Library Designer and Library Object Manager User Manual

Version 5.00.00

Rockwell Automation

User Manual

Original Instructions

Important User Information

Read this document and the documents listed in the additional resources section about installation, configuration, and operation of this equipment before you install, configure, operate, or maintain this product. Users are required to familiarize themselves with installation and wiring instructions in addition to requirements of all applicable codes, laws, and standards.

Activities including installation, adjustments, putting into service, use, assembly, disassembly, and maintenance are required to be carried out by suitably trained personnel in accordance with applicable code of practice.

If this equipment is used in a manner not specified by the manufacturer, the protection provided by the equipment may be impaired.

In no event will Rockwell Automation, Inc. be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.

The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, Rockwell Automation, Inc. cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by Rockwell Automation, Inc. with respect to use of information, circuits, equipment, or software described in this manual.

Reproduction of the contents of this manual, in whole or in part, without written permission of Rockwell Automation, Inc., is prohibited.

Throughout this manual, when necessary, we use notes to make you aware of safety considerations.

WARNING: Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.

ATTENTION: Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence.

IMPORTANT: Identifies information that is critical for successful application and understanding of the product.

These labels may also be on or inside the equipment to provide specific precautions.

SHOCK HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.

BURN HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.

ARC FLASH HAZARD: Labels may be on or inside the equipment, for example, a motor control center, to alert people to potential Arc Flash. Arc Flash will cause severe injury or death. Wear proper Personal Protective Equipment (PPE). Follow ALL Regulatory requirements for safe work practices and for Personal Protective Equipment (PPE).

The following icon may appear in the text of this document.

Tip: Identifies information that is useful and can help to make a process easier to do or easier to understand.

Rockwell Automation recognizes that some of the terms that are currently used in our industry and in this publication are not in alignment with the movement toward inclusive language in technology. We are proactively collaborating with industry peers to find alternatives to such terms and making changes to our products and content. Please excuse the use of such terms in our content while we implement these changes.

Contents

Library Designer **11**

Overview 11

Decoration types 12

Navigate the interface 15

Use inclusions with library objects **21**

Use inclusions 21

View ownership for an object 22

Library Ownership window 22

About library objects **25**

Create an empty library object 26

Add selected objects 27

Copy and paste a library 29

Add a child object or dependency 29

Add a module to a library object 30

Add a drive or an IO module 30

Order programs in a library 31

Create an axis ID for a motion tag 31

Show library object dependencies 31

Hide library object dependencies 32

Include dependent objects 32

Export library object members 33

Export library object parameters 33

Import library object members 34

Import library object parameters 34

Create an empty parameters file 34

Track Logix content changes 35

Publish a library 35

Library Object parameters 36

Validate Libraries 38

Validation Error/Warning list 39

Delete a library object 40

Assign to/Create New window 40

Module Wizard overview 42

Connection Properties settings 43

Advanced connection properties ... 44
Parameters for a library object ... 50
Export Members ... 52
Import Members ... 53
Decorator panel ... 55
Decorator Panel settings ... 55
Change the columns in a tab ... 57
Use find to limit the display ... 57
Use expressions and functions ... 58
Create an expression ... 58
Expression Builder ... 59
Decorative elements tabs ... 61
Predefined functions and operators ... 61
Expression box ... 64
Configure an element substitution ... 65
References window ... 65
Substitution Builder window ... 66
Parameters tab ... 68
Add a parameter ... 69
Edit a parameter ... 69
Add an external reference ... 70
Copy a parameter ... 70
Add a group ... 70
Reassign a parameter to a group ... 71
Move a parameter within a group ... 71
Delete a group ... 72
Add a subobject ... 72
Limit the external references ... 72
View a parameter reference ... 73
Delete a parameter ... 73
Add New / Edit Parameter window ... 73
New / Edit SubObject window ... 78
Parameter Filter Builder window ... 79
Functions tab ... 80
Add a function ... 80
Add branches ... 81

Use a previously created function....82
Copy a function....82
Edit a function....82
View a function reference....83
Delete branches....83
Delete a function....83
Function Builder window....84
Substitution tab....86
Add a substitution....87
Change the order of execution....87
Copy a substitution....88
Edit a substitution....88
Delete a substitution....88
Member Selector window....89
Substitution Builder....89
Predefined tab....90
External References tab....91
Edit an external reference....92
Delete an external reference....92
Reference Builder window....92
Linked Libraries tab....93
Add a linked library....94
Add New / Edit Linked Library....95
Add parameter links....97
Parameters browser....98
Interfaces tab....98
Standardize data structures....99
Add an interface....99
Delete or edit interfaces....100
Add interface members....100
Add New/Edit Interface window....101
Edit Interface window....102
Interface Member window....103
Tags window....104
Library object properties....104
Change the name or description....108

Add a tag as a parameter....109
Add a tag as an external reference....109
Apply decoration to tag values....110
Decorate a tag alarm condition....110
Add parameter connection to a tag....111
Parameterize tag properties....111
Exclude inherited substitutions....112
Set a rule for instantiation....113
Apply a substitution to an element....114
Activate a new element....115
Extended Properties....115
Alarms....115
Edit Alarm....118
Library Object Manager....119
Library Object Manager overview....119
The interface....120
Set the default options....122
Create a list of options....122
Settings dialog box reference....122
Library Repositories overview....125
Add and mount an ACD repository....125
Unmount an ACD repository....125
Open an ACD file in Studio 5000 Logix Designer....126
Open an ACD file in Library Designer....126
Export ACD to L5X....126
Open the ACD file location....127
Remove an ACD repository....127
Add a group....127
Rename a group....127
Remove a group....127
Add a folder repository....128
Remove a folder repository....128
Add the ACM database....128
Remove the ACM database....129
Connection Properties dialog....129
Advanced Properties dialog....130

Create and copy library objects....137
Create a library object....137
Publish a library object....137
Publish multiple library objects....138
Copy multiple library objects....138
Edit library properties....139
Update library description....139
Copy a library object to a solution....140
New library settings....140
Library Import Configuration....141
Edit library properties dialog box....143
Copy to new solution dialog box....145
Library Attachments....147
Add an attachment....147
Assign an attachment reference....147
Edit an attachment....147
Change the include condition....148
Delete an attachment....148
Delete an attachment reference....148
Extract an attachment....148
Import an attachment....149
Export an attachment....149
Attachments....149
Attachment....150
Library Content....151
Review the decoration settings....151
View the Logix code....152
Expression Builder....152
Add display objects....153
FT View substitutions....154
Apply substitutions to a symbol....154
Configure the object tag and path....154
Apply positioning to a symbol....156
Delete a symbol....156
Update a symbol....157
Symbol properties....157

Symbol Builder dialog box....158
Add VBA items....159
Add a code item....159
Delete a code item....160
Update a VBA code item....160
VBA Items Properties....160
VBA Code Builder....160
VBA Item Builder....161
Add FTAE content....161
Add an alarm to a library object....161
Add a message....162
Create a message....163
Add an existing message....164
Delete message content....164
Change the tag update rate....164
Digital tab....165
Status Tags tab....169
Control Tags tab....170
Message Editor dialog box....171
Add Alarms content....172
Add a trigger to a library object....173
Add a message to a trigger....173
Delete a trigger content....173
Delete a message content....174
Triggers properties....174
Message properties....177
Message Builder....179
Add FT Historian content....180
Add a Historian tag to a library object....180
Delete a Historian tag....180
Historian tag properties....181
Add Custom Collections content....183
Add an item to Custom Collections....183
Item properties....183
Add an attribute....184
Add Attachments....184

Library Object Import Wizard **187**

Components on the wizard 187

Add library objects to an ACD file 189

Legal Notices **191**

Library Designer

Overview

Use Library Designer to assign the project, the controller, and any of the Logix objects to one or many library objects. Each library object defines a set of functions, capabilities, and connections. For example, those that support the functions of the valve, motor, and controller modules. Rather than being tied to one application, library objects can be configured to meet the needs of multiple applications. Library Designer allows the publishing of a library directly into an ACM database. Options include the ability to specify the location where the library will be published in the ACM database, and the ability to specify the status of the library, either Published or Pending.

Custom properties called "Decorations" can be added to a library object using Library Designer. Decorations include parameters, subobjects, functions, substitutions, and external references. Decoration lets the library object be configured when it is implemented in a project in Studio 5000® Application Code Manager.

Logix objects can be restricted to a single library object or assigned to multiple library objects, each with a different set of decorations. A library object can contain a single Logix object, or a Logix object can be added as an element of a more complex library object. For example, a P_Alarm Add-On Instruction can be assigned to a valve library object and can also be an element of a Motor or Pump library object.

Each ACD file can support multiple projects, controller libraries, and library objects. The ACD is not required to contain a project or controller library. While the decoration is stored as part of the ACD file, it is treated as a separate layer of information from the base controller code and does not affect code execution.

Decoration controls how the library object is instantiated, including configurations such as naming, tag values, conditional inclusion, and connections to other library objects. One or many distinct instances of a library object can be instantiated within an ACM project and each instance can be separately configured. By using Library Designer, each Logix object can be published directly to the ACM database or to a file in HSL4 format.

The Library Designer performs these tasks:

- Creates library objects.
- Specifies the Logix content that is included in the library object.
- Decorates the library object with parameters, subobjects, substitutions, functions, external references, linked libraries, and interfaces.
- Creates substitutions for text strings that extend to all elements of a library object during instantiation.
- Creates substitution overrides for specific elements.
- Creates mathematical and logical expressions using decorative elements.
- Assigns parameters to be populated by user input, calculated values (functions and expressions), or references to other elements.
- Sets conditions for inclusion of any element of a library object during instantiation.
- Makes tags and tag members accessible to Application Code Manager by adding them as parameters or external references.
- Populates tags based on parameters, functions, and expressions.
- Publishes Logix library objects directly to the Application Code Manager Database or to a file in HSL4 format.

Validation is an important part of any code-based system and the Library Designer and Application Code Manager tools can validate objects within a library structure and controller projects respectively to ensure an error free

library or ACM project. The configurable fields within the Library Designer or Application Code Manager display a validation error or warning icon when an invalid value is specified. For example, If an invalid character is included in a parameter name. The field would be highlighted with an error icon. Hovering over the error icon for error resolution information.

Rockwell Automation recognizes that some of the terms that are currently used in our industry and in this publication are not in alignment with the movement toward inclusive language in technology. We are proactively collaborating with industry peers to find alternatives to such terms and making changes to our products and content. Please excuse the use of such terms in our content while we implement these changes.

Decoration types

Decoration is added to the library object in Library Designer. The decoration can then be applied to any field of any element of the library object that accepts values from the Expression Builder or Substitution Builder. These fields display the ellipsis (...) button to the right.

Each decoration type has a distinct role in configuring a library object.

Parameters

A parameter is an argument exposed for external access and controls how the library object is instantiated.

- Parameters have simple data types: Boolean, string, integer, or real.
- Parameters are set and modified by direct user input (immediate), calculation results, or references to other parameters. Parameters added as a decorative element are only accessible through Application Code Manager, and are not accessible once the completed project is exported to code.

Parameters created in Library Designer have these functions:

- Storing information that is pertinent to the specific instance of the library object, but not functional. For example, the customer contact information for a project.
- Differentiating each instance of a library object in a project. For example, the slot location of module object.
- Configuring each instance of a library object in a project. For example, to set whether a specific instance of a valve object has permissives or interlocks.
- Populating a tag through user input or a specific external reference.

Parameters allow a single instance of base controller code to have many variations and are used in a variety of different applications. Parameters are instantiated once. A parameter must be unique within a library object. A parameter can be copied to other library objects and to library objects of different scope.

Parameters can be collected together into a subobject. A subobject is a grouped set of parameters that can be instantiated multiple times. Examples include the channels of an analog input or the contact information for a project team member. Subobjects can be auto-generated during instantiation or added manually by the user when the library object is brought into an Application Code Manager project.

Functions

A function is an argument that is not exposed to external access. The value of a function is generated by user-defined logic created in Library Designer and by conditions that apply during instantiation.

A function can be either conditional or calculated:

- A conditional function returns one of multiple possible results generated by expressions and based on IF/ELSE/ELSEIF logic. A conditional function allows for multiple branches and nesting.
- A calculated function generates a single value, based on a single expression.

Both of the functions are created using the **Expression Builder**.

Functions are copied between library objects and between library objects of different scope, as long as the decorative elements used in the expressions are common to both library objects.

Functions are saved within Library Object Manager. Saved functions are available to all projects opened in Library Designer.

Substitutions

A substitution is a user-defined rule which, during instantiation, replaces a text string in the name, description, instantiation location, or other attribute of a library object element with a parameter value, calculation result, or referenced value.

IMPORTANT: Substitutions are applied globally based on a simple search-and-replace logic.

When elements are created in the Logix Designer application, and text strings are selected for substitution in the Library Object Manager application, verify that the naming conventions and standards used are sufficient to help prevent conflicts when using substitutions.

Substitution that affects text strings in unexpected locations can make the library object function in unexpected ways or fail to validate.

Substitutions applied at one level of the library object hierarchy extend to all objects at lower levels of the hierarchy, and to all elements that are contained within the library objects. Substitutions applied at a higher level in the hierarchy take precedence over substitutions applied directly to the library object.

Substitutions that are inherited by an element from the containing library object, or from a library object higher in the library object hierarchy (base library), can be overridden at the element level using the Substitution Builder.

Substitutions can be copied and pasted from one library object to another and can be copied and pasted between library objects of different scope.

Predefined parameters

A predefined parameter is one of a set of parameters that are automatically available to all library objects created in the Library Designer. They are defined and scoped by the program. They are the same for all library objects in the hierarchy, as well as for all elements of all library objects, and are available to all substitutions, expressions, and functions. Users cannot create, modify, or delete predefined parameters.

IMPORTANT: Predefined parameters appear generically at all levels of the Project hierarchy. Verify that the use is properly scoped when applying a predefined parameter.

Predefined parameters are populated during instantiation when a library object is added to an Application Code Manager Project.

Predefined parameters cannot be copied or pasted, since they are defined by the Library Designer and are identical for all library objects.

External references

An external reference makes the value of a local tag, controller tag, or tag member within a library object accessible to parameters in other library objects. Used in conjunction with parameters that have been assigned to accept values by reference, external references provide the points of contact between library objects in an Application Code Manager project.

In an Application Code Manager project, link an external reference to a reference-type parameter. The parameter references the value of the external reference when the project is in operation. Reference-type parameters are defined so that the external references that are accessible to them are limited to those that meet certain criteria (filters).

Any tag or tag member can be added as an external reference.

Expressions

An expression is a one-line statement that generates a single calculated result. Expressions return a string, numeric, or Boolean values. Expressions generate values automatically during instantiation.

An expression can be as simple as a single decorative element token, or can involve one or more operations involving one or more decorative elements and operators.

Expressions are used in any field in the Library Designer that accepts a calculated result. These fields display the ellipsis (...) button to the right.

Expressions can incorporate any decorative element that is available to the current library object element, and a set of logical and mathematical operators.

Expressions can be entered manually or created in the **Expression Builder**. The **Expression Builder** is a responsive environment to create, test, and save expressions.

Linked libraries

A linked library is a library object containing elements that are shared with other libraries.

Linked libraries specify the relationships between library objects. The links are applied as decoration to Logix code in place of parameters.

Linked libraries can be configured to share dependencies on Logix content. For instance an AOI or UDT definition is defined in a single library object, then linked to multiple library objects.

Linked libraries assist the Application Code Manager user when configuring an object for instantiation. For instance, a regulatory control valve typically needs an analog input for instantiation.

Parameters, displayed in the bottom pane, may also be shared with linked libraries.

Parameter links are used to read or write the parameter values between a library object and linked library object. Parameter links display the direction of the flow of information.

Interfaces

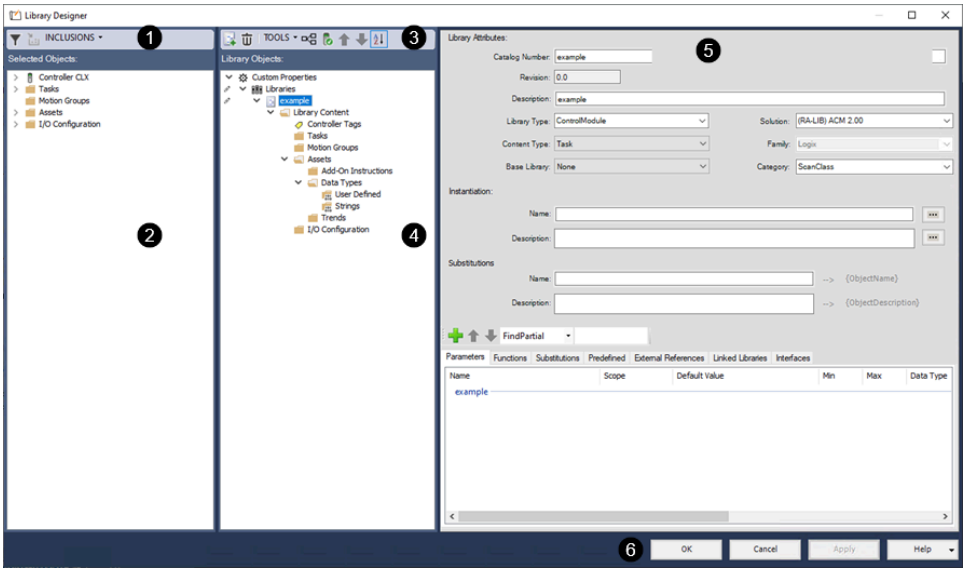
There are two types of interfaces, input and output. Output interfaces are connected to input interfaces.

Output interfaces allow variant tag member structures from different libraries to be 'mapped' to a common interface member name. Output interfaces are used with linked libraries.

Input interfaces are used within substitutions without needing to know the provider's tag structure. Input interfaces act as 'placeholder' substitutions. Input interfaces are typically used with a linked library, but can be configured as unassigned.

Navigate the interface

The Library Designer interface is organized into a three-pane display composed of two panels with toolbars and object trees and a third panel that contains object decorations (properties and parameters).



This table describes the regions and controls on the Library Designer interface.

	Interface element	Description
1	Selected Objects toolbar	The Selected Objects toolbar has three elements: • **Show/Hide Decorated Elements**: Toggles the display of items in the **Selected Objects** tree view that are associated with a library object. The default for these items is to display as blue or green. Click once to hide them and again to restore the display.

	Interface element	Description
		• **Add selected elements to a Library**: Opens the **Assign to/Create New** window for the selected element. This is one of three ways an item in the **Selected Objects** tree view may be moved to the library objects column. • **INCLUSIONS**: Select whether child elements or dependencies (elements referenced by the selected element) are included when the selected item is added to a new or existing library object. Also used to determine whether the Library Designer will restrict items to a single library object or allow them to be associated with multiple library objects. Active Container Mode is also available.
2	Selected Objects tree view	The **Selected Objects** tree view follows the structure of the **Controller Organizer** view or the **Logical Organizer** view in Studio 5000 Logix Designer. The display varies depending on the item selected when the Library Designer was opened. It includes the object or objects that were selected and all referenced elements from the Project. An object may be associated with one or many library objects, depending on the ownership settings for the project. For example, an Add-On Instruction may be associated with both a valve library object and a motor library object. Click › to the left of an item in the tree to display the elements that are contained within it. Click ⌄ to collapse the item. Objects are color-coded to indicate whether all, some, or none of the associations from the initial instance in Studio 5000 Logix Designer have been

	Interface element	Description
		replicated in the library objects added in Library Designer. • Green. Objects that are fully associated within the Library Designer. • Blue. Objects where some, but not all, of the associations have been replicated. • Black. Objects that have no associations.
3	Library Objects toolbar	The **Library Objects** toolbar has five elements: • **Create a New Library**: Opens the **Assign to/Create New** window to create an empty library object. • **Delete Selected Objects**: Deletes the selected library objects. • **TOOLS**: Selects to perform different actions to library objects. • **Show/Hide dependencies for all libraries**: Selects to display or hide the dependent objects. • **Validate Libraries**: Selects to check validation problems. This button is disabled if changes are not applied. • **Move up** and **Move down**: Orders the libraries in the **Library Objects** tree. The order is limited to the object type, for example, project, controller, or Logix library. • **Alphabetical**: Sorts the libraries alphabetically.
4	Library Objects tree view	This column displays a tree view of all library objects included in the current project or ACD file. Library objects are structured in a three-level hierarchy: *Project library object* *Controller library object* *Logix Object library objects: Tasks, Programs, Modules* A project can include project library objects, controller library objects, and multiple Logix object library objects.

	Interface element	Description
		Library objects have a three-level structure: *Catalog Number* *Library Content Folder* *Controller Tags Tasks* *Motion Groups* *Add-On Instructions Data Types* *I/O Configuration* The structure contained in each library object matches the structure of a project created in the Studio 5000 Logix Designer application, and all elements included in the library object are placed at the appropriate location in the project hierarchy. This makes it possible for the library object of a valve that is dependent on controller tags, Add-On Instructions, and data types to include all of these required elements when it is instantiated in an Application Code Manager project. The color of the text in the tree denotes whether the element is included in the library. • Black. Dependent elements included in the library. • Gray. Dependent elements that are not included in the library.
5	Decorator panel on page 55	The decorator panel becomes active when an element within a library object is selected. The decorator panel displays the fields and functions available to add, modify, or delete decoration. The display changes based on the currently selected element and its location within the library object structure. Settings that can be edited display with white backgrounds. Settings that are locked for editing appear dimmed. Settings that can accept calculated values are followed by the ellipsis (...) button. Clicking this button opens the Expression Builder on page 59.

	Interface element	Description
6	Commands	The Library Designer uses standard commands: • **OK**: Closes the Library Designer and saves all changes that have been made since the program was opened. • **Cancel**: Closes the Library Designer without saving the changes. • **Apply**: Updates all library objects in the **Selected Libraries** columns with the most recent changes applied in the decorator panel. It does not close the program. • **Help**: Displays the help menu.

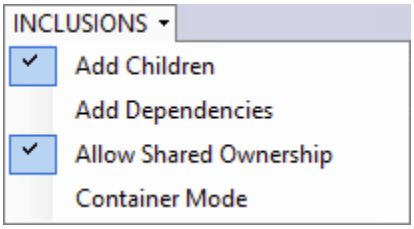
Use inclusions with library objects

Use inclusions

Use inclusion commands to add fully functional objects or specific object elements to a library object.

The **INCLUSIONS** menu on the **Selected Objects** toolbar contains menu commands that determine which elements are included when an item is added to a library object and which library objects that the item may be added to. All commands toggle on and off. Select the menu command once to activate it, and select it again to deactivate it.

All commands affect future selections, and have no effect on selections and inclusions that have already been made to existing library objects. Choices remain active until they are changed, and remain in effect when the Library Designer is closed and reopened. The menu must be reopened after each selection.



This table describes each **INCLUSIONS** menu selection.

Selection	Description
Add Children	When this command is active, all elements contained within the selected item in the **Selected Objects** tree view will be included when it is added to a library object.
Add Dependencies	When this command is active, all elements referenced by the selected item in the **Selected Objects** tree view will be included when it is added to a library object.
Allow Shared Ownership	Determines whether an item in the project hierarchy is associated with multiple library objects or restricted to a single library object. The default is shared ownership, which allows for multiple associations. The setting for this command affects all associations made until it is changed. • When the **Allow Shared Ownership** command is activated, items that were previously limited to a single library object may be added to other library objects. • When the **Allow Shared Ownership** command is deactivated, selected items may be added to one library object. If any of these items is selected and added to another library object, the Library Designer does not allow the association to take place and displays a warning.
Container Mode	Container mode allows for objects to be grouped within a container so that any changes such as additions or deletions

Selection	Description
	made within Studio 5000 to an object that forms part of a library would automatically be included to be part of the library. Container mode can be set for task, program and routine objects residing within a library. When this command is active, all program tags and routines are added to the library object automatically. If it is inactive, the user must manually add the content to the library object.

Example

When a two-state valve object (program) is added to a library object using different inclusion selections:

- With **Add Children** and **Add Dependencies** deactivated, valve program is the only element that is added to the library object.
- With **Add Children** activated, the local tags and routines contained within the valve program are also added to the library object.
- With **Add Children** and **Add Dependencies** both activated, controller tags, Add-On Instructions, and data types that are referenced by the valve program are also added to the library object.

View ownership for an object

The Library Ownership on page 22 window displays a list of all library object associations for the item highlighted in the **Selected Objects** column. The listing includes the object type and the library object or objects that the item is associated with. Items with multiple library object associations show multiple listings.

Tip: When a selected object contains child objects, all of the child objects are included in the **Library Ownership** list.

To view the ownership for a selected object

1. Right-click the item in the **Selected Objects** column and select **Library Ownership**. The **Library Ownership** window opens.
2. Click **Navigate** for a listing to open the owner library and display the decorator panel for the selected item.

Library Ownership window

The **Library Ownership** window shows the usage of an item by library objects. By default, items can be owned by multiple library objects, this is called shared ownership. If an item should be used exclusively by a single library object, disable shared ownership for the item.

This table describes the columns in the **Library Ownership** window.

Column	Description
Name	The name of the item.
Type	The type of the item.
Owner Library	The name of the library that contains the library object that references the item.

Column	Description
Navigate	Opens the owner library and displays the decorator panel for the selected item. Use this to learn the specific association for the listed item, where it is in the library object structure, and the context of its inclusion in the library object.

About library objects

A library object is the class definition of an object. A library object is instantiated. Individual library object files (HSL4) are XML-formatted and registered in the Application Code Manager database. A library object typically defines parameters, subclasses, user interface content, and portions of controller code (for example, Logix) and HMI code (for example, FactoryTalk® View Machine Edition (ME) or FactoryTalk® View Site Edition (SE)).

Library objects contain controller code, as well as decoration (custom properties). Decoration is applied to a library object in the Library Designer. Decoration can be inherited from a library object that is higher in scope. Decoration applied to a library object is inherited by, or available to, all elements that are contained within the library object. Decoration can also be applied directly to an element, overriding inheritance from the library object and from library objects of higher scope.

The project library object and controller library object are added to an ACD file using separate menu commands in the Library Designer. The project and controller library objects are placed higher in the library object hierarchy than any Logix object library objects and have predefined parameters.

IMPORTANT: It is not necessary to create the project library object and controller library object to create library objects from the Logix objects in the ACD file.

Library objects designated as modules in the Library Designer will be brought into Application Code Manager projects as hardware components rather than software components. The Library Designer features a module wizard that ensures that required parameters are included when the new module library object is created.

Parameters, substitutions, functions, external references, and expressions created in the Library Designer can be accessed, reused, and rescoped multiple times using the **Expression Builder**. Functions and expressions that are saved in the Library Designer become program resources and are available to all projects opened in the program. The Library Designer also includes a set of predefined parameters available to all projects.

The Library Designer allows for libraries to be classified by their content type. The largest granularity type would be the task type library followed by the program type library and then the routine type library. With each library type, it is possible to add content without including parent containers. As an example a task type library would include task, program and routine objects, a program type library would include program and routine type objects and a routine type library would contain routine objects. Each library type would not include parent objects. As an example a program type library would not include task type objects, programs from different tasks can be added to the same library however when this library is instantiated in Application Code Manager all the programs will end up under the same task. To keep all of the programs separated, make the library a task type.

Different actions can be performed on the **Library Objects** items based on where the item is in the library object structure.

This list describes the possible actions.

Action	Description
Delete	Deletes the selected item.
Copy	Copies the selected item.

Action	Description
Paste	Available after copying an object from the **Selected Objects** column. Pastes the copied items from the **Selected Objects** column into the selected library object.
Hide/Show dependencies	Hides or displays the dependencies for the selected object.
Export Members	Selects Functions, External References, Linked Libraries, and Interfaces to back up or transfer to another system or library.
Import Members	Selects Functions, External References, Linked Libraries, and Interfaces to import from another system, library, or backup.
Export Parameters	Exports the parameters of the selected object as an Excel file.
Import Parameters	Imports parameters from an Excel file to the selected object.
Create Empty Parameters Spreadsheet	Generates an empty parameters spreadsheet in Excel.
Publish Library	Publishes the selected library object to a specified location. If publishing the library results in duplicated task names, a warning message is displayed.
Log Viewer	Opens the **Log File Viewer** dialog box.
Track Logix Content Changes	Tracks Logix content changes in Logix Designer since the last time that Library Designer was launched.

Create an empty library object

The project library object and controller library object are added to an ACD file using separate menu commands in the Library Designer.

The project and controller library objects are placed higher in the library object hierarchy than any Logix object library objects and have predefined parameters.

To create an empty library object

1. From the **Library Objects** pane, either:
   - On the **Library Objects** toolbar, click the **Create a New Library** button.
   - Right-click the **Libraries** folder in the **Library Objects** column, then select **Add**, then **Library**.

   Tip: Project and controller library objects are placed higher in the **Library Object** hierarchy for the ACD file. Project library objects include default parameters and substitutions. Substitutions added to these library objects extend to other library objects.
   Use the **Move Up** and **Move Down** arrows to reorder the libraries in the **Library Objects** tree. The order is limited to the object type, for example, Project, Controller, or Logix Library.

   The Assign to/Create New on page 40 window opens.
2. Enter the **Catalog Number** (name) and **Description** of the library object in the respective fields.

IMPORTANT: The **Catalog Number** is the only required field and must be unique within the ACD file.

Any library attribute configured in the ACD file will become the primary value and cannot be manually edited when being published. Leave it blank if this is required.

3. Select the **Library Type** and **Content Type** from the menus.
4. Select a base library from the **Base Library** menu if the new library object should inherit substitutions from an existing library object. Select **None** if inheritance should not take place.
5. Select the **Solution**, **Family**, and **Category** from the menus.
6. In the **Instantiation** fields, enter a name and description. These are applied as the name and description for each instantiation of the library object.

Tip: To use a calculated value for this field, click the ellipsis (...) button next to the field to open the **Expression Builder**.

7. Enter initial substitutions in the **Name** and **Description** fields. These substitutions extend to all elements within the objects, and begin the decoration process. Additional substitutions can be added once the library object has been created.
8. Click **OK** to save the new library object.

Add selected objects

Items in the **Selected Objects** pane are added to the **Library Objects** pane to edit their parameters in the decorator pane.

Objects can be added using the **Add selected objects to a Library** toolbar button in the **Selected Objects** pane, by clicking and dragging the objects between the panes, or by copying the object from the selected objects pane to the library objects pane.

When clicking and dragging or copying and pasting objects between panes, placing the selected object on top of the main **Libraries** node opens the **Assign to/Create New** window. Placing the selected object on an existing library branch places the selected object in that branch in the appropriate subfolder for the object type. The new elements inherit all applicable substitutions from the library object.

IMPORTANT: Default substitutions are inherited automatically. Overrides to the default substitutions, and all other decoration, must be applied manually to new elements of an existing library object.

To add a selected object to a library object

1. Click **INCLUSIONS** and select one or more of the following options:
   - Select **Add Children** to include all elements contained within the selected item in the **Selected Objects** tree view when it is added to a library object.
   - Select **Add Dependencies** to include all elements referenced by the selected item in the **Selected Objects** tree view when it is added to a library object.

- Select **Allow Shared Ownership** to allow the selected item in the **Selected Objects**tree view to be added to more than one library object. When this option is inactive, the selected item may only be added to one library object.
- Select **Container Mode** to add all program tags and routines to the library object automatically. When this option is inactive, content must be manually added to the library object.

2. In the **Selected Objects** tree view, select the item or items to add. Hold down the **Shift** or **Ctrl** keys to select multiple items.
3. Click **Add selected objects to a Library**. The **Assign to/Create New** window opens.
4. To add the selection to an existing library object, select the library object from the **Select Library** dropdown menu. The rest of the settings in the dialog box are dimmed because they are defined by the library object.
5. To add the selection to a new library object, check the **Create New** checkbox. The remaining fields in the window, which appear dimmed by default, become active.
   a. In the **Catalog Number** box, enter the library object name, which displays when the library object is registered in the Application Code Manager application.

      IMPORTANT: The **Catalog Number** is the only required field and must be unique within the ACD file. Some fields will be automatically configured depending on the value of certain fields.
      Any library attribute configured in the ACD file will become the primary value and cannot be manually edited when being published. Leave it blank if this is required.

   b. (optional) In the **Description** box, enter the description of the library object.
   c. (optional) In the **Library Type** box, select the library type to identify the library object as a module. A module registers as hardware rather than software in the Application Code Manager application.
   d. (optional) In the **Content Type** box, select the content type: **Task**, **Program**, or **Routine**.
   e. (optional) In the **Base Library** menu, select a base library if you want the new library object to inherit substitutions from an existing library object. Select **None** if you do not want inheritance to take place.
   f. (optional) In the **Solution** box, select the solution.

      Tip: This list is empty if no solution has been set in another library in the same ACD or if the Settings file is also empty.

   g. (optional) In the **Family** box, select a family from the list.
   h. (optional) In the **Category** box, select a category from the list.
   i. (optional) Enter a name and description in the **Instantiation** fields. These will be applied as the name and description for each instantiation of the library object.

      Tip: To use an expression to generate the values, rather than entering a text string, click the ellipsis (...) button to the right of the field to open the **Expression Builder**.

   j. (optional) Enter initial substitutions in the **Name** and **Description** fields. These substitutions extend to all elements within the objects, and begin the decoration process. Additional substitutions can be added once the Library Object has been created.
6. Click **OK** to save the library object.

Copy and paste a library

In Library Designer, before you update an existing library object, you can copy and paste a whole library to create a backup or create a variant of the original library.

To copy and paste a library

1. In **Library Objects**, expand **Custom Properties**, and then expand **Libraries**.
2. Right-click the library object that you want to copy, and then select **Copy**.
3. Right-click **Libraries**, and then select **Paste**.
4. In the **Library Designer** window, select **Apply** to save changes.

Add a child object or dependency

When an object added to a library object in the Library Designer is later modified in Studio 5000 Logix Designer, the modifications must also be added to the library object in Library Designer for the library object to include the new functionality.

For example, if a valve object has new rungs added to a contained ladder logic diagram in Studio 5000 Logix Designer, the new rungs must also be added to the valve library object in Library Designer.

IMPORTANT: Items in the **Selected Objects** column are color-coded to indicate whether they are fully associated, partially associated, or not associated in Library Designer.

- Items that have been newly created in Studio 5000 Logix Designer display as black.
- Items that have newly added associations in Studio 5000 Logix Designer that have not been replicated in Library Designer display as blue.
- Items that are fully associated display as green.

To add a child object or dependency to an existing library object

1. In the **Selected Objects** column, click the items to add. Press the **Shift** or **Ctrl** keys while clicking to select multiple items.
2. After selecting objects. either:
   a. Drag the selection on top of an existing library object in the **Library Objects** column.
   b. Right-click the selection, then select **Copy**. Right-click the **Library Objects** folder, then select **Paste**.

The selection is added at the appropriate locations within the existing library objects structure. The new elements inherit all applicable substitutions from the library object.

IMPORTANT: Default substitutions are inherited automatically. Overrides to the default substitution, and all other decoration, must be applied manually to new elements of an existing library object.

Alternatively, a new library object can be created that includes the modified elements, since any Logix object can be used to create multiple library objects. The two library objects will be saved with different version numbers when they are added to a repository in the Library Object Manager application.

Add a module to a library object

Module objects are listed in the **Selected Object** tree view under the **I/O Configuration** node. These objects may be added to a library object. The Library Designer recognizes the selected object as a module and opens the **Module Wizard**.

To add a module to a library object

1. Select a module object in the **Selected Object** tree view.
2. Add the module object to a library object. The Module is added to the library object and the **Module Wizard** opens.
3. (optional) Double-click to edit the **Default Value** in the initial set of parameters. If the parameter is an **Integer** data type, the **Min** and **Max** values are also editable.

   Tip: The **Parameter Name** is not editable.

4. (optional) Select Apply to include the parameter in the library object.
5. (optional) Double-click to edit the initial set of subobjects. Parameter names appear enclosed in angle brackets.

   Tip: The **Channel Type** is not editable.

6. (optional) Select Apply to include the subobjects in the library object.

Add a drive or an IO module

The following steps describe how to create a module type drive library and an Ethernet IO module library.

To add a drive or an Ethernet IO module to a library object

1. In **Selected Objects**, drag a drive or an Ethernet IO module to the **Libraries** node in **Library Objects**.
2. In **Assign to/Create New**, configure appropriate attributes of this module including **Family** and **Category**, and then click **OK**.

   Tip: When adding an Ethernet IO module, configure the following attribute in **Module Wizard**:
   - For the **SubObjects**, select **Apply** and edit the number in the **Qty** column.

3. In the decorator panel, edit the following parameters and functions:
   - For the **ChassisName** parameter, set the **Default Value** to **{ObjectName}**.
   - For the **ChassisName** parameter, set **Visible** to **False**.
   - Remove the name format in the SubObject, and then delete the **Fn_SlotPad** function and the **Slot** parameter.
   - Add an **Address** parameter, set the **Data Type** and **Reference Type** to **Immediate**, and then set the **Default Value** to **192.168.1.1**.
4. Click the drive or Ethernet IO module, and then configure the following attributes:
   - Set the **ModuleName** to **{ObjectName}**.
   - Set the **ParentModule** to **{ParentName}**.

- Expand the **Port** node and set the **IP Address** to **{Address}**.
- (optional) Expand **Communications > Connections** and set the **RPI** to **{Fn_RPI}**.

5. Click **Apply**.
6. Right-click the library, select **Publish Library**, select a publish destination, and then click **OK**.
7. In **Library Import Configuration > 04 Library Usage Rules**, set the **Upstream Keys** to **CommsType='EtherNet'**.

IMPORTANT: This is a must step when publishing the current library.

8. Click **Apply**.

Order programs in a library

When a library is of content type **Program** in **Library Designer**, you can rearrange the order of the programs in that library as needed. Once you use the library in Application Code Manager and publish the controller to an ACD file, the order of the programs you customized will display in Studio 5000 Logix Designer.

To order programs in a library of content type Program

1. In **Library Objects**, expand the library of content type **Program**.
2. Right-click **Programs**, and then select **Order Programs**.
3. In **Order Programs**, select a program, and then click the **Up** or **Down** button to rearrange the programs.
4. Select **OK**.
5. In the **Library Designer** window, select **Apply** to save changes.

Create an axis ID for a motion tag

Library objects with motion tasks include tags with the data type "Axis_CIP_Drive". An object parameter for the AxisID of that tag is added to the library automatically. The AxisId parameter can be defined using with the parameter expression { TAGNAME_AxisId }. By default Axis IDs have a Min attribute value of 0 and a Max attribute value of 4294967296. The default value of the parameter is set to the current AxisID value of the tag.

To create an axis ID

1. In **Library Objects**, expand the library object **Library Content > Motion Groups > Motions Task** so that the **TAG_AXIS_CIP** object is visible.
2. Right-click the **TAG_AXIS_CIP** object and select **Create New AxisID**.
   An information message displays the new AxisID value. Click **OK**.

Tip: To change the AxisID value, right-click the AxisID value and select **Add\Edit Expression** to open the **Expression Builder**.

Show library object dependencies

If the **Add Dependencies** option is selected when adding the library object, all dependencies are automatically included in the object.

Use the **Show or Hide Dependencies** setting to show or hide the dependent objects.

Showing dependencies may take some time and will impact performance.

To show library object dependencies

1. In the **Library Objects** pane, either:
   - Right-click the library object, and then click **Show dependencies**.
   - Click the library object, and then on the **Library Objects** toolbar, click the **Show dependencies for all libraries** button.

   Tip: Selecting **Show dependencies** when the **Libraries** node is selected will show all dependencies of all library objects in the tree.

2. A confirmation message shows, indicating that showing all library object dependencies will affect performance. Click **OK**.

Hide library object dependencies

If the **Add Dependencies** option is selected when adding the library object, all dependencies are automatically included in the object.

Use **Show dependencies** or **Hide dependencies** to show or hide the dependent objects.

Hiding dependencies may take some time and will impact performance.

To hide library object dependencies

1. In the **Library Objects** pane, either:
   - Right-click the library object, and then click **Hide dependencies**.
   - Click the library object, and then on the **Library Objects** toolbar, click the**Hide dependencies for all libraries** button.

   Tip: Selecting **Hide dependencies** when the **Libraries** node is selected will hide the dependencies of all library objects in the tree.

2. A confirmation message shows, indicating that hiding all library object dependencies will affect performance. Click **OK**.

Include dependent objects

For object elements to be available to the application, they must be included in the library. The color of the object name in the **Library Objects** tree denotes whether the element is included in the library. The name of dependent objects included in the library is displayed using black text while the name of elements that are not included in the library are displayed using gray text. When objects that are not included in the library are selected in the **Library Objects** tree the decorator pane is blank.

To include dependent objects

1. In the **Library Objects** pane, right-click the library object and then click **Include in this library**.

Tip: Select **Shift+Click** to select multiple sequential library objects to include.
Select **Ctrl+Click** to select multiple individual objects to include.

2. The decorator pane updates to show the attributes and parameters of the object and the library.
   If multiple objects are being included, the decorator pane flashes several times and displays a **Name** value of ****Multiple Objects****.
   The object name in the **Library Objects** tree changes from gray to black indicating that it has been included.

Export library object members

Export all or selected Functions, External References, Linked Libraries and Interfaces in a library definition to transfer them to another system, library, or backup. The Export functionality is applicable to individual libraries where each library can be exported into a separate file. Library object parameters must be exported separately.

To export library object members

1. In **Library Objects**, either:
   - Right-click an individual library.
   - Right-click an individual member such as a Function.
2. In the right-click menu, select Export Members on page 52. **Export Members** appears in the center of **Library Designer**.
3. In **Export Members**, select or deselect members to export.
   By default all members and dependencies are selected. Dependencies are included or excluded based on member selections.
4. Select **Export**, then select **Save**.

Export library object parameters

Export parameters to an Microsoft® Excel® spreadsheet to transfer them to another system or to make a backup copy of library objects. All of the libraries can be exported in one process or individual libraries can be exported.

To export library object parameters

1. In the **Library Objects** pane, either:
   - Click the **Libraries** node.
   - Click an individual library.
2. On the **Library Objects** toolbar, click **Tools > Export Parameters**.
3. A green progress bar appears at the bottom of the Library Designer window.
   Once the export process has completed, an Excel spreadsheet is opened containing the exported parameters.
4. To retain the parameters file, in Excel click **File >Save**.

Import library object members

Import all or selected Functions, External References, Linked Libraries and Interfaces to transfer them from another system, library, or a backup. The Import functionality uses a previously created Export file and is applied to individual libraries. Library object parameters must be imported separately.

To import library object members

1. In **Library Objects**, right-click an individual library.
2. In the right-click menu, select **Import Members**.
3. Browse to and select an export file, then select **Open**. Import Members on page 53 appears in the center of **Library Designer**.
4. In **Import Members**, select an Action for each member or dependency.
   The default Action is determined based on the content in the destination library. Dependencies are not automatically included or excluded based on member Actions and must be selected individually.
5. Select **Import**.
6. Review the validation results, then select **Continue**.

Import library object parameters

Import parameters to an Excel spreadsheet to transfer them from another system or to restore a backup copy of library object parameters. The import can include multiple library objects or individual library objects. The library objects must exist to import parameters. This process does not create library objects.

To import library object parameters

1. In the **Library Objects** pane, either:
   - Click the **Libraries** node.
   - Click an individual library.
2. On the **Library Objects** toolbar, click **Tools > Import Parameters**.
3. Browse to and select an import file, then select **Open**.
4. A green progress bar appears at the bottom of the Library Designer window.
   Once the import process has completed, the imported parameters are displayed in the decorations panel of the library object on the **Parameters** tab.

Create an empty parameters file

Use an empty library objects parameters file to enter data for library object parameters manually. The empty parameters file can be scoped to all libraries or to a single library.

To create an empty library object parameters file

1. In the **Library Objects** pane, either:
   - Click the **Libraries** node.
   - Click an individual library.
2. On the **Library Objects** toolbar, click **Tools > Export Parameters**.

3. A green progress bar appears at the bottom of the Library Designer window.
   - If the **Libraries** node was selected, an Excel spreadsheet is opened containing tabs for each library object.
   - If an individual library was selected, and Excel spreadsheet is opened containing a single tab for the selected library object.
4. To retain the parameters file, in Excel click **File > Save**.

Track Logix content changes

Use **Library Designer** to track Logix content changes in **Logix Designer** since the last time Library Designer was launched. When Logix contents have been changed in **Logix Designer**, notifications in the form of an asterix (*) appear in **Library Designer** next to the relative library content. The notifications will be cleared automatically when **Library Designer** is closed and the tracking is updated.

Tracking changes may take some time and will impact performance.

To track Logix content changes

1. Add library objects in the **Selected Objects** column to the **Library Objects** column.
2. On the **Library Objects** toolbar, click **Tools > Track Logix content changes**. Notifications appear when Logix contents are changed in **Logix Designer**.

   Tip: If Logix contents are deleted in **Logix Designer**, the notification appears on the parent folder.

3. Click **OK**.

Publish a library

Publish a library for it to be available for use by other applications. Library objects can be published to an existing Library Object Manager repository, a folder, or an ACM database.

To publish a library

1. In the **Library Objects** pane, either:
   - Select the **Libraries** item to publish all libraries.
   - Press **Shift** and click adjacent library objects.
   - Press **Ctrl** and click multiple library objects.
   - Click a library object.
2. Right-click the highlighted selection and then click **Publish Library**.
   The **Library Destination** window opens.
3. Select one of the following destinations for the published library:
   - **LOM Repository**
     Select from the list of repositories configured in Library Object Manager.
   - **Folder**
     Type the path to a folder to publish the library or click the ellipsis (...) to browse for a folder.
   - **Current ACM Database**
     Automatically connects to the last ACM database instance used.

- **Another ACM Database**

  Click the ellipsis (...) button to open the Connection Properties on page 43 window and connect to a different ACM database.

4. The **Library Import Configuration** window opens.
   - Enter a **Revision Description** for each library object listed.
   - The library attributes configured on the library property in the ACD file will be used as the primary values.
   - Once all of the required parameters are entered, click **OK**.

   Tip: If the **OK** button is disabled, a required parameter has not been specified.

5. After the publication process completes, the **Library Publication** dialog box displays the **Library Published** message. Click **OK** to close the dialog box and return to Library Designer.

Library Object parameters

Library Designer can read data from an Excel file to populate library parameters.

The parameter fields included in the Excel file when creating an empty library object parameters file are:

Parameter	Description
:Library: Library Name	The name of the library object the parameter belongs to. Library Name must match an existing library object.
Param Name	The name of the parameter. This is a required field. It must be unique within the library scope.
Type	The data type of the parameter. • Bool • String • Int • Real • AxisID
Def Value	The default value for the parameter. The value can be entered manually or generated by an expression.
Minimum	The minimum value for the parameter. Required if the parameter data type is Int or Real. The value can be entered manually or generated by an expression.
Maximum	The maximum value for the parameter. Required if the parameter data type is Int or Real. The value can be entered manually or generated by an expression.

Parameter	Description
ControlType	Determines the mode of user interaction with the parameter when the parameter is an immediate reference type. • **TextBox**. The parameter value is directly typed into a text box. • **DropDownList**. The parameter value is selected from a list of possible values.
ControlValues	Required when **DropDownList** is specified as the **ControlType**. Defines the items included in the dropdown list from which the user can select. Enter options as a comma-separated string.
Group	The group in the Parameters tab where the parameter will appear. If the value entered matches an existing group name, the parameter will appear in this group. If the value entered does not match an existing group name, a new group will be created.
Ref Type	Determines whether the parameter is accessible to user input as an entry field, is populated automatically by a calculation, or references other parameters. The options are: • **Immediate**. Parameter is accessible to user input as an entry field. • **Calculated**. Parameter is not accessible to user input. The value is set to the name or description of the selected Parameter. • **Reference**. Parameter references an external reference. The parameter and the external reference function as a consumed and a produced tag. The parameter is linked to the external reference after instantiation. The **Filter** setting can be used to set criteria for the external reference.
ReadOnly	Set to **TRUE** or **FALSE**. Determines whether the user will be able to enter values for the parameter in the Application Code Manager application, or only read values that have been generated.
Disabled	Defines a condition that disables the parameter during instantiation. This field only affects parameters that are included in a subobject.
Auto Inc	Set to **TRUE** or **FALSE**. When set to **TRUE**, the parameter will be populated automatically and the value incremented every time a new subobject is added to an ACM Project.
Linked Param	Select the parameter that this parameter is linked to from the list. The list of available parameters is composed of parameters within the library object where the **Ref Type** is defined as **Reference**.

Parameter	Description
	When the **Ref Type** is set to **Calculated**, this parameter is required.
Linked Property	Select the property within the parameter that contains the value to use. When the **Ref Type** is set to **Calculated**, this parameter is required.
CLX Dependency	Set to **TRUE** or **FALSE**. Determines whether the parameter is dependent upon a controller for it value.
Padding	Set to **TRUE** or **FALSE**. When **TRUE**, single-digit integer values will be padded with zeros.
Filter	When **Reference** has been specified as the **Ref Type** for the parameter, use this field to create a filter expression to limit the external references.
UseCustom	When **Calculated** has been specified as the **Ref Type** for the parameter, use this field to specify a condition which, if true at instantiation, overrides the default and opens the field to user input.
Append	A text string that will be added to the end of the value of the parameter.
Visible	Defines a condition to control whether the parameter is displayed when the library object is added to an ACM Project.
Help	Help text to explain the function of the parameter and the result when specific values are entered. Appears at the bottom of the parameter window when the parameter is selected in the Application Code Manager application.

Validate Libraries

Use **Validate Libraries** to perform a complete validation operation on all library objects in the Library Designer to obtain a list of validation issues. Validation in the Library Designer checks these items:

- Paths used in substitutions are valid.
- References to tags and members are valid.
- Parameters referenced by an interface member exist.

Validation issues encountered are grouped into a list with errors and warnings severity categories. Resolve the issues to ensure that the library is error free when publishing or generating controller code.

To validate libraries

1. In the **Library Objects** pane, on the **Library Objects** toolbar, click the **Validate Libraries** button.
2. A validation progress message is displayed, informing you that validation is occurring.
   Validation may take some time. To stop the validation process, click the **Cancel** button on the progress message.

3. When the validation process is complete, either:
   a. No errors were found. The **No validation problems found** message is displayed. Click **OK** to close the validation message.
   b. Errors were found. A list of issues is displayed in the **Validations Errors/Warnings** dialog box. Review the list of issues.
      - Warnings that have the **Action** field enabled can be set to **Resolve** so that they can be automatically resolved by clicking the **Apply Actions** button in the upper right corner of the **Validation Errors/Warnings** dialog box.
      - Errors must be fixed manually. Rerun the validation process after fixing the error conditions.

Validation Error/Warning list

The validation list contains of a number of columns that provide additional details and possible action selection options for each detected validation issue.

This table describes the information provided in the validation list:

Column	Description
Severity	Identifies the type of issue, either **Warning** or **Error**.
Class	The type of validation issue encountered: • ResolvableReference: A value contains an unresolved reference that cannot be resolved. • ReusedName: The same name exists for the following possible objects: functions, parameters, sub objects, external references, interfaces and library links. • TagNotFound: The referenced element cannot be found. • UnbalancedBraces: Missing brackets in the expression. • UnresolvableReference: Invalid references to external references, invalid field for linked parameter, interface member is not valid, interface reference does not match interface link, parameter does not exist in the linked parameter referenced sub object, referenced interface does not match interface link, referenced object does not contain a matching library link, value does contain a resolvable reference, value does not contain a valid object id. • InvalidCondition: The inclusion condition is invalid. • Incompatible object class: Object class is incompatible with parent.
Library	Name of the library where the validation problem was detected.
Element	The specific element within the library where the problem was detected.
Item	The module within the validation engines that triggered the warning or error message.
Description	A description of the validation issue.

Column	Description
Action	Use this field to resolve warnings automatically. If the validation issue is resolvable, then this field can be set to either Resolve or Ignore. If the issue cannot be resolved automatically, the field will be disabled and will display N/A. Issues with the severity type Error cannot be resolved automatically.
Action Detail	A description of what action will be taken when the action is set to **Resolve** and the **Apply Actions** button is clicked.

Delete a library object

Delete an item or items in the **Library Objects** pane that are no longer needed. Press the Shift key while clicking to select multiple sequential items. Press the Ctrl key while clicking to select multiple individual items.

To delete a library object

1. Select the items to delete, either:
   - Right-click the selected items and then click **Delete**.
   - On the **Library Objects** toolbar, click **Delete**.
2. In the **Warning** dialog, click **Yes** to delete the selected items.

Assign to/Create New window

How do I open the Assign to/Create New window?

- On the **Selected Objects** toolbar, click **Add selected object to a Library**.
- On the **Library Objects** toolbar, click **Create a New Library**.
- Drag or paste items from **Selected Objects** to the top **Libraries** node in **Library Objects**.

The **Assign to/Create New** window is used to populate library objects. It can be used to bring items from the ACD file into existing library objects that can be edited using Library Designer or to create library objects that contain the item from the ACD file. Additionally, when opened from the **Library Object** pane it can be used to create new empty, library objects.

When an existing library is being used, only **Select Library** is active; all **Library Attributes** settings in this window are read-only.

When creating a library object, all of the settings except for **Revision** become active.

This window opens when you click the **Add selected object to a Library** button.

The **Assign to/Create New** window has these settings.

Name	Type	Description
Select Library	Dropdown menu	Choose which of the existing library objects an item should be assigned to. Visible when using items originating from an ACD file or after clicking **Add selected**

Name	Type	Description
		object to a Library on the **Select Objects** toolbar.
Create New	Checkbox	Select to create a library object and assign the item to it. Visible when using items originating from an ACD file or after clicking the **Add selected object to a Library** button on the **Select Objects** toolbar.
Catalog Number	Text entry	The library object name, which appears when the library object is registered in the Application Code Manager application.
Revision	Auto	Dimmed field, displays the system-generated version of the library object.
Description	Text entry	The description of the library object.
Library Type	Dropdown menu	Selects the library type and identifies the library object as a module. A module registers as hardware rather than software in the Application Code Manager application.
Content Type	Dropdown menu	Selects the object type, **Task**, **Program**, or **Routine**.
Base Library	Dropdown menu	Selects an existing library object in the library object hierarchy as a base library. When a base library is selected, the current library object inherits substitutions from the base library, and the **Expression Builder** accesses all custom properties of the base library when used in the current library object. When **None** is selected, the current library object does not inherit custom properties.
Solution	Dropdown menu	This list is empty if no solution has been set in another library in the same ACD or if the **Settings** file is also empty.
Family	Dropdown menu	Selects a family from the list.
Category	Dropdown menu	Selects a category from the list.
Instantiation: Name	Text entry	Sets the name for each instantiation of the library object. To use a calculated value for this field, click the ellipsis (...) button next to the field to open the **Expression Builder**.

Name	Type	Description
Instantiation: Description	Text entry	Sets the default description for each instantiation of the library object. To use a calculated value for this field, click the ellipsis (...) button next to the field to open the **Expression Builder**.
Substitutions: Name	Text entry	Sets the default substitution for a text string in the library object name when the library object is instantiated. This string can be the complete name or a substring within the name. The substitution extends to the names of all elements within the library object. By default, the Library Object Manager application assigns the predefined parameter {ObjectName} as the substitution for the entered string.
Substitutions: Description	Text entry	Sets the default substitution for a text string in the library object description when the library object is instantiated. This string can be the complete description or a substring within the description. The substitution extends to the descriptions of all elements within the library object. By default, the Library Object Manager application assigns the predefined parameter {ObjectDescription} as the substitution for the entered string.

Module Wizard overview

How do I open the Module Wizard?

- Add the module object to a library object of module type. The **Module Wizard** opens.
- Assign to or create a library object of module type. The **Assign to/Create New** window opens, and then select **OK**. The **Module Wizard** opens.
- Right-click a Module that has already been added to a library and select Decorate Library.

The **Module Wizard** generates default parameters and subobjects for the library object based on the controller code for the module. This decoration conforms to the standards of the Studio 5000 Logix Designer design process. Accept the defaults or edit them in the wizard.

- The Library Designer automatically assigns the **Module Type**. This field cannot be edited.
- The Library Designer displays the same structure of the **I/O Configuration** tree as it in Studio 5000 Logix Designer, which makes decorating non-rack modules or devices easier.

- The Library Designer generates an initial set of parameters. The **Parameter Name** field cannot be edited. The **Default Value** field is editable by double-clicking in the field. For parameters with a data type of **Integer**, the **Min** and **Max** are editable by double-clicking in the field. The **Apply** checkbox determines whether the parameter is included in the library object. All boxes are checked by default.
- The Library Designer generates an initial set of subobjects. For example, an analog input module opens in the wizard with Analog Input (AI) and Analog Output (AO) subobjects. The **Channel Type** field is not editable. Other fields are editable; double-click in the field to open it for editing. Parameter names appear enclosed in angle brackets. The **Apply** checkbox determines whether the SubObject will be included in the library object. All boxes are checked by default.

Connection Properties settings

The **Connection Properties** dialog box defines the settings for how Library Designer connects with the ACM database. Use this dialog box to set the data source, the type of user authentication, to specify a different database file, or to modify advanced security settings.

This table describes the settings in the **Connection Properties** dialog box.

Setting	Description
Data source	Database type. Always select Microsoft SQL Server (SqlClient).
Server name	Selects a computer name and SQL Server instance from the list or type a computer name and SQL Server instance in this format: <Computer Name>\<SQL Server Instance>.
Log on to the server	
Use Windows Authentication	Allows SQL Server sign in using Windows authentication. When selected, the logged on Windows user account credential will be sent to SQL Server to authenticate the session.
Use SQL Server Authentication	Allows SQL Server sign in using SQL Server authentication. When selected, provide the username and password for the SQL Server authentication. • **User name**. The SQL Server username, "sa" by default. • **Password**. The SQL Server password associated with the username specified, "ApplicationAdm1n" by default. • **Save my password**. When selected, saves the SQL Server password specified so that it can be used in the next session.
Connect to a database	
Select or enter a database name	Select a database name from the list or enter a database name.
Attach a database file	When selected, specify the identifiers for the database file. SQL Server database files have two names, the operating system file name used to locate the database in the file system and the

Setting	Description
	logical file name used to identify the database within SQL Server transactions. • Type a database file name or use the **Browse** button to use the **Open** dialog to locate the database file by clicking through the file system. • In **Logical name**, type the logical name of the database.
Advanced	Select to configure advanced database properties.
Test Connection	Tests the connection to the database. If a "Test connection succeeded." message is not returned, check that the following settings are correct: • Computer name • SQL Server authentication • Network access (remote SQL Server)

Advanced connection properties

The **Advanced Properties** dialog box provides a means of changing how the connection between the Library Designer and the ACM database passes information.

This table describes the settings in the **Advanced Properties** dialog box. The dialog box is divided into functional areas.

Tip: Applying the recommended settings will improve ACM performance especially for network connections.

Area	Setting	Possible Values	Description
Advanced	MultipleActiveResultSets	True False (default)	When True, multiple result sets can be returned and read from one connection.
	Network Library	**blank (required if local)** Named Pipes (DBNMPNTW) Shared Memory (DBMSLPCN) **TCP/IP (DBMSSOCN) (recommended if networked)** VIA (DBMSGNET)	The network library used to establish a connection to an instance of SQL Server. **Do not use when the SQL Server is resident on the local host computer, value should be blank.**
	Packet Size	**8000 (recommended)**	Size in bytes of the network packets used to communicate with an instance of SQL Server. PacketSize may be a value in the range of 512 bytes and 32767 bytes.

Area	Setting	Possible Values	Description
	Transaction Binding	Implicit Unbind (default) Explicit Unbind	Indicates the binding behavior of connection to the **System.Transactions** namespace. When set to **Implicit Unbind**, the connection detaches from the transaction when it ends, switching back to autocommit mode. When set to **Explicit Unbind** the connection remains attached to the transaction until the transaction is closed. The connection will fail if the associated transaction is not active or does not match the current transaction.
	Type System Version	Latest (default) SQL Server 2012 SQL Server 2008 SQL Server 2005	Indicates which server type system the provider will expose through the DataReader.
Connection Resiliency	ConnectRetryCount	**2 (recommended)**	Number of attempts to restore a connection. The number of reconnections attempted after identifying that there was a connection failure. This must be an integer between 0 and 255. Set to 0 to disable reconnecting on idle connection failures.
	ConnectRetryInterval	**5 (recommended)**	Delay between attempts to restore connection. The amount of time (in seconds) between each reconnection attempt after identifying that there was a connection failure. This must be an integer between 1 and 60.
Context	Application Name	.Net SqlClient Data Provider	The name of the application.
	Workstation ID		The name of the workstation connecting to SQL Server.

Area	Setting	Possible Values	Description
Initialization	ApplicationIntent	ReadWrite (default) ReadOnly	Declares the application workload type when connecting to a server.
	Asynchronous Processing	True False (default)	When true, enables usage of the Asynchronous functionality in the .NET Framework Data Provider.
	Connect Timeout	**30 (recommended)**	The length of time in seconds to wait for a connection to the server before ending the attempt and generating an error. A value of 0 indicates no limit, and should be avoided in a ConnectionString because an attempt to connect waits indefinitely.
	Current Language		The SQL Server Language record name.
Pooling	Enlist	True (default) False	When **True**, sessions in a Component Services environment should automatically be enlisted in a global transaction where required.
	Load Balance Timeout	30 (default)	The minimum amount of time (in seconds) for this connection to live in the pool before being destroyed. When a connection is returned to the pool, its creation time is compared with the current time, and the connection is destroyed if that time span (in seconds) exceeds the value specified by **Load Balance Timeout**. A value of zero (0) causes pooled connections to have the maximum connection timeout.

Area	Setting	Possible Values	Description
	Max Pool Size	**1000 (recommended)**	The maximum number of connections allowed in the pool. Valid values are greater than or equal to 1. Values that are less than **Min Pool Size** generate an error.
	Min Pool Size	1 (default)	The minimum number of connections allowed in the pool. Valid values are greater than or equal to 0. Zero (0) in this field means that no minimum connections are initially opened. Values that are greater than **Max Pool Size** generate an error.
	PoolBlockingPeriod	Auto AlwaysBlock **NeverBlock (recommended)**	Defines the blocking period behavior for a connection pool. When connection pooling is enabled and a timeout error or other sign-in error occurs, an exception will be thrown and subsequent connection attempts will fail for the next five seconds, the "blocking period". If the application attempts to connect within the blocking period, the first exception will be thrown again. Subsequent failures after a blocking period ends will result in a new blocking period that is twice as long as the previous blocking period, up to a maximum of one minute.
	Pooling	**True (recommended)** False	When **True**, the connection object is drawn from the appropriate pool, or if necessary, is created and added to the appropriate pool. Any newly created connection is added to the pool when

Area	Setting	Possible Values	Description
			closed by the application. In the next attempt to open the same connection, that connection will be drawn from the pool. Connections are considered the same if they have the same connection string. Different connections have different connection strings.
Replication	Replication	False (default) True	Used by SQL Server in replication. Set to **True** if replication is supported using the connection.
Security	Authentication	NotSpecified (default) SqlPassword ActiveDirectoryPassword ActiveDirectoryIntegrated	Specifies the method of authenticating with SQL Server.
	Column Encryption Setting	Enabled Disabled (default)	Default column encryption setting for all the commands on the connection.
	Encrypt	True False (default)	When True, SQL Server uses SSL encryption for all data sent between the client and server if the server has a certificate installed.
	Integrated Security	True False (default)	Whether the connection is to be a secure connection or not. When **False**, User ID and Password are specified in the connection. When **True**, the current Windows account credentials are used for authentication.
	Password	*******	Indicates the password to be used when connecting to the data source.
	Persist Security Info	True False (default)	When **False**, security-sensitive information, such as the password, is not returned as part of the connection if the

Area	Setting	Possible Values	Description
			connection is open or has ever been in an open state.
	TrustServerCertificate	**True (recommended)** False	When **True** (and **Encrypt** is set to **True**), SQL Server uses SSL encryption for all data sent between the client and server without validating the server certificate. If **TrustServerCertificate** is set to **True** and **Encrypt** is set to **False**, the channel is not encrypted.
	User ID	sa	Indicates the user ID to be used when connecting to the data source.
Source	AttachDbFilename		The name of the primary file, including the full path name, of an attachable database.
	Context Connection	True False (default)	When **True**, indicates that the connection should be from the SQL Server context. Available only when running in the SQL Server process.
	Data Source	localhost\SQLACM (default)	Indicates the name of the data source to connect to.
	Failover Partner		The name or network address of the instance of SQL Server that acts as a failover partner.
	Initial Catalog	*Initial Database Name*	The name of the initial catalog or database in the data source.
	MultiSubnetFailover	True False (default)	If your application is connecting to a high availability, disaster recovery (AlwaysOn) availability group (AG) on different subnets, setting this value to **True** configures SqlConnection to provide faster detection of and connection to the (currently) active server.
	TransparentNetworkIPResolution	True (default) False	If your application connects to different networks, setting

Area	Setting	Possible Values	Description
			this value to **True** configures SqlConnection to provide transparent connection resolution to the currently active server, independently of the network IP topology. When set to **True**, the application is required to retrieve all IP addresses for a particular DNS entry and attempt to connect with the first one in the list. If the connection is not established within 0.5 seconds, the application will try to connect to all other IP addresses in parallel. When the first IP address answers, the application will establish the connection with the respondent IP address. If **MultiSubnetFailover** is set to **True**, this setting is ignored. If **Failover Partner** is specified, this setting is ignored. The default setting is **False** if **Authentication** is set to either **Active Directory Password** or **Active Directory Integrated**, otherwise the default setting is **True**.
	User Instance	True False (default)	Indicates whether the connection will be redirected to connect to an instance of SQL Server running under the user's account.

Parameters for a library object

Use the **Library Import Configuration** window to define the parameter settings for a library object that is being published from Library Designer into a library repository.

The **Library Import Configuration** window has these settings.

Name	Type	Description
01 Library Status		
Status	Dropdown menu	If the library is saved as Pending, this revision will be replaced next time changes are saved. If the library is saved as Published, a new revision will be created next time changes are saved.
02 Revision History		
Revision Description	Text entry	The revision description of the library object. This is an optional setting.
Reset History	Dropdown menu	Displays different options to keep the current revision history, remove previous revision histories, or clear all revision histories. This is a required setting.
03 Library Details		
CatalogNumber	Text entry	The library object name, which appears together with the revision number in the library object listing when the library object is registered in ACM. This is entered manually. This is a required setting.
Description	Text entry	The description of the library object. This is an optional setting.
Family	Dropdown menu	The Family of the library object. This is a required setting.
Solution	Dropdown menu	The Solution for the library object. This is a required setting.
Library Type	Dropdown menu	The Library Type for the library object. This is a required setting.
Category	Dropdown menu	The Category for the library object. This is a required setting.
Content Type	Text	Displays the type of the object, Task, Program, or Routine.
Owner	Text entry	The user or entity that originally published the library object. This is a required setting.
Major Revision	Text entry (integer)	The major revision number for the library object. For a new library object, this defaults to 1. If you are updating an existing library a new value will be set based on existing library objects with the same

Name	Type	Description
		CatalogNumber, unless the library status is Pending.
Minor Revision	Text entry (integer)	The minor revision number for the library object. For a new library object, this defaults to 0. If you are updating an existing library a new value will be set based on existing library objects with the same CatalogNumber, unless the library status is Pending.
04 Library Usage Rules		
Upstream Keys	Text entry	For module library objects: a rule that limits the upstream hardware components that will be made accessible to the library object when it is added to an ACM Project. The rule is entered manually as a logical expression.
Downstream Keys	Text entry	For module library objects: a rule that limits the downstream hardware components that will be made accessible to the library object when it is added to an ACM Project. The rule is entered manually as a logical expression.

Export Members

How do I open Export Members?

1. In **Library Objects**, either:
   - Right-click an individual library.
   - Right-click an individual member such as a Function.
2. In the right-click menu, select **Export Members**. **Export Members** appears in the center of **Library Designer**.

Use **Export Members** to select Functions, External References, Linked Libraries, and Interfaces to back up or transfer to another system or library.

Column	Description
Name	Displays the name of the item, grouped into Functions, External References, Interfaces, or Linked Libraries.
Export	Selects a member and its dependencies for export.
Type	Displays the type of the item.
Source Library	Displays the name of the source library or library scope.

Column	Description
Details	Opens a read-only decorator panel to display specific associations for the item.
Depends On	Lists associated dependencies of the item.
Used By	Lists associated members of the item.
Description	Displays the description of the item.

Import Members

How do I open Import Members?

1. In **Library Objects**, right-click an individual library.
2. In the right-click menu, select **Import Members**.

Use **Import Members** to select Functions, External References, Linked Libraries, and Interfaces to import from another system, library, or backup.

Column	Description
Name	Displays the name of the item, grouped into Functions, External References, Interfaces, or Linked Libraries.
Action	Selects an action to perform during import. Available actions vary based on the destination library: • **Create**: Import the item and its dependencies. Available when the member does not exist in the destination library. • **Discard**: Do not import the item or its dependencies. Available when the member does not exist in the destination library. • **Overwrite**: Import and replace items in the destination library with items from the import file. Existing item configurations are overwritten. Available when the member exists in the destination library. • **Use Existing**: Do not import the item or its dependencies. Available when the member exists in the destination library. The default Action is determined based on the content in the destination library.
Type	Displays the type of the item.
Source Library	Displays the name of the source library or library scope.
Details	Opens a read-only editor panel to display specific associations for the item.
Depends On	Lists associated dependencies of the item.
Used By	Lists associated members of the item.
Description	Displays the description of the item.

Decorator panel

Any element within a library object that accepts decoration opens the decorator panel when it is selected in the **Library Objects** column. The decorator panel displays the fields for the element where decoration can be applied. The display changes based on the decoration available for the selected element.

Fields in the decorator panel that can be edited display with white backgrounds. Fields that are locked for editing appear dimmed. Fields that can accept calculated values show the ellipsis (...) button to the right that opens the **Expression Builder** windows.

In a typical application, substitutions added to the library object extend to names and descriptions for all elements contained within the library object. This allows for consistent identification of all elements within each instance of the library object that is added to an ACM Project.

Parameters, functions, and expressions can be applied as conditions for instantiation, to populate tags, to populate tag extended parameters, and to configure task, program, or routine names.

Substitutions added to the library object, as well as substitutions added to the library objects higher in the library object hierarchy, are applied automatically. Substitutions can be overridden using the **Substitutions Builder**.

The decorator panel becomes inactive when organizational folders in a library object, or items in the **Selected Objects** column, are selected.

When a module is selected, the decorator panel displays the configurable parameters for the selected module. Various decoration operations are possible and expressions can be added. The decorator will only be enabled for supported modules. Different hardware modules with different catalog numbers can be added to the same library.

Decorator Panel settings

When a library object is selected, the top half of the **Decorator Panel** has these settings.

Name	Field Type	Description
Catalog Number	Text entry	The library object name, which appears when the library object is registered in the Application Code Manager application.
Revision	Auto	Dimmed field, displays the system-generated version of the library object.
Description	Text entry	The description of the library object.
Library Type	Dropdown menu	Selects the library type and identifies the library object as a module. A module registers as hardware rather than software in the Application Code Manager application.
Content Type	Dropdown menu	Selects the object type, **Task**, **Program**, or **Routine**.

Name	Field Type	Description
Base Library	Dropdown menu	Selects an existing library object in the library object hierarchy as a base library. When a base library is selected, the current library object inherits substitutions from the base library. When using the **Expression Builder**, all custom properties of the base library are available to the expression used in the current library object. When "NONE" is selected, the current library object does not inherit Custom Properties.
Solution	Dropdown menu	This list will be empty if no solution has been set in another library in the same ACD or if the Settings file is also empty.
Family	Dropdown menu	Selects a family from the list.
Category	Dropdown menu	Selects a category from the list.
Instantiation: Name	Text entry	Sets the name for each instantiation of the library object. The value can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Instantiation: Description	Text entry	Sets the default description for each instantiation of the library object. The value can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Substitutions: Name	Text entry	Sets the default substitution for a text string in the library object name when the library object is instantiated. This string can be the complete name or a substring within the name. The substitution extends to the names of all elements within the library object. By default, the library object Manager application assigns the Predefined Parameter {ObjectName} as the Substitution for the entered string.
Substitutions: Description	Text entry	Sets the default substitution for a text string in the library object description when the library object is instantiated. This string can be the complete

Name	Field Type	Description
		description or a substring within the description. The substitution extends to the descriptions of all elements within the library object. By default, the library object Manager application assigns the Predefined Parameter {ObjectDescription} as the Substitution for the entered string.

The values for these settings were entered when the library object was created and are displayed by default. Changing some of the attribute settings may automatically set other attributes or limit what is available for selection to ensure consistency.

The bottom half of the decorator panel contains tabs that display the decoration that applies to the library object.

Decoration (custom properties) can only be added at the library object level. Decoration added to a library object extends to all elements contained by the library object.

- Substitutions added to project or controller library objects (base libraries) extend automatically to all elements contained within these library objects and to all library items that are lower in the hierarchy.
- Parameters and functions added to project or controller library objects are available to expressions created in the **Expression Builder** at all levels of these objects and to all library objects that are lower in the hierarchy.
- Decoration inherited from a library object or base library can be overridden at the element level.
- Predefined parameters are available to the **Expression Builder** at all levels of all library objects. The user cannot create, modify, or deleted the parameters.

Change the columns in a tab

Follow these steps to change the columns displayed in the decorator panel tabs.

To change the columns displayed in a tab

1. Right-click the column heading.
2. Select **Columns** to display the submenu listing all columns.
3. Mouse over an inactive column to add it to the display. Mouse over an active column to remove it from the display.

Tip: Column widths can be resized by hovering over the right side of the column until the pointer changes to a double arrow, then clicking the column edge and dragging it to the desired size.

Use find to limit the display

Use find to limit the information displayed on the properties tabs shown in the decorator panel. All columns currently being displayed are searched.

There are two find methods:

- **FindPartial**: Searches for the text string in any part of an entry.
- **FindPrefix**: Searches for the text string at the beginning of an entry.

To use find to limit the display based on a text string

1. In the decorator panel, use the dropdown menu to select the find method from the toolbar.
2. In the text box to the right, enter the text string to search for.
   The tab display reacts dynamically to the text string entered, returning a shorter list of matching information as the text string grows.

Tip: Delete the text in the entry box to return the display to the default information.

Use expressions and functions

Expressions can be used for any parameter that accepts a calculated result.

Expressions can be entered manually or created in the Expression Builder on page 59.

IMPORTANT: When an expression is entered manually, the decorative element and function tokens must be entered in the correct format:

- The token must be enclosed in curly brackets {}.
- The name must contain only alphanumeric characters and underscores.
- The name must match the capitalization of the original.
- Tokens that return a string value must be enclosed in single quotes (apostrophes).

Text entered into an expression that is to be evaluated as a string value must also be enclosed in single quotes.

Expressions can contain functions. The value of a function is generated by user-defined logic created in the Library Designer and by conditions that apply during instantiation. Functions are copied between library objects and between library objects of different scope, as long as the decorative elements used in the expressions are common to both library objects. Predefined functions are available in the **Expression Builder** or use the Function Builder on page 84 to create functions as needed.

Expressions that are created using the **Expression Builder** are added to the **Saved** portion of the **Expressions** tab and becomes available to any project opened in the Library Designer.

Create an expression

An expression is a one-line statement that generates a single calculated result. Use the Expression Builder in the Library Designer to create expressions for properties of a library object that are generated when the object is instantiated. The functions and operators available in the Expression Builder on page 59 provide the ability to test and manipulate the values returned by the decorative elements.

Expressions can be used with general object attributes, such as the object name and description, and for object and subobject parameter settings. Expressions can contain other expressions.

Expressions return a string, numeric, or Boolean values.

To create an expression

1. In the decorator panel, locate the setting that will be calculated using the expression.
2. Click the ellipsis (...) button next to the setting. The **Expression Builder** window opens.
   - To add a parameter, function, expression, predefined element to the expression, double-click the listing in the tab.
   - To use a predefined function or operator in an expression, double-click the item. Hover over the item to get a description of the function usage.

     As items are added, the Expression box displays the expression being created. To edit an expression manually, click within the Expression on page 64 box and type.

     Continue adding items to create the desired expression.
3. In **Result Type**, select **String**, **Boolean**, or **Numeric**.

   IMPORTANT: If the current field where the expression will be applied has a predefined data type, the **Result Type** field will be set to match this data type and is dimmed.
4. Click **Validate**.
   - If the validation passes, the box under **Result Type** displays "Passed" with a green background.
   - If the validation fails, the box under **Result Type** displays an error message describing the syntax error or data type error with a red background.

     Continue editing the expression until it passes validation.
5. Click **Test** to test the expression. The **Expression Test** window opens and displays the current result of the expression.

   Tip: The expression must pass validation before it can be tested.
6. In the **Expression Builder**, click **Save**. The **SaveExpression** dialog box opens. In **Expression Name**, type a name for the expression.

   The expression is added to the **Saved** portion of the **Expressions** tab.

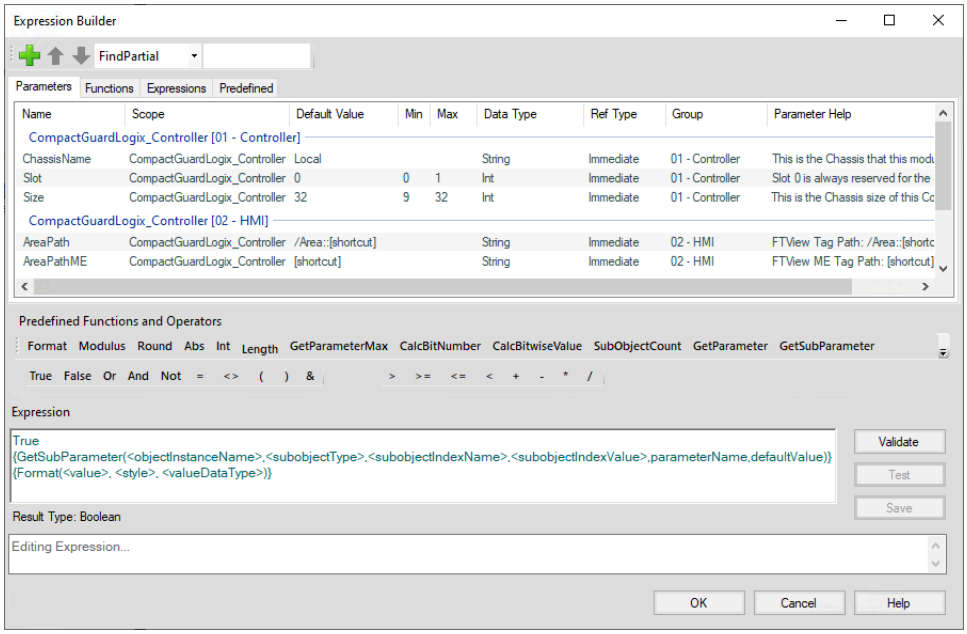
   Tip: Clicking **Save** does not apply the expression to the current field.
7. Click **OK** to close the **Expression Builder** window and apply the expression to the current field.

Expression Builder

How do I open Expression Builder?

- In Library Designer or Library Object Manager, select the ellipsis button (...) next to parameters, values, and other objects that can be modified by or decorated with an expression.

The **Expression Builder** is an environment to create, test, and save expressions.



The **Expression Builder** window includes a command toolbar, decorative element tabs, and functional areas.

Item	Description
	The **Add New** button is only active on the **Parameter** tab and opens the **Add New Parameter** window.
	The **Move Up/Move Down** buttons are not active in the **Expression Builder** window.
	This dropdown menu can initiate a search for parameters that match a text string entered in the adjacent field. There are two different types of finds that can be performed, use the dropdown menu to select the one to use: • **FindPartial**: Searches for the text string in any part of an entry. • **FindPrefix**: Searches for the text string at the beginning of an entry. Find can be used on all of the tabs.
Decorative element tabs on page 61	The decorative elements available to the current expression. These include all elements added to the current library object and all elements added to the library objects that are higher in scope. These elements include: • Parameters • Functions • Expressions • Predefined Functions and expressions can be saved and used in future projects.

Item	Description
Predefined Functions and Operators on page 61	A collection of logical and mathematical operators that can be used to manipulate the values generated by the decorative elements.
Expression on page 64 box and Result Type	The expression appears here as elements are added to it. There are also settings and buttons to set the data type of the expression result, and to validate, test, and save the expression. The **Result Type** displays a color-coded response when the expression is validated and tested.

Decorative elements tabs

The decorative elements tabs display the properties available to the current expression. These include all elements added to the current library object and all elements added to the library objects that are higher in scope. Tabs are used to organize the different actions that can be taken on each element:

- Parameters
  Use this tab to add new parameters to the expression, edit the existing parameters, delete a parameter from the expression, copy an existing parameter, show references to other uses of the parameter, and move the parameter up or down in the instantiation order of the expression.
- Functions
  Use this tab to add new functions to the expression, edit the existing functions, delete a function from the expression, copy an existing function, paste an existing function, and show references to other uses of the function.
- Expressions
  Use this tab to add an existing expression to the selected library object. Using saved expressions eliminates repetitive expression coding when multiple fields use the same expression, or when an existing expression can be used as a template. Double-click an expression from either the **Most Recently Used** or **Saved** lists to add that expression to the Expression box. If needed, expressions can be deleted from the expressions list using this tab. The Library Designer saves a running list of the last ten expressions created in the **Expression Builder**. Saved expressions are available to all projects opened in the Library Designer.
  Both functions and expressions can be saved within the Library Designer and used in future projects.
- Predefined
  Use this tab to add items from the list of global and local parameters that are available to all library objects to the expression. Predefined parameters cannot be modified, deleted, copied, or pasted.

> **IMPORTANT:** Saved and recently used expressions are carried over from previous projects. The decorative elements in these expressions may not be present in the current project.

Predefined functions and operators

The functions and operators available in the Expression Builder on page 59 provide the ability to test and manipulate the values returned by the decorative elements.

Tip: Mouse over or click a predefined function or operator to display a tooltip describing the input options for that function or operator. The tooltip will remain open until manually closed or the **Expression Builder** is closed.

The Expression on page 64 box displays the expression statement as it is created. Elements of the statement can be entered manually or by clicking items in the decorative elements tabs or the functions and operators area. This table describes the functions and operators in the **Expression Builder**.

Name	Description
Functions - Elements are inserted into these functions to replace the <*value*> token.	
Format	Inserts: {Format(<value>,<style>,<valueDataType>)}. Returns the value entered in <value> using the numeric format entered in <style>. Hexadecimal Byte Formatting styles: • (XBB) – Hex Byte Big Endian • (XBL) – Hex Byte Little Endian Standard Microsoft® styles: • (D) Decimal • (X) Hexadecimal • (C) Currency • (E) Scientific • (F) Fixed Point • (G) General (default) • (N) Number • (P) Percent Data type options are: • Int • Real • DateTime
Modulus	Inserts: {Modulus(<value1>,<value2>)}. Returns the remainder after the value of the decorative element inserted as <value1> is divided by the value of the decorative element inserted as <value2>.
Round	Inserts: {Round(<value>)}. Returns the rounded value of the decorative element inserted as <value>.
Abs	Inserts: {Abs(<value>)}. Returns the absolute value of the decorative element inserted as <value>.
Int	Inserts {Int(value)}. Returns the integer portion of a number.
Length	Inserts: {Length(<value>)}. Returns the number of characters of the decorative element inserted as <value>.

Name	Description
GetParameterMax	Inserts: {GetParameterMax({ObjectName},<subobjectType>,<parameter>)}. Applies to parameters that are included in a subobject. Returns the maximum value for the parameter inserted as <parameter> from within all subobjects that match the predefined parameter {SubObjectName} within the library object that matches {ObjectName}.
CalcBitNumber	Inserts: {CalcBitNumber({ObjectName},<subobjectType>,<parameter>,<subobjectIndex>,<resultDataType>,[startIndex],[endIndex])}. Applies to parameters that are included in a subobject. Returns the binary equivalent decimal value of a bit parameter across multiple subjects or within a certain range defined by the index values.
CalcBitwiseValue	Inserts: {CalcBitwiseValue(<Tag Type>, <Mask Value>, <Bit number>: {Expression})}. Returns the calculated value of Tags of type SINT, USINT, INT,UINT, DINT, UDINT,LINT and ULINT when the individual bits are parameterized. Each bit of the tag can have an individual expression applied to it. The instruction operates like an AND instruction on the mask and the parameterized value.
SubObjectCount	Inserts: {SubObjectCount({ObjectName},<subobjectType>}. Returns the number of subobject instances for the subobject inserted as <subobjectType> within the library object that matches the predefined parameter {ObjectName}.
GetParameter	Inserts: {GetParameter(<objectInstanceName>,parameterName, defaultValue)}. Returns the parameter value of the specific object or the defaultValue if the function is unresolvable.
GetSubParameter	Inserts: {GetSubParameter(<objectInstanceName>, <subobjectType>, <subobjectIndexName>, <subobjectIndexValue>,parameterName, defaultValue)}. Returns the parameter value of the specified object or the defaultValue if the function is unresolvable.
InstanceCount	Inserts:{InstanceCount(<librarySolution>, <catalogNumber>, <major>, <minor>)}. Counts and returns the number of instances for a specific library.
Logical and Mathematical Operators	
True	Logical TRUE: used to test the value returned by a decorative element.

Name	Description
False	Logical FALSE: used to test the value returned by a decorative element.
Or	Logical OR.
And	Logical AND.
Not	Logical NOT.
= <>	Equal to and not equal to.
( )	Parentheses, used to set the order of operation for complex expressions.
&	Mathematical AND.
> >= <= <	Greater than, greater than or equal to, less than or equal to, and less than.
+ - * /	Plus, minus, multiplied by, and divided by.

Expression box

An expression is a one-line statement that generates a single, calculated result. Expressions can return a string, numeric, or Boolean value, and generate values automatically during instantiation.

Expressions can incorporate any decorative element available to the current library object element, as well as a set of logical and mathematical operators, text strings, and numeric characters. Decorative elements and functions display as tokens surrounded by curly brackets as shown here: *{Element Name}*.

The **Expression** box displays the expression as elements are added to it. The **Expression** box includes these items:

Name	Description
Expressions window	A blank space in which the expression appears as elements are added to it.
Validate	Click this button to validate the current expression for syntax and data type. If the expression is valid the box at the bottom of the window is shaded green and displays the message "passed". If the expression is not valid, the box is shaded red and displays an error message.
Test	Click this button to test the current expression and display the result of the calculation or an error message if the expression is not valid.
Save	Save the expression. The expression is added to the **Saved** portion of the **Expressions** tab and becomes available to any project opened in the Library Designer. Click **OK** in the **Expression Builder** to apply the saved expression to the current field.

Name	Description
Result Type	Displays the data type of the expression. • String • Boolean • Numeric

Configure an element substitution

Element substitutions are performed to replace an element during instantiation. Elements that can be substituted include modules, tasks, programs, and routines.

To configure an element substitution

1. In the **Library Objects** pane, expand the library object folder that contains the element to be substituted.
2. Click the element, the decorator pane displays the element attributes.
3. Next to an attribute for which to substitute a different value (such as Name, Description, or ParentModule) click the ellipsis (...) button to open the Substitution Builder on page 66 window.
4. Configure the substitution as needed.
5. Click **OK** to save the substitution.

References window

How do I open the References window?

To open the **References** window for a parameter, right-click the parameter to open the contextual menu, then select **Show References**.

Once a parameter, substitution, or function is added to a library object, it may be referenced multiple times within the library object. Parameters, substitutions, or functions added to a project or controller library object may be referenced from within multiple library objects.

Examples of ways references can be used:

- As a part of a field value
- As part of an expression
- As part of a substitution
- As part of a function

The **References** window lists all references to the selected item and navigates to the referencing entities.

The fields in the **References** window.

Name	Description
Library	The library object for the element that references the item.
Used by	The element that references the item. The value is displayed in this format: *Element Type:SubObject Name*.

Name	Description
Details	The specific reference point within the element. For example, a field in the **Edit Parameter** window, a conditional inclusion, or the **Value Expression** for a controller or local tag.
Navigate	Click to open the decorator panel for the element where the reference is located.

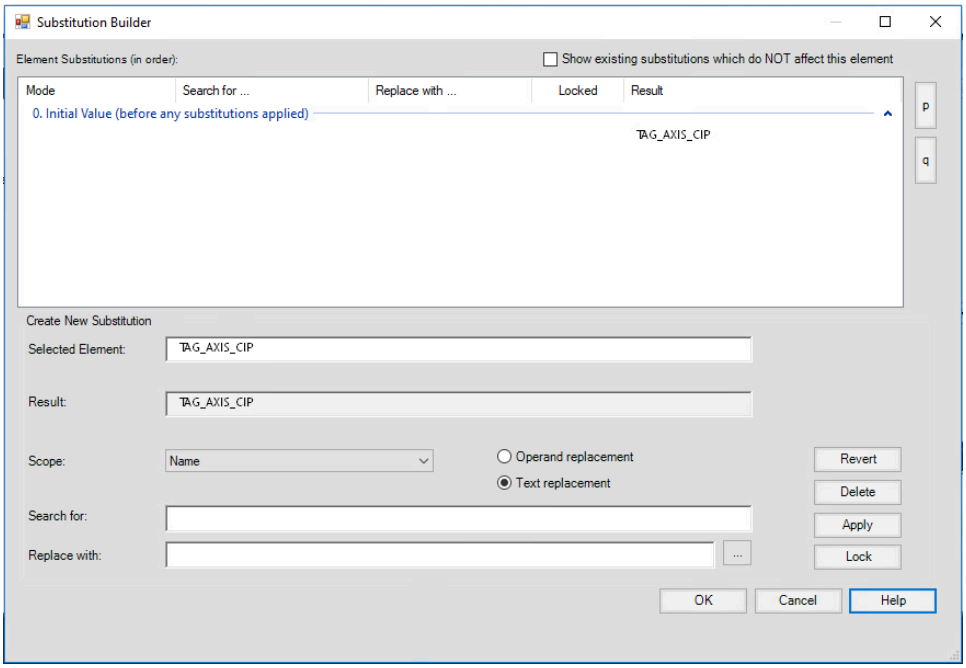
Substitution Builder window

How do I open the Substitution Builder window?

1. In the **Library Objects** pane, expand the library object folder.
2. Click an element, the decorator pane displays the element attributes.
3. Next to an attribute for which to substitute a different value (such as Name, Description, or ParentModule) click the ellipsis (...) button to open the **Substitution Builder** window.

Use the **Substitution Builder** to create substitutions that are specific to the current library element. Create one or many substitutions based on the text strings contained in the original field value.

The **Substitution Builder** has two functional areas: the **Element Substitutions** panel and the **Create New Substitutions** panel.



Element Substitutions

Displays a list of substitutions and their effect on the value of the current field:

- If the **Show existing substitutions which do NOT affect this element** checkbox is not selected, the window limits the display to substitutions that include text strings that are part of the original value of the current field.
- If the **Show existing substitutions which do NOT affect this element** checkbox is selected, the window displays all substitutions that have been added to the current library object and to library objects that are higher in the project hierarchy.

Tip: Displaying all substitutions shows potential conflicts and unexpected replacements for substitutions added to the element.

The display traces the field value from its initial value through any Substitutions that have been applied. The display is grouped as follows:

- **0. Initial Value** displays the original value of the field.
- **1. Library substitution for [Base library object Name]** displays the Substitution applied from the Base library object, if one exists.
- **2. Library substitution for {library object Name]** displays the Substitution applied from the library object, if one exists.
- **3. [Field Name] substitution** displays Substitutions created for the current field.

Tip: Depending on where in the hierarchy substitutions are added, **Groups 1, 2, or 3** may not appear.

Substitutions appear highlighted in yellow.

Create New Substitution

The **Create New Substitution** area displays the fields and commands for creating a substitution.

Name	Field Type	Description
Selected Element	Read-only	The original text string entered in the **Search For** field.
Result	Read-only	The result generated when the **Apply** button is clicked to test the Substitution.
Scope	Dropdown menu	The scope for the new substitution. There are three options: • [Current Field] • library object • base library object If a library object or base library object is selected, the new substitution will replace previous substitutions added to that library object for the same text string.
Search For	Text entry	The text string to replace.

Name	Field Type	Description
Replace With	Text entry	The replacement text. This can be entered manually or generated by an expression.
Operand replacement/ Text replacement	Radio buttons	Sets the condition under which the current tag is instantiated. The default setting is Always. A condition can be entered manually or generated by an expression.
Revert	Button	Reverts to the current substitution for the field.
Delete	Button	Deletes the currently selected line in the Element Substitutions window. **Important**: Deleting a Substitution removes it from the library object.
Apply	Button	Tests the current substitution and adds it to the **Element Substitution** section.
Unlock/Lock	Button	Unlocks an existing substitution, allowing it to be edited, or locks a substitution, helping prevent it from being edited.

Tip: To use a calculated value for this field, click the ellipsis (...) button next to the field to open the **Expression Builder**.

Parameters tab

Parameters are listed alphabetically by default. Group parameters make organization and access more intuitive.

- Group names display in blue. Group names display in this format: `library object Name [Group Name]`.
- Collapse and expand the group using the arrow buttons on the right, or by double-clicking the group name. Parameters display alphabetically within their groups, although the display order can be changed.
- When a new library object is added, an initial group is created in the **Parameters** tab. The group name matches the library object name. Additional groups can be added as parameters are added to the library object.
- Subobjects appear in the **Parameters** tab as a type of group. Subobject names display in blue. Subobject names are displayed in this format: `library object Name.SubObject Type`.

This table describes the commands available on the **Parameters** tab toolbar.

Command	Description
	The **Add New** button opens the **Add New Parameter** window.
	The **Move Up/Move Down** buttons change the position of a selected Parameter item in the tab listing. Parameters are listed in alphabetical order by default. Parameters may be moved

Command	Description
	up or down within their group but cannot be moved between groups using these buttons.
FindPartial	This dropdown menu can initiate a search for parameters that match a text string entered in the adjacent field. There are two different types of finds that can be performed, use the dropdown menu to select the one to use: • **FindPartial**: Searches for the text string in any part of an entry. • **FindPrefix**: Searches for the text string at the beginning of an entry.

Add a parameter

A parameter is an argument exposed for external access and controls how the library object is instantiated. Parameters allow a single instance of base controller code to have many variations and are used in a variety of different applications.

To add a parameter

1. In the decorator panel, either:
   - Click  on the **Parameters** tab toolbar.
   - In the **Parameters** tab, right-click a group name and then click **Add New Parameter**.
     The Add New / Edit Parameter on page 73 window opens.

     
     Tip: The entry fields in the window are functionally grouped. Some fields appear conditionally based on selections made in other fields.
2. Enter the values for the new parameter.
3. Click **OK** to add the new parameter.

Edit a parameter

Parameters allow a single instance of base controller code to have many variations and are used in a variety of different applications.

To edit a parameter

1. In the decorator pane, click the **Parameters** tab, either:
   - Double-click the parameter.
   - Right-click the parameter, then select **Edit**.
2. The **Edit Parameter** window opens.
3. Enter the values for the edited parameter.
4. Click **OK** to update the parameter.

Add an external reference

An external reference makes the value of a local tag, controller tag, or tag member within a library object accessible to parameters in other library objects.

IMPORTANT: While all of the fields in the References Builder on page 92 are open to manual editing, it is recommended to follow this procedure and allow the Library Designer to enter values automatically.

To add an external reference

1. Locate the tag within the library object.

   Tip: Controller tags and local tags are open for use as external references.

2. Double-click the tag name. The name highlights to indicate it has been selected.
3. Right-click the selected tag name, then select **Add External Reference**.
   The **References Builder** window opens.
4. Enter the values for the new external reference or accept the default values generated by Library Object Manager.
5. Click **OK** to add the external reference.

Copy a parameter

A parameter must be unique within a library object. A parameter can be copied to other library objects and to library objects of different scope.

To copy a parameter

1. In the decorator pane, right-click the parameter, then select **Copy**.
   Select multiple parameters by pressing the **Shift** or **Ctrl** key while clicking.
2. In the **Library Objects** pane, click the library object to which to add the parameter.
3. In the decorator pane, right-click the name of the group in the **Parameters** tab and then click **Paste Parameter(s)**.
   The parameter is added to the library object.

   Tip: The parameter retains its original group name, so a new group might also be added along with the parameter.

Add a group

All groups other than the default group must contain at least one parameter. Parameter groups are added, modified, and deleted through the **Group** setting in the **Add new Parameter** and **Edit Parameter** windows.

When the **Group** setting is blank, the default Group name is based on the name of the library object. To use a different default group name, add text to the **Group** setting of the first parameter added to the library object, the text is appended to the library object name. Subsequent parameters added will use that group name by default. If the **Group**

setting is edited to be blank in any parameter, the original default group name based on the library object name is re-established.

To add a group

1. In the decorator panel, either:
   - Click  on the **Parameters** tab toolbar. The **Add New Parameter** window opens.
   - Double-click an existing parameter that does not belong to a group. The **Edit Parameter** window opens.
2. Under the **UserInterface** section, in **Group**, type a name.
3. Click **OK**. A new group is created, and the parameter is included in the new group.

Reassign a parameter to a group

Parameter groups are added, modified, and deleted through the **Group** setting in the **Add New Parameter** and **Edit Parameter** windows. Changing the Group setting of a parameter reassigns it to another group or creates a group if the group specified does not exist. To reassign each parameter in a group, edit each parameter individually.

To reassign a parameter

1. In the decorator pane, click the **Parameters** tab, then either:
   - Double-click the parameter.
   - Right-click the parameter, then select **Edit**.
2. The **Edit Parameter** window opens.
3. Under the **UserInterface** section, in **Group**, type a new name or a name of an existing group.

    **Tip:** When reassigning parameters to existing groups, make sure to type the group name exactly as it appears in the **Parameters** tab.

4. Click **OK** to reassign the parameter.

Move a parameter within a group

Parameters can be collected together into a group. The order of the parameters in the group determines the order in which each parameter is instantiated.

To move a parameter within a group

1. In the decorator panel, on the **Parameters** tab, click the parameter.
2. After selecting the parameter, either:
   - On the **Parameters** toolbar, click **Move Up** or **Move Down**.
   - Right-click the parameter, then select **Move Up** or **Move Down**.

    **Tip:** A parameter cannot be moved outside of its group. To change the group membership, reassign the parameter.

Delete a group

Parameter groups are added, modified, and deleted through the **Group** box in the **Add New Parameter** and **Edit Parameter** windows. To delete a group, all of the parameters in the group must be edited and assigned to other groups. Once a group has no members, it is removed from the parameters list. The first group shown on the parameters tab is the default group and cannot be deleted. When the **Group** setting is blank, the parameter is assigned to the default group.

To delete a group

1. In the decorator pane, click the **Parameters** tab, either:
   - Double-click the parameter.
   - Right-click the parameter, then select **Edit**.
2. The **Edit Parameter** window opens.
3. Under the **UserInterface** section, in **Group**, change the name to a different group or delete the text so that it is blank.
4. Click **OK**.

Tip: If reassigning parameters to existing groups, make sure to type the group name exactly as it appears in the **Parameter** tab. If the text in the **Group** setting does not match an existing group name, a new group is created and the parameter is assigned to it.

Add a subobject

A subobject is a grouped set of parameters that can be instantiated multiple times. Subobjects can be auto-generated during instantiation or added manually by the user when the library object is brought into an Application Code Manager project.

To add a subobject

1. In the decorator panel, right-click any group name, then select **Add New SubObject**.
   The New SubObject on page 78 window opens.
2. Enter the values for the new subobject.
3. Click **OK** to add the new subobject.

Limit the external references

The **Filter** field can be used to limit the external references available to a parameter where the **Reference Type** is set to **Reference**. The parameter references the value of the external reference when the project is in operation. Reference-type parameters are defined so that the external references that are accessible to them are limited to those that meet certain criteria (filters).

To limit the external references available to a parameter

1. In the decorator pane, double-click the parameter to open the **Edit Parameter** window.
2. Under the **UserInterface** section, click the **Filter** setting and then click the ellipsis (...) button to open the Parameter Filter Builder on page 79 window.
3. The **Filters** tab lists the objects that can be used as filters. To add an object to a filter expression, double-click its listing. The object is added in this format:

Library.[Classification Level] = '[Object Name]'

4. To add a logical operator to the expression, either:
   a. Click the operator listing once.
   b. Enter the text manually.
5. Click **Validate** to validate the filter. The filter is validated for internal syntax and data type. The validation bar displays green for a passed validation and red, with an error message, for a failed validation.
6. Click **OK** to accept the filter expression.

View a parameter reference

Use the **References** window to view a list of all references to a parameter. A reference makes the value of a tag or element within a library object accessible to parameters in other library objects. From the **References** window, navigate to the referencing entities to view tag details.

To view a parameter reference

1. In the **Library Objects** pane, click the library object that contains the parameter of interest.
2. In the decorator pane, on the **Parameters** tab, right-click the parameter and then click **Show References**.
   The References on page 65 window opens.
3. Click **Navigate** for the reference listing. The decorator pane updates to display the tag details for that element.
4. Click **OK** to close the **References** window.

Tip: The **References** window must be closed before the items in the decorator panel can be accessed.

Delete a parameter

Delete parameters that are not used. If a parameter is used in a reference, it cannot be deleted. Use the **Show References** command to view parameter references before attempting to delete a parameter.

To delete a parameter

1. Right-click the parameter, then select **Delete**.
   The **Delete Objects** dialog opens.
2. To finish the deletion, click **Yes**.

Add New / Edit Parameter window

How do I open the Add New / Edit Parameter window?

- Click  on the **Parameters** tab toolbar.
- In the **Parameters** tab, right-click a group name and then click **Add New Parameter**.

The settings in the window are functionally grouped. Some settings appear conditionally based on selections made in other settings. This table describes the settings in the **Add New / Edit Parameter** window.

Name	Type	Description
01 General		

Name	Type	Description
Name	Text entry	The name of the parameter. This is a required field.
Parameter Help	Text entry	Help text to explain the function of the parameter and the result when specific values are entered. Appears at the bottom of the parameter window when the parameter is selected in the Application Code Manager application.
Scope	Dropdown menu	The library object that contains the parameter. The list includes all library objects in the library object tree. The default value is the current library object. Parameters can be moved to a different library object by changing the selection here.
Data Type	Dropdown menu	The data type of the parameter. Options are: • Bool (Boolean) • String • Int (Integer) • Real • AxisID
02 Reference		
Reference Type	Dropdown menu	Determines whether the parameter is accessible to user input as an entry field, is populated automatically by a calculation, or references other parameters. The options are: • Immediate: The parameter is accessible to user input as an entry field. • Calculated: The parameter is not accessible to user input. The value is set to the name or description of the selected Parameter. • Reference: The parameter references an external reference. The parameter and the external reference function as a consumed and a produced tag. The parameter is linked to the external reference

Name	Type	Description
		after instantiation. The **Filter** setting can be used to set criteria for the external reference.
These fields appear if **Calculated** *is selected in* **Reference Type**.		
Reference Linked Parameter	Dropdown menu	Select the parameter that this parameter is linked to from the list. The list of available parameters is composed of parameters within the library object where the **Reference Type** is defined as "Reference". This is a required field.
Reference Field	Dropdown menu	Select the field within the parameter that contains the value to use. This is a required field.
Use Custom	Text entry	Specifies a condition which, if true at instantiation, overrides the default and opens the field to user input. The value can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the Expression Builder on page 59.
03 Data		
Default Value	Text entry	The default value for the parameter. The value can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the **Expression Builder**.
This field appears if String is selected in **Data Type** *and the* **Reference Type** *is Immediate.*		
Max Length	Text entry	The maximum length allowed for the value of the parameter. The length can be entered manually in which case the parameter value will be limited to this number of characters in Application Code Manager. Leave blank to allow any length.
Append	Text entry	A text string that will be added to the end of the value of the parameter.
These fields appear if **Integer** *or* **Real** *is selected in* **Data Type**.		
Min	Text entry	The minimum value for the parameter. The value can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the **Expression Builder**.

Name	Type	Description
		This is a required field.
Max	Text entry	The maximum value for the parameter. The value can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the **Expression Builder**. This is a required field.
Engineering Unit	Text entry	A unit of measure added to the value of the parameter to provide additional context.
Invalid	Text entry	An additional condition for validation of parameter values. The condition can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the **Expression Builder**.
This field appears if **Integer** *is selected in* **Data Type**.		
Padding	TRUE-FALSE Dropdown menu	When TRUE is selected, single-digit integer values will be padded with zeros.
This field appears if **Integer** *is selected in the* **Data Type** *field and the parameter is being added to a subobject.*		
AutoIncrement	TRUE-FALSE Dropdown menu	When TRUE is selected, the parameter will be populated automatically and the value incremented every time a new subobject is added to an ACM Project. For example, multiple channel subobjects will be numbered sequentially as they are added.
AutoIncrementOffset	Text entry	When a library object is created, set the offset to increment the autoincrement parameter value.
04 UserInterface		
Group	Text entry	The group in the **Parameters** tab where the parameter will appear. If the value entered matches an existing group name, the parameter will appear in this group. If the value entered does not match an existing group name, a new group will be created.
Filter	Text entry	When **Reference** has been selected as the **Reference Type** for the new parameter, use this field to create a filter expression to limit the external

Name	Type	Description
		references. For example, an input parameter for a valve can be limited to an input channel that is a subobject of a module. The filter expression can be entered manually or generated by the Parameter Filter Builder on page 79. Click the ellipsis (...) button next to the field to open the **Parameter Filter Builder**.
Visible	Text entry	Defines a condition to control whether the parameter is displayed when the library object is added to an ACM Project. The condition can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the **Expression Builder**.
Disabled	Text entry	Defines a condition that disables the parameter during instantiation. The condition can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the **Expression Builder**. This field only affects parameters that are included in a subobject.
Read Only	TRUE-FALSE Dropdown menu	Determines whether the user will be able to enter values for the parameter in the ACM application, or only read values that have been generated.
This field appears if **Immediate** *is selected in the* **Reference Type** *field and* **String**, **Integer**, *or* **Real** *is selected in the Data Type field.*		
Control Type	Dropdown menu	Determines the mode of user interaction with the parameter. • **TextBox**. The parameter value is directly typed into a text box. • **DropDownList**. The parameter value is selected from a list of possible values.
This field appears if **DropDownList** *is selected in the* **Control Type** *field.*		
Control Values	Text entry	Defines the items included in the dropdown list from which the user can select. Enter the options as a comma-separated string. This is a required field when it appears.

New / Edit SubObject window

How do I open the New / Edit SubObject window?

1. In the **Library Objects** pane, select a library.
2. In the decoration pane, on the **Parameters** tab, right-click any group name, then select **Add New SubObject**. The **New / Edit SubObject** window opens.

A subobject is a grouped set of parameters that can be instantiated multiple times. Examples include the channels of an analog input or the contact information for a project team member. Subobjects can be auto-generated during instantiation or added manually by the user when the library object is brought into an Application Code Manager project.

This table describes the fields in the **New / Edit SubObject** window.

Name	Type	Description
Name	Text entry	The name applied to the subobject when added to an Application Code Manager project. The value can be entered manually or generated by an expression.
Type	Text entry	The identity for the SubObject group in the **Parameters Tab**. This is a required field. The name must be unique for the library object.
Description	Text entry	A description of the subobject that appears at the bottom of the Application Code Manager window when its subobject is active. The value can be entered manually or generated by an expression. To use a calculated value for this field, click the ellipsis (...) button next to the field to open the **Expression Builder**.
Auto Populate	Checkbox	When selected, automatically creates multiple instances of the subobject during instantiation. For auto population to occur: • There must be a parameter contained in the subobject with the **Data Type** field set to INT. This field generates sequential identity numbers for each instance of the subobject. • A value must be entered in the **Min** and **Max**fields for the parameter. The **AutoIncrement** field must be set to TRUE.

Name	Type	Description
Pre-configured Instances	Checkbox	Selected when the number of subobjects are needed. Allows multiple subobjects to be created using a pre-configured option.
Auto Increment Parameter	Dropdown menu	Only available when **Auto Populate** is selected. Lists all integer subobject parameters. If no parameters are configured, the list is empty.
Dimension	Text entry	Determines how many instances of the subobject are automatically created on instantiation into the Application Code Manager. The maximum number should not be greater than **Auto Increment Parameter**.
Pre-configured Instances	Checkbox	Select to enable **Add Instance**.
Add Instance	Button	Select to add a pre-configured instance. Dimmed when the **Pre-configured Instances** checkbox is not selected.
Add Parameter	Button	Select to display the **Add New Parameter** dialog.

Parameter Filter Builder window

How do I open the Parameter Filter Builder window?

1. In the decorator pane, click the **Linked Libraries** tab.
2. Click the linked library in which to add the parameter links. The **Add new parameter link** button in the **Parameter Links** toolbar becomes active.
3. In the **Parameter Links** pane, either:
   - Click the **Add new parameter link** button.
   - Right-click an empty row and click **Add**.

Use the **Filter** field to limit the external references available to a parameter.

The **Parameter Filter Builder** is divided into functional areas and contains two tabs.

Name	Description
Filters/Saved Filters Tabs	The **Filters** tab lists objects and elements within the current hierarchy of library objects, as well as their current values. They are grouped, in descending order, by the four classification levels applied when the library object is saved in the Library Object Manager application: • Catalog number • Category • Family

Name	Description
	• Library type The **Saved Filters** tab displays filters previously created for the current ACD file.
Filters and Operators	The most commonly-used elements are listed here for quick access, as well as the logical AND, logical OR, and equals (=) functions.
Expression	The parameter filter appears here as elements are added to it. There is also a button to validate the filter. The validation bar at the bottom displays a color-coded response when the filter is validated.

Functions tab

A function is an argument that is not exposed to external access. The value of a function is generated by user-defined logic created in the Library Designer and by conditions that apply during instantiation.

A function can be either conditional or calculated:

- A conditional function returns one of multiple possible results generated by expressions and based on IF/ELSE/ELSEIF logic. A conditional function allows for multiple branches and nesting.
- A calculated function generates a single value, based on a single expression.

Functions are listed alphabetically under a single group heading named by the library object.

This table describes the commands available on the **Functions** tab.

Command	Description
	The **Add New** button opens the **Function Builder** window.
	The **Move Up/Move Down** buttons are deactivated for this tab.
	This dropdown menu can initiate a search for functions that match a text string entered in the adjacent field. There are two different types of finds that can be performed, use the dropdown menu to select the one to use: • **FindPartial**: Searches for the text string in any part of an entry. • **FindPrefix**: Searches for the text string at the beginning of an entry.

Add a function

A function is an argument that is not exposed to external access. The value of a function is generated by user-defined logic created in the Library Designer and by conditions that apply during instantiation.

To add a function

1. In the decorator panel, either:
   - Click  on the **Function** tab toolbar.
   - In the **Function** tab, right-click a group name and then click **Add new Function**.

     The Function Builder on page 84 window opens.

     Tip: The fields displayed in the **Function Builder** are dynamic depending on the type of function selected.

2. Enter the values for the new function.
   - In **Type**, either:
     - Choose **Calculation**, then specify an **Expression** to evaluate to generate the output value.
     - Choose **Conditional**, then specify a logical condition to evaluate using **If/Else** logic to generate the output value. Expressions can be used as part of the logical statement for both evaluated conditions and output values.
   - In **Result Type**, choose the data type of the functions output.
   - In **Function Scope**, choose the library object that contains the function.
   - To base the new function on an existing function, click **Import** and then select the function.

     Tip: Click the ellipsis (...) button next to a field that supports expressions to use the Expression Builder on page 59 to create the expression.

3. (optional) Click **Save** to save the function as an XML file for reuse purposes.
4. Click **OK** to add the new function.

Add branches

A conditional function returns one of multiple possible results generated by expressions and based on IF/ELSE/ELSEIF logic. A conditional function allows for multiple branches and nesting.

To add branches to a conditional statement

1. Open the function in the Function Builder on page 84.
2. Right-click an item in the conditional statement menu tree.
3. Select the root **Condition** item to add an **ELSEIF** statement to the root level of the statement. The **Function Builder** adds an **Else if this expression is true:** and a **Return this value:** field.

   Tip: Adding ELSEIF statements to the conditional statement causes it to function like a CASE statement.

4. Select an IF or an ELSEIF item to nest an IF statement within it. The **Function Builder** adds an **If this expression is true:**, a **Return this result:**, and an **Else:** field.

   Tip: There is no limit to the number of branches or the number of nested levels in a conditional statement.

Use a previously created function

Use the **Import** button to reuse a previously created function, either to populate the current function fields or as a template to create a function.

To use a previously created function

1. In the Function Builder on page 84 window, click **Import**.
   The **Saved Functions** window opens.
   The **Saved Functions** window lists all functions that were previously created in the current ACD file. Columns list the function name, result type, and function type.

   Tip: If any functions are no longer used, click **Delete** to delete them.

2. Select a saved function and click **OK**. The saved function populates the **Function Builder** window.
3. Click **OK** to reuse the function or edit the fields to create a function.

   IMPORTANT: Function names must be unique within a library object. A warning displays when the function name matches the name of an existing function in the current library object. Rename the function to be unique.

Copy a function

A function can be copied from one library object to another library object.

To copy a function to a different library object

1. In the decoration pane, click the **Functions** tab.
2. Right-click the function, then select **Copy**.
3. In the **Library Objects** pane, click the other library object.
4. In the decoration pane, click the **Functions** tab.
5. Right-click a group name, then select **Paste**.
   The function is added to the library object.

   Tip: If the library object does not contain all of the decorative elements referenced by the function, the Function Builder on page 84 window opens with any statements that contain missing elements outlined in red.
   The function is pasted into the library object even if the missing elements are not resolved.

Edit a function

A function is an argument that is not exposed to external access. The value of a function is generated by user-defined logic created in the Library Designer and by conditions that apply during instantiation. Edit a function when needed, such as to add branches to a conditional statement or update an expression.

To edit a function

1. In the decorator pane, click the **Functions** tab, then either:
   - Double-click the function.
   - Right-click the function, then select **Edit**.

     The **Function Builder** window opens.
2. Edit the function.
3. Click **OK** to save the changes to the function.

View a function reference

Once it has been added, a function may be referenced multiple times within a library object. Functions added to a project or controller library object may be referenced from within multiple library objects.

These are examples of ways a function can be referenced:

- As a field value for a parameter or library object elements
- As part of an expression
- As part of a substitution
- As part of another function

To view a function reference

1. In the **Library Objects** pane, click the library object that contains the function of interest.
2. In the decorator pane, on the **Functions** tab, right-click the function and then click **Show References**. The References on page 65 window opens. Click **Navigate** for the reference listing. The decorator panel opens for that element.
3. Click **OK** to close the **References** window.

Tip: To access the fields in the decorator panel, the **References** window must be closed.

Delete branches

A conditional function allows for multiple branches and nesting. The additional branches and nested conditions can be deleted if they are not needed.

To delete branches from a conditional statement

1. Right-click an item in the conditional statement menu tree.
2. Select **Delete**.

Tip: The root IF or ELSE statements cannot be deleted.

Delete a function

Delete a function if it is no longer used.

Tip: Use the **Show References** command to list references to the function before using the **Delete** command.

To delete a function

1. Right-click the function, then select **Delete**. The **Delete Objects** window opens.
2. To finish the deletion, click **Yes**.

Tip: A warning window displays if the function is referenced.

Function Builder window

How do I open the Function Builder window?

- In either the Library Designer decorator panel or the **Expression Builder** window, click the **Functions** tab, right-click a category name and select **Add New Function**.

The value of a function is generated by user-defined logic created in the Library Designer and by conditions that apply during instantiation. Functions are copied between library objects and between library objects of different scope, as long as the decorative elements used in the expressions are common to both library objects.

Use the settings in the **Function Builder** window to create functions or to modify an existing function after it is imported.

This table describes the settings in the **Function Builder** window.

Name	Setting	Description
Name	Text entry	The name of the function. This is a required field.
Type	Dropdown menu	Determines whether the function generates a single value or uses IF/ELSE logic to generate one of multiple possible values: • **Conditional**: The function uses IF/ELSE logic to generate one of multiple possible values. • **Calculation**: The function generates a single value based on a single statement.
Result Type	Dropdown menu	The data type of the function. Options are: • Boolean • String • Numeric • Integer

Name	Setting	Description
Function Scope	Dropdown menu	The library object that contains the function. The default value is the current library object.
This setting appears if the **Type** selected is **Calculation**.		
Expression	Dropdown menu	A single statement that determines the value generated by the function. The value can be entered manually or generated by an Expression. This is a required field when **Calculated** has been selected in the **Type** field.
These settings appear if the **Type** *selected is* **Conditional**.		
Statement Definition		
Condition	Menu	Provides a visual representation of the conditional statement in menu form. The **Function Builder** window displays, at most, one **If this expression is true:** and one **Return this value:** field at a time. Select each line in the conditional statement menu to access the statement fields for that line. • Click a menu item to make its statement field active. • Click the **Plus Icon** to the left of a menu item to expand the listing. This icon appears when there are nested statements within the item. • Click the **Minus Icon** to the left of a menu item to collapse the listing. This icon appears when there are nested statements within the item. • Use the scroll bar to access additional menu items if the conditional statement menu has grown too large to fit in the window.
If this expression is true	Text entry	The condition being tested. The value can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the **Expression Builder**. This is a required field when **Conditional** is selected in the **Type** field.
Return this value	Text entry	The value used if the condition returns TRUE.

Name	Setting	Description
		The value can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the **Expression Builder**. This is a required field when **Conditional** is selected in the **Type** field.
Else	Text entry	The value used if the condition returns FALSE. The value can be entered manually or generated by an expression. Click the ellipsis (...) button next to the field to open the **Expression Builder**. This is a required field when **Conditional** is selected in the **Type** field.

Substitution tab

A substitution is a user-defined rule that, during instantiation, replaces a text string in the name, description, instantiation location, or other attribute of a library object element with a parameter value, calculation result, or referenced value. Substitutions can be set to search all text in the library object, or restricted to text in operands, which are the instructions in routines.

Keep these considerations in mind when using substitutions.

- Substitutions applied at one level of the library object hierarchy extend to all Objects at lower levels of the hierarchy, and to all elements that are contained within the library objects. For example, a substitution applied to a Project library object extends to the Controller library object and all Logix Object library objects in the same ACD file, as well as to all elements within all of these library objects.
- Substitutions applied to a library object extend to all elements within the library object. Substitutions applied at a higher level in the hierarchy take precedence over substitutions applied directly to the library object.
- Substitutions that are inherited by an element from the containing library object, or from a library object higher in the library object hierarchy (base library), can be overridden at the element level using the **Substitution Builder**.
- Substitutions can be copied and pasted from one library object to another and can be copied and pasted between library objects of different scope.
- Substitutions are not grouped.

> **IMPORTANT:** Substitutions are applied globally based on a simple search-and-replace logic. Care should be taken in both the naming conventions and standards used when elements are created in Logix Designer, and in the text strings selected for substitution in the Library Object Manager application.
> Substitutions that affect text strings in unexpected locations can make the library object function in unexpected ways or fail to validate.

This table describes the commands available on the **Substitions** tab.

Command	Description
	The **Add New** button opens the **Substitution Builder** window.
	The **Move Up/Move Down** buttons change the position of a selected Substitution item in the tab listing. Substitutions are listed in alphabetical order by default.
	This dropdown menu can initiate a search for functions that match a text string entered in the adjacent field. There are two different types of finds that can be performed, use the dropdown menu to select the one to use: • **FindPartial**: Searches for the text string in any part of an entry. • **FindPrefix**: Searches for the text string at the beginning of an entry.

Add a substitution

A substitution is a user-defined rule which, during instantiation, replaces a text string in the name, description, instantiation location, or other attribute of a library object element with a parameter value, calculation result, or referenced value.

To add a substitution

1. In the decorator panel, either:
   - Click  on the **Substitutions** tab toolbar.
   - In the **Substitutions** tab, right-click a group name and then click **Add New Substitution**.
     The Substitution Builder on page 66 window opens.
     All fields are required.
2. Enter the values for the new substitution.
3. Select a scope in the **Scope** field.
4. In the **Original** field, enter an initial text string to be replaced.
5. In the **Replacement** field, enter a replacement string. Strings, numeric characters, and tokens can all be used in the replacement string.
   Click the ellipsis (...) to open the Member Selector on page 89 window and select a decorative element to use as the replacement.
6. In **Search Mode**, specify the type of search to use during the substitution process.
   - Choose **Text** to search for text strings not in tokens.
   - Choose **Operand** to search for an operand in a token.
7. Click **OK** to close the **Substitution Builder**.
   The substitution is added to the **Substitutions** tab.

Change the order of execution

Substitutions applied at one level of the library object hierarchy extend to all objects at lower levels of the hierarchy, and to all elements that are contained within the library objects. Substitutions applied at a higher level in the hierarchy take precedence over substitutions applied directly to the library object.

To change the order of execution for a substitution

1. In the decorator pane, click the **Substitutions** tab.
2. Click the substitution, either:
   - On the **Substitutions** toolbar, click **Move Up** or **Move Down**.
   - Right-click the substitution, then select **Move Up** or **Move Down**.

Tip: Substitutions can also be alphabetically sorted by column content. Click the column name to toggle between ascending and descending sort order.

Copy a substitution

A substitution can be copied to a different library object.

To copy a substitution to a different library object

1. In the decorator panel, click the **Substitutions** tab.
2. Right-click the substitution, then select **Copy**.
3. Open the library object in which to add the substitution.
4. Right-click the name of the group in the **Substitutions** tab, then select **Paste**.
   The substitution is added to the library object.

Tip: The substitution will only take effect in the new location if the text string to be replaced and all decorative elements used in the substitution are contained in the library object.

Edit a substitution

Edit a substitution to change the scope or replacement conditions when needed.

To edit a substitution

1. In the decorator pane, click the Substitutions on page 89 tab, either:
   - Double-click the substitution.
   - Right-click the substitution, then select **Edit**.
     The **Substitution Builder** window opens.
2. Edit the substitution as needed.
3. Click **OK** to save the edits to the substitution.

Delete a substitution

Delete a substitution when it is no longer needed.

To delete a substitution

1. Right-click the substitution, then select **Delete**.
   The **Delete Objects** confirmation window opens.

2. To finish the deletion, click **Yes**.

Tip: The **Warning** window does not display for substitutions.

Member Selector window

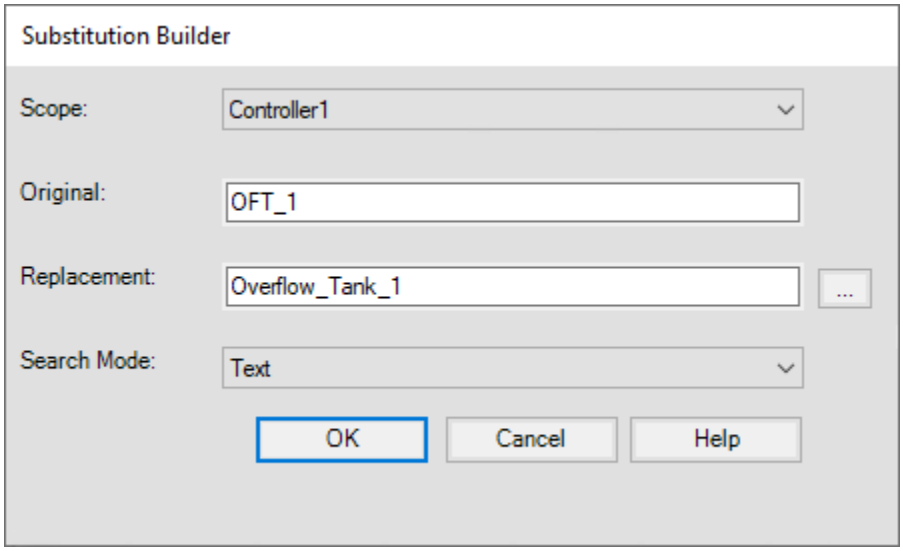
The **Member Selector** window gives quick access to all decorative elements available as replacements. All parameters and functions added to the current library object, including parameters and functions added to the library objects higher in the library object hierarchy, display in the window. Saved functions and predefined parameters also display.

The decorative elements are organized in tabs, with the same features as the tabs in the decorator panel.

New parameters and functions can be added to the project directly from the tabs in the **Member Selector**.

Substitution Builder

When adding a substitution to a library object or editing an existing substitution from the **Substitution** tab the **Substitution Builder** is restricted to just those settings that are applicable to the context of the substitution.



This table describes the fields available:

Name	Field Type	Description
Scope	Dropdown menu	The scope for the substitution. Defaults to the current library object.
Original	Text entry	The text string to replace.
Replacement	Text entry	The replacement text. This can be entered manually or generated by an expression. Click the ellipsis (...) to open the Expression Builder on page 59.

Name	Field Type	Description
Search Mode	Dropdown menu	Defines the type of search conducted to perform the substitution. • **Text**. Searches text strings in the defined scope for the original text and makes the designated replacement. • **Operand**. Search tokens in the defined scope for the original text and makes the designated replacement.

Predefined tab

A predefined parameter is one of a set of parameters that are automatically available to all library objects created in the Library Designer. They are defined and scoped by the program. They are the same for all library objects in the hierarchy, as well as for all elements of all library objects, and are available to all substitutions, expressions, and functions. Users cannot create, modify, or delete predefined parameters.

IMPORTANT: All predefined parameters are available to the **Expression Builder** regardless of the scope of the current element. A predefined parameter used to supply a field value to an object or element of higher scope will return no value.

Care should be taken to scope predefined parameters correctly when they are used to populate field values.

Predefined parameters are populated during instantiation when a library object is added to an Application Code Manager project.

Predefined parameters cannot be copied or pasted, since they are defined by the Library Designer and are identical for all library objects.

Predefined Parameters are listed alphabetically and are grouped by scope.

- Group names display in blue. Collapse and expand the groups using the arrow buttons on the right or by double-clicking the group name. Parameters display alphabetically within their groups. All of the columns are sortable.
- Predefined parameters sort within their groups.
- The items listed cannot be moved up or moved down independently of the sort order.
- Use **FindPartial** or **FindPrefix** to limit the items listed

This table describes the predefined parameters.

Name	Data Type	Description
Global		
ControllerDescription	String	Description of the controller for the current project.

Name	Data Type	Description
ControllerName	String	Name of the controller for the current project.
MotionGroupName	String	Name of the group.
ParentName	String	Name of the Object or element that contains the current element.
ProcessorType	String	Processor type of the controller for the current project.
ProjectDescription	String	Description of the current project.
ProjectName	String	Name of the current project.
SoftwareRevision	String	Software revision number of the controller for the current project.
Local		
ObjectDescription	String	Description of the current library object.
ObjectName	String	Name of the current library object.
ProgramDescription	String	Description of the program that contains the current element.
ProgramName	String	Name of the program that contains the current element.
SubObjectDescription	String	Description of the subobject that contains the current element.
SubObjectName	String	Name of the subobject that contains the current element.
TaskDescription	String	Description of the task that contains the current element.
TaskName	String	Name of the task that contains the current element.
Modules		
ParentChassisName	String	For module library objects only: The chassis name of the current library object.
ParentChassisSize	Integer	For module library objects only: The chassis size of the current library object.
ParentChassisSlot	Integer	For module library objects only: The chassis slot of the current library object.

External References tab

An external reference makes the value of a local tag, controller tag, or tag member within a library object accessible to parameters in other library objects. Used in conjunction with parameters that have been assigned to accept values by reference, external references provide the points of contact between library objects in an Application Code Manager project.

External references are listed alphabetically. They are grouped based on the value in the **Name** field for the external reference.

In an Application Code Manager project, link an external reference to a reference-type parameter. The parameter references the value of the external reference when the project is in operation. Reference-type parameters are defined so that the external references that are accessible to them are limited to those that meet certain criteria (filters).

- Any tag or tag member can be added as an external reference.
- An external reference can be copied to other library objects and to library objects of different scope.

  IMPORTANT: Check that the library object contains the referenced Tag before copying and pasting an External Reference.

- External references are not available to the **Expression Builder**.

Use the **External References** toolbar buttons to perform these actions.

Button	Description
	The **Add New** button opens the References Builder on page 92.
	The **Move Up/Move Down** buttons are deactivated for this tab.
	This dropdown menu initiates a search for external references that match the text string entered in the adjacent field.

Edit an external reference

Edit the external references as needed.

To edit an external reference

1. Right-click the external reference, then select **Edit**. The References Builder on page 92 window opens.
2. Edit the values of the external reference.
3. Click **OK** to update the external reference.

Delete an external reference

Delete external references when no longer needed.

To delete an external reference

1. Right-click the external reference, then select **Delete**. The **Warning** window opens.
2. To finish the deletion, click **Yes**.

Reference Builder window

External references provide the points of contact between library objects in an Application Code Manager project.

This table describes the fields in the **References Builder** window.

Name	Field Type	Description
Name	Text entry	The name of the external reference. This is a required field.
ReferenceScope	Dropdown menu	Determines the scope of the external reference within the current library object. If the library object contains subobjects, these will appear as options on the list, and the external reference can be scoped to them. If the library object has no subobjects, **Object** will be the only option in the list.
Data Type	Dropdown menu	The data type of the external reference. Options are: • String • Integer • Real • Boolean
Value	Text entry	The specific location of the external reference once the library object has been instantiated. By default, it uses the predefined parameter. {ObjectName}. Displays in the format: `{ObjectName}.Tag Name` The value can be entered manually or generated by an expression. To use a calculated value for this field, click the ellipsis (...) button next to the field to open the Expression Builder on page 59.
Description	Text entry	The description of the external reference that appears when it is highlighted in Application Code Manager. The value can be entered manually or generated by an expression. To use a calculated value for this field, click the ellipsis (...) button next to the field to open the **Expression Builder**.

Linked Libraries tab

Linked library objects contain elements that are shared with other libraries. A library link creates a relationship or dependency between two libraries. Library links can be bi-directional and a library can link to multiple other libraries.

Use the Linked Libraries tab in the decorator panel to add linked libraries and parameter links. Library links connect library instances while parameter links allow for information to be consumed between the libraries.

The **Linked Libraries** tab consists of two panes:

- **Linked Libraries**

  Linked libraries, displayed in the top pane of the **Linked Libraries** tab, specify the relationships between library objects. The links are applied as decoration to Logix code in place of parameters.

  Linked libraries can be configured to share dependencies on Logix content. For instance an AOI or UDT definition can be defined in a single library object, then linked to multiple library objects.

  Linked libraries assist the Application Code Manager user when configuring an object for instantiation. For instance, a regulatory control valve typically needs an analog input for instantiation.

  The **Linked Libraries** toolbar can be used to perform these actions:

Action	Description
Add New	Displays the **Add New Linked Library** dialog.
Edit	Displays the **Edit Linked Library** dialog.
Delete	Deletes the selected object.
Show References	Displays the **References** dialog. Libraries linked to the selected object are listed.

- **Parameter Links**

  Parameters, displayed in the bottom pane of the **Linked Libraries** tab, can also be shared with linked libraries.

  Parameter links are used to read or write the parameter values between a library object and a linked library object. Parameter links display the direction of the flow of information.

  The **Parameter Links** toolbar can be used to perform these actions:

Action	Description
Add new parameter link	Displays the **Parameter browser** dialog.
Delete selected	Deletes the selected parameter.

Within the **Parameter Links** list, there are two additional controls.

Control	Description
Ellipsis (...) button	Displays the **Parameter browser** dialog.
Data Flow arrow (—>) button	Click to toggle the data flow direction between the linked parameters.

Add a linked library

A linked library is a library object containing elements that are shared with other libraries. Linked libraries specify the relationships between library objects. The links are applied as decoration to Logix code in place of parameters.

To add a linked library

1. In the decorator panel, either:
   - Click  on the **Linked Libraries** tab toolbar.
   - In the **Linked Libraries** tab, right-click in the top grid and then click **Add New Link**.
     The Add New / Edit Linked Library on page 95 window opens. By default libraries resident in the ACD file are displayed in the list.
2. In **Select the library you want to link to**, click a library object. The object parameters are displayed in the right panel.

    **Tip:** If no libraries are displayed in the list, then it means that the ACD file only contains the one library.
   To link to libraries contained in the ACM database, click **Connect to Database**.
3. In **Link Name**, type a name and then press **Enter**.
4. Populate other fields as needed.
5. Click **OK**.
   The link appears in the list on the **Linked Libraries** tab.

Add New / Edit Linked Library

How do I open Add New / Edit Linked Library?

1. In the **Library Objects** column, select a library.
2. In the decorator panel, select the **Linked Libraries** tab.
3. Right-click in the top list of linked libraries, then select **Add New Link**.

Use **Add New / Edit Linked Library** to define the parameters of the linked library. The window contains two panes, a library pane and a parameter pane. The settings in the parameter pane are functionally grouped.

This table describes the fields in the **Add New / Edit Linked Library** window.

Name	Field Type	Description
Select the library you want to link to:		
Filter	Text Entry	Apply a filter to find the desired selected library object.
Connect to Database	Button	Connect to a different database to access additional libraries.
Libraries in this ACD file	List	Displays selected library objects in the same ACD as the chosen object. Click to select a library object.
Server *****	List	Displays selected library objects from other databases or locations. Click to select a library object.
01 Link Details		
Link Name	Text Entry	Mandatory field. Enter the name of the link to be displayed.

Name	Field Type	Description
Mandatory	Dropdown menu	True/False option. Set this field to True if the linked library must be resolved before creating, updating, or importing an object.
Include Condition	Expression	Specifies the expression used to determine whether a linked library is included in Application Code Manager. If the expression is True, the library is included. If the expression is False, the library is not included. Selecting the ellipsis button **(...)** opens the Expression Builder on page 59 to write, validate, and test the expression. Only editable when **Mandatory** is set to **False**. Default value is **Always**.
02 Library Details		
Catalog Number	Text Entry	Editable field that displays the catalog number of the selected library object. The field auto-populates when a library object is selected. Use an underscore (_) as a wildcard for a single character, or a percentage sign (%) as a wildcard for multiple characters.
Family	Text Entry	Editable field that displays the family of the selected library object. The field auto-populates when a value is assigned in the selected library object. Use an underscore (_) as a wildcard for a single character, or a percentage sign (%) as a wildcard for multiple characters.
Solution	Text Entry	Editable field that displays the solution of the selected library. The field auto-populates when a value is assigned in the selected library object. Use an underscore (_) as a wildcard for a single character, or a percentage sign (%) as a wildcard for multiple characters.
Library Type	Text Entry	Editable field that displays the library type of the selected library. The field auto-populates when a value is assigned in the selected library object.

Name	Field Type	Description
		Use an underscore (_) as a wildcard for a single character, or a percentage sign (%) as a wildcard for multiple characters.
Category	Text Entry	Editable field that displays the category of the selected library. The field auto-populates when a value is assigned in the selected library object. Use an underscore (_) as a wildcard for a single character, or a percentage sign (%) as a wildcard for multiple characters.
Revision	Auto-populate	A rule that defines what version of a library can be linked to. Enter a number if the library version must exactly match, or use the greater than, less than, and equal to signs to specify version numbers greater than, less than or equal to the number specified. For example, `>=2` requires that the linked library version be equal to 2 or greater.
03 Instance Details		
Instance Name	Text Entry	Enter an instance name for the linked library.
Instance Description	Text Entry	Enter a description of the instance.

Add parameter links

Parameter links are used to read or write the parameter values between a library object and linked library object. Parameter links display the direction of the flow of information.

To add parameter links

1. In the decorator pane, click the **Linked Libraries** tab.
2. Click the linked library in which to add the parameter links. The **Add new parameter link** button in the **Parameter Links** toolbar becomes active.
3. In the **Parameter Links** pane, either:
   - Click the **Add new parameter link** button.
   - Right-click an empty row and click **Add**.

     The Parameters browser on page 98 window opens.
4. Select a parameter and click **OK**.

   The new parameter will appear in the **Parameter Links** pane.

Parameters browser

How do I open the Parameters browser window?

1. In the decorator pane, click the **Linked Libraries** tab.
2. Click the linked library in which to add the parameter links. The **Add new parameter link** button in the **Parameter Links** toolbar becomes active.
3. In the **Parameter Links** pane, either:
   - Click the **Add new parameter link** button.
   - Right-click an empty row and click **Add**.

Use the **Parameters browser** window to define the parameter links for a linked library.

The **Parameters browser** window includes these items:

Item	Desription
Library Catalog Number	The library object name, which appears when the library object is registered in the Application Code Manager application.
Library Location	Where the library is located.
FindPartial/FindPrefix	Searches the library for parameters that include the text string typed in the adjacent box. Two methods of search are provided: • FindPartial. Searches each field for any instances of the specified string. • FindPrefix. Searches the beginning of each field in the object for the string.
Parameter list	The parameters available in the library object. If a search string is specified, the parameters are limited to those that meet the search criteria.

Interfaces tab

An interface is an object in a library that provides a link that another library can consume. The link references an object in a library such as a task, program, routine or tag. There are two types of interfaces, input and output.

Output interfaces are connected to input interfaces. Output interfaces allow variant tag member structures from different libraries to be mapped to a common interface member name. Output interfaces are used with linked libraries. Output interfaces allow varied tag member structures from different libraries to be mapped to a common interface member name.

Input interfaces are used within substitutions without needing to know the provider's tag structure. Input interfaces act as placeholder substitutions. Input interfaces are typically used with a linked library, but can be configured as unassigned.

The **Interfaces** tab displays the interfaces linked to the selected object. Interfaces can be created, edited, and deleted at this tab. These commands are executed by clicking the icons or right-clicking the object and selecting the command.

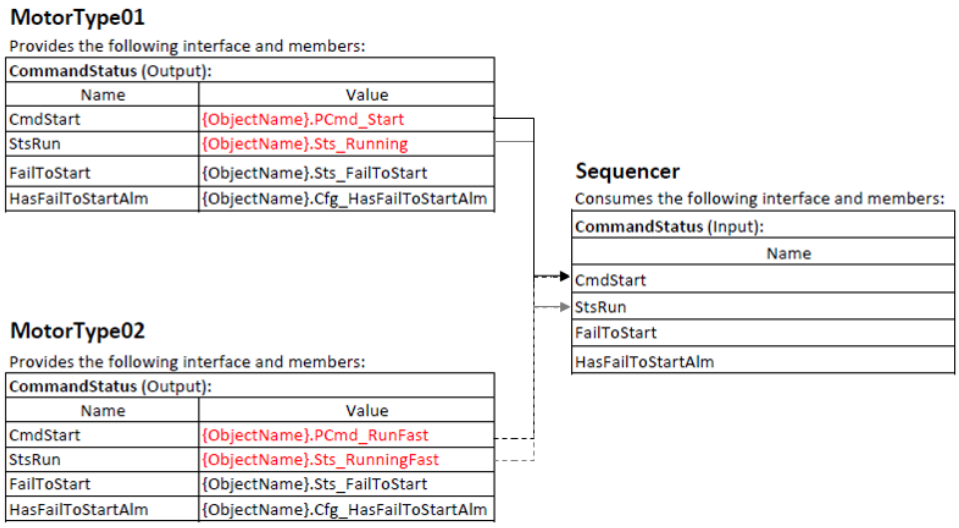
Name	Action
Add New	Displays the Add New Interface dialog.

Name	Action
Edit	Displays the Edit Interface dialog.
Delete	Deletes the selected object.
Show references	Displays the References dialog. Libraries linked to the selected object are listed.

Standardize data structures

An interface is an object in a library that provides a link to a reference object in a library (such as a task, program, routine or tag) that another library would be able to consume.

Consider the following example: MotorType01 and MotorType02 both have a set of output tags, however the tag data structure is different, which could result in invalid code when the ACM controller is generated. To solve this problem use Library Designer to define interfaces that link the correct tag data structures through a common name. The following diagram illustrates this example.



Add an interface

There are two types of interfaces, input and output. Add an interface as needed.

To add an interface

1. Select the **Interfaces** tab.
2. In the **Interfaces** tab, select **Add New**. The Add New Interface on page 101 window opens.
3. In **Name**, type a name for the interface.

4. In **Usage**, select either:
   - **Output**
     a. On the **Interface Members** toolbar, click **Add member** . The **Tags** window opens.
     b. Choose a task, program, routine or tag to add as an interface member.
     c. Click **OK** to return to the **Add New Interface** window.
   - **Input**
     a. On the **Interface Members** toolbar, click **Import members** . The **Select Library Interface** window opens.
     b. Select a library with a configured output interface. Libraries that have available output interfaces are displayed on the right side of the **Select Library Interface** window.
     c. Select available interface members from the **Interface Members** grid in the lower right pane.
     d. Click **OK** to return to the **Add New Interface** window.
5. In **Interface Description**, type a caption to describe the interface.
6. (for an Input interface) In **Keying**, select either:
   - **ExactMatch**. The keys defined for the interface and its members in the project must match the ones defined on the target system for the input to be consumed. The interface and its members can't be set manually.
   - **Disabled**. Keying attributes are not evaluated when determining whether to consume data from the interface. Member references can be selected manually, however only members with the same data type can be selected.
7. In **Key Id**, enter an identifier that allows for automatic linking of interface members. For automatic linking to occur the KeyId of the input interface must match the KeyId of the output interface.
8. In **Revision**, enter the revision number of the interface link.
9. In **Link Name**, type the name of the library associated with this interface.
10. Click **OK**. The **OK** button is not available until the fields are populated.

Delete or edit interfaces

Delete or edit interfaces as needed.

To delete or edit interfaces

1. Select the interface to change.
   a. To delete an interface, on the **Interfaces** toolbar, click the **Delete** button or right-click the interface and select **Delete**.
   b. To edit an Interface, on the **Interfaces** toolbar, click the **Edit** button. The **Edit Interface** window opens. Change the field values where applicable. The **Revision** field must be updated.
2. (optional) To import additional **Interface Members**, select the down arrow.
   The **Select Library Interface** window opens.
   a. Connect to a new database by clicking the **Connect to Database** button.
   b. Select the file to import.
   c. Select **OK** when the desired files are added.
3. Click **OK**.

Add interface members

Add interface members as necessary.

To add interface members

1. To add interface members, select an interface, either:
   a. Click the **Add new member** button located next to **Interface Members**.
   b. Right-click the first field and select **Add New**.
2. The **Tags** window opens.
3. Select the tag that will be added and click **OK**.

Add New/Edit Interface window

How do I open the Add New/Edit Interface window?

1. In the decorator pane, click the **Interfaces** tab.
2. On the **Interfaces** toolbar, either:
   - Click the **Add new interface** button. The **Add New Interface** window opens.
   - Click the **Edit** button. The **Edit Interface** window opens.

Use the **Add New Interface** window to define the parameters of the interface. Use the **Edit Interface** to modify the parameters of the interface. Both windows share the layout and fields. The window contains two panes, a parameter pane and an **Interface Members** pane. The settings in the parameter pane are functionally grouped.

This table describes the fields in the windows.

Name	Field Type	Description
Categorized	Button	Click to change the parameter display to categorize parameters by function.
Alphabetical	Button	Click to change the parameter display to parameters in an alphabetized order.
Property Pages	Button	Click to display the property pages when they are hidden.
01 Interface Detail		
Name	Text entry	The name of the interface. This field is required.
Usage	Dropdown menu	Choose whether the interface in an input interface consumed by the library or output interface produced by the library.
Interface Description	Text entry	(optional) Describe the usage of the interface.
02 Interface Connection		
Key Id	Text entry	The identifier for the key assigned to this interface.
Revision	Text entry	The revision number of this interface.
Interface Members		
Add Member	Button	Click to open the **Tags** window and choose an **I/O Configuration** tag to assign to this interface.

Name	Field Type	Description
Edit	Button	When an interface member is selected, click to open the **Interface Member** window and modify its parameters.
Delete Member	Button	When an interface member is selected, click to delete the interface member from the list.
Import Members	Button	Not available.
List	Grid	List of all the interface members configured for this interface. Displays the name of the member, the type of data it contains, the value of the member and any descriptive text.

Edit Interface window

How do I open the Edit Interface window?

1. In the decorator pane, click the **Interfaces** tab.
2. On the **Interfaces** toolbar, click the **Edit** button.

Use the **Edit Interface** window to modify the parameters of the interface. The window contains two panes, a parameter pane and an **Interface Members** pane. By default, the settings in the parameter pane are categorized by function.

This table describes the items in the **Edit Interface** window.

Name	Field Type	Description
Categorized	Button	Click to change the parameter display to categorize parameters by function.
Alphabetical	Button	Click to change the parameter display to parameters in an alphabetized order.
Property Pages	Button	Click to display the property pages when they are hidden.
01 Interface Details		
Name	Text entry	The name of the interface. This field is required.
Usage	Dropdown menu	Choose whether the interface in an input interface consumed by the library or output interface produced by the library.
Interface Description	Text entry	(optional) Describe the usage of the interface.
02 Interface Connection		
Key Id	Text entry	The identifier for the key assigned to this interface.

Name	Field Type	Description
Revision	Text entry	The revision number of this interface.
Interface Members		
Add member	Button	Click to open the **Tags** window and choose an **I/O Configuration** tag to assign to this interface.
Edit	Button	When an interface member is selected, click to open the Interface Member window and modify its parameters.
Delete member	Button	When an interface member is selected, click to delete the interface member from the list.
Import members	Button	Not available.
List	Grid	List of all the interface members configured for this interface. Displays the name of the member, the type of data it contains, the value of the member and any descriptive text.

Interface Member window

How do I open the Interface Member window?

1. In the decorator pane, click the **Interfaces** tab.
2. In the **Interface Members** section, either:
   - Click the interface member, then on the **Interface Members** toolbar, click **Edit**  .
   - Right-click the interface member and then click **Edit**.
   - Double-click the interface member.

Use the **Interface Member** window to edit the properties of an interface member. Changes that are made apply to other libraries that use the same interface.

Property	Description
Name	The name of the interface member.
Data Type	The data type of the interface member. This field is read-only.
Value	The value of the interface member. This value can be directly entered or generated by an expression. Click the value and then click the ellipsis (...) button to open the Expression Builder on page 59 window.
Description	A description of the interface member.

Tags window

How do I open the Tags window?

1. In the decorator pane, click the **Interfaces** tab.
2. Select an interface, either:
   - Click the **Add new member** button located next to **Interface Members**.
   - Right-click the first field and select **Add New**.

Use the **Tags** window to add tags to an interface member.

The **Tags** window includes these items:

Item	Description
Tag tree	Navigate the tree control to find the controller tag, task tag, or input/output tag to assign as an interface member.
Tag list	Display tags selected by name and data type. The listed tags are added to the interface member by clicking **OK**.

Library object properties

Library object elements support different decorative properties. The decorator panel is divided into two sections:

- Attributes section. Displays the configurable attributes for a library object element.
- Parameters section. Displays the customizable properties (decorations) available for a library object element and any of its extended parameters.

Use the configurable items in the decorator panel to turn static instances of controller tags, local tags, and tag members into parameters or external references. Adding a tag as a parameter opens the tag to values set by the user, or by calculations or references set after the library object has been added to an Application Code Manager project. Adding a tag as an external reference makes the value of the tag available to other library objects after the library object has been added to an Application Code Manager Project.

This table describes the decorative attribute items available for the different library object elements.

Name	Field Type	Supported item	Description
Name	Text entry	Tag Task Program Routine Directive Function Block sheet Sequential function chart Structured text chart Structured text chart line Motion group Motion group axis Add-On Instruction Data type	The name applied to the item when the library object is instantiated. If substitutions appear in the original item name, these are applied by default. This field cannot be edited for Function Block sheet or data type items.

Name	Field Type	Supported item	Description
		Module	
Description	Text entry	Tag Task Program Routine Function Block sheet Sequential function chart Structured text chart Structured text chart line Motion group Motion group axis Add-On Instruction Data type	The description applied to the item when the library object is instantiated. If substitutions appear in the original item description, these are applied by default. This field cannot be edited for data type items.
Description Language	Dropdown menu	Tag Task Program Routine Function Block sheet Sequential function chart Structured text chart Motion group Add-On Instruction Data type	The language used for the description. The default is English. This field cannot be edited for data type items.
Library Object	Read-only	Tag Task Program Routine Function Block sheet Sequential function chart Structured text chart Motion group Add-On Instruction Data type Module	The library object that contains the item.
Logix Path	Read-only link	Tag Task Program Routine Function Block sheet Sequential function chart Structured text chart Motion group Add-On Instruction	A link to the applicable item for the element in Studio 5000 Logix Designer. Click the link to open the screen. Close Library Designer to access Studio 5000 Logix Designer.

Name	Field Type	Supported item	Description
		Data type	
Configure Instantiation Rules: Condition	Text entry	Tag Task Program Routine Rung Function Block sheet Function Block diagram Sequential function chart Sequential function chart element Structured text chart Structured text chart line Motion group Motion group axis Add-On Instruction Data type	Sets the condition under which the current item is instantiated. The default is **Always**. A condition can be entered manually or generated by an expression.
Configure Instantiation Rules: Usage	Dropdown menu	Tag Task Program Routine Rung Function Block sheet Function Block diagram Sequential function chart Sequential function chart element Structured text chart Structured text chart line Motion group Motion group axis	The number of times the item is instantiated. Default options are: • One per Object • Include Once If the containing library object has subobjects, there will also be an option "Once per sub object [SubObject name]" for each subobject. The default value is "One per Object".
Exclude Base Library Substitutions	Checkbox	Tag Task Program Routine Rung Function Block sheet Function Block diagram Sequential function chart Sequential function chart element Structured text chart Structured text chart line Motion group	If selected, allows substitutions added to the base library object for the current library object to be overridden.

Name	Field Type	Supported item	Description
		Motion group axis Add-On Instruction Data type	
Exclude Library Substitutions	Checkbox	Tag Task Program Routine Rung Function Block sheet Function Block diagram Sequential function chart Sequential function chart element Structured text chart Structured text chart line Motion group Motion group axis Add-On Instruction Data type	If selected, allows substitutions added to the current library object to be overridden.
Container Mode	Checkbox	Task Program Routine Rung	When this command is active, all program tags and routines are added to the library object automatically. If it is inactive, the user must manually add the content to the library object.
Catalog No	Read-only	Module	The catalog number of the item.
Major Version	Read-only	Module	The major version for the item.
Minor Version	Read-only	Module	The minor version for the item.
ParentModule	Text entry	Module	The name that is applied to the module when the library object is instantiated. A predefined parameter is assigned by default.

This table describes the actions available from the parameters section of the decoration pane. Actions that are not supported by the current element appear will be unavailable. To see the available actions, right-click the element.

Action	Description
Substitutions	Opens the Substitution Builder.
Select By > Included in Library	Selects all elements that are included in a library.
Select By > Excluded from Library	Selects all elements that are excluded from libraries.

Action	Description
Select By > Usage > One per Object	Selects all elements where the Usage field has been set to "Include Once per Object".
Select By > Usage > One per SubObject	Selects all elements where the Usage field has been set to "Include Once per SubObject".
Select By > Usage > Include Once	Selects all elements where the Usage field has been set to "Include Once".
Select By > Include Condition	Selects all elements where the Condition field matches the option selected in the submenu. The submenu displays all conditional inclusions rules for the current diagram.
Select Same > Usage	Selects additional elements that match the Usage field for the currently selected element.
Select Same > Include Condition	Selects additional elements that match the Condition field for the currently selected element.
Select Same > Both of the Above	Selects additional elements that match both the Usage field and the Condition field for the currently selected element.
Select Same > Tag	Selects additional elements that reference the same Tag as the currently selected element.
Select All	Selects all elements.
Remove	Deactivates the selected elements.
Add	Not Available.
Undo	Undoes the last action.
Redo	Redoes the last action.
Extended Properties	Displays any extended properties set in Studio 5000 Logix Designer that can be modified. Select the ellipsis button **(...)** to open the **Expression Builder**.

Some elements, such as Add-On Instructions and Data Types, do not have configurable parameters in the bottom half of the decorator panel.

Change the name or description

Names are structured when the elements are added in Studio 5000 Logix Designer to allow substitutions to be added once, to the library object, and then extend consistently to all elements contained within the library object.

For example, a motor library object with an identifying string of MX001 in the library object name, and with the predefined parameter {ObjectName} applied as a substitution, might have tags named MX001_Permissives, MX001_Interlock, and MX001_IOFault.

Substitutions can also be added to the project library object or controller library object. These will extend throughout the project hierarchy and will take precedence over substitutions added to the library object, unless the **Base Library** field for the library object has been set to **None**.

Use the decorator panel attributes to override default substitution for individual elements. Use the Substitution Builder on page 66 to create substitutions that are specific to the current element.

To change the element name or description

1. In the **Library Objects** pane, click the item whose element name or description should be changed upon instantiation.
2. In the attributes section of the decoration pane, click the ellipsis (...) button next to **Name** or **Description** to open the **Substitution Builder**.
3. Review the substitutions listed in the **Element Substitutions** window to trace the origin of the current substitution, if one exists.

   Tip: Check **Show existing substitutions which do NOT affect this element** to see all current substitutions. This shows potential conflicts and unexpected replacements for new substitutions.

4. Select a scope in the **Scope** field.
5. In the **Search for** field, enter an initial text string to be replaced.
6. In the **Replace with** field, enter a replacement string.
7. Select a scope for the text search.
8. Click **Apply** to test the substitution. The new substitution appears in the **Element Substitutions** window.
9. Click **OK** to exit the **Substitution Builder**.
10. (optional) In the **Decorator Panel**, check the **Exclude Base Library Substitutions** or **Exclude Library Substitutions** checkbox so the new substitution takes precedence.

Add a tag as a parameter

Adding a tag or tag member as a parameter makes the tag value accessible to the Application Code Manager application.

These tags are supported:

- Tags
- Motion group tags
- Module tags

To add a tag as a parameter

1. In the **Library Objects** pane, select an object that contains a supported tag set.
2. Navigate through the objects until the **Tags** tab is visible in the decorator pane.
3. In the decorator pane, right-click the tag name and select **Add as Parameter**. The Add New Parameter on page 73 window opens.
4. Edit the parameter. Some fields will be filled with default values based on the tag settings.
5. Click **OK** to add the parameter. The new parameter is added to the **Parameters** tab for the module library object. The token for the new parameter appears in the **Value Expression** column for the tag, indicating that the tag now accepts values from the parameter. Decoration tokens are displayed using blue text.

Add a tag as an external reference

Adding a tag or tag member as an external reference makes the tag value accessible to reference-type parameters in Application Code Manager.

These tags are supported:

- Tags
- Motion group tags
- Module tags

To add a tag as an external reference

1. In the **Library Objects** pane, select an object that contains a supported tag set.
2. Navigate through the objects until the **Tags** tab is visible in the decorator pane.
3. In the decorator pane, right-click the tag name and select **Add External Reference**. The **References Builder** window opens.

   Tip: All fields other than the **Description** field are filled in by default. In a typical application, the default values should not be changed.

4. Click **OK** to add the external reference. The new external reference is added to the **External References** tab for the library object.

Apply decoration to tag values

The expression token appears in the **Value Expression** column for the tag, indicating that the tag uses the expression to generate values. Decoration tokens are displayed using blue text.

After decorating the tag value, check the **OPC Tag Export** option to make the tag or tag member available to the ACM Tag Export functionality. The ACM Tag Export functionality is used to create an excel file that can be used to write values to an online Logix controller via OPC.

To apply decoration to a tag value

1. In the **Library Objects** pane, select the item with tags to which decoration will be applied.
2. In the decorator panel, expand the tag tree until the tag value is visible.
3. Right-click the tag listing, then select **Add\Edit Expression**. The Expression Builder on page 59 window opens.
4. Edit the expression using the **Expression Builder**.
5. Click **OK**.

Decorate a tag alarm condition

Decorate a tag-based alarm condition to change and substitute values for a tag's configured alarm.

When published, all associated alarm definitions are included in the library if:

- The related UDT or AOI is included in the library.
- A system or module data type tag with alarm definitions is included in the library.

To apply decoration to a tag alarm condition:

1. In **Library Objects**, select a controller or local tag that has an alarm.
2. In the decorator panel, select Alarms on page 115.
3. Double-click an alarm in the list, edit the properties, then select **OK**.

Add parameter connection to a tag

A program parameter is an argument that is exposed for external access by a program. Pre-define parameter connections to achieve data sharing between programs. Parameter connections are displayed in Library Designer and can be added to local tags. All parameter connections added to a tag member are available at the tag level for ease of use. The conditional expression for Parameter connections is disabled by default.

There are four types of program parameters that can be used with Parameter Connections:

- Input
- Output
- InOut
- Public

For more information about parameter connections, see *Logix 5000 Controllers Program Parameters* in Rockwell Automation Literature Library.

To add a parameter connection to a local tag

1. In **Library Objects**, select a tag that is part of a parameter connection.
2. In the decorator panel, select **Tags**, and then select a tag from the list.
3. Select **Extended Properties > Parameter Connections**.
4. In the **Condition** column, click the ellipsis (...) button to add a condition in the Expression Builder on page 59 dialog box.
5. In the **Connection EndPoint** column, click the ellipsis (...) button to add a connection local tag substitution in the Substitution Builder on page 66 dialog box.
6. Click **OK**.

IMPORTANT: Parameter connections that have been included in a library may become invalid due to certain operations to the EndPoint tags in **Logix Designer**. The following is a basic list of findings related to some actions that were applied.

Action	Unique Identifier (UID)	Customer Object Attributes (COA)
Rename any of the EndPoint Tags	No change	Persisted
Change connection to another EndPoint Tag	No change	Deleted
Change the Usage of an EndPoint Tag	No change	Persisted
Move an EndPoint Tag to another Program	New UID	Persisted
Change the DataType of both EndPoint Tags	No change	Persisted
Copy Paste an EndPoint Tag	New UID	Persisted

Parameterize tag properties

Extended properties set in Logix Designer can be modified in Library Designer to show dynamic, expression-based values. A tag with expression-based extended properties is "parameterized".

Parameterize the extended property of a tag to change the value of the extended property based on the value of an expression.

Pass-through values, such as extended properties inherited from other objects, cannot be parameterized in Library Designer. Pass-through values are not displayed in Extended Properties on page 115.

Pass-through values that are overridden in Logix Designer can be parameterized. Overridden pass-through values are displayed in **Extended Properties**.

To parameterize tag extended properties

1. In **Library Objects**, select a parameter or tag.
2. (optional) In the decorator panel, select the **Description Language** to parameterize an extended property for a specific language.
3. Select **Tags**, then select a tag from the list.
4. Select or expand **Extended Properties**.
5. In **Extended Properties**, select an extended property, then select the ellipsis button **(...)** to open the Expression Builder.
6. Use Expression Builder on page 59 to write, validate, and test the expression, then click **OK**.

Exclude inherited substitutions

Substitutions can be inherited by an element from the containing library object, or from a library object higher in the library object hierarchy (base library). Remove inherited substitutions that are not applicable to an application.

Library object elements supported:

- Tags
- Tasks
- Programs
- Routines
- Ladder logic diagram elements
- Function block diagram elements
- Sequential function chart elements
- Structured text chart elements
- Motion groups
- Add-On Instructions
- Data types
- Modules

To exclude inherited substitutions

1. In the **Library Objects** pane, select a supported library object element. The decorator panel displays the element attributes on the top half of the pane and the element parameters on the bottom half.
2. In the element attributes section of the decorator panel, select either:
   - **Exclude Library Substitutions**: Remove substitutions that have been inherited from the library object. Reverts to the original value for the element.
   - **Exclude Base Library Substitutions**: Remove substitutions inherited from the base library object for the current library object.

   Tip:
   - If substitutions have also been added to the library object, the field switches to the library object substitution.
   - If no substitutions have been added to the library object, the field reverts to the original value for the element.
3. Select **Apply**, then **OK**.

Set a rule for instantiation

By default, library object items are set to instantiate under all conditions, and to instantiate once every time the library object is added to an Application Code Manager project. If needed configure an expression that determines when the element is instantiated and define its usage.

The following library object items support instantiation rules:

- Digital alarms
- Elements
- Function block elements
- Motion groups
- Motion group axis
- Programs
- Routines
- Sequential function chart elements
- Structured text chart lines
- Structured text elements
- Tags
- Tasks

To set a rule for instantiation

1. In the **Library Objects** pane, select a supported library object item. The decorator panel displays the attributes on the top half of the pane and the parameters on the bottom half.
2. In the attributes section of the decorator panel, under **Configure Instantiation Rules**, in the **Condition** box, enter a value or click the ellipsis (...) button to enter an expression using the Expression Builder on page 59.

3. In the **Usage**box, select either:
   - **One per object**: Instantiates once every time a library object is added in the project.
   - **Include Once**: Limits the element to a single instance in the project.

Tip: To revert to the default condition, click **Revert** (red X) to the right of the **Condition** box.

Apply a substitution to an element

Substitutions can be applied to the following elements to replace them when the program is instantiated:

- Directives
- Function block diagram elements
- Structured text chart lines
- Sequential function chart elements
- Rungs

Tip: Rung numbers are applied sequentially and cannot be changed. If inherited substitutions have been excluded, substitutions applied to the rung will extend to directives contained by the rung.

Ladder diagram elements include a resizing slider in the decorator pane to help view more of the large diagrams.

To apply a substitution to an element

1. In the **Library Objects** pane, select the library object, then right-click the element in the parameter section of the decorator pane and select **Substitutions**.

Tip: The **Substitutions** command does not respond if more than one item is selected.

2. In the **Substitution Builder** window, review the substitutions listed in the **Element Substitutions** box to trace the origin of the current substitution, if one exists.

Tip: Check **Show existing substitutions which do NOT affect this element** to see all current substitutions. This shows potential conflicts and unexpected replacements for new substitutions.

3. Select a scope in the **Scope** field.
4. In the **Search for** field, enter an initial text string to be replaced.
5. In the **Replace with** field, enter a replacement string.
6. Select a scope for the text search. **Operand replacement** limits the search to the tokens for operands, so it should only be selected if this is where the substitution should take place. **Text replacement** limits the search to text strings not in tokens.
7. Click **Apply** to test the substitution. The new substitution appears in the **Element Substitutions** box.
8. Click **OK** to exit the Substitution Builder on page 66.
9. (optional) If supported by the element, check the **Exclude Base Library Substitutions** or **Exclude Library Substitutions** checkbox so the new substitution takes precedence.

Activate a new element

When modifications are made in Studio 5000 Logix Designer to an element that has been added to a library object in Library Designer, the element must be updated in Library Designer to include the modifications. New elements will appear in the parameters section of the decoration pane for the element, but they must be activated so that they are recognized by the Library Designer.

This requirement applies to these items:

- Function block diagrams
- Sequential function charts

Tip: The connection wires that have a gray color must be activated.

To activate a new element

- Right-click the connection wire, then select **Add**. The wire color on the display changes from gray to black.

IMPORTANT: Default substitutions are inherited automatically. Overrides to the default substitution, and all other decoration, must be applied manually to new elements of an existing library object.

Extended Properties

How do I open Extended Properties?

1. In **Library Objects**, select a parameter or tag.
2. In the decorator panel, select **Tags**, then select a tag from the list.
3. Select or expand **Extended Properties**.

Use **Extended Properties** to view, edit, or parameterize the extended properties of a tag.

Pass-through values, such as extended properties inherited from other objects, cannot be parameterized in Library Designer. Pass-through values are not displayed in **Extended Properties**.

Pass-through values that are overridden in Studio 5000 Logix Designer can be parameterized. Overridden pass-through values are displayed in **Extended Properties**.

Alarms

How do I open Alarms?

1. In **Library Objects**, select a controller or local tag.
2. In the decorator panel, select **Alarms**.

Use **Alarms** to view and edit configured alarms for a selected tag.

Column	Description	Read Only / Allows Substitution
01 General		
Name	The alarm definition name. Can contain up to 40 characters.	Read-only.

Column	Description	Read Only / Allows Substitution
Used	Marks the alarm as active and ready for evaluation.	Read-only if the alarm has an **Alarm Definition** attribute and the Definition value is set to TRUE.
Input	The alarm's input that can either be a tag or parameter.	Read-only.
02 Condition		
Type	The type of alarm, either digital or analog.	Read-only.
Expression	Determines whether the alarm is triggered when the Input is true or false. For Boolean input, when Expression is set to =1, the alarm is triggered when the input is true. When Expression is set to =0, the alarm is triggered when the input is false. For analog input, the expression can be set to >, >=, =, <, <=, or <>.	Read-only.
Limit	The value that the condition is evaluated against.	Allows substitution.
Target Tag	The target tag for the alarm.	Allows substitution if the alarm does not have an **Alarm Definition** attribute. Only available if the type is **DEV_HI** or **DEV_LO**.
03 Data		
On Delay	Duration in milliseconds from the time that the alarm condition occurs until the alarm notification is set.	Allows substitution.
Off Delay	Duration in milliseconds from the time that the alarm condition ends until the alarm becomes inactive.	Allows substitution.
Deadband	A value added to the alarm limit to determine when the alarm condition ends.	Allows substitution.
Severity	Displays the alarm severity, where 1 is the least severe and 1000 is the most severe. The severity does not affect how the controller processes the alarm, but it can be used by devices that monitor the alarm. The default severity is 500. By default, severity ranges are mapped to the following priorities: • 1...250 are **Low** priority. • 251...500 are **Medium** priority.	Allows substitution.

Column	Description	Read Only / Allows Substitution
	• 501...750 are **High** priority. • 751...1000 are **Urgent** priority.	
Shelve Duration	The number of minutes the controller postpones processing the alarm.	Allows substitution.
Message	The message displayed on the alarm's monitoring device.	Allows substitution.
Tag1	Contains data pertinent to the alarm to be transmitted to subscribers.	Allows substitution. Only available if the alarm has an **AssocTag1** attribute.
Tag2	Contains data pertinent to the alarm to be transmitted to subscribers.	Allows substitution. Only available if the alarm has an **AssocTag2** attribute.
Tag3	Contains data pertinent to the alarm to be transmitted to subscribers.	Allows substitution. Only available if the alarm has an **AssocTag3** attribute.
Tag4	Contains data pertinent to the alarm to be transmitted to subscribers.	Allows substitution. Only available if the alarm has an **AssocTag4** attribute.
04 Class / Group		
FTGroup	Tags the alarm with a group name to help classify and sort alarms.	Allows substitution if the alarm does not have an **Alarm Definition** attribute.
Class	Use classes to group related alarms.	Allows substitution if the alarm does not have an **Alarm Definition** attribute.
05 Advanced		
FactoryTalk View Command	A FactoryTalk View command that can be run on the operator station when the alarm notification appears.	Allows substitution if the alarm does not have an **Alarm Definition** attribute.
Latched	Latches the alarm. A latched alarm remains active after the alarm condition becomes false, until a reset command is received. The reset command is ignored until the alarm condition is false.	Allows substitution.
Acknowledgment Required	Sets if acknowledgment is required for the alarm.	Allows substitution.
Include in Alarm Set roll-up count	Sets if this alarm is included in the rollup counters of an alarm set containing this alarm.	Allows substitution.
Include in Alarm Set operations	Sets if this alarm is included when an operation is performed on an alarm set containing this alarm.	Allows substitution.

Edit Alarm

How do I open Edit Alarm?

1. In **Library Objects**, select a controller or local tag that has an alarm.
2. In the decorator panel, select **Alarms**.
3. Double-click an alarm in the list, edit the properties, then select **OK**.

Use **Edit Alarm** to apply decorations to a tag-based alarm condition. Properties that can be edited appear black. Properties that cannot be edited appear gray.

Click the ellipsis button **(...)** next to a property to open the Expression Builder on page 59 or Substitution Builder on page 66 for the property.

Library Object Manager

Library Object Manager overview

Use Library Object Manager to publish library objects to the ACM database or to a file in HSL4 format. HSL4 files can be distributed individually or as part of a repository.

After publishing a library object, use Library Object Manager to add HMI displays (FactoryTalk® View Site Edition (SE) or FactoryTalk® View Machine Edition (ME)) and Historian (FactoryTalk® Historian Site Edition) components to support requirements. Features added in the Library Object Manager application save to the individual HSL4 file or database entry for the Library object and do not save to the original ACD file.

Each library object file saved from the Library Object Manager application is classified within a four-level hierarchy:

Solution -> Library Type -> Category -> Catalog Number

Each library object file must have a distinct version number per solution. Just as the same Logix object can be used to create one or many library objects within the Library Designer, the same Library object can be used to create one or many distinct library object files within the Library Object Manager application.

Library objects can be quickly distributed, then registered into and configured for multiple projects in multiple locations. Library objects are available to any project that requires the functionality the library object provides.

Using Library Object Manager projects can be built and executed without high-level programming support.

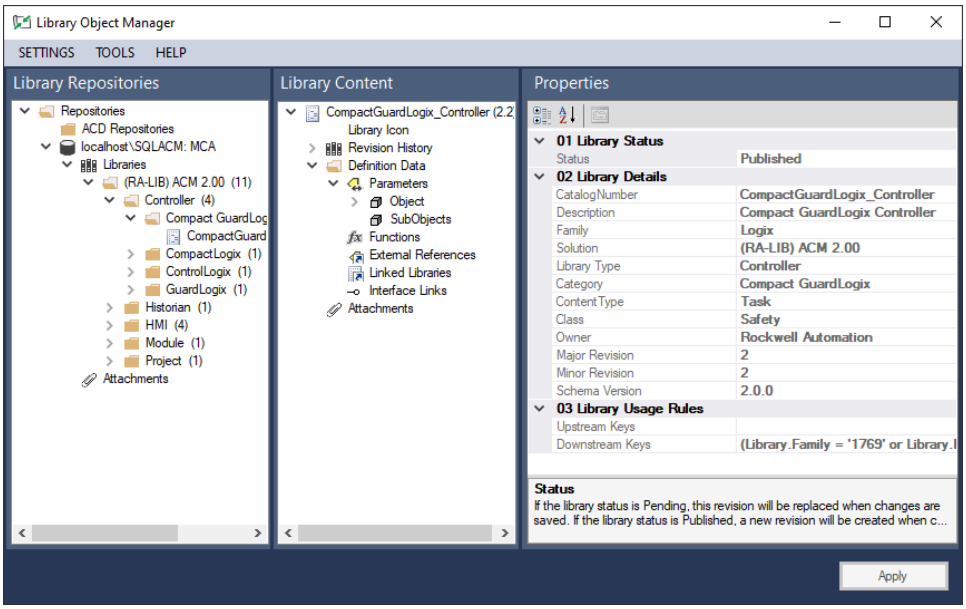
Use the Library Object Manager application to perform these tasks:

- Save library objects as individual files or entities in a database.
- Add non-Logix components to library objects, including FactoryTalk View SE/ME Symbols, FactoryTalk View SE VBA code items, FactoryTalk Historian Tags, FactoryTalk Alarms and Events (FTAE) Digital Alarms, and FactoryTalk View ME (FTVME) Digital Alarms.
- Create repositories for storing and distributing library object files.

Rockwell Automation recognizes that some of the terms that are currently used in our industry and in this publication are not in alignment with the movement toward inclusive language in technology. We are proactively collaborating with industry peers to find alternatives to such terms and making changes to our products and content. Please excuse the use of such terms in our content while we implement these changes.

The interface

The Library Object Manager interface is organized into a three-pane display composed of two panels with object trees and a third panel that contains object properties.



This table describes the regions and controls on the Library Object Manager interface.

Interface element	Description
Menu bar	Use to configure settings for library attributes and view log information.
Library Repositories column	The **Library Repositories** column displays collections of library objects (repositories). There are three kinds of repositories that can be added to the display: • **ACD**: An ACD-based controller code file that includes library objects created in Library Designer. These library objects are still part of the ACD file and must be published to a Folder or Application Code Manager database before they can be added to an Application Code Manager project. ACD files are first added to Library Object Manager, then mounted to make the library objects accessible to the program. • **Folder**: A new or existing Windows folder. Individual library object files can be copied into Folders for remote distribution. • **Application Code Manager**: An instance of the Application Code Manager database. When a library object is copied into a Folder or the ACM Database, Library Object Manager generates an HLS4 file for the library object. Library objects may be copied multiple times. Each time a library object is copied, a distinct version of the library object is created and a distinct HLS4 file is generated.

Interface element	Description
	Click the **+** icon to the left of an item in the tree to display the elements that are contained within it. Click the **-** icon to collapse the element display.
Library Content column	The **Library Content** column displays a tree view of the content of a library object that has been selected in the **Library Repositories** column. The **Library Content** column becomes active when a library object that has been copied to a folder or to the ACM database is selected. By default, the column displays library object content as a read-only display in the **Properties** panel under these headings: • **Revision History**: The information entered when the current version of the library object was created. • **Definition Data**: The decoration added in Library Designer. • **Logix**: The actual Logix code for the library object, displayed as XML. Use the **Library Content** column to add these features to a selected library object: • **FactoryTalk View SE/ME**: A Human Machine Interface (HMI) element. • **FactoryTalk Alarms and Events**: AFactoryTalk Alarms and Events element. • **FactoryTalk View ME Alarms**: A FactoryTalk View ME alarm element. • **FactoryTalk Historian SE**: A FactoryTalk Historian element. These features can only be added after a library object has been copied to a Folder or the ACM Database. They are included in the individual HSL4 library object file, but are not added to the original ACD file.
Properties panel	The **Properties** panel displays the **Revision History**, **Decoration**, and **Logix** code of the selected library object. It also displays the property settings available for editing. When an editable property is selected, click in the **Properties** panel to modify the setting. The property value column will show a cursor to indicate that the setting can be directly entered or a dropdown arrow to select a value from a list. Properties that can accept calculated values also show the ellipsis (...) button. Clicking this button opens the Expression Builder on page 59 or the **Tag Browser**. Click the **Apply** button to accept any changes made to the properties settings.

Set the default options

Use the **Settings** dialog box to set default options for the **Solutions**, **Library Types**, **Categories**, and **Families** items in the **Properties** panel.

To set the default options for library object classification

1. Click **SETTINGS > Library Attributes**.
   The Settings on page 122 dialog box opens displaying the **Solutions**, **Library Types**, **Categories**, and **Families** items.
   A button with an ellipsis (...) follows each item.
2. To define the default options for an item, click the corresponding ellipsis button.
3. The **List Editor** dialog box opens. Enter the options for the items, one per line.
4. Click **OK** to accept the options and return to the **Settings** dialog box. Repeat for all of the items as needed.
5. Click **OK** to save the changes and close the **Settings** dialog box.

Create a list of options

How do I open the List Editor?

1. Click **SETTINGS > Library Attributes**.
2. Click the ellipsis (...) button next to a setting.

The **List Editor** dialog box is used to create a list of options.

To create the list

1. Click in the blank space under **Enter the strings in the list (one per line)**. The cursor appears in the box.
2. Type the item to include in the list. Press **Enter** after each item so that each item is on a different line.
3. After all of the items for the list have been entered, click **OK**.
   The **List Editor** dialog box closes.

Settings dialog box reference

Use the **Settings** dialog box to set default options for the **Solutions**, **Library Types**, **Categories**, and **Families** items in the **Properties** panel.

This table describes the items in the **Settings** dialog box. To configure an item or create a list of items, click the corresponding ellipsis (...) button.

Item	Description
Solutions	The library object repository for a set of library objects.
Library Types	The general classification for a library object based on its function. For example, module, value, or motor.
Categories	A more specific classification for a library object, based on its function. For example, digital, analog, communication, or controller type.
Families	The specific identifier for the base catalog number of the item. For example, 1734, 1738, Logix, or Project.

Item	Description
Default the Publish Status to Pending	Select to set pending as a default library status.

Library Repositories overview

Library repositories are collections of library objects. Library Object Manager uses three kinds of repositories.

- ACD: An ACD-based controller code file that includes library objects created in the Library Designer. These library objects are still part of the ACD file and must be published to a folder or Studio 5000® Application Code Manager database before they can be added to an Application Code Manager project. ACD files are first added to the Library Object Manager application, then mounted to make the library objects accessible to the program.
- Folder: A new or existing Windows folder on the local computer or on a shared network drive. Individual library object files can be copied into remote folders for distribution.
- ACM: Connects Library Object Manager to an instance of the Application Code Manager database.

Library objects can be copied between the different library repositories as needed.

Add and mount an ACD repository

Create library repositories using an ACD file. ACD files contain controller code for Rockwell Automation® Logix platforms, including ControlLogix, CompactLogix, and FlexLogix. When basing a library repository off an ACD file, the ACD file must also be mounted in Library Object Manager before use.

IMPORTANT: An ACD file must not be in use when it is mounted in Library Object Manager. Before mounting the ACD file, make sure that the ACD file is not open in another application.

To add and mount an ACD repository

1. In the **Library Repositories** column, right-click **Repositories** then select **Add Repository > ACD**. The **Select an ACD File** dialog box opens.
2. Navigate to the ACD file to add and double-click the listing or click **Open**.
3. The ACD file opens. The file displays with a red "X" next to the name, and that the library objects are not accessible. To make the library objects accessible, the ACD file must be mounted.
4. In the **Library Repositories** column, right-click the ACD file and select **Mount**.
5. The Red "X" no longer displays and the library objects are added to the repository.

Unmount an ACD repository

While the ACD file is mounted in Library Object Manager, it cannot be opened in Logix Designer. When Library Object Manager is closed, the ACD file is automatically unmounted. However, to continue using Library Object Manager with other projects while editing an ACD file repository in Logix Designer, unmount the ACD file in Library Object Manager first.

To unmount an ACD repository

- In the **Library Repositories** column, right-click the ACD repository and then select **Unmount**.
  The repository remains in the **Library Repositories** column, but the library objects are no longer accessible.

Open an ACD file in Studio 5000 Logix Designer

Move directly between the Library Object Manager application, the Library Designer, and the Studio 5000 Logix Designer application when working with a file. The ACD file must be unmounted before opening it in the Logix Designer application.

Prerequisites

- Add and mount an ACD repository on page 125.
- Unmount an ACD repository on page 125.

To move from the Library Object Manager application to the Studio 5000 Logix Designer application

1. In the **Library Repositories** column, right-click the ACD repository and then select **Open ACD**.
   Studio 5000 Logix Designer starts and the file opens.
2. After completing modifications to the file, close it in the Studio 5000 Logix Designer application.
3. In the **Library Repositories** column, right-click the ACD file and select **Mount**.

IMPORTANT: Modifications to the ACD file are saved to the file, but are not saved to library objects that have already been published to folders or the ACM database. Generate new versions of the library objects to incorporate the most recent modifications.

Open an ACD file in Library Designer

Move directly between the Library Object Manager application and the Library Designer when working with an ACD file. The ACD file must be mounted in Library Object Manager to be opened in the Library Designer.

Prerequisites

- Add and mount an ACD repository on page 125.

To move between the Library Object Manager application to the Library Designer

1. In the **Library Repositories** column, right-click the ACD repository and then select **Launch Library Designer**.
2. The file opens in the Library Designer.
   Complete modifications to the file, then close the Library Designer to return to the Library Object Manager application with the modifications saved.

Export ACD to L5X

In some situations, an L5X file is preferred over an ACD file for transferring controller code and library objects. Use Library Object Manager to export an ACD repository to the L5X format.

To export an ACD repository to a L5X format

1. In the **Library Repositories** column, right-click the ACD file and select **Mount**.
2. Right-click the ACD repository and then select **Export to L5X**.
3. The **Save As** window opens. Navigate to the folder location to save the file and click **Save**.

Open the ACD file location

Use Library Object Manager to open the location of an ACD file quickly.

To open the file location of an ACD repository

1. In the **Library Repositories** column, right-click the ACD repository.
2. Select **Open File Location**.

Remove an ACD repository

Remove an ACD repository if the repository is no longer needed.

To remove an ACD repository

1. In the **Library Repositories** column, right-click the repository.
2. Select **Remove**.

   The repository is closed in the Library Object Manager application and its listing is removed from the **Library Repositories** column.

Add a group

Adding a group allows you to manage the ACD files under **ACD Repositories**.

To add a group

1. In **Library Repositories**, expand **Repositories**.
2. Right-click **ACD Repositories**, and then select **Add > Group**.
3. In **Add Group**, enter the name of the group.
4. Select **OK**. You can locate the ACD files to the group you want through the drag-and-drop editing.

Rename a group

Once a group is created, you can rename it as needed.

To rename a group

1. Right-click the group, and then select **Rename**.
2. Enter the new name in the name box.
3. Press **Enter**.

Remove a group

You can delete a group that is no longer used.

To delete a group

1. Right-click a group, and then select **Delete**.
2. When prompted, select **Yes**.

   The **Group** folder and its descendant groups are deleted. However, all the ACD files under this group and its descendant groups remain unaffected and are relocated to the ACD Repository root folder.

Add a folder repository

Library objects in the folder repository are organized according to a four-level hierarchy:

Solution -> Library Type -> Category -> Catalog Number(version number)

The catalog number and version number uniquely identify the library object. When ACM data is added to a repository, objects are displayed using this organizational method.

After adding the folder repository, add libraries and library objects.

To add a folder repository

1. In the **Library Repositories** column, right-click **Repositories** then select **Add Repository > Folder**. The **Browse For Folder** window opens.
2. Select a folder or create a folder.
   - Select a folder, and then click **OK**.
   - Click **Make New Folder**. A new folder is added to the current directory in the window. The name is highlighted. Change the name and click **OK**.

Remove a folder repository

Remove a folder repository if the folder is no longer needed.

To remove a folder repository from the Library Object Manager application

1. In the **Library Repositories** column, expand **Repositories**.
2. Right-click the folder and select **Remove**.
   The folder is closed in the Library Object Manager application and its listing is removed from the **Library Repositories** column. All library objects that were added to the folder are still present in the folder location and can be registered in Application Code Manager.

Add the ACM database

Library objects in the ACM Database repository are organized according to a four-level hierarchy:

Solution -> Library Type -> Category -> Catalog Number(version number)

The catalog number and version number uniquely identify the library object. When the ACM data is added to a repository, objects are displayed using this organizational method.

To add the ACM database as a repository

1. In the **Library Repositories** column, right-click **Repositories**.
2. Select **Add Repository > ACM**. The **Connection Properties** window opens.
3. Select the default ACM Database or use a different ACM data file.
   - To select the default ACM Database, click **OK**.
   - To select a different data server, select from the dropdown menu in the **Server name** setting, or enter a different server name and SQL server instance. Then click **Refresh**. When the refresh is complete, click **OK**.

- To select a different database from the current server, select from the **Select or enter a database name** dropdown menu. Then click **OK**.
- To select a data file from outside the server, select **Attach a database file** and click **Browse**. Navigate to the data file. Click **OK** to load it, then enter a name in the **Logical name** setting. Click **OK**.

4. (optional) To see detailed information about the data connection, click **Advanced**. The Advanced Properties dialog on page 130 opens.
5. To test the database connection, click **Test Connection**. The **Test results** window displays whether the connection was successful.

Tip: The **OK** button is unavailable until a successful connection is made.

6. Click **OK** to add the ACM database as a library repository.
   The ACM Database along with the library objects in the database are added to the tree view within the **Library Repositories** column.

Remove the ACM database

Remove an ACM database repository if the database is no longer needed.

To remove the ACM database from the Library Object Manager application

1. In the Library Repositories on page 125 column, expand **Repositories**.
2. Right-click the ACM database and select **Remove**. The ACM database is removed from the **Library Repositories** column.

Connection Properties dialog

The **Connection Properties** dialog box defines the settings for how Library Object Manager connects with the ACM database. Use this dialog box to set the data source, the type of user authentication, to specify a different database file, or to modify advanced security settings.

This table describes the settings in the **Connection Properties** dialog box.

Setting	Description
Data source	Database type. Always select Microsoft SQL Server (SqlClient).
Server name	Selects a computer name and SQL Server instance from the list or type a computer name and SQL Server instance in this format: <Computer Name>\<SQL Server Instance>.
Log on to the server	
Use Windows Authentication	Allows SQL Server sign in using Windows authentication. When selected, the logged on Windows user account credential will be sent to SQL Server to authenticate the session.

Setting	Description
Use SQL Server Authentication	Allows SQL Server log on using SQL Server authentication. When selected, provide the username and password for the SQL Server authentication. • **User name**. The SQL Server username, "sa" by default. • **Password**. The SQL Server password associated with the username specified, "ApplicationAdm1n" by default. • **Save my password**. When selected, saves the SQL Server password specified so that it can be used in the next session.
Connect to a database	
Select or enter a database name	Select a database name from the list or enter a database name.
Attach a database file	When selected, specify the identifiers for the database file. SQL Server database files have two names, the operating system file name used to locate the database in the file system and the logical file name used to identify the database within SQL Server transactions. • Type a database file name or use the **Browse** button to use the **Select SQL Server Database File** dialog to locate the database file by clicking through the file system. • In **Logical name**, type the logical name of the database.
Advanced	Select to configure advanced database properties.
Test Connection	Tests the connection to the database. If a "Test connection succeeded." message is not returned, check that the following settings are correct: • Computer name • SQL Server authentication • Network access (remote SQL Server)

Advanced Properties dialog

The **Advanced Properties** dialog box provides a means of changing how the connection between Library Object Manager and the ACM database passes information.

This table describes the settings in the **Advanced Properties** dialog box. The dialog box is divided into functional areas.

Tip: Applying the recommended settings will improve ACM performance especially for network connections.

Area	Setting	Possible Values	Description
Advanced	MultipleActiveResultSets	True False (default)	When True, multiple result sets can be returned and read from one connection.

Area	Setting	Possible Values	Description
	Network Library	**blank (required if local)** Named Pipes (DBNMPNTW) Shared Memory (DBMSLPCN) **TCP/IP (DBMSSOCN) (recommended if networked)** VIA (DBMSGNET)	The network library used to establish a connection to an instance of SQL Server. **Do not use when the SQL Server is resident on the local host computer, value should be blank.**
	Packet Size	**8000 (recommended)**	Size in bytes of the network packets used to communicate with an instance of SQL Server. PacketSize may be a value in the range of 512 bytes and 32767 bytes.
	Transaction Binding	Implicit Unbind (default) Explicit Unbind	Indicates the binding behavior of connection to the **System.Transactions** namespace. When set to **Implicit Unbind**, the connection detaches from the transaction when it ends, switching back to autocommit mode. When set to **Explicit Unbind** the connection remains attached to the transaction until the transaction is closed. The connection will fail if the associated transaction is not active or does not match the current transaction.
	Type System Version	Latest (default) SQL Server 2012 SQL Server 2008 SQL Server 2005	Indicates which server type system the provider will expose through the DataReader.
Connection Resiliency	ConnectRetryCount	**2 (recommended)**	Number of attempts to restore a connection. The number of reconnections attempted after identifying that there was a connection failure. This must be an integer between 0 and 255. Set to 0 to disable reconnecting on idle connection failures.

Area	Setting	Possible Values	Description
	ConnectRetryInterval	**5 (recommended)**	Delay between attempts to restore connection. The amount of time (in seconds) between each reconnection attempt after identifying that there was a connection failure. This must be an integer between 1 and 60.
Context	Application Name	.Net SqlClient Data Provider	The name of the application.
	Workstation ID		The name of the workstation connecting to SQL Server.
Initialization	ApplicationIntent	ReadWrite (default) ReadOnly	Declares the application workload type when connecting to a server.
	Asynchronous Processing	True False (default)	When true, enables usage of the Asynchronous functionality in the .NET Framework Data Provider.
	Connect Timeout	**30 (recommended)**	The length of time in seconds to wait for a connection to the server before ending the attempt and generating an error. A value of 0 indicates no limit, and should be avoided in a ConnectionString because an attempt to connect waits indefinitely.
	Current Language		The SQL Server Language record name.
Pooling	Enlist	True (default) False	When **True**, sessions in a Component Services environment should automatically be enlisted in a global transaction where required.
	Load Balance Timeout	30 (default)	The minimum amount of time (in seconds) for this connection to live in the pool before being destroyed. When a connection is returned to the pool, its creation time is compared with the current

Area	Setting	Possible Values	Description
			time, and the connection is destroyed if that time span (in seconds) exceeds the value specified by **Load Balance Timeout**. A value of zero (0) causes pooled connections to have the maximum connection timeout.
	Max Pool Size	**1000 (recommended)**	The maximum number of connections allowed in the pool. Valid values are greater than or equal to 1. Values that are less than **Min Pool Size** generate an error.
	Min Pool Size	1 (default)	The minimum number of connections allowed in the pool. Valid values are greater than or equal to 0. Zero (0) in this field means that no minimum connections are initially opened. Values that are greater than **Max Pool Size** generate an error.
	PoolBlockingPeriod	Auto AlwaysBlock **NeverBlock (recommended)**	Defines the blocking period behavior for a connection pool. When connection pooling is enabled and a timeout error or other sign-in error occurs, an exception will be thrown and subsequent connection attempts will fail for the next five seconds, the "blocking period". If the application attempts to connect within the blocking period, the first exception will be thrown again. Subsequent failures after a blocking period ends will result in a new blocking period that is twice as long as the

Area	Setting	Possible Values	Description
			previous blocking period, up to a maximum of one minute.
	Pooling	**True (recommended)** False	When **True**, the connection object is drawn from the appropriate pool, or if necessary, is created and added to the appropriate pool. Any newly created connection is added to the pool when closed by the application. In the next attempt to open the same connection, that connection will be drawn from the pool. Connections are considered the same if they have the same connection string. Different connections have different connection strings.
Replication	Replication	False (default) True	Used by SQL Server in replication. Set to **True** if replication is supported using the connection.
Security	Authentication	NotSpecified (default) SqlPassword ActiveDirectoryPassword ActiveDirectoryIntegrated	Specifies the method of authenticating with SQL Server.
	Column Encryption Setting	Enabled Disabled (default)	Default column encryption setting for all the commands on the connection.
	Encrypt	True False (default)	When True, SQL Server uses SSL encryption for all data sent between the client and server if the server has a certificate installed.
	Integrated Security	True False (default)	Whether the connection is to be a secure connection or not. When **False**, User ID and Password are specified in the connection. When **True**, the current Windows account

Area	Setting	Possible Values	Description
			credentials are used for authentication.
	Password	*******	Indicates the password to be used when connecting to the data source.
	Persist Security Info	True False (default)	When **False**, security-sensitive information, such as the password, is not returned as part of the connection if the connection is open or has ever been in an open state.
	TrustServerCertificate	**True (recommended)** False	When **True** (and **Encrypt** is set to **True**), SQL Server uses SSL encryption for all data sent between the client and server without validating the server certificate. If **TrustServerCertificate** is set to **True** and **Encrypt** is set to **False**, the channel is not encrypted.
	User ID	sa	Indicates the user ID to be used when connecting to the data source.
Source	AttachDbFilename		The name of the primary file, including the full path name, of an attachable database.
	Context Connection	True False (default)	When **True**, indicates that the connection should be from the SQL Server context. Available only when running in the SQL Server process.
	Data Source	localhost\SQLACM (default)	Indicates the name of the data source to connect to.
	Failover Partner		The name or network address of the instance of SQL Server that acts as a failover partner.
	Initial Catalog	*Initial Database Name*	The name of the initial catalog or database in the data source.
	MultiSubnetFailover	True False (default)	If your application is connecting to a high availability, disaster recovery

Area	Setting	Possible Values	Description
			(AlwaysOn) availability group (AG) on different subnets, setting this value to **True** configures SqlConnection to provide faster detection of and connection to the (currently) active server.
	TransparentNetworkIPResolution	True (default) False	If your application connects to different networks, setting this value to **True** configures SqlConnection to provide transparent connection resolution to the currently active server, independently of the network IP topology. When set to **True**, the application is required to retrieve all IP addresses for a particular DNS entry and attempt to connect with the first one in the list. If the connection is not established within 0.5 seconds, the application will try to connect to all other IP addresses in parallel. When the first IP address answers, the application will establish the connection with the respondent IP address. If **MultiSubnetFailover** is set to **True**, this setting is ignored. If **Failover Partner** is specified, this setting is ignored. The default setting is **False** if **Authentication** is set to either **Active Directory Password** or **Active Directory Integrated**, otherwise the default setting is **True**.
	User Instance	True False (default)	Indicates whether the connection will be redirected to connect to an instance of

Area	Setting	Possible Values	Description
			SQL Server running under the user's account.

Create and copy library objects

After adding repositories to Library Object Manager, populate the repositories with library objects. Create library objects using Library Object Manager or copy from other sources including ACD repositories or shared folders.

Multiple library objects can be imported at one time.

Create a library object

Create a library object from within a folder repository or the ACM Database. This library object does not include <CLX> content.

To create a library object

1. In the **Library Repositories** column, right-click the database icon and then select **New Library**. The New Library on page 140 window opens.
2. Enter the information for the new library object.
3. Click **OK**.

Publish a library object

Publish a library from an ACD repository to an ACM repository or a folder repository by copying the library from the tree node to the required repository node. Copy existing library objects from one repository to another for collaboration or to restore from a backup.

To publish a library object from the ACD repository to a folder repository or the ACM database

1. Select the library object in the ACD Repository and drag it on top of the icon for the folder repository or ACM database.

   Tip: The pointer changes to the "unavailable" icon until the pointer is over a valid repository icon.

   The Library Import Configuration on page 141 dialog box opens.

   IMPORTANT: When publishing into the ACM database the new library object will be compared to all library objects with the same catalog number that are currently stored in the ACM database, and many of the items in the **Library Import Configuration** dialog box will be completed using the information from the ACD or ACM database. Information that originates from the ACD cannot be edited in the **Library Import Configuration** dialog box.
   If publishing the library results in duplicated names, a warning message is displayed. Click **OK** to proceed with publishing and accept the duplicate names.

2. Enter the information for the library object.
3. Click **OK** to add the library object to the repository.

Publish multiple library objects

Multiple library objects can be published from an ACD repository at one time. Some objects may require additional information before they can be imported.

To publish multiple library objects from the ACD Repository to a folder repository or the ACM database

1. Select multiple library objects.
   - Hold down the **Shift** key to select a contiguous block of library objects.
   - Hold down the **Ctrl** key to select individual library objects that are not contiguous.
2. Place the mouse within one of the selected items and drag all of the library objects on top of the folder repository or the ACM database. The Library Import Configuration on page 141 window opens.

   IMPORTANT: When publishing into the ACM database the new library object will be compared to all library objects with the same catalog number that are currently stored in the ACM database, and many of the items in the **Library Import Configuration** dialog box will be completed using the information from the ACD or ACM database. Information that originates from the ACD cannot be edited in the **Library Import Configuration** dialog box.
   If publishing the library results in duplicated names, a warning message is displayed. Click **OK** to proceed with publishing and accept the duplicate names.

3. All of the selected library objects are listed under the **New Libraries** heading.
   a. Select an item from the list to display its settings and enter the information for the library object.
   b. Click **OK** once all items have been entered to add all of the library objects at the same time.

   IMPORTANT: A red warning icon next to a library object in the list indicates that required information is not present for the library object.
   Library objects cannot be imported until all required settings are complete.
   If the required information cannot be supplied, clear the checkbox for that library object to omit it from the import list.

Copy multiple library objects

Multiple library objects can be imported at one time. Library objects copied from a repository are copied intact. They cannot be edited during the import process.

To copy multiple library objects between folder repositories and the Application Code Manager database

1. Select multiple library objects.
   a. Hold down the **Shift** key to select a contiguous block of library objects.
   b. Hold down the **Ctrl** key to select individual library objects that are not contiguous.
   c. Select a Solution, Library Type, or Category listing to select all of the contained library objects, or select the **Libraries** listing to select all of the library objects in the repository.
2. Place the mouse within one of the selected items and drag all of the library objects on top of the folder repository or the ACM database. The **Library Import Configuration** dialog box opens.
3. Click **OK** to complete the copy process and publish the library.

IMPORTANT: If publishing the library results in duplicated task names, a warning message is displayed. Click **OK** to proceed with publishing and accept the duplicate names.

Edit library properties

Use the Edit Library Properties on page 143 dialog box to edit the properties of the selected library objects. Both pending and published library objects' properties can be edited.

To edit library properties

1. Select one or multiple library objects.
2. Right-click one of the selected items and select **Edit Library Properties**.
3. In the **Edit Library Properties** dialog box, edit the properties as needed.

Tip: If a parent library object node is selected, select **Show all Library versions**.

4. Click **OK**.

Update library description

Library Object Manager supports the library description update of published libraries without creating a library version.

To update library description

1. In the **Library Repositories** column, expand a library repository > **Libraries** > library folders.
2. Right-click the **Libraries** node, a library folder, or a library object and select **Update Description**.
3. In the **Update Library Description** dialog box, edit the library description.

Tip: Select the **Libraries** node or a library folder will update all child library objects' descriptions at the same time.

4. Click **OK**.

Copy a library object to a solution

Library Object Manager supports the copying of pending or published libraries from single solutions to new solutions.

To copy a library object to a new solution

1. In the **Library Repositories** column, expand a library repository > **Libraries** > library folders.
2. Right-click a library folder or a library object and select **Copy to New Solution**.
3. In the Copy to New Solution on page 145 dialog box, edit the new solution name and other properties as needed.
4. Click **OK**.

New library settings

The **New Library** dialog box has these settings.

Name	Type	Description
01 Library Status		
Status	Dropdown menu	If the library is saved as Pending, this revision will be replaced next time changes are saved. If the library is saved as Published, a new revision will be created next time changes are saved.
02 Revision History		
Revision Description	Text entry	A description of the updates made to the current library object, compared to previous versions of the library object. Appears in the Revision History screen when the Object is registered in the Application Code Manager (ACM) application. This is entered manually. This is a required setting.
03 Library Details		
CatalogNumber	Text entry	The library object name, which appears together with the revision number in the library object listing when the library object is registered in ACM. This is entered manually. This is a required setting.
Description	Text entry	The description of the library object. This is a required setting.
Family	Dropdown menu or text entry	The family for the library object. This is a required setting.
Solution	Dropdown menu or text entry	The solution for the library object. This is a required setting.

Name	Type	Description
Library Type	Dropdown menu or text entry	The library type for the library object. This is a required setting.
Category	Dropdown menu or text entry	The category for the library object. This is a required setting.
Content Type	Text	Displays the type of the object. For example, task, program, or routine.
Owner	Text entry	The user or entity that originally published the library object. This is a required setting.
Major Revision	Text entry (integer)	The major revision number for the library object. For a new library object, this defaults to 1.
Minor Revision	Text entry (integer)	The minor revision number for the library object. For a new library object, this defaults to 0.
04 Library Usage Rules		
Upstream Keys	Text entry	For module library objects: a rule that limits the upstream hardware components that will be made accessible to the library object when it is added to an ACM Project. The rule is entered manually as a logical expression.
Downstream Keys	Text entry	For module library objects: a rule that limits the downstream hardware components that will be made accessible to the library object when it is added to an ACM Project. The rule is entered manually as a logical expression.

Library Import Configuration

Use the **Library Import Configuration** window to define the parameter settings for a library object that is being imported.

The **Library Import Configuration** window has these settings.

Name	Type	Description
01 Library Status		
Status	Dropdown menu	If the library is saved as Pending, this revision will be replaced next time changes are saved. If the library is saved as Published, a new revision will be created next time changes are saved.

Name	Type	Description
02 Revision History		
Revision Description	Text entry	The revision description of the library object. This is an optional setting.
Reset History	Dropdown menu	Displays different options to keep the current revision history, remove previous revision histories, or clear all revision histories. This is a required setting.
03 Library Details		
CatalogNumber	Text entry	The library object name, which appears together with the revision number in the library object listing when the library object is registered in ACM. This is entered manually. This is a required setting.
Description	Text entry	The description of the library object. This is an optional setting.
Family	Dropdown menu	The Family of the library object. This is a required setting.
Solution	Dropdown menu	The Solution for the library object. This is a required setting.
Library Type	Dropdown menu	The Library Type for the library object. This is a required setting.
Category	Dropdown menu	The Category for the library object. This is a required setting.
Content Type	Text	Displays the type of the object, Task, Program, or Routine.
Owner	Text entry	The user or entity that originally published the library object. This is a required setting.
Major Revision	Text entry (integer)	The major revision number for the library object. For a new library object, this defaults to 1. If you are updating an existing library a new value will be set based on existing library objects with the same CatalogNumber, unless the library status is Pending.
Minor Revision	Text entry (integer)	The minor revision number for the library object. For a new library object, this defaults to 0. If you are updating an existing library a new value will be set based on

Name	Type	Description
		existing library objects with the same CatalogNumber, unless the library status is Pending.
04 Library Usage Rules		
Upstream Keys	Text entry	For module library objects: a rule that limits the upstream hardware components that will be made accessible to the library object when it is added to an ACM Project. The rule is entered manually as a logical expression.
Downstream Keys	Text entry	For module library objects: a rule that limits the downstream hardware components that will be made accessible to the library object when it is added to an ACM Project. The rule is entered manually as a logical expression.

Edit library properties dialog box

The **Edit Library Properties** dialog box has these settings.

Name	Type	Description
01 Library Status		
Status	Dropdown menu	If the library is saved as Pending, this revision will be replaced next time changes are saved. If the library is saved as Published, a new revision will be created next time changes are saved.
02 Revision History		
Revision Description	Text entry	The revision description of the library object. This is an optional setting.
Reset History	Dropdown menu	Select different options to keep current revision history, remove lower revision history, or clear all revision history. This is a required setting.
03 Library Details		
CatalogNumber	Text entry	The library object name, which appears together with the revision number in the library object listing when the library object is registered in ACM. This is entered manually. This is a required setting.

Name	Type	Description
Description	Text entry	The description of the library object. This is an optional setting.
Family	Dropdown menu	The Family of the library object. This is a required setting.
Solution	Dropdown menu or text entry	The Solution for the library object. This is a required setting.
Library Type	Dropdown menu	The Library Type for the library object. This is a required setting.
Category	Dropdown menu	The Category for the library object. This is a required setting.
Content Type	Text	Displays the type of the object, Task, Program, or Routine.
Owner	Text entry	The user or entity that originally published the library object. This is a required setting.
Major Revision	Text entry (integer)	The major revision number for the library object. For a new library object, this defaults to 1. If you are updating an existing library a new value will be set based on existing library objects with the same CatalogNumber, unless the library status is Pending.
Minor Revision	Text entry (integer)	The minor revision number for the library object. For a new library object, this defaults to 0. If you are updating an existing library a new value will be set based on existing library objects with the same CatalogNumber, unless the library status is Pending.
04 Library Usage Rules		
Upstream Keys	Text entry	For module library objects: a rule that limits the upstream hardware components that will be made accessible to the library object when it is added to an ACM Project. The rule is entered manually as a logical expression.
Downstream Keys	Text entry	For module library objects: a rule that limits the downstream hardware components that will be made accessible to the library object when it is added to an

Name	Type	Description
		ACM Project. The rule is entered manually as a logical expression.
05 Library Repository		
Repository Name	Text	The name of the library repository. This property is read-only.
Repository Path	Text	The path of the library repository. This property is read-only.

Copy to new solution dialog box

The **Copy to New Solution** dialog box has these settings.

Name	Type	Description
01 Library Status		
Status	Dropdown menu	If the library is saved as Pending, this revision will be replaced next time changes are saved. If the library is saved as Published, a new revision will be created next time changes are saved.
02 Revision History		
Revision Description	Text entry	The revision description of the library object. This is an optional setting.
Reset History	Dropdown menu	Select different options to keep current revision history, remove lower revision history, or clear all revision history. This is a required setting.
03 Library Details		
CatalogNumber	Text entry	The library object name, which appears together with the revision number in the library object listing when the library object is registered in ACM. This is entered manually. This is a required setting.
Description	Text entry	The description of the library object. This is an optional setting.
Family	Dropdown menu	The Family of the library object. This is a required setting.
Solution	Dropdown menu or text entry	The Solution for the library object. This is a required setting.
Library Type	Dropdown menu	The Library Type for the library object. This is a required setting.

Name	Type	Description
Category	Dropdown menu	The Category for the library object. This is a required setting.
Content Type	Text	Displays the type of the object, Task, Program, or Routine.
Owner	Text entry	The user or entity that originally published the library object. This is a required setting.
Major Revision	Text entry (integer)	The major revision number for the library object. For a new library object, this defaults to 1. If you are updating an existing library a new value will be set based on existing library objects with the same CatalogNumber, unless the library status is Pending.
Minor Revision	Text entry (integer)	The minor revision number for the library object. For a new library object, this defaults to 0. If you are updating an existing library a new value will be set based on existing library objects with the same CatalogNumber, unless the library status is Pending.
04 Library Usage Rules		
Upstream Keys	Text entry	For module library objects: a rule that limits the upstream hardware components that will be made accessible to the library object when it is added to an ACM Project. The rule is entered manually as a logical expression.
Downstream Keys	Text entry	For module library objects: a rule that limits the downstream hardware components that will be made accessible to the library object when it is added to an ACM Project. The rule is entered manually as a logical expression.
05 Library Repository		
Repository Name	Text	The name of the library repository. This property is read-only.
Repository Path	Text	The path of the library repository. This property is read-only.

Library Attachments

Library attachments are supporting files that are imported, exported, and published along with a library. They do not contain any executable code. Use attachments to deliver additional details with the library, such as specifications or usage documentation.

Add an attachment

Add an attachment to an Application Code Manager database before referencing and publishing them using a library.

To add an attachment

1. In **Library Object Manager**, select **Attachments**.
2. Click **Add New Attachment**.
3. In Attachment on page 150, enter a name and description for the attachment.
4. In **File**, enter a path or click the ellipsis button **(...)** to browse for a path to a file.
5. (optional) In **Revision Description**, enter a revision message describing the changes to this attachment for this revision.
6. Click **OK**.

Assign an attachment reference

Assign an attachment reference to a library so that Library Object Manager exports and publishes the attachment along with the library.

To assign an attachment reference to a library

1. In **Library Object Manager**, from **Library Repositories**, select a library.
2. In **Library Content**, right-click **Attachments**, then select **Add**.
3. In **Add Attachments**, select one or more attachments to reference in the library.
4. (optional) Click the ellipsis button **(...)** next to **Include Condition** to change when Library Object Manager should include the attachment in the library.
5. Click **OK**.

Edit an attachment

Edit an attachment in an Application Code Manager database to change the properties of an attachment. Editing an attachment does not modify its unique File ID.

IMPORTANT: Editing an attachment in an Application Code Manager database updates the attachment every library that references it.

To edit an attachment

1. In **Library Object Manager**, select **Attachments**.
2. Select the attachment from the list.
3. Click **Edit**.
4. In Attachment on page 150, change the properties of the attachment, then click **OK**.

Change the include condition

Change the include condition of an attachment reference to determine when Application Code Manager extracts and publishes the attachment from the project. The attachment is only extracted if the include condition evaluates to true.

To change the include condition of an attachment reference

1. In **Library Object Manager**, from **Library Repositories**, select a library.
2. In **Library Content**, expand **Attachments**, then select an attachment reference from the list.
3. In the properties pane, select the ellipsis button **(...)** next to **Include Condition** to open Expression Builder on page 59.
4. In **Expression Builder**, modify, test, and save the include condition, then select **OK**.
5. Click **Apply**.

Delete an attachment

Delete one or more attachments from an Application Code Manager database when they are no longer needed.

To delete an attachment

1. In **Library Object Manager**, remove references to the attachments from all libraries.
2. In **Library Object Manager**, select Attachments on page 149.
3. Select one or more attachments from the list.
4. Click **Delete**.

Delete an attachment reference

Delete an attachment reference from a library in order to stop Library Object Manager from exporting and publishing the attachment along with the library.

To delete an attachment reference from a library

1. In **Library Object Manager**, from **Library Repositories**, select a library.
2. In **Library Content**, expand **Attachments**, then select an attachment reference from the list.
3. Right-click the attachment reference, then select **Delete**.
4. When prompted, select **Yes**.

Extract an attachment

Extract an attachment to copy its original file from an Application Code Manager database to a folder on the computer. Extracted files are saved with file names matching their **File Name** property.

To extract an attachment

1. In **Library Object Manager**, select **Attachments**.
2. Select an attachment from the list.
3. Click **Extract Files**.
4. In **Browse For Folder**, select a folder where the file should be extracted, then click **OK**.

Import an attachment

Import an attachment from an .HZ1 file to add it to an Application Code Manager database.

To import an attachment from an .HZ1 file

1. In **Library Object Manager**, select Attachments on page 149.
2. Select **Import from .HZ1 Attachment Files**.
3. In **Import HZ1 attachment file**, browse and select one or more files, then click **Open**.
4. (optional) If prompted, select whether to overwrite existing attachments in the Application Code Manager database.
   - Selecting **Yes** overwrites attachment files. All files are imported.
   - Selecting **No** cancels the import. No files are imported.

Export an attachment

Export an attachment to an .HZ1 file to transfer it to another database. Exported .HZ1 files are saved with filenames matching their File ID property, such as *8779ddbb-bd37-438c-ab4e-3b6b75051333.hz1*.

To export an attachment to an .HZ1 file

1. In **Library Object Manager**, select Attachments on page 149.
2. Select one or more attachments from the list.
3. Select **Export to .HZ1 Attachment Files**.
4. In **Browse For Folder**, select a folder where the file should be exported, then select **OK**.

Attachments

How do I open Attachments?

- In **Library Repositories**, select **Attachments**.

Use **Attachments** to add, edit, remove, import, or export file attachments to or from an Application Code Manager database.

Button or Column	Description
Add New Attachment	Opens **Attachment** to add an attachment file to the database.
Edit	Opens **Attachment** to edit the properties of an existing attachment file.
Delete Attachments	Removes the selected attachment files from the database.
Extract Files	Extracts the selected attachment files to a folder location. The output files are extracted to their original filenames and formats.
Export to .HZ1 Attachment Files	Exports the selected attachment files to a folder location. The output files are exported with filenames matching the file ID in the .HZ1 format.
Import from .HZ1 Attachment Files	Imports one or more .HZ1 format attachment files from a folder location.

Button or Column	Description
Filter	Limits the attachment list to attachments with properties matching the characters entered in the **Filter** field. The filter function uses exact character matching and is case insensitive. Wildcard characters are not supported.
Select	Toggles whether a row is selected before deleting or exporting the attachments.
Name	Displays the name of the attachment.
Description	Displays the description of the attachment.
File Name	Displays the original file name of the attachment.
File ID	Displays the unique file identification number of the attachment.
Revision Description	Displays the latest revision description of the attachment.
Modified Date	Displays the last modified date of the attachment.
Modified By	Displays the last user that modified the attachment.

Attachment

How do I open Attachment?

Either:

- In **Attachments**, select **Add New Attachment**.
- In **Attachments**, select an existing attachment, then select **Edit**.

Use **Attachment** to set or edit the properties of an attachment.

Field	Description
File ID	Displays the unique file identifier of the attachment.
Name	Sets the name of the attachment.
File	Sets the filename of or path to the attached file.
Description	Sets the description of the attachment.
Revision Description	Sets the description of the last change or revision to the attachment.
Last Updated	Displays the last date and time that the attachment was revised.
Last User Updating File	Displays the last user that revised the attachment.

Library Content

After creating repositories and library objects, add and configure content for library objects. Library content refers to the properties and parameters defined for a library object.

Read-only library content:

- **Revision History**: The information entered when the current version of the library object was created.
- **Definition Data**: The decoration added in the Library Designer.
- **Logix**: The actual Logix code for the library object, displayed as XML.

In Library Object Manager use the **Library Content** column to add these features to a selected library object:

- **FactoryTalk View SE/ME**: A Human Machine Interface (HMI) element, used for displays.
- **FactoryTalk Alarms and Events**: A FactoryTalk Alarms and Events element, used for digital alarms.
- **FactoryTalk View ME Alarms**: A FactoryTalk View ME alarm element, used for digital alarms.
- **FactoryTalk Historian SE**: A FactoryTalk Historian element, used for tags.

These features can only be added after a library object is copied to a folder or the ACM Database. They are included in the individual HSL4 library object file, but are not added to the original ACD file.

Tip: After modifying a library object using Library Object Manager, an asterisk (*) appears next to the object name until their repository is removed. When the repository is removed from the Library Object Manager application, updates are saved to the HLS4 file.

When working with library objects, different tools are provided to help with the object configuration. These tools are opened by clicking the ellipsis (...) button next to a configurable item. The following tools are part of Library Object Manager.

- Expression Builder
- File Browser
- Symbol Builder
- Tag Editor
- Message Editor
- Message Builder

Review the decoration settings

The Library Content on page 151 column displays all parameters, functions, and external references added to a library object in the Library Designer. All settings for decorative elements can be reviewed but not edited in the Library Object Manager application.

To review the decoration for a library object

1. Select a library object in a folder or database repository. The **Library Content** column and **Properties** panel activate.
2. Right-click the **Definition Data** listing and then select **Expand All**.

The column displays listings for all parameters, functions, and external references added to the library object.

3. Click a listing for a decorative element to review its settings in the **Properties** panel.

View the Logix code

View the Logix code of a library object to review programs, tasks, routines, and linked libraries associated with the object.

To view the Logix code for a library object.

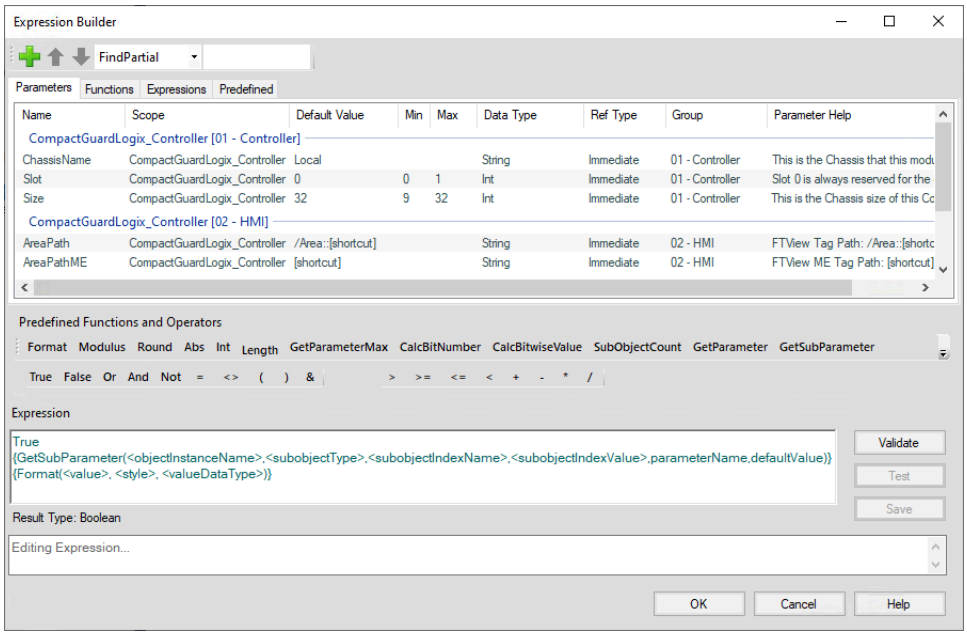
1. Select a library object in a folder or database depository. The Library Content on page 151 column and **Properties** panel activate.
2. Click the **Logix** item in the **Library Content** column. The complete, line-by-line code for the library object displays in the **Properties** panel.

Expression Builder

How do I open Expression Builder?

- In Library Designer or Library Object Manager, select the ellipsis button (...) next to parameters, values, and other objects that can be modified by or decorated with an expression.

The **Expression Builder** is an environment to create, test, and save expressions.



The **Expression Builder** window includes a command toolbar, decorative element tabs, and functional areas.

Item	Description
	The **Add New** button is only active on the **Parameter** tab and opens the **Add New Parameter** window.
	The **Move Up/Move Down** buttons are not active in the **Expression Builder** window.

Item	Description
	This dropdown menu can initiate a search for parameters that match a text string entered in the adjacent field. There are two different types of finds that can be performed, use the dropdown menu to select the one to use: • **FindPartial**: Searches for the text string in any part of an entry. • **FindPrefix**: Searches for the text string at the beginning of an entry. Find can be used on all of the tabs.
Decorative element tabs on page 61	The decorative elements available to the current expression. These include all elements added to the current library object and all elements added to the library objects that are higher in scope. These elements include: • Parameters • Functions • Expressions • Predefined Functions and expressions can be saved and used in future projects.
Predefined Functions and Operators on page 61	A collection of logical and mathematical operators that can be used to manipulate the values generated by the decorative elements.
Expression on page 64 box and Result Type	The expression appears here as elements are added to it. There are also settings and buttons to set the data type of the expression result, and to validate, test, and save the expression. The **Result Type** displays a color-coded response when the expression is validated and tested.

Add display objects

Different display objects (symbols) can be added to a library object to enable those displays to be used in HMI displays created using FactoryTalk® View Studio.

To add FactoryTalk View content to a library object

1. Right-click the library object in the **Library Content** column and select **Add Section > FT View**. An **FT View** folder, with subfolders for **Substitutions**, **SE Symbols**, **SE VBA Items**, and **ME Symbols**, is added to the library object.
2. Add a symbol to a library object. Repeat this process for all symbols needed for the library object:
   a. Right-click the **SE Symbols** or **ME Symbols** folder and select **Add**. The Symbol Builder on page 158 dialog box opens.
   b. In **Select Display Export File**, enter the file location for the exported FactoryTalk View Symbol file and then press **Enter** or click the ellipsis (...) button to open **Select a Display Export File** dialog and navigate to the file, and then click **Open**. The symbols contained in the file are added to the **Select Root Node/s of Symbol** list.

c. In the **Select Root Node/s of Symbol** list, select the symbol or group to import.

d. (optional) In the **Select Substitutions to Apply** list, clear any substitutions that should not be applied.

Tip: In a typical application, it is not necessary to clear any of the items in the **Select Substitutions to Apply** list.

e. Click **OK** to import the symbol.

FT View substitutions

After adding a FactoryTalk View (**FT View**) section to the **Library Content** pane, configure substitutions and symbols for the objects.

Substitutions are used to quickly update FactoryTalk View objects with different text and other display objects as needed by an application. Substitutions can be created from the **Substitutions** property pane or from a symbol property pane. When substitutions are created from the symbol property pane, they are applied to that element immediately. When a substitution is created in the **Substitutions** property pane, it is not applied to an element immediately, but is available to be applied when the symbol is updated.

The **Substitutions** node under the **FT View** node in the **Library Content** pane displays a list of substitutions in the **Properties** pane. Substitutions can be added, edited, deleted or moved up and down from this view. None of these changes will be implemented into the displays though until they are updated.

Substitutions configured in Library Object Manager are not stored as decoration to the original data, but are applied into the original data. This means adding, removing, or editing a substitution can result in erroneous changes to the original data.

Apply substitutions to a symbol

Use substitutions to update elements of a symbol display.

To apply substitutions to a symbol

1. After the symbol has been added, review the XML code in the **Source** panel of the **Properties** column.
2. Highlight the item to be replaced, then right-click it and select **Apply Substitution**. The **Substitution Builder** dialog box opens with the highlighted text automatically placed in the **Original Text** box.
3. Click in the **Replacement** box and enter the text to use instead of the original text. To use a calculated value as the replacement value, click the ellipsis (...) to open the Expression Builder window and create an expression.
4. Click **OK** to apply the substitution to the current symbol.
   The substitution is listed in the **Substitutions** folder for use with other symbols.

Configure the object tag and path

The **Object Tag** and **Path** parameters must be set for correct substitution when the display is instantiated in an ACM project. This can be done in the FactoryTalk View Studio application, before the display is exported, or using Library Object Manager after the display has been added.

Tip: If Controller references are created in the FactoryTalk View Studio application as FactoryTalk View parameters rather than direct references, the reference functionality will be handled by the FactoryTalk View application during actual operation, and the substitutions described in this procedure will not be necessary.

To configure the parameters in the FactoryTalk View Studio application

Tip: A Global Object is used in this procedure. The same procedure applies to other objects, although the display screens, Object Tag name, and XML code may be different.

1. In the **Graphics** editor, on a graphic display, right-click a display element object and select **Global Object Parameter Values**. The **Global Object Parameter Values** window opens.
2. Enter the following value in the **Object Tag** parameter.
   {{AreaPath}{ObjectName}}

   IMPORTANT: The value entered must include the second pair of curly brackets enclosing the {AreaPath} and {ObjectName} tokens.
   The **Path** Parameter is also a part of the **Object Tag** Parameter.

3. Enter the following value in the **Path** Parameter.
   {{AreaPath}}

   IMPORTANT: The value entered must include the second pair of curly brackets enclosing the {AreaPath} token.

4. Export the display object.
5. Import the display objects into Library Object Manager. Review the XML code for the display after it is added to a library object. Find the **Parameters** list and locate the values for the **Object Tag** and **Path** parameters. The values set in the FactoryTalk View Studio application are correct and should not be edited. To change them for some reasons, use Application Code Manager to update them.

To configure the parameters using Library Object Manager

1. After the symbol has been added, review the XML code in the **Source** panel of the **Properties** column. Find the Parameters list and locate an instance of the parameter used to define the path value.
2. Highlight the parameter and then right-click and select **Apply Substitution**. The **Substitution Builder** dialog box opens with the highlighted text automatically placed in the **Original Text** box.
3. Click in the **Replacement** box and enter the predefined parameter token **{ControllerName}** then click **OK**.
4. In the XML code in the **Source** panel of the **Properties** column, find the Parameters list and locate an instance of the parameter used to define the current value for the Object Tag Parameter.
5. Highlight the parameter and then right-click and select **Apply Substitution**. The **Substitution Builder** dialog box opens with the highlighted text automatically placed in the **Original Text** box.
6. Click in the **Replacement** box and enter the predefined parameter token **{ObjectName}** then click **OK**.
   The substitutions are now listed in the **Substitutions** folder.

When the library object is instantiated in the ACM application, the name configured for the instance substitutes for the Symbol's {ObjectName} token. The {AreaPath} token substitution takes place when the value is configured in the AreaPath parameter for the controller.

Apply positioning to a symbol

The **Top** and **Left** Parameters for each symbol should be checked and, if necessary, modified in the **Properties** panel.

Symbols are positioned within a display based on the coordinates of their upper left corner, using absolute or relative positioning.

In absolute positioning, the symbol is positioned relative to the upper left corner of the screen, while in relative positioning, the symbol is positioned based on an offset from its individual bounding box.

Absolute positioning guarantees that the overall configuration of the symbols in a display will match the original configuration created in FactoryTalk View Studio.

Relative positioning adds symbols to a display starting at the upper left corner, moving left to right, then down one row when horizontal space requires. Each symbol is positioned within a bounding box that includes the Symbol's height, width, and x and y offsets. Symbols are added to the display in the same order as they were added to the library object in the Library Object Manager application.

IMPORTANT: Relative positioning is required if multiple instances of a symbol appear in a display; absolute positioning would cause all instances to be stacked on top of each other.

Prerequisites

- Add the symbol to the **SE Symbols** or **ME Symbols** folder, selecting all of the position options (Left, Top, Line, and Arc) in the **Apply Substitutions** area.

To apply positioning to a symbol

1. After the symbol has been added, review the XML code in the **Source** panel of the **Properties** column.
2. Locate the **Left** and **Top** Parameters in the XML code.
   - In absolute positioning, the values for the **Left** and **Top** parameters are numeric. Check that they match the desired display position for the symbol, or modify if necessary.
   - In relative positioning, the values are expressions using the following formats. Check that the left and top offsets are consistent with other symbols in the library object. Modify if necessary.
     left = "*{Calc([bounding box offset] + {SymbolWidth}*{LeftIndex})}*"
     top = "*{Calc([bounding box offset] + {SymbolHeight}*{TopIndex})}*"

     Tip: A small offset value of 5...10 provides the most efficient use of screen space and the most predictable configuration of multiple symbols in a display.

Delete a symbol

Delete a symbol if it is no longer used in a display.

To delete a symbol from a library object

1. In the **Library Content** column, expand the **FT View** section, expand either the **SE Symbols** or **ME Symbols** folder where the symbol is located.
2. Right-click the symbol and select **Delete**.

Update a symbol

If the properties of a symbol have changed since it was added to Library Object Manager, update it to use the current version.

To update a symbol

1. In the **Library Content** column, expand the **FT View** section, expand either the **SE Symbols** or **ME Symbols** folder where the symbol is located.
2. Right-click the symbol and select **Update**. The Symbol Builder on page 158 dialog box opens.
3. **Select Display Export File** is preset to the original location of the symbol file. If this location has changed, enter the file location for the exported FactoryTalk View Symbol file and then press **Enter** or click the ellipsis (...) button to open the **Select a Display Export File** dialog and navigate to the file, and then click **Open**. The **Select Root Node/s of Symbol** list updates to show the symbols contained in the file.
4. In the **Select Root Node/s of Symbol** list, select the symbol or group to import.
5. (optional) In the **Apply Substitutions** list, clear any substitutions that should not be applied.
6. Click **OK** to update the symbol.

Symbol properties

When a symbol is selected, the Library Object Manager Properties panel updates to display the appropriate properties.

This table describes the symbol properties.

Property	Type	Description
Symbol Name	Text entry	The name that is applied to the symbol when the library object is instantiated. In a typical application, the default name should be used. Alternatively, a name can be entered manually in the Symbol Name setting.
Include Condition	Text entry	Sets the condition under which the current symbol is instantiated. The default is Always. A condition can be entered manually or generated by an expression. To create an Expression, click the ellipsis (...) button to open the Expression Builder. By default, Symbols are set to instantiate under all conditions and to instantiate once every time the library object is

Property	Type	Description
		added to an ACM project. With FactoryTalk View Symbols, there may be several options that are appropriate for different applications. In this case, add a parameter token to the Include Condition property that allows the end user to select the symbol that best meets their need. Follow these steps to set a rule for instantiation. 1. Enter a new parameter token in the Include Condition setting, as well as the condition (for example, "{symbolstyle} = 1") for inclusion of the symbol. 2. Repeat the process for all other symbol options.
Source Text	Text entry	The editable XML code of the symbol. The XML code includes all substitutions applied when the symbol was imported.

IMPORTANT: Default substitutions in the XML code include the Name attribute and the Left and Top attributes.

Symbol Builder dialog box

How do I open the Symbol Builder dialog box?

1. In the **Library Content** column, expand the **FT View** section, then click the **SE Symbols** or **ME Symbols** folder to select it.
2. Right-click the folder and select **Add**. The **Symbol Builder** window opens.

The **Symbol Builder** is used to configure how a display object is added to a library object.

This table describes the configuration settings in the **Symbol Builder** dialog box.

Setting	Type	Description
Select Display Export File	Text entry or browse	The name of the exported FactoryTalk View graphics file that holds the symbols to import. Use the ellipsis (...) button to browse to the file location.
Select Root Node/s of Symbol	List	A list of all symbols in the file. Symbols can be selected individually or as a group.

Setting	Type	Description
		However, only one symbol or group can be selected for each import.
Select Substitutions to Apply	Checkboxes	When selected, apply substitutions for symbol attributes. There are two types of substitutions: • **Predefined**. The default substitutions that are applied by Library Object Manager when a symbol is imported. • **Saved**. Custom user-defined substitutions developed in response to a specific implementation requirement.

Add VBA items

Add FactoryTalk View SE VBA items to a library object to add and edit code items in Library Object Manager.

To add FactoryTalk View SE VBA items to a library object

1. In **Library Repositories**, select a library content.
2. In **Library Content**, expand the **FT View** section.
3. Right-click the **SE VBA Items** folder and select **Add**.
4. In the **Select Display Export File** box, click the ellipsis (...) button to browse an export file.
5. In the **Select VBA Item/s** box, select a display or a class item.
6. Click **OK**.

Add a code item

Add a VBA code item to edit the code in Library Object Manager.

Prerequisites

- Add SE VBA Items to a library object.

To add a common/symbol code item to a library object

1. Right-click the **Common Code** or **Symbol Code** subfolder and select **Add**, the VBA Code Builder on page 160 dialog box opens.
2. Select a display or a class item and click **OK**. A common/symbol code item is added to the library object.

Tip: The **Name Prefix** substitution in the **VBA Code Builder** dialog box will be check by default when adding a symbol code.

3. Configure the properties as needed.

Delete a code item

Delete a common/symbol code item if it is no longer needed.

To delete a common/symbol code item from a library object

1. In **Library Content**, expand the **FT View** section and the**SE VBA Items** folder.
2. Expand a display or a class section and the code item subfolder.
3. Right-click the code item and select **Delete**.

Update a VBA code item

Add code, delete code or modify the code to upgrade the code item.

To update a VBA code item

1. Right-click the VBA code item and select **Update**. The **VBA Code Builder** dialog box opens.
2. In the **Select Subroutine/s to Include** box, select the code item.
3. In the **Select Substitutions to Apply** box, check one or multiple substitutions and click **OK**.
4. Configure the properties as needed.
5. Click **Apply**.

VBA Items Properties

When a VBA Item is selected, the **Properties** column of Library Object Manager displays the properties of the item. All of the following properties could be edited.

Name	Description
VBA display and class items	
Collection Item Name	Displays the item name in Library Object Manager, could be referenced by related items and used for documentation generation queries.
Include Condition	The condition of this item. Click the ellipsis (...) button to edit the condition.
Encrypted Data (Determines Name and Type of VBA Item)	Displays the name and type of the item.
VBA code items	
Collection Item Name	Displays the item name in Library Object Manager, could be referenced by related items and used for documentation generation queries.
Include Condition	The condition of this item. Click the ellipsis (...) button to edit the condition.
VBA Code	Displays the code of this item. Edit the code or make substitutions in this box.

VBA Code Builder

Use the VBA Code Builder to select a common/symbol code to add or update.

Item	Description
Select Display Export File	Select a display export file of VBA items, click the ellipsis (...) button to select another file.
Select Subroutine/s to include	A list of subroutines in the selected file. Select display or class subroutines to add or update.
Select Substitutions to Apply	A list of substitutions in the library, including: • **Predefined** A **Name Prefix** substitution • **Saved** Substitutions saved in the library

VBA Item Builder

Use the **VBA Item Builder** dialog box to choose which VBA items to be added.

Item	Description
Select Display Export File	Select a display export file of VBA items, click the ellipsis (...) button to select another file.
Select VBA Item	A list of the VBA items in the selected file. Select a display or a class item to add.

Add FTAE content

Add FactoryTalk Alarms and Events content to library objects to support Logix digital alarm applications. There is separate content for the digital alarm definition, the alarm message, and the alarm tag update rate.

IMPORTANT: Refer to the FactoryTalk Alarms and Events System Configuration Guide (Rockwell Automation Publication FTAE-RM001) for more information on the screens and settings associated with FactoryTalk Alarms and Events content.

To add FactoryTalk Alarms and Events content to a library object

- Right-click the library object in the **Library Content** column and select **Add Section > FTAE**. An **FTAE** folder, with subfolders for **Digital Alarms**, **Messages**, and **Tag Update Rates**, is added to the library object.

Add an alarm to a library object

In a Logix system, digital alarms are configured using tags that are either on or off. The alarm configuration defines a trigger condition by comparing the value of the tag to the configured alarm state. An alarm can be triggered if the digital alarm is in one of these two states:

- The input tag is equal to zero.
- The input tag is not equal to zero.

Prerequisites

- Add FactoryTalk Alarms and Events content to a library object on page 161.

To add a digital alarm to a library object

1. Right-click the **Digital Alarms** subfolder and select **Add**. The **Tag Browser** dialog box opens.
2. In **Select the Alarm Trigger Tag**, type the tag name or part of the tag name to filter the tags listed. To choose a tag that is a member of an array, expand the tag group name to see all of the members of the array.
3. Scroll in the list box and double-click the tag name that will trigger the digital alarm.
   The tag **Data Type** must be Boolean (**BOOL**).
4. Click **OK**. The tag is added as an object in the **Digital Alarms** folder.
5. Configure the properties as needed.

Add a message

Alarm messages describe alarm conditions. Each alarm can be associated with only one message.

To add a message to a digital alarm

1. In **Library Repositories**, select a library object that includes Logix tags.
2. In **Library Content**, expand the **FTAE** folder and the **Digital Alarms** folder.
3. In the **Digital Alarms** folder, select the alarm object, the **Properties** pane displays the digital alarm properties.
4. Under **Message**, click the **New** button. The Message Editor on page 171 opens.
5. Enter the message text in the space provided.
6. To add tag variables to the message text, place the cursor where the variable should be inserted then use the **Add Variable** controls to specify the tag:
   a. In **Variable**, choose the type of tag to insert.
   b. If **Data Type** is configurable for the tag selected, in **Data Type**, choose **Numeric** or **String**.
      If **Data Type** is **Numeric**:
      - In **Number of Digits**, specify the number of digits to display.
      - In **Left Fill**, choose whether to fill the space to the left of the number with zeros, spaces, or nothing.
   c. If **Data Type** is **String**:
      - Select **Used Fixed Width** to enable specification of a character width for the variable.
      - In **Number of Characters**, specify the number of characters to display.
   d. Click **Add** to add the variable to the message.
7. To add an expression evaluation to the message, place the cursor at the point in the message text where you want the expression to be evaluated and then click the ellipsis (...) button to open the **Expression Builder** and create the expression.
   a. Add parameters and functions or add previously created or predefined expressions to the message shown in the **Expression** setting.
   b. Click **Validate** to confirm the expression syntax.
   c. Click **Test** to evaluate the expression and confirm that the correct information is being generated.
   d. If creating an expression or editing an existing expression, click **Save** to save the expression.
   e. For new expressions, the **Save Expression** dialog box appears. In **Expression Name**, type a name for this expression and then click **OK** to close the **Save Expression** dialog box.
   f. Click **OK** to close the **Expression Builder** and add the expression to the message.
8. In **ID**, enter an identification number for the message.

IMPORTANT: The value within the **ID** setting for each message must be unique for all digital alarm messages within a project. Consider assigning blocks of ID numbers for each library object.

9. Click **OK** to apply the completed message to the selected digital alarm.

Create a message

Use the **Message Editor** to create messages that can then be assigned to different alarm conditions. The ID specified for each message must be unique for all digital alarm messages within a project, but multiple messages can be configured using the same ID. Consider assigning blocks of ID numbers for each library object to ensure that conflicts are not inadvertently introduced.

To create a message using the Message Editor

1. In **Library Repositories**, select a library object that includes Logix tags.
2. In **Library Content**, expand the **FTAE** folder and then right-click the **Messages** folder and select **Add**. The Message Editor on page 171 opens.
3. Enter the message text in the space provided.
4. To add tag variables to the message text, place the cursor where the variable should be inserted then use the **Add Variable** controls to specify the tag:
   a. In **Variable**, choose the type of tag to insert.
   b. If **Data Type** is configurable for the tag selected, in **Data Type**, choose **Numeric** or **String**.
      If **Data Type** is **Numeric**:
      - In **Number of Digits**, specify the number of digits to display.
      - In **Left Fill**, choose whether to fill the space to the left of the number with zeros, spaces, or nothing.
   c. If **Data Type** is **String**:
      - Select **Used Fixed Width** to enable specification of a character width for the variable.
      - In **Number of Characters**, specify the number of characters to display.
   d. Click **Add** to add the variable to the message.
5. To add an expression evaluation to the message, place the cursor at the point in the message text where you want the expression to be evaluated and then click the ellipsis (...) button to open the **Expression Builder** and create the expression.
   a. Add parameters and functions or add previously created or predefined expressions to the message shown in the **Expression** setting.
   b. Click **Validate** to confirm the expression syntax.
   c. Click **Test** to evaluate the expression and confirm that the correct information is being generated.
   d. If creating an expression or editing an existing expression, click **Save** to save the expression.
   e. For new expressions, the **Save Expression** dialog box appears. In **Expression Name**, type a name for this expression and then click **OK** to close the **Save Expression** dialog box.
   f. Click **OK** to close the **Expression Builder** and add the expression to the message.
6. In **ID**, enter an identification number for the message.
7. Click **OK**. The completed message appears in the list of messages in the **Properties** pane.

Add an existing message

Alarm messages can be created independently from alarms and then added to the alarm.

To add an existing message to a digital alarm

1. In **Library Repositories**, select a library object that includes Logix tags.
2. In **Library Content**, expand the **FTAE** folder and the **Digital Alarms** folder.
3. In the **Digital Alarms** folder, select the alarm object, the **Properties** pane displays the digital alarm properties.
4. Under **Message**, click the **Browse** button. The **Message Browser** opens.
5. The **Message Browser** displays all messages within the current ACD file. Select a message and click **OK**.

IMPORTANT: The value within the ID setting for each message must be unique for all digital alarm messages within a project. Messages added using the **Message Browser** retain their original ID.

Delete message content

If a message is no longer needed, it can be deleted from Library Object Manager.

To delete a message from a library object

1. In **Library Repositories**, select a library object that includes Logix tags.
2. In **Library Content**, expand the **FTAE** folder and then click the **Messages** folder.
3. The **Properties** panel displays a listing of all existing messages when the **Messages** subfolder is selected.
4. Right-click the message listing and then click **Delete**.
   If the message is used in a digital alarm, a confirmation dialog appears listing the alarms that use the message. Click **Yes** to continue with the deletion. The alarms using the message no longer have an alarm message.

Change the tag update rate

When the **Tag Update Rates** folder is selected in Library Object Manager the **Properties** pane displays information about how frequently a tag's value is updated. Only tags that are monitored for alarm conditions, used as alarm limits or tag values, or referenced as associated tags, are listed in the **Properties** pane.

Changing the update rate of a tag does not directly change the configuration of an alarm and does not affect the state of the alarm.

To change the tag update rate for a digital alarm

1. In **Library Repositories**, select a library object that includes Logix tags.
2. In **Library Content**, expand the **FTAE** folder and the **Digital Alarms** folder.
3. In the **Digital Alarms** folder, click the **Tag Update Rates** subfolder, the **Properties** pane displays a table associating each **Update Rate** with a **Tag Name**.
4. Right-click a tag name, point to **Change Update Rate**, then select an update rate from the list.
   The **Update Rate** displayed for that **Tag Name** in the **Properties** pane is changed.

Digital tab

How do I access Digital Alarm properties?

1. In **Library Repositories**, select a library object that includes Logix tags.
2. In **Library Content**, expand the **FTAE** folder and the **Digital Alarms** folder.
3. In the **Digital Alarms** folder, select the alarm object, the **Properties** pane displays the digital alarm properties.

A digital alarm monitors a tag for one of the following alarm conditions:

- If the value of the tag is equal to zero.
- If the value of the tag is not equal to zero.

When a digital alarm is selected, the **Properties** column of Library Object Manager displays three tabs—**Digital**, **Status Tags**, and **Control Tags**.

The ellipsis (...) buttons following the text entry settings provide additional dialog boxes to assist in specifying the property.

Click **Apply** to apply changes made in these tabs.

This table describes the settings in the **Digital** tab.

Name	Setting Type	Description
Name	Text entry or Expression	The name that will be applied to the digital alarm when the library object is instantiated. This name will appear in the Application Code Manager screens. The setting is populated with a default name based on the Tag name and library object for the selected Tag. In a typical application, the default name should be used. Alternatively, a name can be entered manually in the **Name** setting. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Include Condition	Text entry or Expression	Sets the condition under which the current Digital Alarm is instantiated. The default is **Always**. A condition can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Input Tag	Text entry or Tag	The **ItemID** that will be applied to the Digital Alarm when the library object is instantiated. The setting is populated with

Name	Setting Type	Description
		a default name based on the Tag name and library object for the selected Tag. To browse to an alarm trigger tag, click the ellipsis (...) button to open the **Tag Browser**.
Condition	Dropdown menu	The condition that triggers the digital alarm: • **Source <> 0** The alarm becomes active if the value of the input tag is not equal to zero. • **Source = 0** The alarm becomes active if the value of the input tag is equal to zero.
Latched	Checkbox	Determines whether the alarm remains in effect if the condition is no longer met. When selected, the alarm remains In Alarm, even if its alarm condition returns to normal. To allow a latched alarm to return to normal after its alarm condition returns to normal, the operator must reset the alarm. To allow the alarm to return to normal after its alarm condition returns to normal without requiring operator intervention, clear the checkbox.
Severity	Text entry or Tag	The severity required for the event to activate the alarm. Assign a severity level to the alarm to indicate the urgency of the In Alarm condition. The severity value can be a controller tag value or an integer value. If the severity value is an integer value, the range of values is 1 through 1000, where 1 is the least severe, and 1000 is the most severe. Ranges of alarm severities are mapped to four alarm priorities. Do not use the alarm input tag itself to set its severity value. Input tags may not be updated correctly, and can at times be assigned bad or uncertain quality values (for example, when the connection to a controller is lost, or if a data server is rebooted).

Name	Setting Type	Description
		To browse to a severity tag, click the ellipsis (...) button to open the **Tag Browser**.
Acknowledge required	Checkbox	Determines whether an acknowledgment is required to turn off the alarm. If the alarm does not require acknowledgment by an operator, clear the checkbox. Most alarms are configured to require acknowledgment, but there may be events that are not displayed to operators that do not require acknowledgment. Alarms configured to not require acknowledgment are always in the acknowledged state, even when they are in alarm.
Minimum duration	Text entry	The minimum duration required for the event to activate the alarm. Specify the minimum amount of time that the alarm condition must be true before the alarm condition becomes active. Set the duration from 0 to 600 seconds. Use this setting to eliminate false alarms.
Show Alarm as Tag	Checkbox	Determines whether the alarm is shown as a tag. Tags are used to monitor the status and operate on alarms programmatically at run time. Clients can read and write to the alarm tags to monitor and change alarm states. Select the checkbox to enable tags for this alarm. Clear the checkbox to disable tags for this alarm.
Message	Text entry, Message, or Expression	The text displayed when the digital alarm is triggered. Enter a text message up to 255 characters long that describes the alarm condition. Click **New** to open the **Message Editor** and create a message. Click **Browse** to select a pre-existing message from the **Message Browser**. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.

Name	Setting Type	Description
ID	Read-only	The unique numeric ID for the message in the **Message** setting.
New	Button	Opens the **Message Editor** to create a message.
Edit	Button	Opens the **Message Editor** to edit an existing message.
Browse	Button	Opens the **Message Browser**.
Associated Tags	Text entry or Tag	Up to four tags that may be included in the text of the digital alarm message. In many cases, it is useful to have additional process information associated with an alarm. At runtime, the values of associated tags can be embedded in alarm messages. To browse to a tag, click the ellipsis (...) button next to the tag to open the **Tag Browser**.
Alarm Class	Text entry or Expression	The class for the digital alarm. Type a classification string for the alarm that allows filtering of alarms at run time. For example, use an alarm class to associate alarms together by function, such as those that monitor for valves that fail to open and close. The alarm class text can be up to 40 characters long. To create an expression, click the ellipsis (...) button to open the Expression Builder on page 59.
FactoryTalk View Command	Text entry or Expression	The FactoryTalk command that is executed when the alarm is triggered. Type a FactoryTalk View command, up to 1000 characters long, to associate with the alarm. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Alarm Group	Text entry or Expression	The Alarm group that the alarm will be associated to. The setting is populated with a default reference type parameter based on the tag name for the selected tag when an

Name	Setting Type	Description
		alarm is created. Can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.

Status Tags tab

When a digital alarm is selected, the **Properties** column of Library Object Manager displays three tabs—**Digital**, **Status Tags**, and **Control Tags**.

The ellipsis (...) buttons following the text entry settings provide additional dialog boxes to assist in specifying the property.

Click **Apply** to apply changes made in these tabs.

This table describes the settings in the **Status Tags** tab.

Name	Setting Type	Description
Disabled Tag	Text entry or Tag	Assign a tag whose value is set to 1 when the alarm state is Disabled, and to 0 when the alarm state is Enabled. To browse to select the disabled tag, click the ellipsis (...) button to open the **Tag Browser**.
Suppressed Tag	Text entry or Tag	Assign a tag whose value is set to 1 when the alarm state is Suppressed, and to 0 when the alarm state is Unsuppressed. To browse to select the suppressed tag, click the ellipsis (...) button to open the **Tag Browser**.
In Alarm Tag	Text entry or Tag	Assign a tag whose value is 1 when the alarm state is In Alarm, and is 0 when the alarm state is Normal. To browse to select the in alarm tag, click the ellipsis (...) button to open the **Tag Browser**.
Acknowledged Tag	Text entry or Tag	Assign a tag whose value is 1 when the alarm state is Acknowledged, and is 0 when the alarm state is Unacknowledged. To browse to select the acknowledged tag, click the ellipsis (...) button to open the **Tag Browser**.

Name	Setting Type	Description
Shelved Tag	Text entry or Tag	Assign a tag whose value is set to 1 when the alarm state is Shelved and to 0 when the alarm state is Unshelved. To browse to select the shelved tag, click the ellipsis (...) button to open the **Tag Browser**.

Control Tags tab

When a digital alarm is selected, the **Properties** column of Library Object Manager displays three tabs—**Digital**, **Status Tags**, and **Control Tags**.

Alarm control tags allow monitoring of alarms by using tags in the controller. When a tag is assigned as a control tag, when the value of the tag changes to positive, the server automatically acknowledges, disables, enables, suppresses, unsuppresses, shelves, or unshelves all alarms associated with the control tag.

The **Control Tags** tab includes **Auto Reset** checkboxes. Select each box to automatically reset the corresponding tag to 0 when the tag goes back into alarm. Otherwise the tag must be manually reset. Keep in mind, there are no restrictions for using the same tag in multiple places. However, when using the same tag as a control tag and status tag with the auto reset enabled, the alarm state could be incorrect. To avoid problems, clear the **Auto Reset** checkbox when using the same tag as a control tag and status tag.

The ellipsis (...) buttons following the text entry settings provide additional dialog boxes to assist in specifying the property.

Click **Apply** to apply changes made in these tabs.

This table describes the settings in the **Control Tags** tab.

Name	Setting Type	Description
Disable Tag	Text entry or Tag	Select the tag to disable an alarm. Disabling an alarm stops the alarm condition from being evaluated and places it in the default state of Normal. To browse to select the disable tag, click the ellipsis (...) button to open the **Tag Browser**.
Enable Tag	Text entry or Tag	Select the tag to enable an alarm. Enabling an alarm adds the alarm condition to the events that are reported to the alarm reporting objects. To browse to select the enable tag, click the ellipsis (...) button to open the **Tag Browser**.
Suppress Tag	Text entry or Tag	Select the tag to suppress an alarm. Suppress an alarm that is not needed

Name	Setting Type	Description
		temporarily, for example, because the alarm is caused by another alarm that is already being attended to. Suppressed alarms continue to be logged and evaluated, but must be unsuppressed to return to active alarm status. To browse to select the suppress tag, click the ellipsis (...) button to open the **Tag Browser**.
Unsuppress Tag	Text entry or Tag	Select the tag to unsuppress an alarm. To browse to select the unsuppress tag, click the ellipsis (...) button to open the **Tag Browser**.
Acknowledge Tags - All Levels	Text entry or Tag	Assign a tag to acknowledge all levels of alarms when the tag value is positive. To browse to select the acknowledge all levels tag, click the ellipsis (...) button to open the **Tag Browser**.
Shelve Tags - All Levels	Text entry or Tag	Assign a tag to shelve all levels of alarms when the tag value is positive. To browse to select the shelve all levels tag, click the ellipsis (...) button to open the **Tag Browser**.
Shelve Duration	Text entry or Tag	Type a value or assign a tag to specify for which period (in minutes) the alarm should be shelved. To browse to select the shelve duration tag, click the ellipsis (...) button to open the **Tag Browser**.
Unshelve All Tag	Text entry or Tag	Assign a tag to unshelve all alarms when the tag value is positive. To browse to select the unshelve all tag, click the ellipsis (...) button to open the **Tag Browser**.

Message Editor dialog box

Alarm messages describe alarm conditions. Use the **Message Editor** to create the messages needed for different alarms. Each alarm can be associated with only one message.

This table describes the settings in the **Message Editor**.

Name	Setting Type	Description
Message	Text entry	The message appears here as it is compiled. Text can be entered directly in this setting. Variable tags are added when **Add** is clicked.
Add Variable		
Variable	Dropdown menu	A list of tags that can be added to the message. This includes a number of default variables, as well as up to four variables added defined by **Associated Tags** settings in the **Properties** panel.
Data Type	Dropdown menu	Where applicable based on the Tag selected, offers a choice of numeric or string for the data type of the Tag value within the message.
These settings appear if **String** is selected in the **Data Type** setting.		
Use Fixed Width	Checkbox	Limits the amount of information displayed for the tag value within the message to the value entered in the **Number of Characters** setting.
Number of Characters	Text entry	Sets the maximum number of characters to display for the tag value within the message.
These settings appear if **Numeric** is selected in the **Data Type** setting.		
Number of Digits	Text entry	Sets the number of digits to display for the tag value within the message.
Decimal Places	Text entry	Sets the number of decimal places to display for the tag value within the message.
Left Fill	Dropdown menu	Determines whether values will be filled in to match the value set for **Number of Digits**. Values used to fill in the space can be either zeros or spaces.
Add	Button	Adds the currently selected tag to the message.
ID	Text entry	The unique numeric ID for the message.

Add Alarms content

Add FactoryTalk View ME Alarms content to library objects to support plant control applications. Added contents include triggers and messages.

To add FactoryTalk View ME Alarms content to a library object

- In the **Library Content** column, right-click the library object and select **Add Section > ME Alarms**. A **ME Alarms** folder, with **Triggers** and **Messages** subfolders, is added to the library object.

Add a trigger to a library object

FactoryTalk View ME reads the tag's or expression's value, and generates an alarm when the value matches an alarm message's trigger value.

Prerequisites

- Add FactoryTalk View ME Alarms content to a library object on page 172.

To add a trigger to a library object

1. Right-click the **Triggers** subfolder and select **Add**. The **Trigger** dialog box opens.
2. In **Tag or Expression**, type a tag name or an expression, or click the ellipsis (...) button to choose a tag and click **OK**.
3. Click **OK**.
4. Configure the properties as needed.

Add a message to a trigger

Trigger messages describe alarm conditions and properties.

Prerequisites

- Add a trigger to a library object on page 173.

To add a message to a trigger

1. In **Library Repositories**, select a library object that includes FactoryTalk View ME Alarms content.
2. In **Library Content**, expand the **ME Alarms** folder and select the **Messages** subfolder.
3. Open the Message Builder on page 179 dialog box with one of the following methods:
   - Right-click **Messages** and select **Add**.
   - Click  in the Properties on page 177 column.
   - Right-click in the table of the Properties on page 177 column and select **Add**.
4. Fill in **Message Details** in the **Message Builder** dialog box.
5. Click **OK**.

Delete a trigger content

Delete a trigger content when it is no longer needed.

To delete a trigger content from a library object

1. In **Library Repositories**, select the library object that includes the trigger content to delete.
2. In **Library Content**, expand the **ME Alarms** folder and the **Triggers** subfolder.
3. Right-click the trigger content and select **Delete**.
4. Select **Yes**.

Delete a message content

Delete one or more messages when they are no longer needed.

To delete a message content from a library object

1. In **Library Repositories**, select the library object that includes the message content to delete.
2. In **Library Content**, expand the **ME Alarms** folder and click the **Messages** subfolder.
3. In the **Properties** panel, select one message item or select **Ctrl** to select multiple items.
4. Delete items with one of the following methods:
   - Right-click the selected items and select **Delete**.
   - Click  .
5. In the **Delete Message** dialog box, click **Yes**.

Triggers properties

How do I access Triggers properties?

1. In **Library Repositories**, select a library object that includes **ME Alarms**.
2. In **Library Content**, expand the **ME Alarms** folder and the **Triggers** folder.
3. In the **Triggers** folder, select the trigger object.

When a trigger is selected, the **Properties** column of Library Object Manager displays the properties of the trigger.

The ellipsis (...) buttons following the text entry settings provide additional dialog boxes to assist in specifying the property.

Click **Apply** to apply changes made in this column.

This table describes the settings in the **Properties** column.

Name	Setting Type	Description
Tag or Expression	Text entry or expression	Displays the tag name or expression associated with the connection. To create an expression, click the ellipsis (...) button to open the Expression Builder on page 59.
Include Condition	Text entry or expression	Sets the condition under which the current trigger is initiated. The default is **Always**. A condition can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.

Name	Setting Type	Description
Trigger Type	Dropdown menu	Select the type of data the trigger's tag or expression uses: • **Value**—integer or floating point values. Floating point values are rounded to the nearest integer. Use with analog or digital tags. • **Bit**—a bit array consisting of one or more bit positions. Use this trigger type to generate multiple alarm messages with a single tag (or array tag) or expression. Each bit in the array whose value changes from 0 to 1 triggers an alarm if a message is configured for the bit. • **LSBit** (least significant bit)—an analog or digital tag, or a direct reference bit array consisting of one or more bit positions. Use this trigger type to trigger alarms based on a priority sequence that is determined by bit position. When multiple bits in the array change from 0 to 1, only the alarm with the lowest bit position is triggered. The LSB trigger type property does not work with tags that use the Default data type. The default data type is Floating Point, and LSB does not support Floating Point. Bit and LSB trigger types can be used with a digital HMI tag or a direct reference using the array syntax.
Trigger Label	Text entry or expression	Type a label for the trigger, up to 40 characters long. Do not use commas.
Use ack all value	Checkbox	Select this checkbox to send a value to the data source when the operator presses the Ack All (Acknowledge All Alarms) button or when a RemoteAckAll connection occurs.
Ack All Value	Text entry	In the box, type the integer value to send.

Name	Setting Type	Description
Optional trigger connections	Text entry or expression	Specify the following optional trigger connections: • Handshake Tag—a write connection. Assign a digital tag to this connection to notify the data source when the trigger connection's value changes. • Ack Tag—a write connection. Assign a digital or analog tag to this connection to notify the data source when an alarm is acknowledged by the operator. • Remote Ack Tag/Exp—a read connection. Assign an analog tag or an expression to this connection to allow the data source to acknowledge alarms. • Remote Ack Handshake Tag—a write connection. Assign a digital tag to this connection to notify the data source when a remote acknowledgment occurs. • Message Tag—a write connection. Assign a tag to this connection. When the alarm is triggered, if the **Message to Tag** option is selected on the **Messages** tab, up to 82 characters of the message text is written to this connection. The maximum string length supported is 82 characters. Make sure that the tag supports the type of data sent in the alarm message. • Message Notification Tag—a read/write connection. Assign a digital tag to this connection if you want to use message handshaking. This connection notifies the operator that an alarm message has been sent. • Message Handshake Tag/Exp—a read connection. Assign a digital or analog tag or an expression to this connection. This connection

Name	Setting Type	Description
		provides the run-time terminal with confirmation that alarm messages are received by the data source. Do not configure an alarm's Trigger and Remote Ack settings to use the same tag. Doing so may cause unpredictable alarm acknowledgment behaviors.

Message properties

How do I access Message properties?

1. In **Library Repositories**, select a library object that includes **ME Alarms**.
2. In **Library Content**, expand the **ME Alarms** folder.
3. Select the **Messages** folder.

When the **Messages** folder is selected, the **Properties** column of Library Object Manager displays the properties of messages.

Click **Apply** to apply changes made in this column.

This table describes the elements in the **Properties** column.

Item	Description
	Add a message.
	Edit details of the selected message.
	Delete one or multiple messages.
Filter	Filter the messages.
Clear all identifiers	Clear all the alarm identifiers assigned to the alarm messages.
Trigger	Select the trigger for which to define messages.
Trigger Value	Enter a nonzero integer value or bit position. Bit position starts at 1. When the trigger's value changes to this value, or when the bit in this position changes from zero to 1, the associated message is generated. This column cannot be blank.
Message	Enter the message, up to 256 characters. To create a line break in the message, type *\n*. To show the character sequence in the message, type *\\n*. You cannot use tag placeholders (*#n*) in the variables that are embedded in an alarm message. PanelView Plus 7 Standard applications are limited to 500 alarm messages, and PanelView Plus 6 Compact applications are limited to 200 alarm messages.

Item	Description
	If a message uses an embedded variable to show the value of an alarm, and if that alarm is in alarm when the application starts up, the alarm banner will show question marks in place of the embedded variable. This is because the alarm system cannot determine when the alarm was triggered, and therefore cannot determine what the value of the variable was at that time. After the value has been reset to 0, the embedded variable value will show accurately the next time when the alarm is triggered.
Alarm Identifier	This column only shows when the Use Alarm Identifier checkbox is selected. Enter a value (1-32767) in this column to identify the alarm message. If this column is blank, the alarm identifier value is zero.
Display	Select this checkbox to open the alarm graphic display that is selected on the **Advanced** tab of the **Alarm Setup** editor when this alarm is triggered. The message is added to the alarm history and can be shown in any alarm list, alarm banner, and alarm status list objects even if this checkbox is clear and the graphic display is not opened. When the **Bit** trigger type is used, and multiple bits are triggered at the same time, the highest bit triggered message will be shown on the alarm banner display even if the **Display** checkbox of particular bits message is clear. Filtering alarms on the display object that opens automatically when alarms occur (specified on the **Triggers** tab) is not recommended. It is assumed that the user will want this display to register all alarms. If you filter alarms on this display for some reason, be sure to clear the **Display** checkboxes for all the alarms that have been filtered out. Otherwise the display will open automatically when the alarm occurs because the **Display** option for the alarm is selected, but the alarm object (banner or list) will be blank, because the alarm that was triggered has been filtered out for that object.
Audio	For applications running on Windows 7 SP1, Windows 10, Windows Server 2012 and Windows Server 2016 operating systems, select this checkbox to sound the run-time computer's internal beeper when the alarm is triggered. The beeper beeps continuously, five seconds on and five seconds off, until the alarm is silenced.
Print	Select this checkbox to print the alarm message when this alarm is triggered. Be sure that a printer is set up on the run-time terminal.

Item	Description
	You can print alarms in landscape mode with a line printer, but only one alarm is printed per page. If you set up alarm messages in multiple languages, all messages, trigger labels, and times are printed in the current application language.
Message to tag	Select this checkbox to send the text of the message to the message connection specified on the **Triggers** tab when the alarm is triggered. If you set up alarm messages in multiple languages, make sure that the data source can receive the message in Unicode format or convert it into ASCII characters for all the languages you will use.
Background	Click this button to select a color for the background of the message text.
Foreground	Click this button to select a color for the message text.

Message Builder

Use the **Message Builder** to fill in message details. The **Message Builder** dialog box includes a command toolbar, message details, and descriptions.

Item	Descriptions
	Sort message details by category.
	Sort message details by alphabetical.
	Open property pages.
Message Details	Fill in the following message details: • **Trigger**: Select one or all triggers. • **Trigger Value**: Edit or click the browse button to select one or more trigger value expressions in **Expression Builder**. This box shouldn't be empty. • **Message**: Edit or click the browse button to select one or more message expressions in **Expression Builder**. • **Alarm Identifier**: Edit or click the browse button to select one or more alarm identifier expressions in **Expression Builder**. • **Display**: Select **True** or **False**. • **Audio**: Select **True** or **False**. • **Print**: Select **True** or **False**. • **Message**: Select **True** or **False**. • **Background**: click the browse button to select a color. • **Foreground**: click the browse button to select a color.
Descriptions	Description of the selected item in **Message Details**.

Add FT Historian content

FactoryTalk Historian can be used to monitor tags on Logix objects. FactoryTalk Historian collects time series data from process equipment, manufacturing devices, and other main data sources that are important to an operation.

With this data when a specific event such as a downtime event, a bad batch, or an alarm occurs, process variables in the FactoryTalk Historian for the same time frame as the event can be reviewed and any correlations that might explain the event can be investigated and additional steps taken as needed to address any issues identified.

Follow these steps to add FactoryTalk Historian content.

- Right-click the library object in the **Library Content** column and select **Add Section > FT Historian.**
  An **FT Historian** folder, with the subfolder **Historian Tags**, is added to the library object.

Add a Historian tag to a library object

FactoryTalk Historian can be used to monitor tags on Logix objects. Select a library object that includes Logix tags to add Historian tags that monitor process variables.

Prerequisites

- Add FactoryTalk Historian content to a library object.

To add a Historian tag to a library object

1. Right-click the **Historian Tags** subfolder and select **Add**. The **Tag Browser** window opens.
2. In **Select the Tag to Log**, type the tag name or part of the tag name to filter the tags listed. To choose a tag that is a member of an array, expand the tag group name to see all of the members of the array.
3. Scroll in the list box and double-click the tag name to log.
4. Click **OK**. The Tag is added as an object in the **Historian Tags** folder.
5. Configure the properties as needed.

Delete a Historian tag

Delete a Historian tag when it is no longer needed.

To delete a Historian tag from a library object

1. In **Library Repositories**, select the library object that includes the Historian tag to delete.
2. In **Library Content**, expand the **FT Historian** folder and then expand the **Historian Tags** folder.
3. Right-click the object and then click **Delete**.
   A confirmation dialog appears. Click **Yes** to continue with the deletion.

   IMPORTANT: When a Historian Tag is added to a library object, a corresponding parameter is also added. When a Historian Tag is deleted, the corresponding library object parameter must be deleted as well.

4. Locate the parameter for the deleted Historian Tag in this subfolder in the **Library Content** column: **Definition Data > Parameters > Object > Historian Configuration**.

5. Right-click the parameter and select **Delete**.
   A confirmation dialog appears. Click **Yes** to continue with the deletion.

Historian tag properties

A number of the settings in the **Properties** panel are populated with default values. In a typical application, it is not necessary to change these values.

This table describes the settings in the **Properties** panel when a Historian Tag is selected.

Name	Setting Type	Description
Tag Name	Text entry or expression	The name that will be applied to the Historian Tag when the library object is instantiated. This name will appear in the Application Code Manager screens. The setting is populated with a default name based on the Tag name and library object for the selected tag. Can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the Expression Builder on page 59.
Include Condition	Text entry	Sets the condition under which the current Historian Tag is instantiated. The default is **Always**. A condition can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Scan Class	Text entry	The scan class for the Historian Tag. The scan class consists of a period in seconds that defines how often FactoryTalk Historian collects data for the object. The setting is populated with a default value based on the tag name for the selected tag. Can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Description	Text entry	A description for the tag. Can be entered manually or generated by an expression.

Name	Setting Type	Description
		To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Engineering Units	Text entry	The engineering unit for the tag. Can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Instrument Tag	Text entry	The ItemID that will be applied to the Historian Tag when the library object is instantiated. The setting is populated with a default name based on the tag name and library object for the selected tag. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
FTLD Interface Number	Text entry	The FactoryTalk Historian Live Data (FTLD) Interface number for the Historian Tag. The setting is populated with a default value based on the selected tag. Can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.
Point Type	Dropdown menu	The data type for the Historian point. Options are: • Digital • Float16 • Float32 • Float64 • Int16 • Int32 • String • Timestamp • Blob
Typical Value	Text entry	The typical value for the Historian tag. Can be entered manually or generated by an expression. To create an expression, click the ellipsis (...) button to open the **Expression Builder**.

Name	Setting Type	Description
Zero	Text entry	The zero value for the Historian Tag. This setting is only active if a **Float** or **Int** option is selected for **Point Type**.
Span	Read-only	The span for the Historian tag. This setting is only active if a **Float** or **Int** option is selected for **Point Type**.

Add Custom Collections content

You can create custom collections that can be used in documentation generation.

To add Custom Collection content

- Right-click the library object in the **Library Content** column and select **Add Section > Custom Collections**.
  An object named **Custom Collections** is added to the library object.

Add an item to Custom Collections

Once you apply the changes in **Library Object Manager**, the item you add to **Custom Collections** will display in the **Properties** panel of Application Code Manager.

To add an item in Custom Collections

1. In **Library Content**, Right-click **Custom Collections**, and then select **Add**.
   A folder named **NewCollection** is added and you can configure its properties in the Properties on page 183 panel.
2. Right-click **NewCollection**, and then select **Add**.
   An item named **NewItem** is added and you can configure its properties in the **Properties** panel.

Item properties

The following table displays the options in the **Properties** panel of an item object.

Options	Descriptions
Item Name	Displays the name of the item. A condition can be entered manually or generated by an expression. To create an Expression, click the ellipsis (...) button to open the **Expression Builder**.
Include Condition	Displays the Include Condition of the item. The default is **Always**. A condition can be entered manually or generated by an expression. To create an Expression, click the ellipsis (...) button to open the **Expression Builder**.
Add New Attribute	Opens **Attribute Builder**.
Edit Attribute	Opens **Attribute Builder**.
Delete Attributes	Deletes an attribute.

Options	Descriptions
Filter	Enter the key words or the name of an attribute to search it quickly.
Name	Displays the name of an attribute.
Value	Displays the value of an attribute.

Add an attribute

Use the **Attribute Builder** dialog box to add an attribute to an item.

To add an attribute

1. In the **Properties** panel of an item, select **Add New Attribute**.
2. In the **Attribute Builder** dialog box, enter the name of the attribute and the value can be entered manually or generated by an expression.
3. Select **OK**.

Add Attachments

How do I open Add Attachments?

- In **Library Content**, right-click **Attachments**, and then select **Add**.

Use **Add Attachments** to add, edit, remove, import, or export file attachments to or from an Application Code Manager database.

Button or Column	Description
Add New Attachment	Opens **Attachment** to add an attachment file to the database.
Edit	Opens **Attachment** to edit the properties of an existing attachment file.
Extract Files	Extracts the selected attachment files to a folder location. The output files are extracted to their original filenames and formats.
Export to .HZ1 Attachment Files	Exports the selected attachment files to a folder location. The output files are exported with filenames matching the file ID in the .HZ1 format.
Import from .HZ1 Attachment Files	Imports one or more .HZ1 format attachment files from a folder location.
Filter	Limits the attachment list to attachments with properties matching the characters entered in the **Filter** field. The filter function uses exact character matching and is case insensitive. Wildcard characters are not supported.
Select	Toggles whether a row is selected before deleting or exporting the attachments.
Name	Displays the name of the attachment.
Description	Displays the description of the attachment.

Button or Column	Description
File Name	Displays the original file name of the attachment.
File ID	Displays the unique file identification number of the attachment.
Include Condition	Displays the Include Condition of the attachment. The default is Always. To create an Expression, click the ellipsis (...) button to open the **Expression Builder**. Double-click the selected attachment to open **Attachment**, and then you can configure the attachment. A condition can be entered manually or generated by an expression.
Extraction Path	Displays the extraction path of the attachment. To create an Expression, click the ellipsis (...) button to open the **Expression Builder**. Double-click the selected attachment to open **Attachment**, and then you can configure the attachment. A path can be entered manually or generated by an expression.
Revision Description	Displays the latest revision description of the attachment.
Modified Date	Displays the last modified date of the attachment.
Modified By	Displays the last user that modified the attachment.

Library Object Import Wizard

How do I open the Library Object Import Wizard?

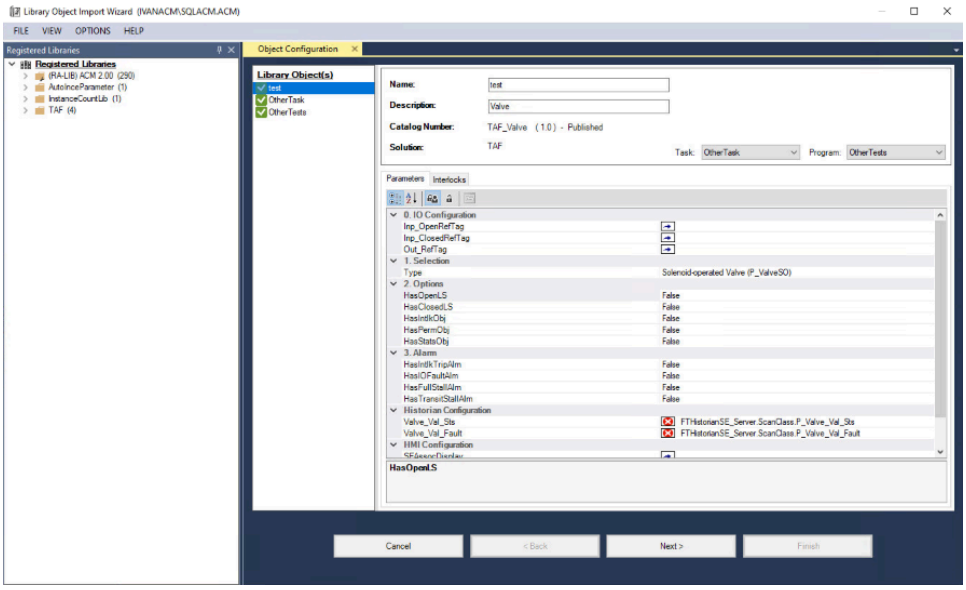
- In **Logix Designer**, right-click the ACD file object and select **Plug-Ins > Import Library Objects**.

Use the **Library Object Import Wizard** to connect to an ACM database and add library objects to the ACD file on page 189.

The Library Object Import Wizard includes:

- **Registered Libraries** that display all libraries (classes) in the connected ACM database in a tree view. Select library objects to add them to the ACD file.
- **Content Configuration** that display the properties and parameters of the selected objects. Edit the properties and parameters of the objects and link the objects to other objects or the ACD file.

Components on the wizard



This table describes each section of the Library Object Import Wizard on page 187.

Item	Description
Menu bar	
FILE	• **Connect** Edit database connection properties.

Item	Description
	• **Exit** Exit the **Library Object Import Wizard**.
VIEW	• **Restore Default Layout** Select to restore the default layout of the **Library Object Import Wizard**.
OPTIONS	• **Include Project Data** Select to include all instances with their parameter values. All libraries (compressed) will be included as part of the Controller's Custom Properties.
Pane	
Registered Libraries	Displays all libraries (classes) in the connected ACM database in a tree view. Select library objects to add them to the ACD file. Right-click any branch in the **Registered Libraries** tree view to view the following commands: • **Register** Registers one or more library files (HSL4). Required when a library object is provided as an HSL4 file and needs to be incorporated into ACM. • **Extract Attached Files** Extracts the attached files in the selected library to a folder. • **Refresh** Refreshes the tree view.
Object Configuration	Displays the properties and parameters of the selected objects. Edit the properties and parameters of the objects and link the objects to other objects or the ACD file. This pane includes: • **Library Object(s)** • Properties of the selected library object
Object Configuration > Unresolved parameters	Displays parameters, which might substitute predefined or base library parameters. Assign new values to these parameters and click **Next**.
Object Configuration > Merge Actions (Multiple objects only)	Displays merge actions of all parameters. Use this pane to add dependencies, overwrite dependencies, or merge existing Tasks, Programs, and Routines. Merge actions include: • **Add** Imports the new content. • **Merge** (Existing Task, Program, and Routines only) Merges the content and appends content to the bottom. • **Skip** Skips the content importing. • **Use Existing**

Item	Description
	Uses the existing data type or AOI. • **Overwrite** Overwrites the existing data type or AOI with the new content.
Object Configuration > Content Preview	Displays a preview of the content, which will be inserted to the ACD.

Add library objects to an ACD file

Use the Library Object Import Wizard on page 187 to add library objects to an ACD file.

To add library objects to an ACD file

1. In **Logix Designer**, right-click the ACD file object and select **Plug-Ins > Import Library Objects**.
2. In the **Library Object Import Wizard**, do one of the following:
   - In **Registered Libraries**, select one or more library objects. Drag the objects to the **Library Object(s)** column.
   - In **Registered Libraries**, double-click a library node.
   - In the **Library Object(s)** column, right-click the blank cell and select **Add New**.
3. Configure the parameters of every object and click **Next**.

   Tip:
   - When adding one object, all parameters are shown in the **ACD Tag Browser**.
   - When adding two or more objects, link the current library objects to other library objects with one of the following methods:
     - In **IO Configuration**, select **Link to Existing instance** to link to other library objects added to the **Library Object(s)** column.
     - In **IO Configuration**, select **Link to an ACD instance** to link to the existing controller or program tags. Choose a tag in the **ACD Tag Browser** dialog box.
   - Click the **Override Calculated Parameters** button to unlock the calculated parameters.

4. (optional) Configure values of unresolved parameters and click **Next**.

   Tip:
   - Double-click the unresolved parameter to open the Edit Unresolved Name Substitution dialog box.
   - In some cases such as a conditional expression, the unresolved parameter value may be part of the substitution to simplify resolving the value. Double-click in the Text field in the occurrences section that will automatically populate the value field for further editing.

5. (optional) Merge the new content with the existing content and click **Next**.
6. Configure the content preview and click **Next**.
7. (optional) Check the import log file if the import process fails. Close the **Library Object Import Wizard** and open the ACD file to discard the failed changes.

Legal Notices

Rockwell Automation publishes legal notices, such as privacy policies, license agreements, trademark disclosures, and other terms and conditions on the Legal Notices page of the Rockwell Automation website.

Software and Cloud Services Agreement

View and sign the Rockwell Automation Software and Cloud Services Agreement here.

Third-Party Software Licenses

View a full list of all third-party software used in this product in C:\Program Files\Common Files\Rockwell\Help \Application Code Manager\ReleaseNotes\OPENSOURCE\third-party_attributions.txt, including:

- Open-source software
- Commercial software
- Other third-party free software

You may obtain Corresponding Source code for open-source packages included in this product from their respective project websites. Alternatively, you may obtain the complete Corresponding Source code by contacting Rockwell Automation via the Contact form at the bottom of the Rockwell Automation website: http://www.rockwellautomation.com/global/about-us/contact/contact.page. Please include "Open Source" as part of the comments for your General Inquiry request.

Rockwell Automation Support

Use these resources to access support information.

Technical Support Center	Find help with how-to videos, FAQs, chat, user forums, and product notification updates.	rok.auto/support
Knowledgebase	Access Knowledgebase articles.	rok.auto/knowledgebase
Local Technical Support Phone Numbers	Locate the telephone number for your country.	rok.auto/phonesupport
Literature Library	Find installation instructions, manuals, brochures, and technical data publications.	rok.auto/literature
Product Compatibility and Download Center (PCDC)	Get help determining how products interact, check features and capabilities, and find associated firmware.	rok.auto/pcdc

Documentation feedback

Your comments help us serve your documentation needs better. If you have any suggestions on how to improve our content, complete the form at rok.auto/docfeedback.

Waste Electrical and Electronic Equipment (WEEE)

At the end of life, this equipment should be collected separately from any unsorted municipal waste.

Rockwell Automation maintains current product environmental information on its website at rok.auto/pec.

Rockwell Otomasyon Ticaret A.Ş. Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400 EEE Yönetmeliğine Uygundur

Connect with us.

rockwellautomation.com — expanding **human possibility**™

AMERICAS: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
EUROPE/MIDDLE EAST/AFRICA: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
ASIA PACIFIC: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

LOGIX-UM006K-EN-E - March 2025
Supersedes Publication LOGIX-UM006J-EN-E - March 2024