navigate to the Recipe page.

4. Set the following values for your ingredients:

Brew Temp: 96

Brew Time: 4

Ground Beans: 18

Water: 280

5. In the Manage section of the toolbar, click the Save As button.
6. Name the recipe Perfect Cup
7. Set the version number to 1
8. Let's make coffee: Click the Run button to start the batch.
9. Leave the Batch Number as 1 and write a note when prompted.
Fortunately, the simulator will ignore the time you specified and finish within a few seconds. The results are displayed at the bottom of the Recipe page.

Feel free to experiment with alternate recipes (the simulator won't provide samples for tasting). Each time you change the ingredient amounts and use Save As, you create a new recipe, or (if configured in the parent Recipe Book tag) a new version of an existing recipe.

What if you want to double or triple the batch?

You need to add a [Recipe Proportions Tag](#) to each ingredient that can scale. There are two ways to specify proportions: by multiples or as a percentage of a whole.

Note: Proportions are sometimes specified as decimal values (1.5 for a batch and a half). Can your PLC work with floating point values? In this exercise, the simulator is configured for standard Modbus and doesn't support /float addressing. We're stuck with integers for our coffee proportions because that is all that our simulated PLC will support.

Exercise 13-8 Make double batches

1. Open the Tag Browser and navigate to the Ingredients tags of your Coffee Recipe book.
2. Add a Recipe Proportions tag as a child of Ingredients.
3. Name it Cups
Description: Cups of coffee.
4. Open the Settings tab.
The default is what we want: Ingredients are multiplied at batch run time. You will be prompted.
5. Save the tag.
6. For each of Water, Ground Beans, and Brew Time in turn, open the properties dialog....
7. In the I/O tab, select Cups as the Proportions tag.
(The brewing time may be a bit off for multiple cups, but we're keeping this example as simple as possible. You can define more than one multiplier in a recipe.)
8. Close the Tag Browser after you have linked the three specified Ingredients to the Proportions tag.

9. Return to the Recipe Page and run another batch.
You will be prompted for all multipliers used by ingredients in this recipe.
(Whether the ingredient amount is zero or not.)

Again, the simulator has been designed to ignore the time ingredient.

Notes:

The amount for each ingredient in a recipe or recipe version is stored in a file that is controlled and maintained as part of the application's version control system. While ingredient values can be set individually by working with your Recipe Ingredient tags in the Recipe Page, it is also common to import ingredient values from either of:

- A Microsoft Access file
- A Microsoft Excel file

See: [Import / Export Recipes](#)

You can also start the retrieve process from the Tag Browser by a Recipe Book then clicking the Import button.

14 Expressions, Part Two

In the chapter, Expressions Part One, you learned how to create expressions, which you then went on to use in tags and widgets.

In this chapter, you'll learn how to create expressions well and gain some practice time during which you can explore the possibilities of how you can employ expressions in your applications.

Other Tag Properties

Tags have other properties that can be read in addition to their values. For example, device drivers maintain seven logged variables that are available to your expressions¹. Access any property of a tag, including its value, by adding a backslash and the property name.

For example:

```
[Tag Name]\Description
```

(Returns a phrase identifier key, not the text of the description. [Multilingual Expressions](#))

```
[Device Driver Name]\Quality
```

```
[I/O Tag Name]\HighAlarmUnacked
```

```
[I/O Tag Name]\ScaledMax
```

Note: Remember that expressions can read values and can calculate using values, but they do not write values. To set one of these properties automatically, use a Tag Parameter Expression in the tag's configuration dialog.

You might wonder how to find the property names. There are two. The easiest is to use the Idea Studio and linked tag properties. An advanced method is to use the Source Debugger.

Use the Idea Studio to find tag properties:

1. Open the Idea Studio.
2. From the file menu, select New >> Tag Widget.
3. In the Select Tag Types dialog, choose the tag for which you want to discover property names.
(You are advised to use this technique with only one tag type at a time.)
4. Accept the default name for the widget.
5. Drag a square (or any shape) to the widget.
6. Open the square's properties dialog.
7. Set the data source of the fill color to Linked Tag Property.

¹Use care with driver properties and expressions: Some are meant to be read once, such as for a report. Reading them continuously in Steady State may slow your application.

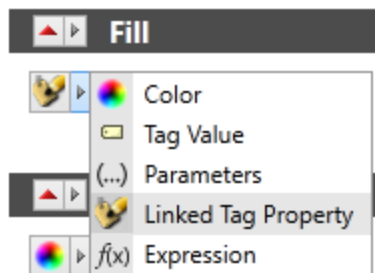


Figure 14-1 Changing the data source of a square's fill property

8. Expand the Linked Tag Properties drop-down to browse the parameters list.

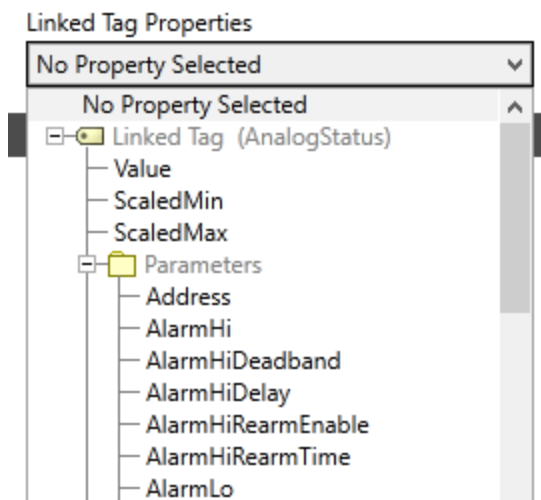


Figure 14-2 Selecting parameters of a linked tag property

Use the Source Debugger to find tag properties:

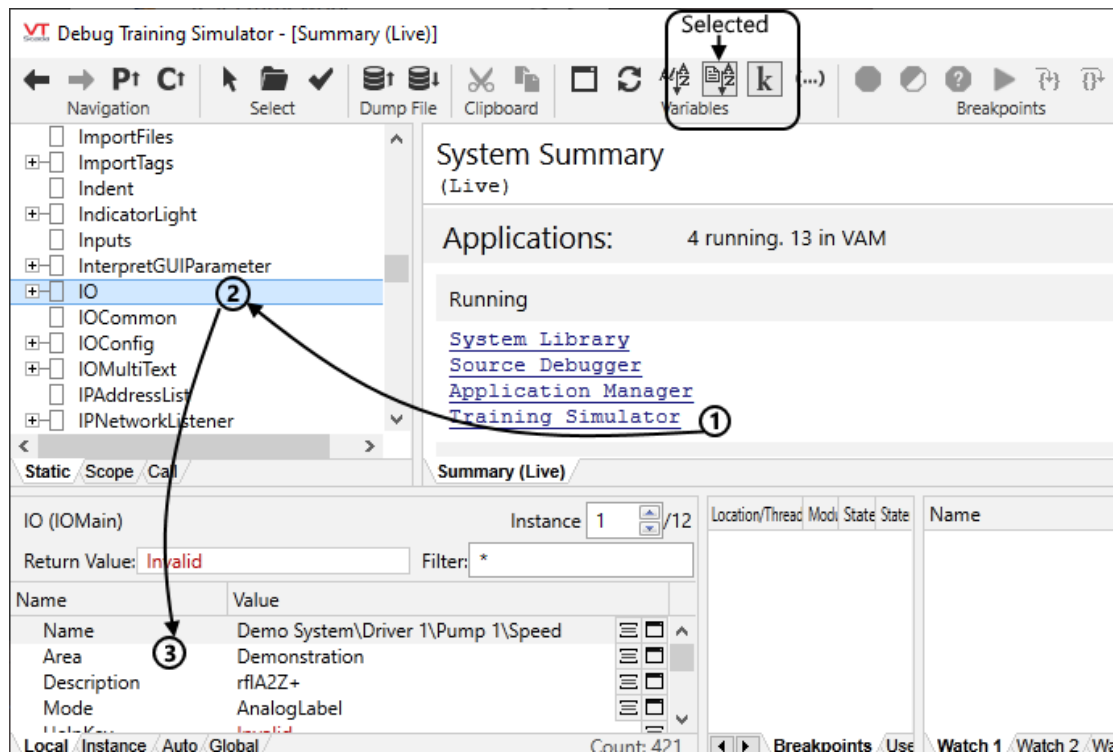


Figure 14-3 The Source Debugger is a very powerful tool for examining applications

To find the names of properties within a tag, follow the numbers in the preceding figure. It will help if the Sort Tree tool is selected in the toolbar as indicated.

1. Click to select the running application.
2. This is the "static tree list". Find and select the tag type that you want to examine.
3. This is the "module content window" with the "local" mode selected. Scroll to view the property names.

Note: Names that begin with a symbol such as # or @ are for use by VTScada. They are unlikely to be useful in your expressions.

Caution: The Source Debugger and other diagnostic tools have code-level access to running applications. This is part of the reason the Security Best Practices topic in the VTScada documentation advises you to keep the VAM hidden from unauthorized users while your application runs.

Exercise 14-1 Expressions that use tag properties

1. On the Station 1 page, there is a label, "Driver Status".
2. Replace that static text with an expression that adds the driver's current quality value.

Use Application Properties

Tip: A frequent request is for a way to let an operator change certain properties of tags, but not give that operator Tag Modification privileges. Among other reasons for why you might want to read application properties in your expressions, the following will show you how to achieve this goal.

Expressions can read¹ application properties that are declared in the [System] section. To do so, write:

```
\PropertyName
```

in your expression, substituting the actual property name for "PropertyName". For example:

```
\DispMgrResizable
```

```
\DispMgrFullScreen
```

Properties stored in either the [Application] or the [Layer] sections are also available, but the syntax to retrieve them is somewhat longer:

```
\AppLayer.LayerSettings.PropertyName
```

Examples include:

```
\AppLayer.LayerSettings.Name
```

```
\AppLayer.LayerSettings.DefaultLanguage
```

Exercise 14-2 Moderate your scan intervals

The default scan interval for most tags is one second. That might be right for some equipment (or even too slow), but for most analog values it's far too frequent. Scanning only every two or three seconds (instead of one) might be perfectly adequate most of the time, and will reduce the load on your network traffic and Historian enormously.

But, maybe you want to be able to change the scan rate depending on circumstances. In that case...

1. Open the Edit Properties page of the Application Configuration dialog.
2. Ensure that you are viewing the Advanced Mode, not the Basic Mode.
3. Insert a new property as shown:

¹No setting of values, at least not in a steady-state expression. The assignment operator, =, is not legal syntax in this context.

VT Add Property [X]

Property Name
CustomScanRate

Section
System

Value
3

Workstation
-- default --

Comment
Custom scan / poll rate

OK Cancel

Figure 14-4 Adding a new label property

4. Save your work and close the Application Properties dialog.
5. Open the Tag Browser.
6. Navigate to find \Station 1\PLC1\Level
7. Open the properties dialog and the I/O tab.
8. Right-click on the Scan Interval field to add a tag parameter expression.
9. Enter the expression as shown. Be sure to deselect the Optimize option. (*)

VT Tag Parameter Expression [X]

Enter New Expression
\CustomScanRate

☐ Optimize to only evaluate at tag initialization

OK Cancel

Figure 14-5 An expression for the scan interval

10. Click OK to save your work for the editor, then the tag.
The Scan Interval field should turn blue and show the number 3.
11. Close the Tag Browser.
12. Open the Idea Studio and add an Edit Property widget to the Station Status page.
Tools >> Standard Library >> Edit Property
(Typically, you would create a new page for this sort of operational control, but we're saving steps.)
13. Configure the widget as shown:

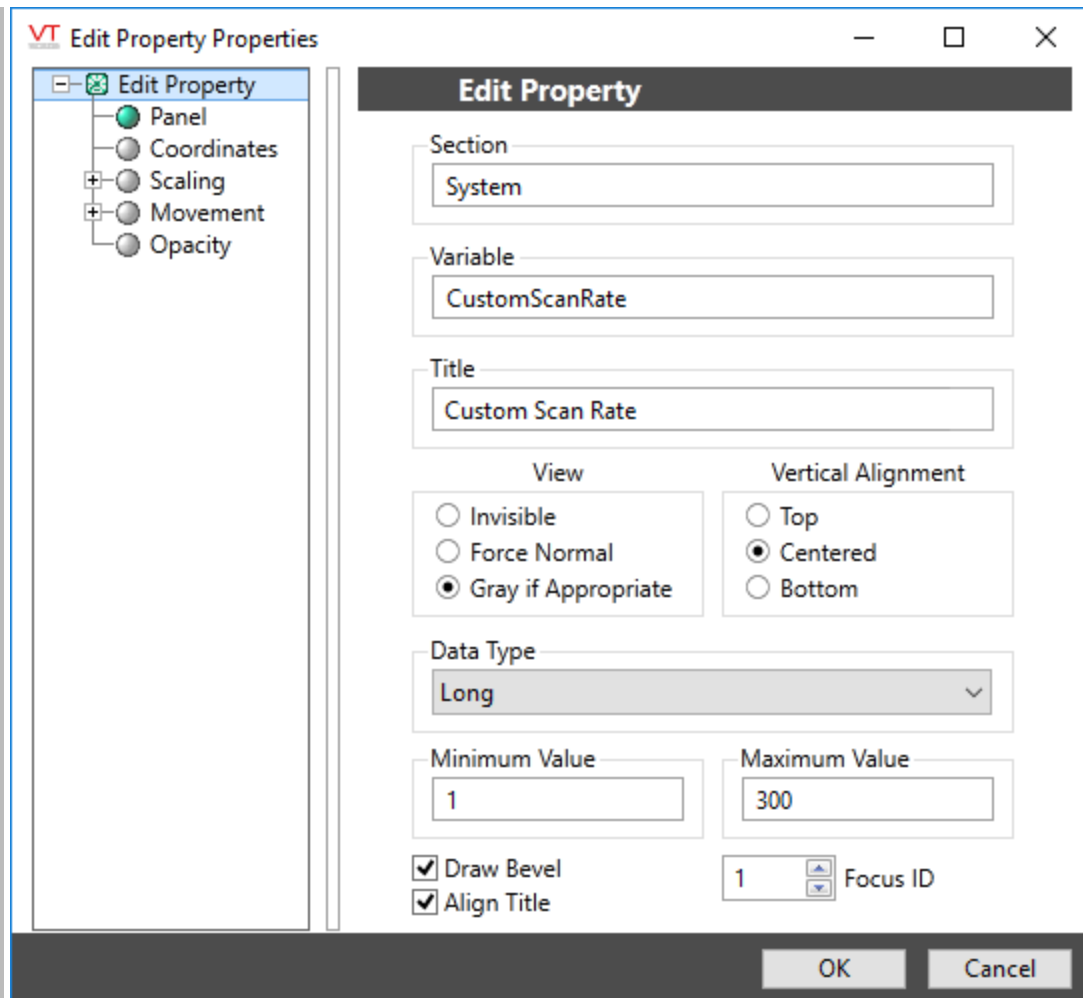


Figure 14-6 Configuration of your Edit Property widget. Note the data type & min and max values.

14. Save your work and run the application. You should be able to adjust your scan interval as required. (To see the effect, you would need to have the tag's properties dialog open while changing the value, and would need to change back and forth between tabs to see the changed value.)

Caution: (*) What you've just seen is potentially dangerous. A tag with a non-optimized parameter expression (and all of its child tags) will restart each time the parameter changes. It also costs slightly more than other tags in terms of memory and CPU load. And, there's a risk that the parameter could be changed to an inappropriate value or changed frequently. A non-optimized parameter expression is valuable in the right circumstances, but use with extreme care!

Time and Date Functions in Expressions

You might use time and date functions for many reasons: as part of a display, to act as a trigger, or to query history. The following is offered as a guide, without listing all the available [Time and Date](#) functions.)

In code, VTScada uses numeric timestamps that represent the number of seconds since January 1, 1980 (known as the Unix epoch). For any given time, note that there is both a UTC value and the local time zone value. Any operation that reads from or writes to the Historian must use UTC. You and your operators will probably prefer working with local time, expressed in a form such as "Tuesday, February 4, 2025 at 9:58 am. Tools are available for all formats.

If your expression needs to find the current time or date, use one of:

Now (interval)	Returns the number of seconds elapsed so far today, updated every (interval) seconds.
Time (seconds, format)	Formats the number of seconds since midnight into a form that's easier for a human to read and understand. For example, <code>Time (Now (1) , 4)</code> The format codes can be found in the reference section of the documentation.
Today ()	Returns the number of days since January 1, 1970.
Date (days, format)	Translates the number of days since Jan 1, 1970 into a human-friendly format. For example, <code>Date (Today () , 4)</code>

The function, [CurrentTime](#)(), will return the total elapsed time in seconds to three decimal points since January 1, 1970. You can control whether the return value uses the local time or the UTC time with an optional parameter, where the default is to return the local time. Again, all Historian operations must use UTC time exclusively.

But note that [CurrentTime](#)() works only in script mode and not in steady state. There are two options for obtaining the current time when working in steady state. The first method is to calculate it to the nearest second:

```
Today() * 86400 + Now(1)
```

The second method is to create a [Workstation Status Tag](#) and below that add an I/O and Calculations tag in analog mode, with the address `Expression:CurrentTime()` The scan interval of the I/O tag will set the frequency at which the expression is refreshed, but note that this will be relative to whenever the tag is restarted, not a even interval of seconds.

Tip: When creating expressions for reporting purposes, including work with the HDV, it is common to require start time and end time boundaries. For these you should consider using the [TimeUtils](#) library, which works in both script and steady-state.

To create a time-based trigger, consider using any of:

AbsTime (enable, interval, off-set)	Becomes TRUE when any repetition of intervals since midnight, plus an offset, is reached.
TimeArrived (timestamp)	Returns TRUE when the specified UTC timestamp is reached.
TimeOut (enable, seconds)	Returns TRUE when the enable parameter has been TRUE for an interrupted number of seconds.
RTimeOut (enable, seconds)	Returns TRUE when the enable parameter has been TRUE for a cumulative number of seconds.

More About Formatting

Having the number of elapsed seconds since a given reference point is useful to a computer, but not easy for humans to read. The `Time()` and the `Date()` functions are available to turn those seconds into a more friendly form. To use them, you will need to refer to a table of available formats to select the one you want.

In general, the `Time()` function looks like so: `Time(Timestamp, Format Flag)`

To obtain the current time, you could use any of the following (examples assume that it's 9:35 p.m.)

```
Time( Now(1), 2) ---> 21:35:00
```

```
Time( Now(1), 7) ---> 09:35 PM
```

The formatting codes and formatting strings can be found in the reference section of the documentation.

Examples:

Assuming that today is Monday, Aug 13th, 2012, then:

```
Date( Today(), 3)
```

... yields, "08-13-12"

```
Date( Today(), "dddd MMM dd, yyyy")
```

... yields, "Monday Aug 13, 2012"

Exercise 14-3 Time and Date

Preparation: Create a Calculation tag and use its expression editor for the following:

1. Enter the following expressions into the expression editor, closing the editor after each to see the result:

`Now(1)`

`Now(5)`

`Time(Now(2), 2)`

`Today()`

`Date(Today(), 2)`

Turn a timestamp into a human-friendly date and time

An example earlier in this topic showed how to calculate a timestamp with `Today()` and `Now()`. To convert in the opposite direction to turn a timestamp into a date and a time, you need to know how many times 86400 goes evenly into the number. Simple division, truncating to the nearest integer will give you that.

```
Int( SomeTimestamp / 86400 )
```

You also need to find out how many seconds remain for the portion of time measured into the day. You can do that with the modulus division operator, `%`, which returns the remainder. For example...

```
SomeTimestamp % 86400
```

...returns the number of seconds into the current day, much like the Now() function.

Triggers in Expressions

The AbsTime() function (for example) will return a TRUE value when a given time interval has passed. This is a handy feature if you want to schedule an event to occur every hour on the hour, or perhaps every morning. Note that, when used in an expression (as opposed to a VTScada script module), the AbsTime function does not reset itself to false.

The format for the AbsTime() function is:

```
AbsTime(Enable, Interval, Offset).
```

- The first parameter, Enable, allows you to switch the function on and off. If Enable evaluates to FALSE, then the AbsTime function will be switched off. Otherwise, it will be on.
- The second parameter, Interval, gives the time in seconds to wait between events. Note that this is an absolute time, not an elapsed time. (Hence the name of the function.)
If you want an expression to run every 24 hours, you would set the interval to 86400. If hourly, set the interval to 3600. In any case, the interval must be greater than 0.
- The third parameter is an offset, measured in seconds to wait after the interval has been reached. If you want your expression to run at five minutes past the hour, every hour, then you would set the interval to 3600 and the offset to 300.

Resetting functions in expressions

AbsTime (and other triggering functions) would be much more useful in expressions if you could make them reset automatically. Fortunately, there's an way to do that: Enclose the trigger within a Latch() function.

The Latch() takes two parameters. When the first parameter becomes TRUE, the overall Latch function becomes TRUE. When the second parameter becomes TRUE, the overall Latch function, including both parameters, resets to become FALSE.

Example:

```
Latch( AbsTime(1, 10, 0), AbsTime(1, 10, 5) )
```

On any ten-second mark (0, 10, 20, 30 ...) the first parameter will become TRUE. Latch will become TRUE. On any ten-second mark, offset by five seconds (5, 15, 25, 35 ...) the second parameter will become TRUE. This will reset the Latch to FALSE. The result is a metronome of sorts, toggling on and off every five seconds.

Tip: If using an expression that sets opacity to zero, it may be helpful to also check whether you're viewing that object in the Idea Studio or in the operator view. Do so with ParentWindow()\Editing For example:
 ParentWindow()\Editing || Latch(AbsTime(1, 1, 0), AbsTime(1, 1, .5))

Exercise 14-4 Strobe Light

1. Draw a filled circle on a page.
2. In the properties dialogue for that circle, open the Opacity option.
3. Use an expression similar to the earlier example that will make the opacity toggle between true and false.

Noticing when a value changes - Watch()

A very common situation is that an expression needs to react when a change occurs. The "log on change" feature of the logger tag is an example of this at work.

The Watch() function provides you with a means to do this in your calculation tags. The format is as follows:

```
watch( InitialValue, Parameter1, ...)
```

You can watch as many parameters as you need. The InitialValue is the only mandatory parameter, and is commonly set to either 1 or 0 to specify the Watch function's initial state. For most expressions outside of a VTScada script code module, the initial value is typically set to 0.

Whenever any of the watched values change, the Watch() function will return True.

It's important to understand that there is no built-in deadband for specifying that a value must change by a certain amount before a Watch() is triggered. Even the smallest change will count.

Note that, as seen with the AbsTime function, the value does not automatically re-set back to its initial value when used in an expression. The Watch() function is best used in combination with the Latch() function, described earlier.

The Watch() function makes a very useful trigger, as it allows you to monitor any other tag (or expression, or ...) anywhere in your application.

Keep the Else's Under Control

Writing long "If Else - Else - Else ..." expressions is not efficient and can be avoided. To prevent excessively nested expressions, the property [MaxNestedExpressionDepth](#) limits you to 200 function calls (including IfElse) within a single expression. There is no need to test this limit, regardless of the complexity of your application.

Alternatives to nested IfElse structures:

A common task is to build a string based on the current value of a tag (or several tags). For this example, suppose that you need to build a message or address that varies with a tag's current state. A long and inefficient method might be to write:

```
[SomeTagName] == 0 ? "Message (or) Address 0"
:
[SomeTagName] == 1 ? "Message (or) Address 1"
:
[SomeTagName] == 2 ? "Message (or) Address 2"
:
etc. for 100 or more messages...
```

There are alternatives:

Option 1

A much easier method is to use a CASE statement in steady state. All statements will execute, but only the result of the statement matching the requested index will be returned.

1. Open the Idea Studio and then open the page, Pump Control.
2. Place Text above the Hand-Off-Auto control. (Ordinary text, not a widget.)
3. When the Edit Text dialog opens, select "Text Expression".
4. Remove the default value of "".
(Ignore the red background that appears. VTScada is complaining about the empty expression.)
For the remainder, you will build an expression in pieces. Do not press enter or OK until instructed to do so.
5. Type `Case (`
6. Expand the Function Selector tool and then select the Tag Browser tool.
7. Navigate to and select the tag, [`<Station 1\PLC1\Pump 1\Switch Position>`]
8. Build the rest of the expression until the whole looks like the following:

```
Case( [<Station 1\PLC1\Pump 1\Switch Position>],
      { 0 } "Hand",
      { 1 } "Off",
      { 2 } "Auto")
```

9. Save your work and put the page into Operational mode.
10. Operate the switch to verify that your expression worked.

Option 2

Another option is somewhat more complicated to create, but has the benefit that the messages are stored in application properties instead of code and therefore can be added to or edited by anyone with the Configuration privilege. It is not necessary to write more code when messages change or new messages are created.

1. Create a set of application properties for each situation:

```
Situation0      = Message or Address 0
Situation1      = Message or Address 1
Situation2      = Message or Address 2
(etc.)
```

2. Write an expression that uses the Variable function to concatenate the tag value onto the prefix "Situation" to return the appropriate property value.

```
Variable(Concat("Situation", [<SomeTagName1>]))
```

No comparisons or If-Else statements are required. This single line of code can return any message that you have created. For more complicated situations, use Concat to build the appropriate property name or combine several values into one message or address.

Multilingual Expressions

In VTScada, all text for the user-interface is stored as phrases, each with a matching phrase key. This is also true for nearly(*) all text that you add to your application including tag descriptions, engineering units, labels and more.

(* Exceptions include tag names, tag areas and user-defined application properties, all of which are stored exactly as written.)

For example, anywhere that displays the English label "1 Day", actually uses the key "1DayLabel". This allows VTScada to translate the user interface to another language such as French.

en.csv	fr.csv
1 #LanguageID,en-us	1 #LanguageID,fr
2 #LanguageName,English	2 #LanguageName,Français
3 #TwilioVoice,alice	3 #TwilioVoice,alice
4 1DayLabel,1 Day,1,One day	4 1DayLabel,1 jour
5 1HourLabel,1 Hour,1,One hour	5 1HourLabel,1 heure
6 1MinuteLabel,1 Minute,1,One minute	6 1MinuteLabel,1 minute
7 1MonthLabel,1 Month,3,One month	7 1MonthLabel,1 mois
8 10ver2ArrowLabel,1/2 Arrow,3	8 10ver2ArrowLabel,1/2 Flèche

Figure 14-7 English and French phrases for the VTScada user interface

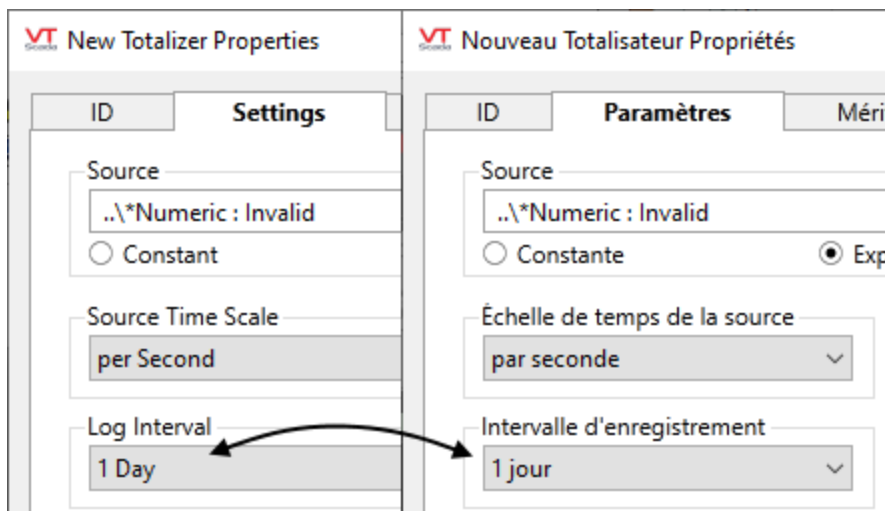


Figure 14-8 1DayLabel as used in a Totalizer tag.

The advantage is clear for those who need a user-interface that can be displayed in a language other than English. But when creating expressions that involve text and any part of the user interface, you must be prepared to work with phrase keys rather than directly with text.

Note: VTScada comes with French, Simplified Chinese and Traditional Chinese phrases for most user-interface text. You must create your own translations for your text.

Get a phrase, given a phrase key

A phrase is one or more words. Because one language may use words in different order compared to another language, it is best to store complete labels as phrases rather than attempt to assemble labels from individual words. For example, for a phrase key such as "GreenButtonLabel", the English phrase might be "The green button", while the French phrase would be "Le bouton vert".

Given a key, you can find the matching phrase in the currently selected language by using the `\GetPhrase` function like so:

```
\GetPhrase("GreenButtonLabel")
```

A parameterized phrase has one or more spots (parameters) in the form %0, %1, etc., that can be replaced on the fly. The advantage is that the parameter placement can be shuffled within each translation. For example, if the expected parameter to ParmButtonLabel is a color name, the English phrase might be "The %0 button", but the French phrase would be "Le bouton %0". In this case, your expression should use the \GetParmPhrase function like so:

```
\GetParmPhrase("ParmButtonLabel", "GreenLabel")
```

In both of the examples just shown, the parameters are enclosed in quotation marks because it is the name of the key that is being passed to the function. You could also use variables that hold those names, which would be more typical for the parameter being passed to the phrase.

Phrases in tag parameters

Tag parameters (such as the Description parameter) present a special case: These are typically snapshot expressions that evaluate only when the tag starts or re-starts. Changing from one language to another does not cause your tags to restart.

The solution to this problem is not to deselect the Optimize option in the parameter expression editor. Rather, it is to use a \ParmPhrase structure instead of a \GetPhrase or \GetParmPhrase function. (In situations where a non-optimized tag parameter must be created, use [MakeParmPhrase](#))

Note: \ParmPhrase is not a function and you will not find it in the function reference (other than a page that will tell you "This is not a function"). When a \ParmPhrase structure is used in certain tag parameters, the translation will update automatically whenever you change the current language. This is true for the following:
Description, Units, Position text on the selector switch, Equipment Type, IngredientName, Recipe Tag Mode, Pump OnText and OffText, the PLC alarm tag's PLCType and AlarmType, and the I/O tag mode.

A parameter phrase structure (\ParmPhrase) has a form much like the \GetParmPhrase function. For example, if your tag's description should describe the color of a button, you would create a parameter expression like so:

```
\ParmPhrase("ParmButtonLabel", "GreenLabel")
```

If any of the parameters that you pass to \ParmPhrase are not phrase keys, then they are simply passed through and not translated.

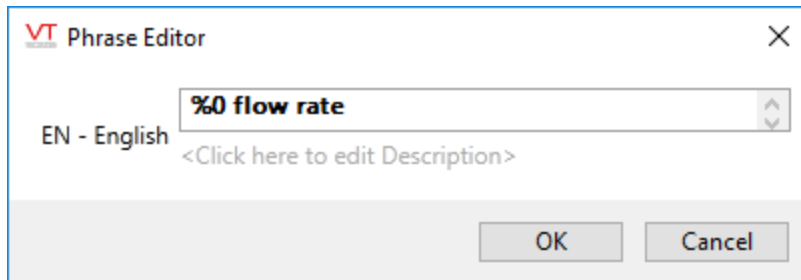
Caution: Do not use \ParmPhrase structures alone outside of tag parameter configuration. Elsewhere, a \ParmPhrase structure may be passed to a \GetPhrase or \GetParmPhrase function call in place of a phrase key, but there is seldom an advantage to doing so.

Create phrases and obtain their keys

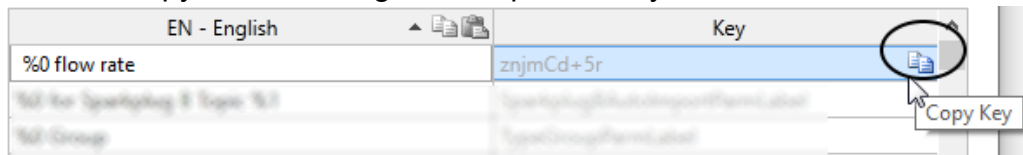
New phrases and phrase keys are created automatically whenever you save a new phrase, such as a new tag description or a new label on a page. Having created a new phrase, you can then use the Languages page in the Application Configuration dialog to search for the matching key. A more efficient way to create new phrases and keys is to work directly in the Languages page. An example follows:

1. Open the Languages panel of the Application Configuration dialog.
2. Ensure that the Key column is displayed.
Click the View button and select Key from the pop-up dialog if it is not.
3. Click the Insert button.
The Phrase Definitions dialog opens.

4. Click where indicated to add the new phrase.
5. Place a %0 to mark the position for the first replaceable parameter. (%1 for the second, etc.)



6. Click where indicated to add a description that will help translators know the meaning of the phrase.
7. [Optional] If you have added another language to your application, create the phrase in that language now.
8. Save the phrase.
Your new phrase should be selected in the list. *It will not be saved until you click Apply, but do not click that button yet.*
9. Click the copy tool, to the right of the phrase key.



The key shown here is an example. Yours will differ.

10. Use the key in your expression.
If creating several phrases, it may be helpful to keep a text editor handy, copying phrases and keys to it as you create them.
11. Click Apply to save the new phrase and its key.
Because the list is sorted alphabetically by phrase or key, the new entry is unlikely to be visible on the screen after you click Apply.

Example: A tag description that reflects the parent tag:

This is a relatively common task. You have a tag structure and would like the description of each child tag in each instance to identify which instance it belongs to. Perhaps (for example) you would like the description to read "Primary pump in the XX region" where XX should be the Area name for the tag.

Begin by opening the Language page in the Application Configuration dialog and creating the phrase, "Primary pump in the %0 region", as described earlier. For the sake of this example assume that the assigned key is named "EU43y4gag".

Create translations if required.

Open the tag's properties dialog and create an expression as shown:

The screenshot shows a form with the following fields:

- Area:** A dropdown menu with "Northern" selected.
- Description:** A text box containing "Primary pump in the Northern region".
- Help Search Key:** A text box with a tooltip overlay. The tooltip contains the text: "Tag Parameter Expression. \ParmPhrase('EU43y4gag', Area)" and "Modification not yet saved".
- Type:** A text box that is currently empty.

Figure 14-9 "EU43y4gag" will not exist as a key in your application. A tag's Area is never translated.

15 Best Practices for Scaling Up Your System

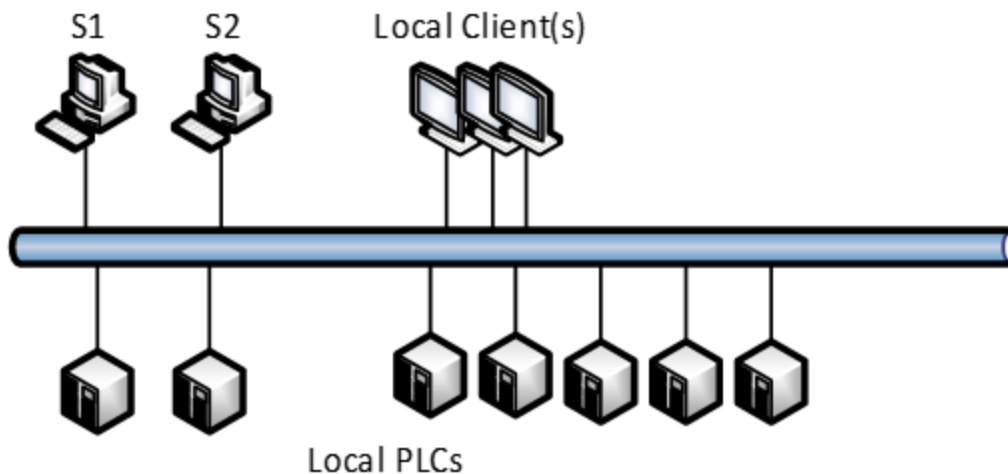
All systems require care and that's especially true of large systems. But how big is "large"? There are several ways to define this:

- Total number of I/O
- Update rates for I/O
- Number of connected devices
- Number of users
- Data storage volume and duration
- External data reporting needs

It may be more accurate to define "large" as "A system that needs extra design considerations for loading" rather than to set hard numbers for any dimension. Each system has its own set of requirements that will influence the design, including:

- Physical distribution
- Logical distribution
- Number of I/O & update rates
- Fault tolerance
- Security
- Data storage
- External data access (viewing & reporting)

Variation 1: In-plant systems



Everything is under one roof with multiple PLCs, RTUs and controllers. There may be large amounts of I/O, updating rapidly. All users need access to all information at all times. Examples include manufacturing facilities, food and beverage production, and thermal power generating stations.

System Overview

These installations are typically not network-limited and therefore high data rates between servers and PLCs are possible. The key to this architecture is to provide a sufficient number of servers and sufficient resources to those servers to keep up with high data rates and high volume logging. These systems are often mission critical and therefore external access may be restricted and redundancy & fault tolerance will be important.

Physical Distribution

Servers will usually be in one or more secure environments within the plant. Operator workstations may be in one or more control rooms. Dedicated operator terminals for certain equipment or process control may be located on the plant floor. A separate control LAN is suggested for security.

Number of I/O & Update Rates

Local PLCs will support very fast polling, meaning alarms and operator screens can be very responsive. This will result in high loads on the I/O and Historian servers. Having dedicated servers for both I/O and Historians will help manage these loads within the system. For very high loads, a single server may not be able to keep up and therefore services may need to be distributed across multiple machines.

For example, in a three server setup, PLCs are broken equally into 6 groups with the following I/O server lists respectively, such that each server will still have an equal load under any failure scenario

- Driver A1
 - Server A
 - Server B
 - Server C
- Driver A2
 - Server A
 - Server C
 - Server B
- Driver B1
 - Server A
 - Server C
 - Server B
- Driver B2
 - Server B
 - Server A
 - Server C
- Driver C1
 - Server C
 - Server A
 - Server B

- Driver C2
 - Server C
 - Server B
 - Server A

Operator Workstations and Terminals

These can be Thin Clients or Thick Clients. There are benefits to either choice.

Thin Clients are more cost effective for full-system workstations, but their loads will be shared by the Thin Client Server, so many simultaneous connections could overwhelm the server and result in them slowing each other down. Dedicated or additional Thin Client servers can help address this. Realms could be used to limit a user to the parts of the system that a terminal or workstation is controlling.

Thick Clients may be a bit more responsive and won't load the servers as much when in use. Tag Area filtering, Master/Subordinate, or start conditions tied to workstation settings can be used to limit the tags running, which will reduce the load. It may also allow for more cost-effective operator terminals if the number of tags needed is very small relative to the full system. (For example, a 200-tag runtime license is more affordable than a 100K Thin Client).

Fault Tolerance

Be sure to provide sufficient backup servers such that all functions can continue properly during designed-for failure scenarios. As an example, if dedicated I/O and Historian servers are needed for the system to perform well, then they should not also serve as backups for each other. Dedicated servers should fail to dedicated backups rather than ask one remaining server to handle both loads. The loads on the remaining server during a failure will be too high.

Data Storage

High update rates will mean that a large amount of historical data can be generated. Large disks and sensible deadbands are recommended. It may be necessary to delete data older than a certain length of time.

External Data Access

External network access may not be permitted into this type of system. If it is permitted, a separate DMZ could be used for thin client connections or queries from business systems. VPN connections to the DMZ are often preferred to public access over TLS. If alarm notifications are used they will often be air-gapped. (For example, using SMS modems instead of Twilio.)

Three Server Configuration Examples for Variation 1 (In-plant system)

Drivers: I/O Primary, Historian Primary

Logging: Historian Primary, I/O Primary

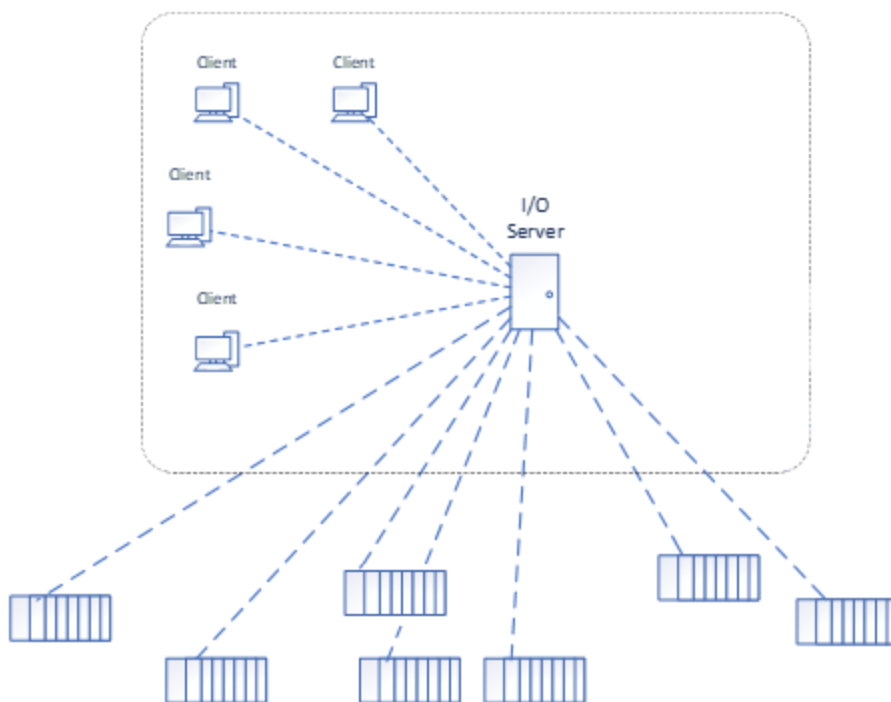
This would result in all load on a single server if either server goes offline. If we've identified that load as too high during normal operation (which is why it was split across two servers) then we don't want that to happen during a server failure. Adding dedicated backups can solve the problem. (but takes us to 4 servers).

Drivers: I/O Primary, I/O Secondary**Logging: Historian Primary, Historian Secondary**

This could be tightened to three servers if we're just designing to account for a single server failure. In this scenario a General Secondary server could serve as the backup for whichever Primary server is offline.

Drivers: I/O Primary, General Secondary**Logging: Historian Primary, General Secondary**

Consider power and networking for failure scenarios as well. Dual-homed networks, UPS's and redundant power supplies can all help protect against single points of failure in your systems.

Variation 2: Distributed I/O systems

In a distributed I/O system there is centralized monitoring of remote assets. There is no local control of field devices but there is a direct interface. Examples include wastewater lift station monitoring, and oil & gas well monitoring.

These systems are characterized by more limited data connectivity between the Remote RTUs and VTScada servers. RTUs will likely be connected via cellular, radio, satellite, or dial up, which will limit how often and how much data can be retrieved from them. These systems will have a lower rate of I/O changes and can often support higher tag counts with less hardware. The key is to manage the more limited connection to the remote devices without losing the responsive user experience of a real-time system.

Physical Distribution

Servers are usually located in a control room or a cloud environment. Clients may be co-located with the servers or (especially in a cloud environment) could all be thin clients.

Data Retrieval

The choice of communication protocol is going to make an important difference. With verbose protocols that require a lot of negotiation to send updates, the data rates from devices are going to be very low. It can be worth considering options that support poll for exception or report by exception schemes to ensure good data resolution and more timely updates for things like alarms, without excessive bandwidth requirements. Such schemes can also back-fill missing data if the link between the central servers and the RTUs goes offline.

Polling schemes may benefit from the Polling Driver, which ensures that blocks of communications happen within the same connection session. This is especially useful for shared communication media like radios or dial up, where each site can only be polled one at a time. The Poll Driver can also be directed to provide higher data update rates while a user is working on a particular site using the Fast Poll widget.

It may not be optimal to set all the sites to the same polling rate. Consider polling more important sites faster and less important sites slower to improve the overall system performance. Polling rates should be chosen such that you do not end up with long queues in the poll driver (or port semaphore / comm link sequencer). Long queues tend to make the system feel less responsive because operator actions may have to wait a long time for their turn.

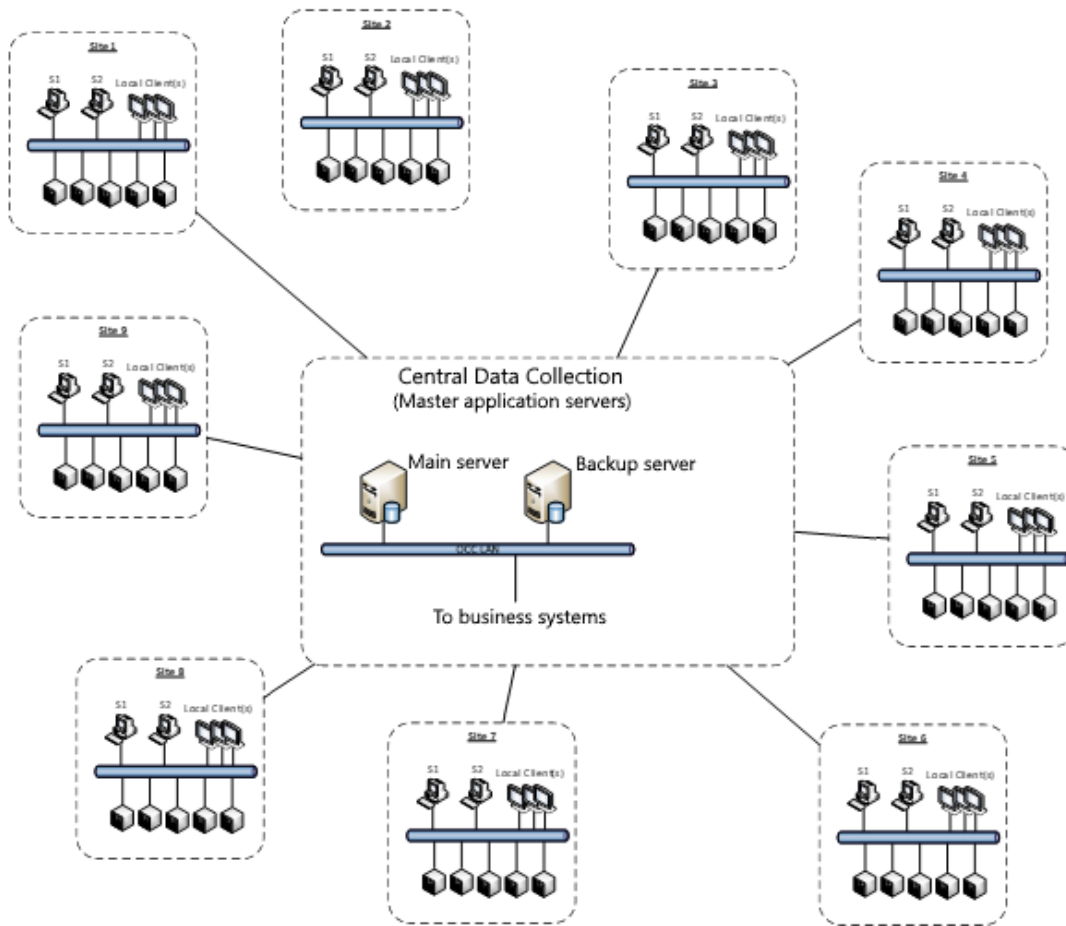
External Access

If large numbers of clients are all connecting using Thin Clients, there may be more load serving for those thin clients than in a plant environment. Dedicated Thin Client servers may be needed to handle this load. Also, those thin clients could be located in a separate DMZ network in order to provide additional security.

Fault Tolerance

All the same things apply here as for Variation 1, except there will probably be fewer servers in total because of the lower data rates.

The remote nature of the communications networks makes it highly likely that some sites will lose communications as a normal part of the operations. Poor cell networks, radio interference and the like, can cause some sites to drop off. Redundant access for control and alarms at critical sites using the Driver Multiplexor can help mitigate this risk. Local event buffering in protocols like DNP3 can help reduce the risk of losing data during such outages.

Variation 3: Geographically distributed, distributed control systems

This system is characterized by geographically distributed assets at multiple sites, each of which has a local control center. There is centralized data collection & storage for display and reporting. Examples include water & wastewater plants for a single utility or oil & gas applications with multiple well field data aggregation. A system of this type might also be mixed in with a distributed I/O system for lift station monitoring.

These systems allow high data rates from the local servers and their connected PLCs / RTUs, but can suffer from low bandwidth / high latency connections between the remote locations and the central servers. The key is to keep the high throughput local to the remote locations and reduce the traffic between servers over the WAN.

Physical Distribution

Each remote location will have at least one server to poll the local devices. These can support fast poll rates and data storage and can raise alarms locally without having to send traffic to the central servers.

The central location may be the main control center for operation of the system, and could serve as the central server for alarm notification and a gateway for remote thin client access. This can simplify configuration at the remote locations by removing the need for external access (DMZ servers, URLs, TLS certificates) or additional hardware (cell modems).

Network Bandwidth

Historian and driver deadbands will ensure the high data rates on the local servers don't result in high data rates between the local servers and the central servers.

In a Master / Subordinate architecture, each remote location should be a subordinate application. All subordinates and the master application would run on the central server (s). This keeps only the local tags running at the remote locations, which reduces the data synchronization between the central servers and the remote locations. It also allows tag counts to be lower on the remote servers, which can make the system less expensive.

If a Master / Subordinate architecture is not appropriate, it's still best to run only the local subset of tags at the remote locations. This can be accomplished through tag area filtering or workstation-specific start conditions.

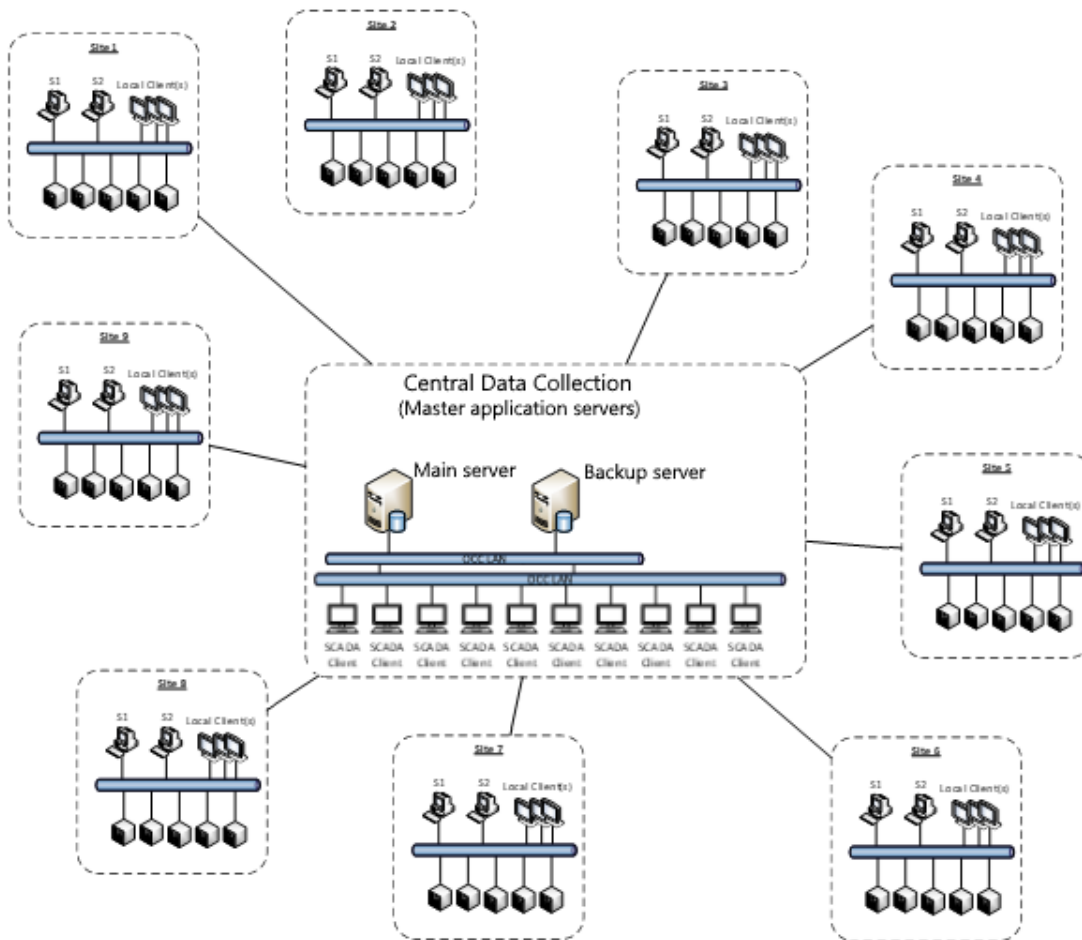
RPC link tolerances can be configured to account for unreliable connections between remote locations. These can reduce servership failover, which can consume a lot of bandwidth.

Fault Tolerance

If the remote locations only have one server, the central server could serve as a backup server, at the cost of additional network traffic in a failover scenario. If that increased network traffic cannot be handled, it may be better to put a second server at the remote locations.

The central server may also need a dedicated backup if it's performing mission-critical tasks like alarm notifications, thin clients, or providing the main operator interface. These central server(s) or central backup(s) could be in a cloud environment.

If the WAN is also prone to downtime, a redundant connection might be helpful. If outages are brief, VTScada's resynchronization and historian back-fill may be sufficient. If not, local alarm notification methods and external access (thin clients) directly to the remote locations may be needed to ensure operational continuity.

Geographically distributed, central control systems:

A system with a centralized control center and geographically distributed assets at multiple sites that are remote from the control center. There is local display and possibly control at the individual sites. Examples include Renewable power generation and Pipelines.

Options and tools for scaling up

To scale up a system, the following options and tools are available:

Load distribution (running portions of the application on different servers).

This can provide detailed control of your servers and reduce the load on all.

System partitioning

There are several ways that this can be achieved:

Master / subordinate applications can reduce the application size on servers at remote sites to only that needed by the site. Advantages are that:

- The system is partitioned by default and no filtering or tag start control is needed.
- At any site, there is no access to configuration of the system as a whole.

- Complex server lists are easier to maintain.
- Each application creates an alarm database and historians by default.

Disadvantages of master / subordinate applications are that:

- It is more difficult to maintain security because all subordinate applications must use common settings. Windows Security Integration is recommended.
- Custom tag types and widgets just exist in an OEM layer that is shared by all applications.
- There is higher overhead on the master server, using slightly more memory.

Master / subordinate systems work well for:

- Distributed applications across multiple facilities.
- Multiple individual control centers with no interactions.
- Multiple application maintenance teams.

Single Application: Using a single application works well when you have a single facility, information is shared by most users, and you have a single team maintaining the application. It can be easier to manage security because all applications are in one application. A shared OEM layer is not required (but might be used for reasons of security or control over SCADA assets). Also, one ChangeSet can contain the entire application (although it may become large).

Disadvantages of a single application are that system partitioning can require extra setup, information for all system components can be exposed and that server lists can become complex. Some of these disadvantages can be mitigated by using multiple historians & alarm databases to distribute the load so that remote sites are handling only the data related to the site.

Realm Area Filtering can restrict users' access to tags and to security realms. Tag Area Filtering can restrict which tags load on any given workstation. A modern variation on Tag Area Filtering is to use tag structures with start conditions. Put a start condition on top-level tags and use an expression that checks the current workstation to prevent tags meant for Site A from loading on the server at Site B.

Server Hardware

In general:

- Use fast, solid state drives
- Choose high clock speeds and fewer cores over more cores with lower speeds.
- Install "lots" of RAM.

Further suggestions are available on the VTScada website.

Network

Requirements can vary widely depending on data throughput. Bandwidth estimates should be done as part of the design and Trihedral can help you with this.

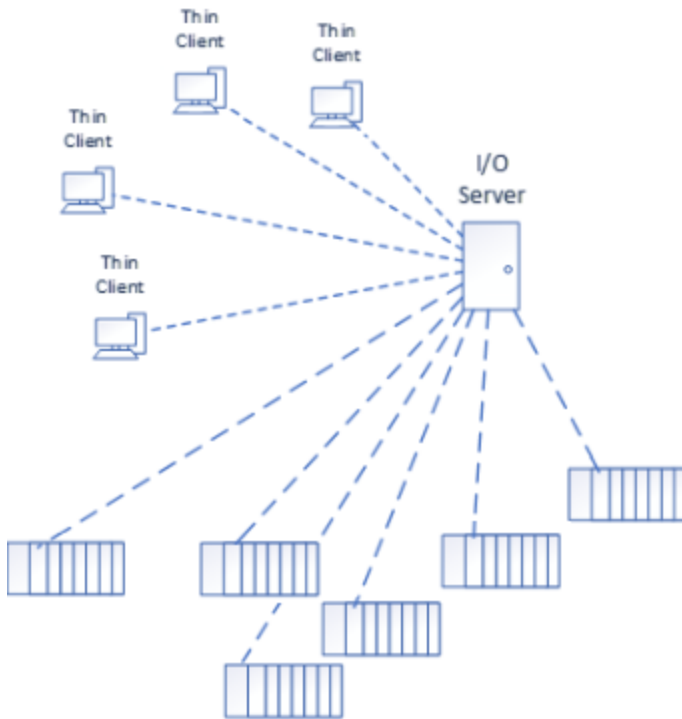
Software Settings

Set appropriate deadbands on both I/O reads and on logging

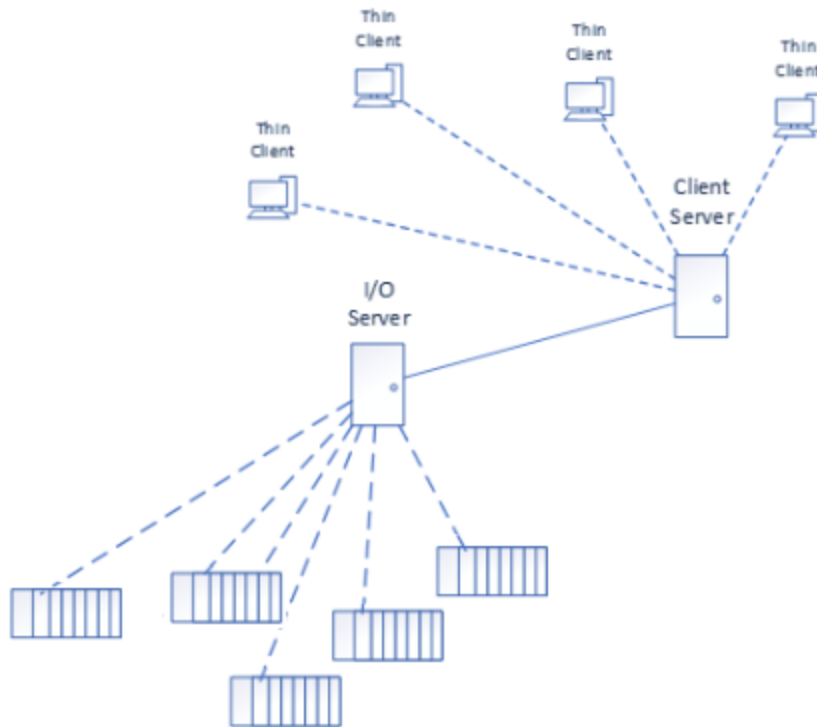
Minimize the use of non-optimized tag parameters.

Example - Adding load distribution to a simple system:

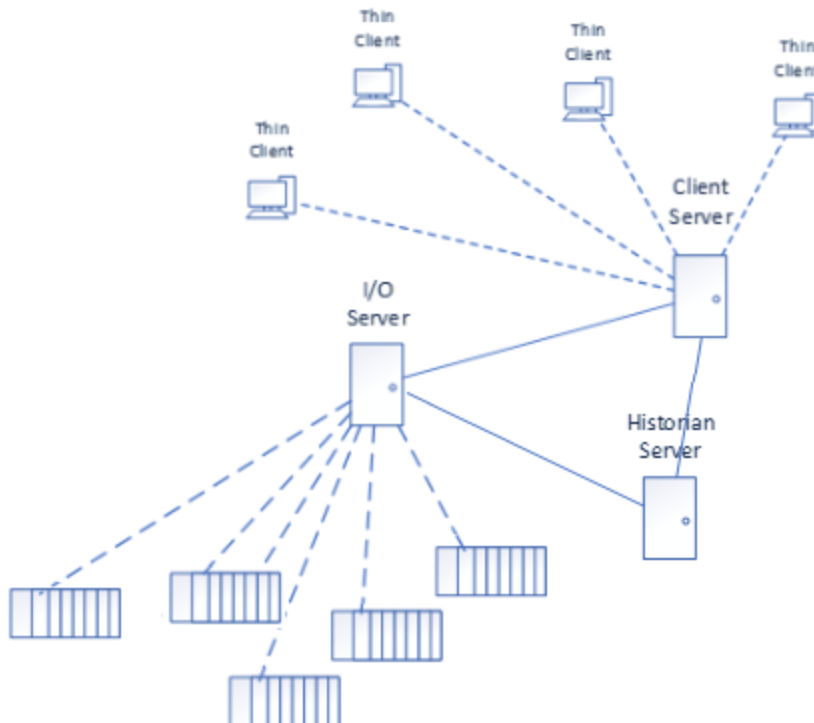
This example illustrates how a company might scale up from a simple system. The starting point is a single server with local PLCs / RTUs, and a number of thin client connections on the same network.



As a first step, you might add a second server, dedicating one to the I/O and the other to the clients.



To further distribute the load, you might then add a historian server.



As the system grows, dedicated I/O servers might be added as required.

Appendices

Bonus topics, to be covered or not during a course depending on student interest.

A Tag Tables

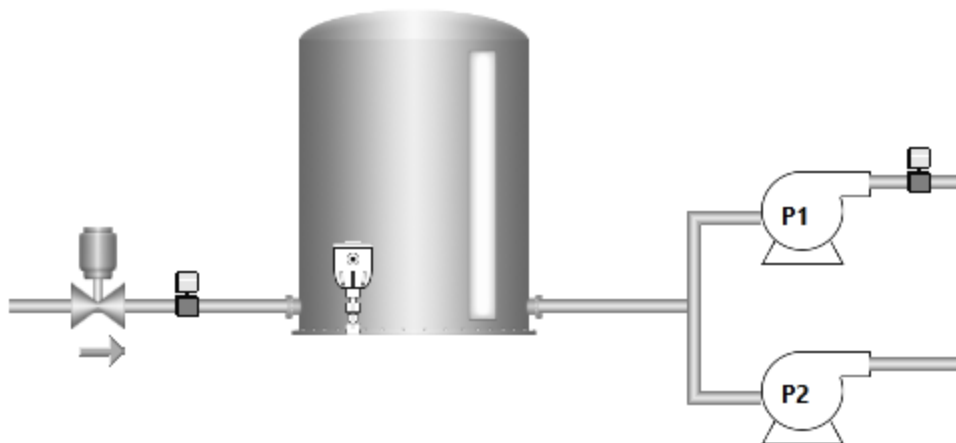
Port: TCP/IP - Address and port vary. See note in the first exercises of this workbook.

Driver: Modbus Compatible - Open Modbus TCP

Note: Tag names or descriptions may vary slightly from those used in the course exercises, to provide clarity in the table.
Only the Level tag has different scales for unscaled versus scaled process data max

Tag	PLC Address	Point Description	Units	Scale	Address field
Analog					
Level	40001	level	in.	0 - 10000 0 - 100	Read
High Level	40011	High level setpoint	in.	0 - 100	Write
Low Level	40015	Low level setpoint	in.	0 - 100	Write
High Level FB	40013	High setpoint feedback	in.	0 - 100	Read
Low Level FB	40017	Low setpoint feedback	in.	0 - 100	Read
High Alarm	40019	High level alarm setpoint	in.	0 - 100	Write
High Alarm FB	40021	High alarm setpoint feedback	in.	0 - 100	Read
Low Alarm	40023	Low level alarm setpoint	in.	0 - 100	Write
Low AlarmFB	40025	Low alarm setpoint feedback	in.	0 - 100	Read
Inflow Rate	40003/float	Rate of flow into tank	gpm	0 - 200	Read
Pump 1 Speed Set	40037	Pump 1 speed setpoint	rpm	0 - 1200	Write
Pump 1 Speed FB	40039	Pump 1 speed setpoint feedback	rpm	0 - 1200	Read
Pump 1 Speed	40035	Pump 1 speed	rpm	0 - 1200	Read
Pump 1 Amps	40031	Pump 1 current drawn	Amps	0 - 50	Read
Pump 1 Flow	40033	Pump 1 flow rate	gpm	0 - 200	Read
Pump 1 Switch Position	40041	Pump 1 switch position feedback	-	0 - 2	Read
Pump 2 Speed Set	40057	Pump 2 speed setpoint	rpm	0 - 1200	Write
Pump 2 Speed FB	40059	Pump 2 speed setpoint feedback	rpm	0 - 1200	Read
Pump 2 Speed	40055	Pump 2 speed	rpm	0 - 1200	Read

Pump 2 Amps	40051	Pump 2 current drawn	Amps	0 - 50	Read
Pump 2 Flow	40053	Pump 2 flow rate	gpm	0 - 200	Read
Pump 2 Switch Position	40061	Pump 2 switch position feedback	-	0 - 2	Read
Digital / Selector					
Pump 1 Running	3	Pump 1 running / stopped			Read
Pump 1 Start	4	Pump 1 start (Hand)			Write
Pump 1 Stop	5	Pump 1 stop (OFF)			Write
Pump 1 Auto	6	Pump 1 auto			Write
Pump 1 Fault	7	Pump 1 fault			Read
Pump 2 Running	23	Pump 2 running / stopped			Read
Pump 2 Start	24	Pump 2 start (Hand)			Write
Pump 2 Stop	25	Pump 2 stop (OFF)			Write
Pump 2 Auto	26	Pump 2 auto			Write
Pump 2 Fault	27	Pump 2 fault			Read



B Document Your Application

After creating an application, it's not a bad idea to provide instruction for the people who will be using it. There are several ways that you can do this. The following is a list of your options from easiest to more challenging:

Place helpful text on your pages.

Use standard, single-line [Text](#) for short notes and [Multi-Line Text Widget](#) for paragraphs.

Use [Tool Tip Widgets](#)

These reduce screen clutter while still providing a brief note that is visible only when someone hovers over the widget.

Create context-sensitive labels with [Multi-Text Widgets](#).

These can contain up to 16 messages, each of which is shown at progressively higher values of a linked tag.

Link [Program Spawn Widgets](#) to PDF files.

Use any text editor that can save to the PDF format to create concise instructions with all imperative details. There is no limit to how many program spawn widgets you can draw, so keep the content of each file focused on a specific task. When an operator clicks the widget, your file will open using the default PDF viewing program installed on your workstation. (This is usually your web browser, unless otherwise configured.)

When configuring the widget, there is no need to specify a viewing program. Simply enter the full path and name of the file to be opened. The label on each button should tell the user what they can expect to see after they click.

Create a user-defined help widget and a parameterized page.

You can create your own help pop-up that will display custom text that you store in tag instances. See [Create a Help Widget](#) for an instructional tutorial.

Use one of three methods to implement F1 help in your system.

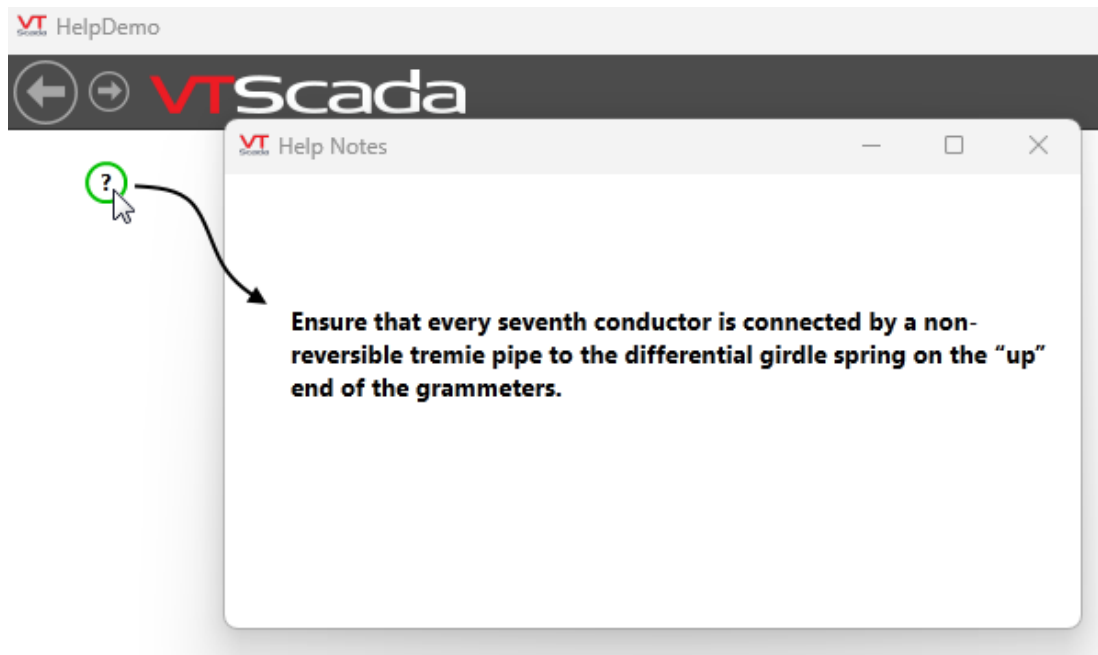
These are advanced configuration options and two of the three require the purchase of a help authoring program, but they provide a way for you to implement the "press F1 for help" paradigm instead of using a label or button.

Refer to the reference topic, [Custom Help Files](#).

Create a Help Widget

With a user-defined widget, a parameterized page, and a short expression, you place a "Help" button anywhere that is helpful to display instructions or other information that you store in any tag.

Note: This advanced tutorial topic assumes that you already know how to create [Parameterized Pages](#) and how to [Create Widgets](#). If not, you are advised to learn to use those features before proceeding. This advanced tutorial skips steps that someone familiar with those technologies would know.



Elements:

Tags

You will use the Help Key field of your tags to store the instructions. This is a non-standard use of the property and therefore somewhat at risk of breaking should Trihedral change the definition of that property in a future release. However, due to the fact that this property is widely employed for non-standard use, that risk is remote.

User-defined widget

In the preceding example, the visible portion of this is simply a circle and a question mark. The widget is defined as a "Tag Widget" and configured such that it can be linked to a wide selection of tag types.

Within the widget is a page hotbox that uses an expression to pass contents of the linked tag's HelpKey field to a parameterized page.

Parameterized page

This is a pop-up page containing a Multi-Line Text widget. The widget is configured to display whatever is passed to the page via its parameter.

For visual clarity, the widget in the preceding example uses the Analog Font.

Steps:

A) Create the Parameterized Page

1. Use the Idea Studio to create a new pop-up page.
 - a. Give the page a descriptive name and title such as "Help Notes".
 - b. Do NOT add it to the menu system.

2. Add a parameter to the page.
 - a. Give the parameter a name such as "TagWithHelp".
 - b. Set the parameter type to Tag.
 - c. Select all the tag types that you might want to use with this system. (You can edit the parameter later to add more.)
3. Add a Multi-Text widget to the page. (Important - do not use any text or widget other than the Multi-Text.)
 - a. Size the widget to accommodate any message that you might want to display.
 - b. Select a font that is relatively easy to read. (You might create your own.)
4. In the Home ribbon of the Idea Studio, click the Source Code tool to open the page in a text editor.
5. [Important] Close the Idea Studio before making any changes to the page using the text editor.
6. Within the text editor, look for the line that begins, `Scope(\Code, "Library", TRUE)\DrawMultiText(`
7. Modify this line as follows. *The example shows only a portion of the line, emphasizing what to modify in bold text. Do not delete code that follows this portion of the example.*

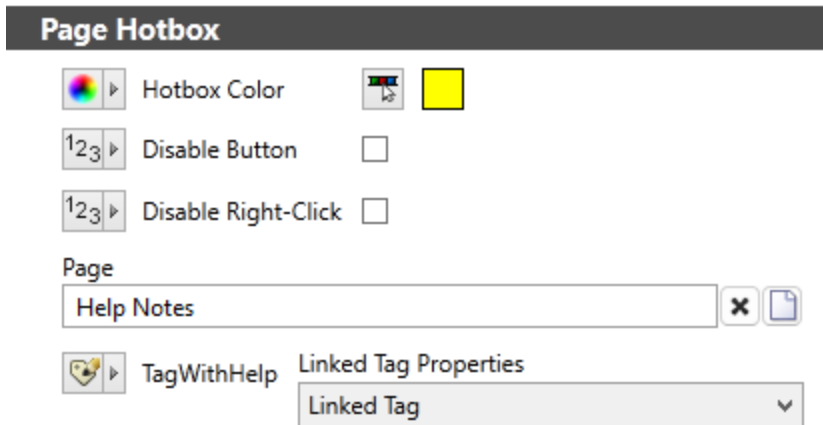
```
Scope(\Code, "Library", TRUE)\DrawMultiText(TagWithHelp\HelpKey, 0, 0,
```

8. Save the text file.
9. In the VAM, use the Import File Changes button for your application.

B) Create the Help Widget

1. Working in the Idea Studio, use the File menu to create a new Tag Widget.
 1. Select all the tag types that you might want to use with this system. The selection set should match that of the parameter you created earlier.
 2. Give the widget a name such as "Help Reference".
 3. Deselect the "Include..." options.
2. In the upper left corner of the widget, draw whatever shape you want to use to represent "click here for help". The example at the top of this topic used a circle with diameter 20 pixels and a question mark.
3. Overlay the symbol with a Page Hotbox widget.
4. In the Page Hotbox widget, select the "Help Notes" page.

5. Configure the parameter to use Linked Tag as shown:



6. Save your work.

Put it All Together

1. Open the properties dialog of a tag (choose a type that was included in the lists earlier).
2. Type a message in the Help Key parameter, then close the dialogs.
3. Draw your new widget on a page, then link it to the tag you just modified.
4. Close the Idea Studio, then click on your widget. A window should pop open, containing the message you put in to the tag's Help Key field.
5. Refine the appearance of the page and widgets to suit.

C Languages

VTScada ships with English phrases for all parts of the program and with both French and Simplified Chinese for (most) phrases that are visible to operators. Translations are not provided for phrases used by development tools including the Tag Browser, Application Configuration dialogs and Idea Studio.

Use the Languages panel of the Application Configuration dialog to create new translations for your application, or to modify phrases used in any installed language. All phrases (labels) used throughout VTScada may be translated here, including those for the development tools.

Your account must have the Application Configuration privilege to be able to modify phrases. For long phrases, a compressed font will be used automatically so that you can see more characters. Hover over a phrase to see a tool-tip containing the full text in your default system font. Phrases and descriptions can not contain line breaks.

Changes are not saved until you click the Apply button.

Select your language for the VTScada Application Manager (VAM)

Your selection here affects only the labels in the VAM. Application names remain unchanged. Remember that Trihedral provides translations only for operator-interface phrases. Many of the tasks performed with the VAM are for developers and therefore no translation is provided.

1. Expand the menu, located at the top, left of the VAM.
2. Open the Settings dialog
3. Select your preferred language.

Enable a language other than English for your application

1. Open the Application Configuration dialog.
2. Select the Languages panel.
3. Select all languages that you want to make available in your application.

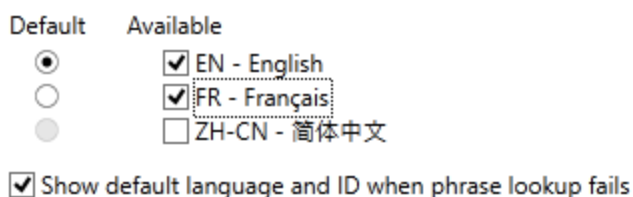


Figure C-1 Detail from Edit Properties, showing FR - Français added to the allowed languages.

Switch between languages

When more than one language is available, operators can switch between one language and another using a selector as shown. Their choice will be remembered for future sessions. The user interface will switch immediately. Note that any untranslated phrases will be shown in the default language, along with the matching phrase key.

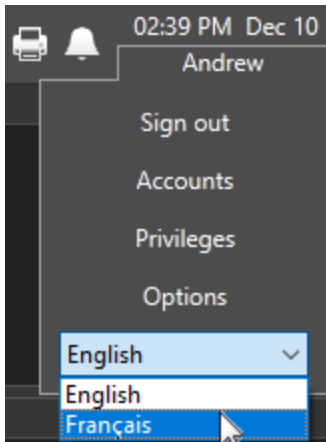


Figure C-2 After signing in, switch between languages by selecting your name, then your preferred language.

How it works:

All text used in VTScada and in your application is stored in language databases (csv files). This includes the entire VTScada user interface, text that you place on any page or widget, and tag properties such as area, description, and engineering units. All text.

For most users, this is invisible. With the possible exception of a few characters such as degree symbols that need to be updated to UTF-8, you should not notice any difference from earlier versions of VTScada. Continue to work as you always have.

Note: Those who write expressions that generate text will need to be aware of, and use, phrase-handling functions. Refer to Multilingual Expressions in the documentation.

Note: If you want to display your application in a language other than US English, use the information that follows in this topic.
When making any change to text that you have written (in any language) always use the phrase editor rather than editing the text directly.

Language CSV files

VTScada ships with three language files: en.csv, fr.csv and cn.csv, all stored in the \Languages folder. Do not edit these files because they will be over-written by later updates to VTScada.

When you change a phrase, or when you add a new CSV file for another language, your work is saved to the \Languages sub-folder of your application. This file is part of your application and will not be damaged by VTScada updates.

Choosing how to edit language CSV files

Caution: DO NOT USE EXCEL TO EDIT LANGUAGE .csv FILES.

By default, when Excel opens a file, it will alter the structure of the file for its purposes. This will destroy the file so far as VTScada is concerned.

If you choose to disregard this warning, you can mitigate the effects by ensuring that all cells in your worksheet are formatted as text and then *importing* the .CSV file.

Caution: Your editor must use the UTF-8 character set.
Failure to heed this warning will result in VTScada showing non-printable characters.

Use only a text editor that can be configured for the Unicode UTF-8 character set. In recent versions of Windows, this includes Notepad but that was not the case in earlier versions. Check before editing.

Add languages

Your application is uni-lingual until you add a new language .csv file. None of the multilingual features will be visible before this step.

To add a new language:

1. Create a folder named **Languages** within your application's folder, if one does not already exist.
2. Using a text editor that can be configured to use Unicode UTF-8, create a new file within the Languages folder of your application.
3. Name the file XX.CSV, where XX are two or more letters identifying the language you want to add.

Examples: **FR.CSV** or **en-GB.CSV**

4. Within the file, add a line that will fully identify the language and the variant. For example en-us for US English, en-gb for English, United Kingdom, or fr-ca for Canadian French.
5. Add a second line, which is the language name as it appears in that language. You may need to configure Windows to set an alternate keyboard configuration.

Example for a French Canadian language file, **FR.CSV**:

```
#LanguageID,fr-ca
#LanguageName,Français
```

6. Save the file.
7. Use the Import File Changes button in the VTScada Application Manager (VAM) to add the file to your application.
8. Open the Application configuration dialog to the Display tab of the Edit Properties page (Basic mode).
Your new language should be available.

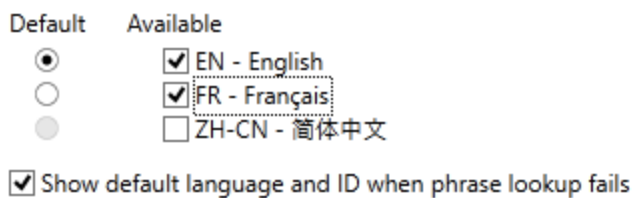


Figure C-3 Detail from Edit Properties, showing FR - Français added to the allowed languages.

9. Select the new language in the list of those available.

Translated phrases for the new language may be added at your convenience. In general, you should ensure that the new language was added successfully before adding phrases to it.

Add and Edit Phrases using the Languages Panel of Application Configuration.

- 1. Open the Application Configuration dialog.
- 2. Select the Languages panel.

Default

Available

☒

☐

☐

☐

☒ EN - English

☒ FR - Français

☐ ZH-CN - 简体中文

☐ ZH-TW - 繁體中文

☒ Show default language and ID when phrase lookup fails

View

Filter

☐ Hide OEM Phrases

☐ Hide Completed Phrases

Export

Sync

Insert

Edit

Apply

Figure C-4 Detail from the Languages panel, showing only the controls

Search and Filter

View

Opens the language View dialog. Use this to select the columns to be displayed in the editing panel.

Tip: The Key column is useful if you have created your own phrases and need to know how to refer to them in your label expressions. Use the "View" button to display this column.

The Key column is essential if you are searching for VTScada labels and properties such as "DialerLocation" by name. Use the arrows within the dialog to change the order in which columns are shown.

VT View

×

Columns

Enable	Name
<input checked="" type="checkbox"/>	EN - English
<input type="checkbox"/>	FR - Français
<input type="checkbox"/>	Key

↑

↓

OK

Cancel

Figure C-5 The languages View dialog.
The Key column is not displayed by default, but is often helpful.

Filter

Enter a word or phrase to search for and limit the display to only those occurrences of the matching value. Leading and trailing wildcards are added automatically if neither is specified. Press <Enter> or the filter button after typing a value into this field.

Caution: New phrases are not shown in a filter until after you select the Apply button. Do not create the same phrase twice, thinking that it somehow failed to save.

Hide OEM Phrases

When selected, the display is limited to only those phrases defined in the current application. Phrases from underlying layers (including VTScada) are hidden.

Hide Completed Phrases

Relevant only when at least two languages are selected. Completed phrases are those for which a translation is available in all selected languages.

Note: If you select the Hide Completed Phrases option while viewing only one language, then the grid will be empty.

Edit Phrases

There are two ways to edit a phrase:

- Click twice on the phrase in the list, then begin editing.
This is the most straight-forward method, but does not provide a way to edit the description attached to the phrase.
- Click once on the phrase, then click the Edit button to open the Phrase Editor dialog. All languages that you have selected for viewing will be shown.

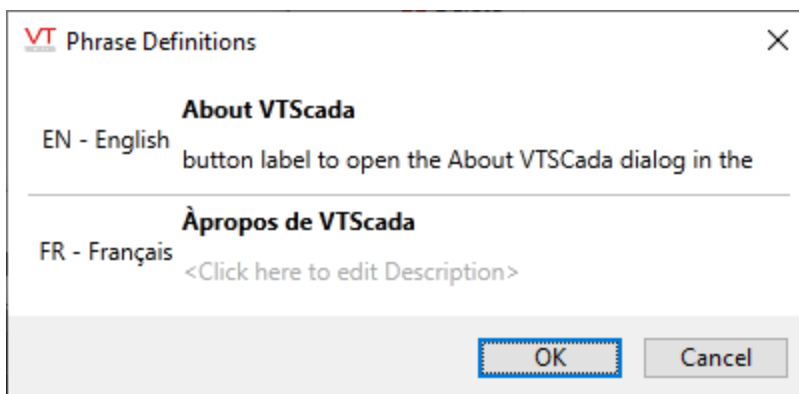


Figure C-6 The Phrase Definition Editor. Click where indicated to open an editing field.

Descriptions are used to guide translators, providing context for words that might have several definitions.

Insert Phrases

Also opens the Phrase Editor dialog. All fields will be empty until you create your phrase. Phrases that contain commas will automatically be saved within quotation marks. Phrases that contain quotation marks will have those symbols "escaped" by doubling the quotation mark.

Each new phrase is assigned an automatically-generated key. You will need this value if using the `\GetPhrase` function to display your text in a label elsewhere in VTScada. Note the copy icon, available for each selected phrase.

Phrases cannot be deleted. There is no mechanism to search out every place in code where a phrase might be used, thus making it likely that a deleted phrase would break some part of the user interface.

EN - English	Key
1 Hour	1HourLabel
1 Minute	1MinuteLabel
1 Month	1MonthLabel
1/2 Arrow	1Over2ArrowLabel

Tip: The Key column is useful if you have created your own phrases and need to know how to refer to them in your label expressions. Use the "View" button to display this column.
The Key column is essential if you are searching for VTScada labels and properties such as "DialerLocation" by name. Use the arrows within the dialog to change the order in which columns are shown.

Copy / Paste Columns

EN - English Use these to translate multiple phrases with Google Translate®. To use:

1. Select the Hide Completed Phrases option.
2. Click the copy button in the column for which you have entries to be translated. Phrases will be copied until the first blank is reached or until a preset maximum buffer size is reached. If the first row in this column is blank, nothing will be copied.
3. Open an Internet browser to a translation tool such as translate.google.com
4. Paste the copied text into the translation window.
5. Edit the translation as needed.
6. Select and copy the translated text.
7. In the VTScada languages list, paste the translations into the appropriate column.

Export / Sync

Use the Export command to send phrases to either a comma-separated value file or an Excel file for editing outside VTScada. Entries in the file are sorted alphabetically by key.

Use this file when contracting the services of a translator to generate phrases in a new language. Use the Sync command to import the edited file, synchronizing it with the phrases in your application.

Caution: Use caution if editing a CSV file in Excel. The default save option is likely to change the formatting in a way that is not compatible with VTScada.

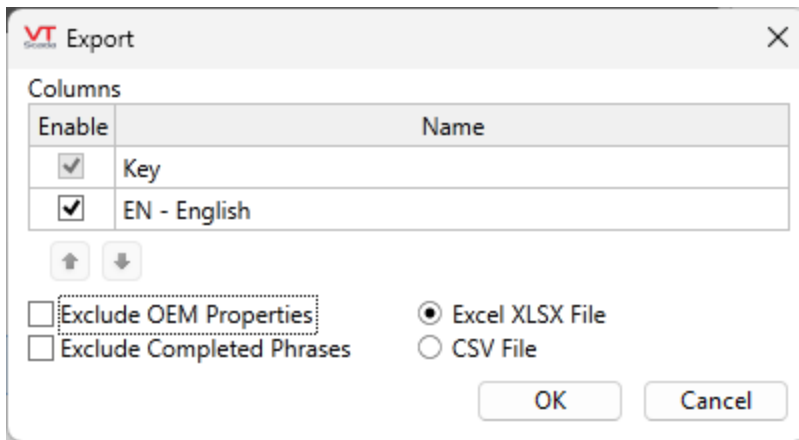


Figure C-7 Choose the target format and what information to include when exporting

Caution: The exported file will contain a version identification code. Do not delete this information. You cannot synchronize your application with a file that does not contain a valid version identifier.

The exported file will always include the key for each phrase. Take care not to edit the key. When synchronizing the file back into your application, phrases are matched to keys. A new phrase is created if no matching key exists.

Within the CSV file, wrap phrases in double quotes if they include a comma, a literal quotation mark, or use double quotation marks to signify a single quotation mark. For example, "Show a "+" rather than label" will be displayed as Show a "+" rather than label. (Do not wrap phrases in quotation marks when using the phrase editor.)

Phrases are never deleted when synchronizing. Entries that have no key value are ignored.

```

1 Key,Category,EN,EN|Comment,FR,FR|Comment
2 #LanguageID,,en-us,,fr-ca,
3 #LanguageName,,English,,Français,
4 MyPowerLabel,,Power Supply Panel,Label for power supply,,
5 1DayLabel,1,1 Day,,,
6 1HourLabel,1,1 Hour,,,
7 1MinuteLabel,1,1 Minute,,,

```

Figure C-8 When adding new entries to the CSV file, start after the first three rows.

Note:

The first three lines in the exported file are reserved for use by VTScada. Do not edit these except to add columns for new languages.

The second column is reserved for use by VTScada. In general, do not edit, and leave blank for new entries. This holds a category number for each phrase, which is used only when translating the core VTScada user-interface phrases. Category numbers are used to set the priority of work when translating phrases by identifying where in the user-interface the phrase is used. Category values are assigned as follows, where operator-facing phrases are the highest priority category.

1: Operator

- 2: Demo pages
- 3: Configuration
- 4: Activation
- 5: System
- 6: Trace
- 7: Debug

% Codes

Numeric % codes such as %0 and %1 are non-defined replaceable parameters. These are used where VTScada will substitute information into a generic phrase. For example, "Alarms muted until %0 %1". This message is displayed when an operator mutes alarms for a defined length of time. When displayed, the time when muting ends will be shown instead of %0 and the date when muting ends will be shown instead of %1.

Text % codes such as %N, %A, and others are defined replaceable parameters.

Phrase Editor

When more than one language is available for an application, this dialog will open automatically each time that VTScada detects that you type a new word or phrase in any field that will become part of the user interface. (Tag names and area properties are used in code, and therefore never translated.) You can disable the automatic opening feature of the editor, instead relying on the editor button, shown circled in the following image.

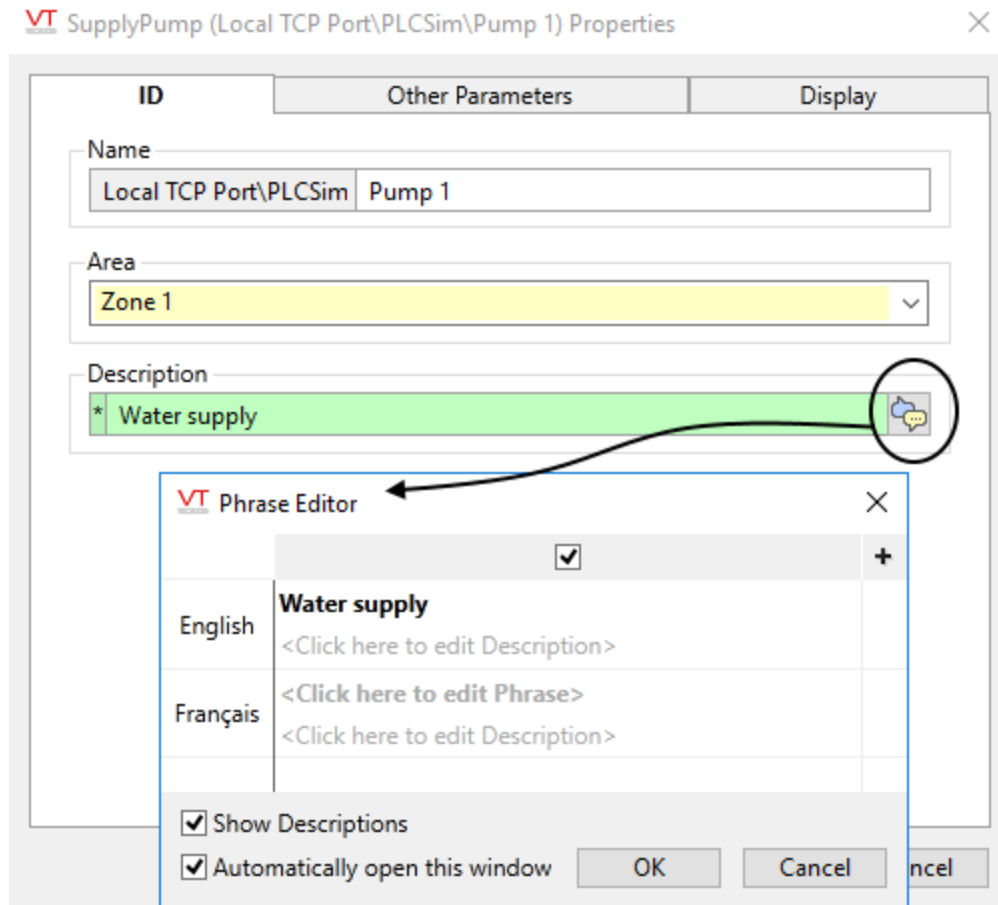


Figure C-9 Opening the Phrase Editor for the Description field of a tag.

The language at the top of the list is the default for your application, as set in the Display tab of the Edit Properties dialog. Other languages are available as configured in the Edit Properties dialog.

Click as indicated to edit descriptions and phrases. Descriptions are encouraged as they are useful to guide translators, or to help other developers select which definition of a given phrase or word applies. (See following image.)

It is often the case that one word or phrase can have several definitions (a homograph). For these, click the plus sign (circled in following image) to add new columns as needed for each definition. Each column matches a unique phrase with a unique key. For the default language (top row), every entry must be the same. Editing one will result in all changing to match. VTScada will automatically find all matching phrases in your default language. The selected column indicates which translation is used in a given instance.

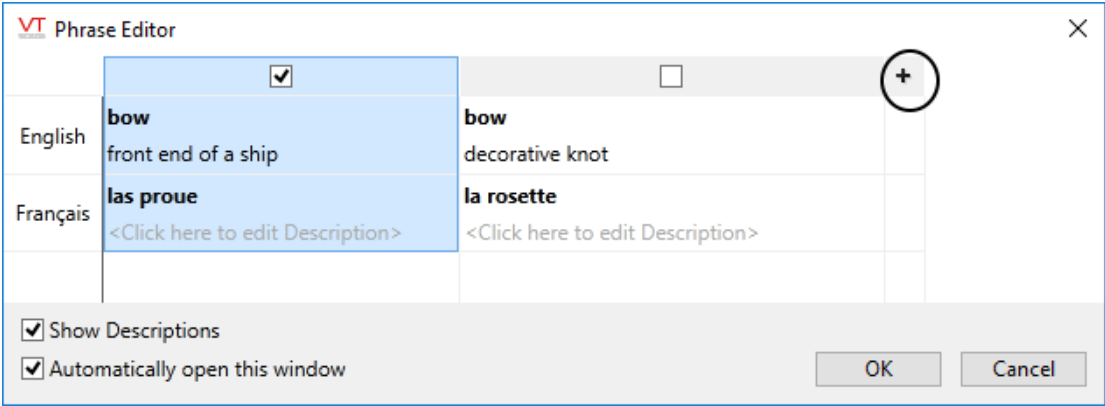


Figure C-10 Example shows a second column added for a homograph.

C Page Shuffling

You can configure your application to show a series of pages in order, automatically. (In other words, a slide show.) To do so, you must configure a set of related application properties; there is no user-interface tool for this feature. Examples are provided with the property descriptions.

Begin by creating the list of pages to be included in the automated display. (PageShuffleList, PageShuffleEnable) Set the display time for each page (PageShuffleDelay) (either one time for all, or a distinct time for each) and a duration for which shuffling will be disabled when an operator action such as a key-press or mouse movement is detected (PageShufflePauseDuration). You can also set a list of realms for which shuffling is enabled (PageShuffleRealms). All selected realms will display the same set of pages for the same duration - there are no realm-specific configuration options.

PageShuffleList

Comma-separated list of page modules to cycle between. Do not specify page titles. This list must contain the declared module name for each page, which can be obtained from AppRoot.SRC or (in most cases) from the name of the source code file holding that page.

For example, the Overview page is declared as `Overview` in AppRoot.SRC and is also the name of the source code file:

```
[ (PAGES)
  Overview Module "Pages\Overview.SRC";
]
```

Figure C-11 (Code snippet from AppRoot.SRC of most applications)

Section: System

Default: PageShuffleList =

*No restart required (Settings.Dynamic property)

Example:

PageShuffleList = Page1,Page2("TagName"),Page3

- where Page2 has its first parameter set to TagName

PageShuffleEnable

Controls whether automatic page shuffling is enabled. Only specified pages will shuffle. (Where "shuffle" means "display in sequence automatically with a specified delay")

Section: System

Default: PageShuffleEnable = 0

*No restart required (Settings.Dynamic property)

PageShuffleDelay

Controls the length of time each page is displayed when shuffling is enabled. Either a single delay to display all pages in PageShuffleList or a comma-separated list of delays to apply to each page (in which case it must contain the same number of entries as PageShuffleList).

(Where "shuffle" means "display in sequence automatically with a specified delay")

Section: System

Default: PageShuffleDelay =

*No restart required (Settings.Dynamic property)

Example:

PageShuffleDelay = 30,45,30

- length of time to show each page

PageShufflePauseDuration

Amount of time to pause page shuffling when mouse or keyboard actions are detected.

(Where "shuffle" means "display in sequence automatically with a specified delay")

Section: System

Default: PageShufflePauseDuration = 60

*No restart required (Settings.Dynamic property)

Example:

PageShufflePauseDuration: 60

- pause the shuffling for 1 minute if the mouse is moved

PageShuffleRealms

Tip: In this context, "Realms" refers to Security Realms, not thin client realms.

Either a single realm name or a comma separated list of realms for which the shuffling is enabled. To add the logged off and non-realm users to the list add a blank entry after a comma at the end of the list. The order of the realms does not relate to the order of the PageShuffleList and PageShuffleDelay settings

(Where "shuffle" means "display in sequence automatically with a specified delay")

Note: PageShuffleRealms is optional. If specified, it will work only when a user in the matching security realm is signed in.
If not specified, page shuffling applies to all realms.

Section: System

Default: PageShuffleRealms =

*No restart required (Settings.Dynamic property)

Example:

```
PageShuffleRealms = WEST
```

- only users in the WEST realm will see shuffling

Notes

E Index

.Dynamic 19

.Startup 19

A

Account 112

accounts 115

Add Parameter Expression 60

add parameter expresson 59

add start condition 59

addin 231

add-in 231

administrator options 131

AlarmDatabaseGroups 289

AlarmListFormats.XML. 291

alwaysShowShelved 295

AppLayer 320

Application privilege 112

application privileges 121

automatic log-off 132

B

bulk download 196

C

ColumnFormats 293

Configuration 18

Connector Properties 55

Control Locks 299

ControlLockIsOwnerHook 304

create new type 52

Create New Type 65

custom map icons 199

custom privileges 121

D

data logging 206
database - historian options 213
DreamReport 207

E

excel add-in 231
Export 355
Expressions 28
extra 295

F

False 39
filtering, concepts 138
filtering, tag area 151
Find 247
folders widget 108

G

grid view 221
Group 80

H

Hidden property sections 19

I

Idea Studio 6
Import File Changes 19
inheritance 159

L

layer 157
Linked Tag Properties 89
Lock Level 302
locked tag 299
Logged Off VIC Sessions 132
Logger tags 208
logging, overview 206

M

map tile source 194

master 169
Multilingual Expressions 327
multiple historians 218
multiwrite tag 76
MySQL 213

O

Ocean Data Systems 207
ODBC connection 254
ODBC server 249
ODBC server configuration 251
ODBCSingleTable 218
OEM layer 157
operator change 320
operator precedence 35
operator priority 35
operators defined 35
optimized expression 58
Optimized Expressions 60
Oracle 213
Override 59

P

page shuffle 360
PageShuffleDelay 361
PageShuffleEnable 360
PageShuffleList 360
PageShufflePauseDuration 361
PageShuffleRealms 361
parameterized page 98
password options 133
PickValid 38
plot view 221
PopupStandard 292
precedence - rules of 35

Q

Quick-Access Toolbar 8

R

read-only workstation 128
Realm Area Filtering 139
realm area filtering 139
Realm Filtering 139
realm filtering 139
realm tag filtering 139
REALMAREAS, config.ini section 143
redefine type 68
refresh 246
remove start condition 59
rename privileges 124
Report tag 226
report tools library 231
Role 112
Rule 112
rule scope 124

S

satellite maps 194
Section names 19
security rules 124
Security, Best Practices 112
SecurityManager-Admin 19
Settings.Dynamic 18
Settings.Startup 18
Setup.INI 18, 25
Shared Security 133
shuffle 360
Site Details page 182
Site List page 177
Site Map 190
Site Properties 55
Sites 177
slide show 360
slippy map 193

SlippyMapTiles 195
SQLite 213
SQLServer 213
Start Tag Expressions 61
station mask 130
subordinate 169
Syntax 30
System privilege 112
SyTech Incorporated 207

T

Tag Area Filtering 151
tag parameter expression 58
Tag Widget 80
Themes 26
tiles 194
True 39
two-factor authentication 135
Type - definition 52
type - new application 157

U

ungroup 81
Update Location 192
Update Site Location 191
update type 68
UPS 25

V

Valid 38
VAM 4
VTScada ODBC Driver 249

W

Workstation.Dynamic 18
workstation.dynamic 23
Workstation.Startup 18
workstation.startup 23

x

XLReporter 207