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.

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.

<table>
<thead>
<tr>
<th>WARNING:</th>
<th>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.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTENTION:</td>
<td>Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Atentions help you identify a hazard, avoid a hazard, and recognize the consequence</td>
</tr>
<tr>
<td>Important:</td>
<td>Identifies information that is critical for successful application and understanding of the product.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>SHOCK HAZARD:</th>
<th>Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.</th>
</tr>
</thead>
<tbody>
<tr>
<td>BURN HAZARD:</td>
<td>Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.</td>
</tr>
<tr>
<td>ARC FLASH HAZARD:</td>
<td>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).</td>
</tr>
</tbody>
</table>
# Table of contents

## Library Designer
- Library Designer overview ........................................................................................................ 11
  - Decoration types ........................................................................................................... 12
  - Navigate the Library Designer interface ...................................................................... 16

## Use inclusions with library objects
- Use inclusions with library objects ......................................................................................... 19
  - View ownership for a selected object ............................................................................. 20
  - Library Ownership window ......................................................................................... 20
  - Library Ownership Conflicts window ........................................................................... 21

## Library objects
- Library objects ............................................................................................................................. 23
  - Create a new empty library object ................................................................................. 24
  - Add selected objects to library objects ........................................................................... 25
  - Copy and paste a library ................................................................................................. 28
  - Add a child object or dependency to an existing library object .................................... 28
  - Add a module to a library object .................................................................................... 29
  - Order programs in a library when content type is "Program" ........................................ 30
  - Create a new axis ID for a motion tag ............................................................................ 30
  - Show library object dependencies ............................................................................... 31
  - Show/Hide un-resolved dependencies ......................................................................... 31
  - 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 library object parameters file ............................................................. 35
  - Publish a library ............................................................................................................ 36
  - Library Object parameters ............................................................................................ 37
  - Validate Libraries .......................................................................................................... 38
  - Validation Error/Warning list ......................................................................................... 40
  - Delete a library object .................................................................................................... 40
  - Assign to/Create New window ....................................................................................... 41
  - Module Wizard overview ............................................................................................... 42
  - Connection Properties settings ....................................................................................... 43
  - Advanced Connection Properties settings ...................................................................... 44
  - Library Import Configuration window ............................................................................ 48
  - Export Members ............................................................................................................. 48
  - Import Members .............................................................................................................. 49
Chapter 4

Decorator Panel

Decorator panel .................................................................................................................. 51
Decorator Panel settings ............................................................................................. 51
Change the columns displayed in a tab ........................................................................ 53
Use find to limit the display based on a text string ..................................................... 53
Use expressions and functions .................................................................................. 54
Configure an element substitution ................................................................................ 61
References window ...................................................................................................... 62
Library Elements - Substitution Builder window ....................................................... 62
Decorator Panel - Parameters tab ................................................................................ 65
Add a new parameter .................................................................................................... 66
Edit a parameter .......................................................................................................... 66
Add an external reference ............................................................................................ 67
Copy a parameter ........................................................................................................ 67
Add a new group ........................................................................................................... 68
Reassign a parameter to a group .................................................................................. 69
Move a parameter within a group ................................................................................ 69
Delete a group .............................................................................................................. 70
Add a subobject .......................................................................................................... 70
Limit the external references available to a parameter ................................................ 71
View a parameter reference ....................................................................................... 72
Delete a parameter ...................................................................................................... 72
Add New / Edit Parameter window ................................................................................ 73
New / Edit SubObject window ...................................................................................... 75
Parameter Filter Builder window ................................................................................... 76
Decorator Panel - Functions tab ................................................................................... 77
Add a new function ....................................................................................................... 78
Add branches to a conditional statement .................................................................... 79
Use a previously created function ............................................................................. 80
Copy a function ........................................................................................................... 80
Edit a function .............................................................................................................. 81
View a function reference ............................................................................................. 82
Delete branches from a conditional statement ............................................................ 82
Delete a function .......................................................................................................... 83
Function Builder window ............................................................................................. 83
Decorator Panel - Substitution tab ............................................................................... 85
Add a substitution ........................................................................................................ 86
Change the order of execution for a substitution ........................................................ 87
Copy a substitution to a different library object .......................................................... 88
Edit a substitution ........................................................................................................ 88
Delete a substitution .................................................................................................... 89
Member Selector window .............................................................................................. 89
Substitution tab - Substitution Builder window .......................................................... 89
Decorator Panel - Predefined tab ................................................................................ 90
Decorator Panel - External References tab .................................................................... 92
Add New / Edit Parameter window ................................................................................ 93
# Table of contents

Delete an external reference ................................................................. 93
Reference Builder window ................................................................. 93
Decorator Panel - Linked Libraries tab .................................................... 94
  Add a linked library ........................................................................ 95
  Add parameter links ....................................................................... 96
  Add New / Edit Linked Library window ........................................ 97
Parameters browser window ............................................................... 98
Decorator Panel - Interface tab .......................................................... 99
  Use interfaces to standardize data structures ................................... 100
  Add an interface ........................................................................... 101
  Delete or edit interfaces ................................................................. 102
  Add interface members .................................................................. 103
  Add New/Edit Interface window .................................................. 103
  Edit Interface window .................................................................. 104
Interface Member window ................................................................. 105
  Tags window .................................................................................. 106
Decorator Panel - Library Object Properties ......................................... 106
  Change the name or description ..................................................... 111
  Add a tag as a parameter ............................................................... 112
  Add a tag as an external reference .................................................. 113
  Apply decoration to tag values ...................................................... 113
  Apply decoration to a tag alarm condition ..................................... 114
  Parameterize tag extended properties .......................................... 114
  Exclude inherited substitutions ....................................................... 115
  Set a rule for instantiation ............................................................. 116
  Apply a substitution to an element ................................................ 117
  Activate a new element .................................................................. 118
  Extended Properties ....................................................................... 119
  Alarms ............................................................................................ 119
  Edit Alarm ....................................................................................... 121

## Chapter 5

**Library Object Manager**

Library Object Manager ............................................................................. 123
Navigate the Library Object Manager interface .................................... 124
  Set the default options for library object classification ...................... 125
  Create a list of options .................................................................... 126
  Settings dialog box reference .......................................................... 126

## Chapter 6

**Library Repositories**

Library Repositories ................................................................................ 129
  Add and mount an ACD repository .................................................. 129
  Unmount an ACD repository ......................................................... 130
  Move between the Library Object Manager application and the Studio 5000 Logix Designer application ................................................. 131
Move between the Library Object Manager application and the Library Designer ................................................................. 131
Export an ACD repository to a L5X format .......................................................................................................................... 132
Remove an ACD repository from the Library Object Manager application ............................................................................. 132
Add a group ........................................................................................................................................................................... 133
Rename a group .................................................................................................................................................................... 133
Delete a group .............................................................................................................................................................. 133
Add a folder repository .............................................................................................................................................. 134
Remove a folder repository from the Library Object Manager application .................................................................................. 134
Add the ACM database as a repository ........................................................................................................................... 135
Remove the Application Code Manager database from the Library Object Manager application ........................................... 136
Connection Properties settings .......................................................................................................................................... 136
ACM Database - Advanced Connection Properties settings ................................................................................................ 137
The Library Object Manager Application: Adding Repositories and Copying Library Objects .................................................... 141
Create a library object ............................................................................................................................................................ 141
Copy a library object from the ACD repository to a folder repository or the ACM database .......................................................... 141
Copy multiple library objects from the ACD repository to a folder repository or the ACM database ........................................ 142
Copy multiple library objects between folder repositories and the ACM database ................................................................. 143
New Library settings ......................................................................................................................................................... 144
Library Import Configuration window .................................................................................................................................. 145

Chapter 7

Library Attachments

Library Attachments ................................................................................................................................................................. 147
Add an attachment ................................................................................................................................................................. 147
Assign an attachment to a library .......................................................................................................................................... 148
Edit an attachment ............................................................................................................................................................. 148
Change the include condition of an attachment reference .................................................................................................. 149
Delete an attachment ......................................................................................................................................................... 149
Delete an attachment reference from a library ....................................................................................................................... 150
Extract an attachment ......................................................................................................................................................... 150
Import an attachment from an .HZ1 file .............................................................................................................................. 150
Export an attachment to an .HZ1 file .................................................................................................................................... 151
Attachments .............................................................................................................................................................................. 151
Attachment.............................................................................................................................................................................. 152
Chapter 8

Library Content

Library Content ................................................................................................................ 155
Review the decoration settings of a library object ....................................................... 156
View the Logix code of a library object ........................................................................ 157
Expression Builder ....................................................................................................... 157
Add FactoryTalk View SE/FactoryTalk View Machine Edition content to a library object .............................................................. 158
FT View substitutions ................................................................................................. 159
Apply substitutions to a symbol ................................................................................. 160
Configure the object tag and path parameters for a symbol ........................................ 160
Apply positioning to a symbol .................................................................................... 162
Delete a symbol from a library object .......................................................................... 163
Update a symbol ........................................................................................................... 164
Symbol properties ....................................................................................................... 164
Symbol builder dialog box ......................................................................................... 165
Add FactoryTalk Alarms and Events content to a library object ................................ 166
Add a digital alarm to a library object ......................................................................... 166
Add a new message to a digital alarm .......................................................................... 167
Create a message using the Message Editor ............................................................... 169
Add an existing message to a digital alarm ................................................................. 170
Delete message content from a library object ............................................................ 171
Change the tag update rate for a digital alarm ............................................................ 171
Digital alarm properties - Digital tab .......................................................................... 172
Digital alarm properties - Status Tags tab ................................................................... 174
Digital alarm properties - Control Tags tab ............................................................... 175
Message Editor dialog box ......................................................................................... 176
Add FactoryTalk Historian content to a library object .............................................. 177
Add a Historian tag to a library object ........................................................................ 177
Delete a Historian tag from a library object ............................................................... 178
Historian tag properties ............................................................................................. 179
Add Custom Collections content to a library object ................................................. 180
Add an item to Custom Collections .......................................................................... 180
Item properties ............................................................................................................. 181
Add a new attribute .................................................................................................... 181
Add Attachments ......................................................................................................... 181

Appendix A

Legal Notices

Legal Notices .................................................................................................................. 183
Library Designer

Library Designer overview

Use the 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. Example: those that support function 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. The 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 the Library Designer. Decorations include parameters, sub-objects, functions, substitutions, and external references. Decoration lets the library object be configured when it is implemented in a project in the ACM application.

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. 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 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. 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 the Application Code Manager application 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. 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.

See also

Navigate the Library Designer interface on page 16

Decorator Panel on page 51

Decoration types

Decoration is added to the library object in the 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 the Application Code Manager application, and are not accessible once the completed project is exported to code.

Parameters created in the Library Designer have these functions:

• Storing information that is pertinent to the specific instance of the library object, but that is not functional. Example: the customer contact information for a project.
• Differentiating each instance of a library object in a project. Example: the slot location of module object.
• Configuring each instance of a library object in a project. 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 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.

Both types of function 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 the Library Object Manager application. Saved functions are available to all projects opened in the 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 naming conventions and standards used are sufficient to prevent conflicts when using substitutions.

Substitution which 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 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 parameter value 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.

See also

Use expressions and functions on page 54
Substitution Builder window on page 62

Navigate the Library Designer 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.

<table>
<thead>
<tr>
<th>Interface element</th>
<th>Description</th>
</tr>
</thead>
</table>
| **1** **Selected Objects toolbar** | The Selected Objects toolbar has three elements:  
  • **Show/Hide Decorated Elements**: Toggles the display of items in the **Selected Object** tree view which are associated with a library object. Default is for these items to display as blue or green. Click once to hide them and again to restore the display.  
  • **Add selected elements to a Library**: Opens the **New Library** window for the selected element. This is one of three ways an item in the **Selected Object** 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. |
| **2** **Selected Objects tree view** | The **Selected Objects** tree view follows the structure of the Controller Organizer view or the Logical Organizer view in the Studio 5000 Logix Designer application. 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. Example: an Add-On Instruction may be associated with both a valve library object and a motor library object.  
  Click the + icon to the left of an item in the tree to display elements that are contained within it. Click the - icon to collapse the item.  
  Objects are color coded to indicate whether all, some, or none of the associations from the initial instance in the Studio 5000 Logix Designer application have been replicated in the library objects added in the Library Designer.  
  • Green. Objects which are fully associated within the Library Designer.  
  • Blue. Objects where some, but not all, of the associations have been replicated.  
  • Black. Objects which have no associations. |
| **3** **Library Objects toolbar** | The **Library Objects** toolbar has two elements:  
  • **Create a New Library**: Opens the **New Library** window to create a new empty library object.  
  • **Delete Selected Objects**: Deletes the selected library objects. |
### 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
    - Logic Object library objects: Tasks, Programs, Modules

A project can include one project library object, one controller library object, and multiple Logix object library objects. 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.

### Decorator panel

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 which are locked for editing appear dimmed. Settings which can accept calculated values are followed by the ellipsis (...) button. Clicking this button opens the Expression Builder.

### 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.

---

### See also

- **Use inclusions with library objects** on page 19
- **Library objects** on page 23
- **Decorator panel** on page 51
- **Expression Builder window** on page 56
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 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.

<table>
<thead>
<tr>
<th>Selection</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Children</td>
<td>When this command is active, all elements contained within the selected item in the Selected Object Tree View will be included when it is added to a library object.</td>
</tr>
<tr>
<td>Add Dependencies</td>
<td>When this command is active, all elements referenced by the selected item in the Selected Object Tree View will be included when it is added to a library object.</td>
</tr>
<tr>
<td>Allow Shared Ownership</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>• When the <strong>Allow Shared Ownership</strong> command is activated, items which were previously limited to a single library object may be added to other library objects.</td>
</tr>
<tr>
<td></td>
<td>• When the <strong>Allow Shared Ownership</strong> 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.</td>
</tr>
<tr>
<td>Container Mode</td>
<td>Container mode allows for objects to be grouped within a container so that any changes such as additions or deletions made within Studio 5000 to an object which 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.</td>
</tr>
</tbody>
</table>
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.

See also

Library Ownership window on page 20

Library objects on page 23

View ownership for a selected object

The Library Ownership 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 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.

See also

Library Ownership window on page 20

Library Ownership Conflicts window on page 21

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.
Use inclusions with library objects

### Column Description

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the item.</td>
</tr>
<tr>
<td>Type</td>
<td>The type of the item.</td>
</tr>
<tr>
<td>Owner Library</td>
<td>The name of the library that contains the library object that references the item.</td>
</tr>
<tr>
<td>Navigate</td>
<td>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 located in the library object structure, and the context of its inclusion in the library object.</td>
</tr>
</tbody>
</table>

### See also

- View ownership for a selected object on page 20
- Library Ownership Conflicts window on page 21

### Library Ownership Conflicts window

Ownership is also tracked to verify that parent and child objects remain associated with the same library object.

For example, if a main task and its child program are associated with a library object in the Library Designer, and the child program is reassigned to a different task within the Studio 5000 Logix Designer application, the Library Designer displays the **Library Ownership Conflicts** window the next time it is opened.

The **Library Ownership Conflicts** window displays a grid with this information.

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the library object.</td>
</tr>
<tr>
<td>Type</td>
<td>The type of library object.</td>
</tr>
<tr>
<td>Conflict</td>
<td>The library ownership relationships that are in conflict.</td>
</tr>
</tbody>
</table>
| Action          | The method to use to resolve the conflict. There are two choices available in the Action menu:  
- **Delete decoration**: Removes the child object from the library object it is associated with. The child object can then be reassigned in the Library Designer. The object displays in the Selected Object column as black, indicating it is no longer decorated.  
- **Re-Assign decoration**: The child object is reassigned to the same library object as its parent object. All existing decoration is removed or replaced by decoration inherited from the parent object. Only available when the parent and child objects are associated with different library objects. |
| Resolution      | The impact that taking the selected action will have on the library object. |

### To resolve Library Ownership conflicts

- In the **Library Ownership Conflicts** window, select the appropriate **Action** and then click **OK**.

### See also

- Ownership on page 20
- View ownership for a selected object on page 20
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 (example: Logix) and HMI code (example: FactoryTalk View Machine Edition or FactoryTalk View 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 library objects separated, make the library a task type.

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

This list describes the possible actions.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete</td>
<td>Deletes the selected item.</td>
</tr>
<tr>
<td>Paste</td>
<td>Available after copying an object from the <strong>Selected Objects</strong> column. Pastes the copied item(s) from the <strong>Selected Objects</strong> column into the selected library object.</td>
</tr>
<tr>
<td>Hide/Show un-resolved dependencies</td>
<td>Hides or displays the dependencies for the selected object.</td>
</tr>
<tr>
<td>Export Parameters</td>
<td>Exports the parameters of the selected object as an Excel file.</td>
</tr>
<tr>
<td>Import Parameters</td>
<td>Imports parameters from an Excel file to the selected object.</td>
</tr>
<tr>
<td>Create Empty Parameters Spreadsheet</td>
<td>Generates an empty parameters spreadsheet in Excel.</td>
</tr>
<tr>
<td>Publish Library</td>
<td>Publishes the selected library object to a specified location. If publishing the library results in duplicated task names, a warning message is displayed.</td>
</tr>
</tbody>
</table>

**See also**

- [Decoration types](#) on page 12
- [Use expressions](#) on page 54
- [Create a new empty library object](#) on page 24
- [Add selected objects to library objects](#) on page 25
- [Add a child object or dependency to an existing library object](#) on page 28

**Create a new 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 a new 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.
- Only one project and controller library object can exist within an ACD file. If the ACD file already contains a project or controller library object, these commands are dimmed in the menu.

The Assign to/Create New 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.

3. Select the **Content Type** and **Library 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.

**See also**

- Assign to/Create New window on page 41
- Delete a library object on page 40

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 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 sub-folder 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 Object 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 Object tree view when it is added to a library object.
   
   - Select Allow Shared Ownership to allow the selected item in the Selected Object 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 automatically add all program tags and routines to the library object. When this option is inactive, content must be manually added to the library object.

2. In the Selected Object tree view, select the item or items to add. Hold down the SHIFT or CONTROL 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 pull-down 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.

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
- 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 **Ellipsis** 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.

**Note:** This feature is only available if the product has a Standard activation license and is not available in Lite mode.

**To copy and paste a library**

1. In **Library Objects**, expand **Custom Properties**, and then expand **Libraries**.
2. Right-click the library object 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.

See also

- Inclusions menu on page 19
- Expression Builder window on page 56

Add a child object or dependency to an existing library object

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

Example: if a valve object has new rungs added to a contained Ladder Logic Diagram in the Studio 5000 Logix Designer application, the new rungs must also be added to the valve library object in the 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 the Library Designer.

- Items which have been newly created in the Studio 5000 Logix Designer application display as black.
- Items which have newly added associations in the Studio 5000 Logix Designer application which have not been replicated in the Library Designer display as blue.
- Items which 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 Object** 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.

**See also**

- Add a module to a library object on page 29
- Create a new empty library object on page 24
- Delete a library object on page 40

**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 subobject(s) in the library object.

See also

Add selected objects to library objects on page 25
Module Wizard overview on page 42

Order programs in a library when content type is "Program"

When a library is of content type "Program" in the Library Designer, you can rearrange the order 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 order of programs.

4. Select OK.

5. In the Library Designer window, select Apply to save changes.

See also

Create a new empty library object on page 24

Create a new 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 automatically added to the library. 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 a new 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.

**See also**

[Expression Builder window](#) on page 56

If the inclusion option **Add Dependencies** was selected when the library object was added, all dependencies were automatically included with the object.

Use the **Show Dependencies** setting to display the dependent objects.

**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 **Settings > Show Dependencies**.

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

2. A confirmation message is displayed, informing you that showing all library object dependencies will affect performance. Click **OK**.

   Showing dependencies may take some time.

**See also**

[Include dependent objects](#) on page 32

[Library objects](#) on page 23

**Show/Hide un-resolved dependencies**

If the inclusion option **Add Dependencies** was selected when the library object was added, all dependencies were automatically included with the object. If the **Add Dependencies** option was not selected, then dependencies (child objects) referenced by the library objects are considered un-resolved because the value of those objects has not been determined.
Display the un-resolved dependent objects to assess the impact of these objects on the library objects.

**To show or hide un-resolved dependencies**

1. In the **Library Objects** pane, right-click the library object, and then click **Show un-resolved dependencies**. The tree updates to show the child objects.

2. Right-click the library object again, and then click **Hide un-resolved dependencies**. The tree updates to exclude the child objects.

**See also**

- Include dependent objects on page 32
- Library objects on page 23

---

### 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:** Shift+Click to select multiple sequential library objects to include.
   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 has been included.

**See also**

- Load library object dependencies on page 31
- Library objects on page 23
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. 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.

See also

Export Members on page 48
Export library object parameters on page 33
Import library object members on page 34

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


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**.

**See also**

- Import library object parameters on page 34
- Create an empty library object parameters file on page 35
- Library Object parameters on page 37

**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** 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**.

**See also**

- Import Members on page 49
- Import library object parameters on page 34
- Export library object members on page 33

**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. 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.

**See also**

- Export library object parameters on page 33
- Create an empty library object parameters file on page 35
- Library Object parameters on page 37

**Create an empty library object parameters file**

Use an empty library objects parameters file to manually enter data for library object parameters. 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**.
See also

Library Object parameters on page 37
Import library object parameters on page 34
Export library object parameters on page 33

Publish a library

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

To publish a library

1. In the Library Objects pane, either:
   a. Select the Libraries item to publish all libraries.
   b. Press Shift and click adjacent library objects.
   c. Press Ctrl and click multiple library objects.
   d. Click a library object.

2. Right-click the highlighted selection and then click Publish Library

   The Library Destination window opens.

3. Select the destination of the published library, either:
   a. LOM Repository

      Select from the list of repositories configured in Library Object Manager.
   b. Folder

      Type the path to a folder to publish the library or click the ellipsis (…) to browse for a folder.
   c. Current ACM Database

      Automatically connects to the last ACM database instance used.
   d. Another ACM Database

      Click the ellipsis (…) button to open the Connection Properties window and connect to a different ACM database.
4. The **Library Import Configuration** window opens.

- Enter a **Revision Description** for each library object listed.
- For new library objects, also add a **Description** and a **Solution**.
- Once all of the required parameters are entered, click **Apply**.

  **Tip:** If the **Apply** 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.

See also

- [Library Object parameters](#) on page 37
- [Validate Libraries](#) on page 38
- [Connection Properties settings](#) on page 43

**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:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library: Library Name</td>
<td>The name of the library object the parameter belongs to. Library Name must match an existing library object.</td>
</tr>
<tr>
<td>Param Name</td>
<td>The name of the parameter. This is a required field. It must be unique within the library scope.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the parameter.</td>
</tr>
<tr>
<td></td>
<td>• Bool</td>
</tr>
<tr>
<td></td>
<td>• String</td>
</tr>
<tr>
<td></td>
<td>• Int</td>
</tr>
<tr>
<td></td>
<td>• Real</td>
</tr>
<tr>
<td></td>
<td>• AxisID</td>
</tr>
<tr>
<td>Def Value</td>
<td>The default value for the parameter. The value can be entered manually or generated by an expression.</td>
</tr>
<tr>
<td>Minimum</td>
<td>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.</td>
</tr>
<tr>
<td>Maximum</td>
<td>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.</td>
</tr>
<tr>
<td>ControlType</td>
<td>Determines the mode of user interaction with the parameter when the parameter is an immediate reference type.</td>
</tr>
<tr>
<td></td>
<td>• <strong>TextBox</strong>. The parameter value is directly typed into a text box.</td>
</tr>
<tr>
<td></td>
<td>• <strong>DropDownList</strong>. The parameter value is selected from a list of possible values.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ControlValues</td>
<td>Required when <code>DropDownList</code> is specified as the <code>ControlType</code>. Defines the items included in the drop down list from which the user can select. Enter options as a comma-separated string.</td>
</tr>
<tr>
<td>Group</td>
<td>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.</td>
</tr>
</tbody>
</table>
| 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. 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 which 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`. 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. |

### See also

- [Export library object parameters](#) on page 33
- [Import library object parameters](#) on page 34
- [Create an empty library object parameters file](#) on page 35

### Validate Libraries

Use **Validate Libraries** to perform a complete validation operation on all library objects in the Library Designer to obtain a list 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 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, if you need 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 top right corner of the Validation Errors/Warnings dialog box.
   • Errors must be fixed manually. Rerun the validation process after fixing the error conditions.

See also

Validation Error/Warning list on page 40
Decorator Panel - Substitution tab on page 85
Decorator Panel - External References tab on page 92
Decorator Panel - Interface tab on page 99
The validation list contains a number of columns which provide additional details and possible action selection options for each detected validation issue.

This table describes the information provided in the validation list:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severity</td>
<td>Identifies the type of issue, either Warning or Error.</td>
</tr>
<tr>
<td>Class</td>
<td>The type of validation issue encountered:</td>
</tr>
<tr>
<td></td>
<td>• ResolvableReference: A value contains an unresolved reference that cannot be resolved.</td>
</tr>
<tr>
<td></td>
<td>• ReusedName: The same name exists for the following possible objects: functions, parameters, sub objects, external references, interfaces and library links.</td>
</tr>
<tr>
<td></td>
<td>• TagNotFound: The referenced element cannot be found.</td>
</tr>
<tr>
<td></td>
<td>• UnbalancedBraces: Missing brackets in the expression.</td>
</tr>
<tr>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td></td>
<td>• InvalidCondition: The inclusion condition is invalid.</td>
</tr>
<tr>
<td></td>
<td>• Incompatible object class: Object class is incompatible with parent.</td>
</tr>
<tr>
<td>Library</td>
<td>Name of the library where the validation problem was detected.</td>
</tr>
<tr>
<td>Element</td>
<td>The specific element within the library where the problem was detected.</td>
</tr>
<tr>
<td>Item</td>
<td>The module within the validation engines that triggered the warning or error message.</td>
</tr>
<tr>
<td>Description</td>
<td>A description of the validation issue.</td>
</tr>
<tr>
<td>Action</td>
<td>Use this field to automatically resolve warnings. If the validation issue is resolvable then this field can be set to either Resolve or Ignore.</td>
</tr>
<tr>
<td></td>
<td>If the issue cannot be automatically resolved, the field will be disabled and will display N/A. Issues with the severity type Error cannot be automatically resolved.</td>
</tr>
<tr>
<td>Action Detail</td>
<td>A description of what action will be taken when the action is set to Resolve and the Apply Actions button is clicked.</td>
</tr>
</tbody>
</table>

See also

- Validate Libraries on page 38
- Use expressions and functions on page 54
- References window on page 62

Delete a library object

Delete 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, then either:
• Right-click the selected items and then click **Delete**.

• On the **Library Objects** toolbar, click **Delete**.

The **Warning** window displays.

2. Click **Yes** to delete the selected items.

**See also**

- **Add selected objects to library objects** on page 25
- **Add a module to a library object** on page 29
- **Create a new empty library object** on page 24

**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 new 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 new library object, all of the settings except for **Revision** become active.

This window opens when the Add selected object to a Library button

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

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Library</td>
<td>Pull-down</td>
<td>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 <strong>Add selected object to a Library</strong> on the <strong>Select Objects</strong> toolbar.</td>
</tr>
<tr>
<td>Create New</td>
<td>Check box</td>
<td>Select to create a new library object and assign the item to it. Visible when using items originating from an ACD file or after clicking the <strong>Add selected object to a Library</strong> button on the <strong>Select Objects</strong> toolbar.</td>
</tr>
</tbody>
</table>
### Table: Library Object Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog Number</td>
<td>Text entry</td>
<td>The library object name, which appears when the library object is registered in the Application Code Manager application.</td>
</tr>
<tr>
<td>Revision</td>
<td>Auto</td>
<td>Dimmed field, displays system-generated version of the library object.</td>
</tr>
<tr>
<td>Description</td>
<td>Text entry</td>
<td>The description of the library object.</td>
</tr>
<tr>
<td>Library Type</td>
<td>Pull-down</td>
<td>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.</td>
</tr>
<tr>
<td>Content Type</td>
<td>Pull-down</td>
<td>Selects the object type, <strong>Task</strong>, <strong>Program</strong>, or <strong>Routine</strong>.</td>
</tr>
<tr>
<td>Base Library</td>
<td>Pull-down</td>
<td>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 <strong>Expression Builder</strong> accesses all custom properties of the base library when used in the current library object. When <strong>None</strong> is selected, the current library object does not inherit custom properties.</td>
</tr>
<tr>
<td>Solution</td>
<td>Pull-down</td>
<td>This list is empty if no solution has been set in another library in the same ACD or if the <strong>Settings</strong> file is also empty.</td>
</tr>
<tr>
<td>Family</td>
<td>Pull-down</td>
<td>Selects a family from the list.</td>
</tr>
<tr>
<td>Category</td>
<td>Pull-down</td>
<td>Selects a category from the list.</td>
</tr>
<tr>
<td>Instantiation: Name</td>
<td>Text entry</td>
<td>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 <strong>Expression Builder</strong>.</td>
</tr>
<tr>
<td>Instantiation: Description</td>
<td>Text entry</td>
<td>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 <strong>Expression Builder</strong>.</td>
</tr>
<tr>
<td>Substitutions: Name</td>
<td>Text entry</td>
<td>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 <strong>ObjectName</strong> as the substitution for the entered string.</td>
</tr>
<tr>
<td>Substitutions: Description</td>
<td>Text entry</td>
<td>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 <strong>ObjectDescription</strong> as the substitution for the entered string.</td>
</tr>
</tbody>
</table>

### See also

- **Add a module to a library object** on page 29
- **Create a new empty library object** on page 24
- **Add selected objects to library objects** on page 25

### Module Wizard overview

**How do I open the Module Wizard?**

Add the module object to a library object. The **Module Wizard** opens.
The **Module Wizard** generates default parameters and subobjects for the library object based on the controller code for the module. This decoration conforms to 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. 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.

**See also**

*Add a module to a library object* on page 29

**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.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data source:</td>
<td>Database type:</td>
</tr>
<tr>
<td></td>
<td>Always select Microsoft SQL Server (SqlClient)</td>
</tr>
<tr>
<td>Server name:</td>
<td>Selects a computer name and SQL Server instance from the list or type a</td>
</tr>
<tr>
<td></td>
<td>computer name and SQL Server instance in the following format:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;Computer Name&gt; \ &lt;SQL Server Instance&gt;</code></td>
</tr>
<tr>
<td>Log on to the server</td>
<td></td>
</tr>
<tr>
<td>Use Windows Authentication</td>
<td>Allows SQL Server log on using Windows authentication. When selected the logged on Windows user account credential will be sent to SQL Server to authenticate the session.</td>
</tr>
</tbody>
</table>
### Library objects

#### Setting Description

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use SQL Server Authentication</strong></td>
<td>Allows SQL Server logon using SQL Server authentication. When selected the user name and password must be provided for authentication by the SQL Server.</td>
</tr>
<tr>
<td><strong>User name</strong></td>
<td>The SQL Server user name, “sa” by default.</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>The SQL Server password associated with the user name specified, “ApplicationAdm1n” by default.</td>
</tr>
<tr>
<td><strong>Save my password</strong></td>
<td>When selected, saves the SQL Server password specified so that it can be used in the next session.</td>
</tr>
</tbody>
</table>

#### Connect to a database

<table>
<thead>
<tr>
<th>Select or enter a database name:</th>
<th>Select a database name from the list or enter a database name.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Attach a database file:</strong></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>- Type a database file name or use the <strong>Browse</strong> button to use the <strong>Open</strong> dialog to locate the database file by clicking through the file system.</td>
</tr>
<tr>
<td></td>
<td>- In <strong>Logical name</strong>, type the logical name of the database.</td>
</tr>
</tbody>
</table>

#### Advanced Properties settings

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.

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

<table>
<thead>
<tr>
<th>Advanced</th>
<th>Select to configure advanced database properties.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Test Connection</strong></td>
<td>Tests the connection to the database.</td>
</tr>
<tr>
<td></td>
<td>If a “Test connection succeeded.” message is not returned, check that the following settings are correct:</td>
</tr>
<tr>
<td></td>
<td>- Computer name</td>
</tr>
<tr>
<td></td>
<td>- SQL Server authentication</td>
</tr>
<tr>
<td></td>
<td>- Network access (remote SQL Server)</td>
</tr>
</tbody>
</table>

---

**See also**

- [Add a linked library](#) on page 95
- [Publish a library](#) on page 36
<table>
<thead>
<tr>
<th>Area</th>
<th>Setting</th>
<th>Possible Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced</td>
<td>MultipleActiveResultSets</td>
<td>True, False (default)</td>
<td>When True, multiple result sets can be returned and read from one connection.</td>
</tr>
<tr>
<td></td>
<td>Network Library</td>
<td>blank (required if local)</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Named Pipes (DBINMPNW)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared Memory (DBMSLPCN)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP/IP (DBMSSOCN) (recommended if networked)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>VIA (DBMSGNET)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Packet Size</td>
<td>8000 (recommended)</td>
<td>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 and 32767 bytes.</td>
</tr>
<tr>
<td></td>
<td>Transaction Binding</td>
<td>Implicit Unbind (default), Explicit Unbind</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>Type System Version</td>
<td>Latest (default), SQL Server 2012, SQL Server 2008, SQL Server 2005</td>
<td>Indicates which server type system the provider will expose through the DataReader.</td>
</tr>
<tr>
<td>Connection Resiliency</td>
<td>ConnectRetryCount</td>
<td>2 (recommended)</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>ConnectRetryInterval</td>
<td>5 (recommended)</td>
<td>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.</td>
</tr>
<tr>
<td>Context</td>
<td>Application Name</td>
<td>.Net SqlClient Data Provider</td>
<td>The name of the application.</td>
</tr>
<tr>
<td></td>
<td>Workstation ID</td>
<td></td>
<td>The name of the workstation connecting to SQL Server.</td>
</tr>
<tr>
<td>Initialization</td>
<td>ApplicationIntent</td>
<td>ReadWrite (default), ReadOnly</td>
<td>Declares the application workload type when connecting to a server.</td>
</tr>
<tr>
<td></td>
<td>Connect Timeout</td>
<td>30 (recommended)</td>
<td>The length of time in seconds to wait for a connection to the server before terminating 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.</td>
</tr>
<tr>
<td></td>
<td>Current Language</td>
<td></td>
<td>The SQL Server Language record name.</td>
</tr>
</tbody>
</table>
### Library objects

<table>
<thead>
<tr>
<th>Area</th>
<th>Setting</th>
<th>Possible Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pooling</td>
<td>Enlist</td>
<td>True (default)</td>
<td>When True sessions in a Component Services environment should automatically be enlisted in a global transaction where required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>False</td>
<td></td>
</tr>
<tr>
<td>Load Balance Timeout</td>
<td>0 (default)</td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>Max Pool Size</td>
<td>1000 (recommended)</td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>Min Pool Size</td>
<td>0 (default)</td>
<td></td>
<td>The minimum number of connections allowed in the pool. Valid values are greater than or equal to 0. Zero (0) in this field means no minimum connections are initially opened. Values that are greater than Max Pool Size generate an error.</td>
</tr>
<tr>
<td>PoolBlockingPeriod</td>
<td>Auto</td>
<td></td>
<td>Defines the blocking period behavior for a connection pool. When connection pooling is enabled and a timeout error or other login error occurs, an exception will be thrown and subsequent connection attempts will fail for the next five seconds, the &quot;blocking period&quot;. 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.</td>
</tr>
<tr>
<td></td>
<td>AlwaysBlock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>NeverBlock</td>
<td>(recommended)</td>
<td></td>
</tr>
<tr>
<td>Pooling</td>
<td>True (recommended)</td>
<td>False</td>
<td>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.</td>
</tr>
<tr>
<td>Replication</td>
<td>Replication</td>
<td>False (default)</td>
<td>Used by SQL Server in replication. Set to True if replication is supported using the connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>True</td>
<td></td>
</tr>
<tr>
<td>Security</td>
<td>Authentication</td>
<td>NotSpecified (default)</td>
<td>Specifies the method of authenticating with SQL Server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SqlPassword</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ActiveDirectoryPassword</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ActiveDirectoryIntegrated</td>
<td></td>
</tr>
<tr>
<td>Column Encryption Setting</td>
<td>Enabled</td>
<td></td>
<td>Default column encryption setting for all the commands on the connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disabled (default)</td>
<td></td>
</tr>
<tr>
<td>Encrypt</td>
<td>True</td>
<td></td>
<td>When True, SQL Server uses SSL encryption for all data sent between the client and server if the server has a certificate installed.</td>
</tr>
<tr>
<td></td>
<td>False (default)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integrated Security</td>
<td>True</td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>False (default)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Password</td>
<td>******</td>
<td></td>
<td>Indicates the password to be used when connecting to the data source.</td>
</tr>
</tbody>
</table>
### Area Setting Possible Values Description

<table>
<thead>
<tr>
<th>Area</th>
<th>Setting</th>
<th>Possible Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Persist Security Info</td>
<td>True, False (default)</td>
<td>When <strong>False</strong>, 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.</td>
</tr>
<tr>
<td></td>
<td>TrustServerCertificate</td>
<td>True (recommended), False</td>
<td>When <strong>True</strong> (and <strong>Encrypt</strong> is set to <strong>True</strong>), SQL Server uses SSL encryption for all data sent between the client and server without validating the server certificate. If <strong>TrustServerCertificate</strong> is set to <strong>True</strong> and <strong>Encrypt</strong> is set to <strong>False</strong>, the channel is not encrypted.</td>
</tr>
<tr>
<td></td>
<td>User ID</td>
<td>sa</td>
<td>Indicates the user ID to be used when connecting to the data source.</td>
</tr>
<tr>
<td>Source</td>
<td>AttachDbFilename</td>
<td></td>
<td>The name of the primary file, including the full path name, of an attachable database.</td>
</tr>
<tr>
<td></td>
<td>Context Connection</td>
<td>True, False (default)</td>
<td>When <strong>True</strong>, indicates the connection should be from the SQL Server context. Available only when running in the SQL Server process.</td>
</tr>
<tr>
<td></td>
<td>Data Source</td>
<td>localhost\SQLACM (default)</td>
<td>Indicates the name of the data source to connect to.</td>
</tr>
<tr>
<td></td>
<td>Failover Partner</td>
<td></td>
<td>The name or network address of the instance of SQL Server that acts as a failover partner.</td>
</tr>
<tr>
<td></td>
<td>Initial Catalog</td>
<td>Initial Database Name</td>
<td>The name of the initial catalog or database in the data source.</td>
</tr>
<tr>
<td></td>
<td>MultiSubnetFailover</td>
<td>True, False (default)</td>
<td>If your application is connecting to a high-availability, disaster recovery (AlwaysOn) availability group (AG) on different subnets, setting this value to <strong>True</strong> configures SqlConnection to provide faster detection of and connection to the (currently) active server.</td>
</tr>
</tbody>
</table>
|                  | 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 others 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 re-directed to connect to an instance of SQL Server running under the user’s account. |

**See also**

- [Connection Properties settings](#) on page 43
- [Publish a library](#) on page 36
- [Add a linked library](#) on page 95
Library Import Configuration window

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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01 Library Status</td>
<td>Pull-down menu</td>
<td>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.</td>
</tr>
<tr>
<td>CatalogNumber</td>
<td>Text entry</td>
<td>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. If you are copying into the ACM Database, the setting value is generated by the database and cannot be edited.</td>
</tr>
<tr>
<td>Description</td>
<td>Text entry</td>
<td>The description of the library object. This is a required setting. If you are copying into the ACM Database, the setting value is generated by the database and cannot be edited.</td>
</tr>
<tr>
<td>Family</td>
<td>Pull-down menu</td>
<td>The Family of the library object. This is a required setting. If you are copying into the ACM Database, the setting value is generated by the database and cannot be edited.</td>
</tr>
<tr>
<td>Solution</td>
<td>Pull-down menu</td>
<td>The Solution for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Library Type</td>
<td>Pull-down menu</td>
<td>The Library Type for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Category</td>
<td>Pull-down menu</td>
<td>The Category for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Content Type</td>
<td>Text</td>
<td>Displays the type of the object, Task, Program, or Routine.</td>
</tr>
<tr>
<td>Owner</td>
<td>Text entry</td>
<td>The user or entity that originally published the library object. This is a required setting. If you are copying into the ACM Database, the setting value is generated by the database and cannot be edited.</td>
</tr>
<tr>
<td>Major Revision</td>
<td>Text entry (integer)</td>
<td>The major revision number for the library object. For a new library object, this defaults to 1. If you are copying into the ACM Database, the database generates a default entry based on existing library objects with the same CatalogNumber.</td>
</tr>
<tr>
<td>Minor Revision</td>
<td>Text entry (integer)</td>
<td>The minor revision number for the library object. For a new library object, this defaults to 0. If you are copying into the ACM Database, the database generates a default entry based on existing library objects with the same CatalogNumber.</td>
</tr>
</tbody>
</table>

04 Library Usage Rules

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upstream Keys</td>
<td>Text entry</td>
<td>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. Rule is entered manually as a logical expression.</td>
</tr>
<tr>
<td>Downstream Keys</td>
<td>Text entry</td>
<td>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. Rule is entered manually as a logical expression.</td>
</tr>
</tbody>
</table>

See also

Publish a library on page 36

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 backup or transfer to another system or library.

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Displays the name of the item, grouped into Functions, External References, Interfaces, or Linked Libraries.</td>
</tr>
<tr>
<td>Export</td>
<td>Selects a member and its dependencies for export.</td>
</tr>
<tr>
<td>Type</td>
<td>Displays the type of the item.</td>
</tr>
<tr>
<td>Source Library</td>
<td>Displays the name of the source library or library scope.</td>
</tr>
<tr>
<td>Details</td>
<td>Opens a read-only decorator panel to display specific associations for the item.</td>
</tr>
<tr>
<td>Depends On</td>
<td>Lists associated dependencies of the item.</td>
</tr>
<tr>
<td>Used By</td>
<td>Lists associated members of the item.</td>
</tr>
<tr>
<td>Description</td>
<td>Displays the description of the item.</td>
</tr>
</tbody>
</table>

**See also**

- Export library object members on page 33
- Import library object members on page 34

**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.

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Displays the name of the item, grouped into Functions, External References, Interfaces, or Linked Libraries.</td>
</tr>
</tbody>
</table>
| 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. |
See also

Import library object members on page 34

Export library object members on page 33
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 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.

See also

Decorator Panel settings on page 51

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

Decorator Panel settings

Decorator panel
### Decorator panel

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog Number</td>
<td>Text entry</td>
<td>The library object name, which appears when the library object is registered in the Application Code Manager application.</td>
</tr>
<tr>
<td>Revision</td>
<td>Auto</td>
<td>Dimmed field, displays system-generated version of the library object.</td>
</tr>
<tr>
<td>Description</td>
<td>Text entry</td>
<td>The description of the library object.</td>
</tr>
<tr>
<td>Library Type</td>
<td>Pull-down</td>
<td>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.</td>
</tr>
<tr>
<td>Content Type</td>
<td>Pull-down</td>
<td>Selects the object type, <strong>Task</strong>, <strong>Program</strong>, or <strong>Routine</strong>.</td>
</tr>
<tr>
<td>Base Library</td>
<td>Pull-down</td>
<td>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 <strong>Expression Builder</strong> 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.</td>
</tr>
<tr>
<td>Solution</td>
<td>Pull-down</td>
<td>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.</td>
</tr>
<tr>
<td>Family</td>
<td>Pull-down</td>
<td>Selects a family from the list.</td>
</tr>
<tr>
<td>Category</td>
<td>Pull-down</td>
<td>Selects a category from the list.</td>
</tr>
<tr>
<td>Instantiation: Name</td>
<td>Text entry</td>
<td>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 <strong>Expression Builder</strong>.</td>
</tr>
<tr>
<td>Instantiation: Description</td>
<td>Text entry</td>
<td>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 <strong>Expression Builder</strong>.</td>
</tr>
<tr>
<td>Substitutions: Name</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Substitutions: Description</td>
<td>Text entry</td>
<td>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.</td>
</tr>
</tbody>
</table>

The values for these settings were entered when the library object was created and are displayed by default. They are all available for editing.

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. They cannot be created, modified, or deleted by the user.

See also

Decorator Panel - Parameters tab on page 65
Decorator Panel - Functions tab on page 77
Decorator Panel - Substitution tab on page 85
Decorator Panel - Predefined tab on page 90
Decorator Panel - External References tab on page 92

Change the columns displayed 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 on the column edge and dragging it to the desired size.

See also

Use find to limit the display based on a text string on page 53

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 drop-down 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.

**See also**

*Change the columns displayed in a tab* on page 53

**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**.

**Important:** When an expression is entered manually, 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 which return a string value must be enclosed in single quotes (apostrophes).

Text entered into an expression which 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** 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.

**See also**

*Expression Builder window* on page 56
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 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 manually edit an expression, click within the Expression 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.
Chapter 4  Decorator panel

• 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 Save Expression 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.

See also

Expression Builder window on page 56

Decoration types on page 12

Decorative elements tabs on page 58

Predefined functions and operators on page 59

Expression box on page 60

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.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Add New" /></td>
<td>The Add New button is only active on the Parameter tab and opens the Add New Parameter window.</td>
</tr>
<tr>
<td><img src="image" alt="Move Up/Move Down" /></td>
<td>The Move Up/Move Down buttons are not active in the Expression Builder window.</td>
</tr>
</tbody>
</table>
| ![FindPartial](image) | The find drop-down 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 drop down 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

The decorative elements available to the current expression. These include all elements added to the current library object and all elements added to 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

A collection of logical and mathematical operators that can be used to manipulate the values generated by the decorative elements.
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression Box and Result Type</td>
<td>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 <strong>Result Type</strong> displays a color-coded response when the expression is validated and tested.</td>
</tr>
</tbody>
</table>

**See also**

- Decorative elements tabs on page 58
- Predefined functions and operators on page 59
- Expression box on page 60

**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 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.

See also

Decoration types on page 12
Predefined functions and operators on page 59
Expression box on page 60

Predefined functions and operators

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

Tip: Mouse over a predefined function or operator to display a tooltip describing the input options for that function or operator.

The Expression 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 tab or the Functions and Operators area. This table describes the functions and operators in the Expression Builder.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Functions</strong></td>
<td>Elements are inserted into these functions to replace the &lt;value&gt; token.</td>
</tr>
<tr>
<td>Format</td>
<td>Inserts: (Format(&lt;value&gt;,&lt;style&gt;,&lt;valueDataType&gt;))</td>
</tr>
<tr>
<td></td>
<td>Returns the value entered in &lt;value&gt; using the numeric format entered in &lt;style&gt;. Style options are:</td>
</tr>
<tr>
<td></td>
<td>• (D) Decimal</td>
</tr>
<tr>
<td></td>
<td>• (X) Hexadecimal</td>
</tr>
<tr>
<td></td>
<td>• (C) Currency</td>
</tr>
<tr>
<td></td>
<td>• (E) Scientific</td>
</tr>
<tr>
<td></td>
<td>• (F) Fixed Point</td>
</tr>
<tr>
<td></td>
<td>• (G) General (default)</td>
</tr>
<tr>
<td></td>
<td>• (N) Number</td>
</tr>
<tr>
<td></td>
<td>• (P) Percent</td>
</tr>
<tr>
<td></td>
<td>Data type options are:</td>
</tr>
<tr>
<td></td>
<td>• Int</td>
</tr>
<tr>
<td></td>
<td>• Real</td>
</tr>
<tr>
<td></td>
<td>• DateTime</td>
</tr>
<tr>
<td>Modulus</td>
<td>Inserts: (Modulus(&lt;value1&gt;,&lt;value2&gt;))</td>
</tr>
<tr>
<td></td>
<td>Returns the remainder after the value of the decorative element inserted as &lt;value1&gt; is divided by the value of the decorative element inserted as &lt;value2&gt;.</td>
</tr>
</tbody>
</table>
### Name | Description
--- | ---
**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>`.

**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.

**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)}`
Returns the parameter value of the specific object.

**GetSubParameter** | Inserts: `{GetSubParameter(<objectInstanceName>,<subobjectType>,<subobjectIndexName>,<subobjectIndexValue>,parameterName)}`
Returns the parameter value of the specified object.

### Logical and Mathematical Operators

**True** | Logical TRUE: used to test value returned by a decorative element.

**False** | Logical FALSE: used to test value returned by a decorative element.

**Or** | Logical OR

**And** | Logical AND

**Not** | Logical NOT

**= <>** | Equal to, not equal to

**( )** | Parentheses, used to set order of operation for complex expressions

**&** | Mathematical AND

**> >= <= <** | Greater than, greater than or equal to, less than or equal to, less than

**+ - * /** | Plus, minus, multiplied by, divided by

### See also

- **Expression Builder window** on page 56
- **Expression box** on page 60

### 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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression Window</td>
<td>A blank space in which the expression appears as elements are added to it.</td>
</tr>
<tr>
<td>Validate</td>
<td>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.</td>
</tr>
<tr>
<td>Test</td>
<td>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.</td>
</tr>
<tr>
<td>Save</td>
<td>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.</td>
</tr>
<tr>
<td>Result Type</td>
<td>Displays the data type of the expression.</td>
</tr>
<tr>
<td></td>
<td>• String</td>
</tr>
<tr>
<td></td>
<td>• Boolean</td>
</tr>
<tr>
<td></td>
<td>• Numeric</td>
</tr>
</tbody>
</table>

See also

Use expressions and functions on page 54

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 attribute for which to substitute a different value (such as Name, Description, or ParentModule) click the ellipsis (...) button to open the Substitution Builder window.
4. Configure the substitution as needed.
5. Click OK to save the substitution.

See also

Add a substitution on page 86
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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library</td>
<td>The library object for the element that references the item.</td>
</tr>
<tr>
<td>Used in</td>
<td>The element that references the item. The value is displayed in this format: Element Type:SubObject Name</td>
</tr>
<tr>
<td>Details</td>
<td>The specific reference point within the element. Examples: a field in the Edit Parameter window, a conditional inclusion, or the Value Expression for a controller or local tag.</td>
</tr>
<tr>
<td>Navigate</td>
<td>Click to open the decorator panel for the element where the reference is located.</td>
</tr>
</tbody>
</table>

See also

- Delete a parameter on page 72
- View a parameter reference on page 72

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 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 new substitution.

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected Element</td>
<td>Read only</td>
<td>The original text string entered in the Search For field.</td>
</tr>
<tr>
<td>Result</td>
<td>Read only</td>
<td>The result generated when the Apply button is clicked to test the Substitution.</td>
</tr>
<tr>
<td>Scope</td>
<td>Pull-down menu</td>
<td>The scope for the new substitution. There are three options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• [Current Field]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• library object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• base library object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If library object or base library object are selected, the new substitution will replace previous substitutions added to that library object for the same text string.</td>
</tr>
<tr>
<td>Search For</td>
<td>Text entry</td>
<td>The text string to replace.</td>
</tr>
<tr>
<td>Replace With</td>
<td>Text entry</td>
<td>The replacement text. This can be entered manually or generated by an expression.</td>
</tr>
<tr>
<td>Operand replacement/ Text replacement</td>
<td>Radio buttons</td>
<td>Sets the condition under which the current tag is instantiated. Default is Always. A condition can be entered manually or generated by an expression.</td>
</tr>
<tr>
<td>Revert</td>
<td>Button</td>
<td>Reverts to the current substitution for the field.</td>
</tr>
<tr>
<td>Delete</td>
<td>Button</td>
<td>Deletes the currently selected line in the Element Substitutions window.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Important:</strong> Deleting a Substitution removes it from the library object.</td>
</tr>
<tr>
<td>Apply</td>
<td>Button</td>
<td>Tests the current substitution and adds it to the Element Substitution section.</td>
</tr>
<tr>
<td>Unlock/Lock</td>
<td>Button</td>
<td>Unlocks an existing substitution, allowing it to be edited, or locks a substitution, preventing it from being edited.</td>
</tr>
</tbody>
</table>
**Tip:** To use a calculated value for this field, click the ellipsis (...) button next to the field to open the Expression Builder.

**See also**

- Configure a module element substitution on page 61
- Decorator Panel - Substitution tab on page 85
- Expression Builder window on page 56

**Decorator Panel - 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 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.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Add New" /></td>
<td>The Add New button opens the Add New Parameter window.</td>
</tr>
<tr>
<td><img src="image" alt="Move Up" /> <img src="image" alt="Move Down" /></td>
<td>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 up or down within their group but cannot be moved between groups using these buttons.</td>
</tr>
</tbody>
</table>
| ![Find Partial](image) | The find drop-down 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 drop down 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. |

**See also**

- Add a new parameter on page 66
- Change the columns displayed in a tab on page 53
Add a new 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 new 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 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.

See also

- Parameter Filter Builder window on page 76
- Add New / Edit Parameter window on page 73

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, then 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.

See also

- Add a new parameter on page 66
- Parameter Filter Builder window on page 76
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 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 the Library Object Manager application.

5. Click **OK** to add the external reference.

See also

- Reference Builder window on page 93
- Edit an external reference on page 93
- Delete an external reference on page 93

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 keys 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 library object.

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

**See also**

- Add a new parameter on page 66
- Add New / Edit Parameter window on page 73
- Parameter References window on page 62
- Parameter Filter Builder window on page 76

**Add a new 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 reestablished.

**To add a new 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.

**See also**

- Add a new parameter on page 66
- Edit a parameter on page 66
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 new group if the group specified does not exist. If every parameter in a group needs to be reassigned, each parameter must be edited 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.

See also

- [Delete a group](#) on page 70
- [Add a subobject](#) on page 70

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.

See also

Copy a parameter on page 67
Reassign a parameter to a group on page 69

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, 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, 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.

See also

Add a new group on page 68
Edit a parameter on page 66

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 / Edit SubObject** window opens.

2. Enter the values for the new subobject.

3. Click **OK** to add the new subobject.

See also

**New / Edit SubObject window** on page 75

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 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** 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 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 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.

See also

Parameter References window on page 62

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 window opens.

2. To finish the deletion, click Yes.
See also

Add a new parameter on page 66

View a parameter reference on page 72

Delete a substitution on page 89

Delete an external reference on page 93

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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>01 General</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Text entry</td>
<td>The name of the parameter. This is a required field.</td>
</tr>
<tr>
<td>Parameter Help</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Scope</td>
<td>Pull-down menu</td>
<td>The library object that contains the parameter. The list includes all library objects in the library object tree. Default value is the current library object. Parameters can be moved to a different library object by changing the selection here.</td>
</tr>
<tr>
<td>Data Type</td>
<td>Pull-down menu</td>
<td>The data type of the parameter. Options are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Bool (Boolean)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Int (Integer)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Real</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AxisID</td>
</tr>
</tbody>
</table>

02 Reference
### Name Type Description

**Reference Type** Pull-down 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**: Parameter is accessible to user input as an entry field.
- **Calculated**: Parameter is not accessible to user input. 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.

*These fields appear if **Calculated** is selected in **Reference Type**.*

**Reference Linked Parameter** Pull-down 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** Pull-down 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**.

**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**.

**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**. 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 pull-down 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 pull-down 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. Example: multiple channel subobjects will be numbered sequentially as they are added.
### Name | Type | Description
--- | --- | ---
04 UserInterface  |  |  

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
<td>Text entry</td>
<td>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.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter</td>
<td>Text entry</td>
<td>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 references. 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. Click the ellipsis (...) button next to the field to open the Parameter Filter Builder.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visible</td>
<td>Text entry</td>
<td>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.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disabled</td>
<td>Text entry</td>
<td>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 which are included in a subobject.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Read Only | TRUE-FALSE pull-down 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. |

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Control Type | Pull-down 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. |

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Values</td>
<td>Text entry</td>
<td>Defines the items included in the drop down list from which the user can select. Enter options as a comma-separated string. This is a required field when it appears.</td>
</tr>
</tbody>
</table>

See also

Add a new parameter on page 66

New / Edit SubObject window on page 75

Parameter References window on page 62

Expression Builder window on page 56

Parameter Filter Builder window on page 76

### 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Type</td>
<td>Text entry</td>
<td>The identity for the SubObject group in the Parameters Tab. This is a required field. The name must be unique for the library object.</td>
</tr>
<tr>
<td>Description</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Auto Populate</td>
<td>Checkbox</td>
<td>When selected, automatically creates multiple instances of the subobject during instantiation. For auto population to occur:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A value must be entered in the Min and Max fields for the parameter. The AutoIncrement field must be set to TRUE.</td>
</tr>
<tr>
<td>Pre-configured Instances</td>
<td>Checkbox</td>
<td>Selected when the number of subobjects are needed. Allows multiple subobjects to be created using a pre-configured option.</td>
</tr>
<tr>
<td>Auto Increment Parameter</td>
<td>Pull-down</td>
<td>Only available when Auto Populate is selected. Lists all integer subobject parameters. If no parameters are configured, list is empty.</td>
</tr>
<tr>
<td>Dimension</td>
<td>Text entry</td>
<td>Determines how many instances of the subobject are automatically created on instantiation into the Application Code Manager. Maximum number should not be greater than Auto Increment Parameter.</td>
</tr>
<tr>
<td>Pre-Configured Instances</td>
<td>Checkbox</td>
<td>Select to enable Add Instance.</td>
</tr>
<tr>
<td>Add Instance</td>
<td>Button</td>
<td>Select to add a pre-configured instance. Dimmed when Pre-configured Instance is not selected.</td>
</tr>
<tr>
<td>Add Parameter</td>
<td>Button</td>
<td>Select to display the Add New Parameter dialog.</td>
</tr>
</tbody>
</table>

**See also**

Add a subobject on page 70

Add a new parameter on page 66

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 on 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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  
  - 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. |

**See also**

- [Edit a parameter on page 66](#)
- [Limit the external references available to a parameter on page 71](#)

**Decorator Panel - 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.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="add_new.png" alt="Add New" /></td>
<td>The <strong>Add New</strong> button opens the <strong>Function Builder</strong> window.</td>
</tr>
<tr>
<td><img src="move_up_down.png" alt="Move Up/Move Down" /></td>
<td>The <strong>Move Up/Move Down</strong> buttons are deactivated for this tab</td>
</tr>
</tbody>
</table>
| ![Find Partial](find_partial.png) | The find drop-down 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 drop down 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. |

**See also**

- [Change the columns displayed in a tab](#) on page 53
- [Use find to limit the display based on a text string](#) on page 53

**Add a new 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 new function**

1. In the decorator panel, either:
   
   - Click ![Function](function.png) on the **Function** tab toolbar.
   
   - In the **Function** tab, right-click a group name and then click **Add new Function**.  

   The **Function Builder** 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
  - or -
  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 to create the expression.

3. Click OK to add the new function.

**See also**

Add branches to a conditional function statement on page 79

Parameter Filter Builder window on page 76

Function Builder window on page 83

Expression Builder window on page 56

Copy a function on page 80

---

Add branches to a conditional statement

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.

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.

See also

Add a new function on page 78
Edit a function on page 81
Delete branches from a conditional statement on page 82
Function Builder window on page 83

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 new function.

To use a previously created function

1. In the Function Builder 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.

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 new 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.

See also

Add branches to a conditional statement on page 79
Function Builder window on page 83
Add a new function on page 78
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 Function(s)**.

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** 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.

**See also**

- [Add a new function](#) on page 78
- [Edit a function](#) on page 81
- [Function Builder window](#) on page 83

**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.
See also

Add branches to a conditional statement on page 79

View a function reference on page 82

Add a new function on page 78

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 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.

See also

References window on page 62

Delete a function on page 83

Delete branches from a conditional statement

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.

**See also**

- [Edit a function](#) on page 81
- [Function Builder window](#) on page 83

**Function Builder window**

How do I open the Function Builder window?

- In the 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 new functions or to modify an existing function after it is imported.
This table describes the settings in the **Function Builder** window.

<table>
<thead>
<tr>
<th>Name</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Text entry</td>
<td>The name of the function. This is a required field.</td>
</tr>
<tr>
<td>Type</td>
<td>Pull-down menu</td>
<td>Determines whether the function generates a single value or uses IF/ELSE logic to generate one of multiple possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Conditional</strong>: Function uses IF/ELSE logic to generate one of multiple possible values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Calculation</strong>: Function generates a single value based on a single statement.</td>
</tr>
<tr>
<td>Result Type</td>
<td>Pull-down menu</td>
<td>The data type of the function. Options are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Boolean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Numeric</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Integer</td>
</tr>
<tr>
<td>Function Scope</td>
<td>Pull-down menu</td>
<td>The library object that contains the function. Default value is the current library object.</td>
</tr>
<tr>
<td>Expression</td>
<td>Pull-down menu</td>
<td>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.</td>
</tr>
<tr>
<td>Statement Definition</td>
<td></td>
<td>Provides a visual representation of the conditional statement in menu form. The <strong>Function Builder</strong> 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.</td>
</tr>
<tr>
<td>Condition</td>
<td>Menu</td>
<td>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.</td>
</tr>
<tr>
<td>If this expression is true</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Return this value</td>
<td>Text entry</td>
<td>The value used if the condition returns TRUE. 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.</td>
</tr>
<tr>
<td>Else</td>
<td>Text entry</td>
<td>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.</td>
</tr>
</tbody>
</table>
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. 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. 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 which 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 Substitutions tab.
### Command Description

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Add New" /></td>
<td>The <strong>Add New</strong> button opens the <strong>Substitution Builder</strong> window.</td>
</tr>
<tr>
<td><img src="image" alt="Move Up/Move Down" /></td>
<td>The <strong>Move Up/Move Down</strong> buttons change the position of a selected Substitution item in the tab listing. Substitutions are listed in alphabetical order by default.</td>
</tr>
</tbody>
</table>
| ![Find Partial](image) | The Find drop-down 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 drop down 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. |

### See also

- Change the columns displayed in a tab on page 53
- Change the default location for a substitution on page 87
- Copy a substitution to a different library object on page 88
- Add a substitution on page 86
- Add decorative elements to a substitution on page 89

### 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 **Add New** on the **Substitutions** tab toolbar.
   - In the **Substitutions** tab, right-click a group name and then click **Add new Substitution**.

   The **Substitution Builder** 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** 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.

**See also**

- [Edit a substitution](#) on page 88
- [Delete a substitution](#) on page 89
- [Member Selector window](#) on page 89
- [Substitution Builder window](#) on page 62

### Change the order of execution for a substitution

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 and then 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.

**See also**

- [Add a substitution](#) on page 86
- [Use find to limit the display based on a text string](#) on page 53
**Copy a substitution to a different library object**

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 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.

**See also**

- **Add a substitution** on page 86
- **Change the order of execution for a substitution** on page 87

**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** tab, then 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.

**See also**

- **Add a substitution** on page 86
- **Delete a substitution** on page 89
- **Module Elements - Substitution Builder window** on page 62
- **Substitution tab - Substitution Builder window** on page 89
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.

See also

Decorator Panel - External References tab on page 92

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 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.

See also

Decorator Panel - Functions tab on page 77
Decorator Panel - Parameters tab on page 65
Decorator Panel - Predefined tab on page 90
Decorator Panel - Linked Libraries tab on page 94
Decorator Panel - Interfaces tab on page 99
When adding a new 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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>Pull-down menu</td>
<td>The scope for the substitution. Defaults to the current library object</td>
</tr>
<tr>
<td>Original</td>
<td>Text entry</td>
<td>The text string to replace.</td>
</tr>
<tr>
<td>Replacement</td>
<td>Text entry</td>
<td>The replacement text. This can be entered manually or generated by an expression. Click the ellipsis (...) to open the <strong>Expression Builder</strong>.</td>
</tr>
<tr>
<td>Search Mode</td>
<td>Pull-down menu</td>
<td>Defines the type of search conducted to perform the substitution.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Text</strong>. Searches text strings in the defined scope for the original text and makes the designated replacement.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Operand</strong>. Search tokens in the defined scope for the original text and makes the designated replacement.</td>
</tr>
</tbody>
</table>

See also

- [Add a substitution](#) on page 86
- [Module Elements - Substitution Builder window](#) on page 62
- [Expression Builder window](#) on page 56

**Decorator Panel - 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 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Global</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ControllerDescription</td>
<td>String</td>
<td>Description of the controller for the current project.</td>
</tr>
<tr>
<td>ControllerName</td>
<td>String</td>
<td>Name of the controller for the current project.</td>
</tr>
<tr>
<td>MotionGroupName</td>
<td>String</td>
<td>Name of the group.</td>
</tr>
<tr>
<td>ParentName</td>
<td>String</td>
<td>Name of the Object or element that contains the current element.</td>
</tr>
<tr>
<td>ProcessorType</td>
<td>String</td>
<td>Processor type of the controller for the current project.</td>
</tr>
<tr>
<td>ProjectDescription</td>
<td>String</td>
<td>Description of the current project.</td>
</tr>
<tr>
<td>ProjectName</td>
<td>String</td>
<td>Name of the current project.</td>
</tr>
<tr>
<td>SoftwareRevision</td>
<td>String</td>
<td>Software revision number of the controller for the current project.</td>
</tr>
<tr>
<td><strong>Local</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ObjectDescription</td>
<td>String</td>
<td>Description of the current library object.</td>
</tr>
<tr>
<td>ObjectName</td>
<td>String</td>
<td>Name of the current library object.</td>
</tr>
<tr>
<td>ProgramDescription</td>
<td>String</td>
<td>Description of the program that contains the current element.</td>
</tr>
<tr>
<td>ProgramName</td>
<td>String</td>
<td>Name of the program that contains the current element.</td>
</tr>
<tr>
<td>SubObjectDescription</td>
<td>String</td>
<td>Description of the subobject that contains the current element.</td>
</tr>
<tr>
<td>SubObjectName</td>
<td>String</td>
<td>Name of the subobject that contains the current element.</td>
</tr>
<tr>
<td>TaskDescription</td>
<td>String</td>
<td>Description of the task that contains the current element.</td>
</tr>
<tr>
<td>TaskName</td>
<td>String</td>
<td>Name of the task that contains the current element.</td>
</tr>
</tbody>
</table>
### Decorator Panel - 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 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.

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Add New" /></td>
<td>The <strong>Add New</strong> button opens the <strong>References Builder</strong>.</td>
</tr>
<tr>
<td><img src="image" alt="Move Up" /> <img src="image" alt="Move Down" /></td>
<td>The <strong>Move Up/Move Down</strong> buttons are deactivated for this tab.</td>
</tr>
<tr>
<td><img src="image" alt="Find Pull-Down" /></td>
<td>The find pull-down menu initiates a search for external references that match the text string entered in the adjacent field.</td>
</tr>
</tbody>
</table>

**See also**

[Change the columns displayed in a tab on page 53](#)

[Use find to limit the display based on a text string on page 53](#)
Use find to limit the display based on a text string on page 53

Decorator Panel - External References tab on page 92

Reference Builder window on page 93

**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 window opens.

2. Edit the values of the external reference.

3. Click **OK** to update the external reference.

See also

Add an external reference on page 67

Reference Builder window on page 93

**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**.

See also

Add an external reference on page 67

Edit an external reference on page 93

Reference Builder window on page 93

**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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Text entry</td>
<td>The name of the external reference. This is a required field.</td>
</tr>
</tbody>
</table>
### Name Field Type Description

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReferenceScope</td>
<td>Pull-down menu</td>
<td>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, <strong>Object</strong> will be the only option in the list.</td>
</tr>
<tr>
<td>Data Type</td>
<td>Pull-down menu</td>
<td>The data type of the external reference. Options are: <strong>String</strong>, <strong>Integer</strong>, <strong>Real</strong>, <strong>Boolean</strong></td>
</tr>
<tr>
<td>Value</td>
<td>Text entry</td>
<td>The specific location of the external reference once the library object has been instantiated. By default, 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 <strong>Expression Builder</strong>.</td>
</tr>
<tr>
<td>Description</td>
<td>Text entry</td>
<td>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 <strong>Expression Builder</strong>.</td>
</tr>
</tbody>
</table>

**See also**

- [Add an external reference](#) on page 67
- [Expression Builder window](#) on page 56

**Decorator Panel - 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:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add New</td>
<td>Displays the <strong>Add New Linked Library</strong> dialog.</td>
</tr>
<tr>
<td>Edit</td>
<td>Displays the <strong>Edit Linked Library</strong> dialog.</td>
</tr>
<tr>
<td>Delete</td>
<td>Deletes the selected object.</td>
</tr>
<tr>
<td>Show References</td>
<td>Displays the <strong>References</strong> dialog. Libraries linked to the selected object are listed.</td>
</tr>
</tbody>
</table>

- **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 parameter value between a library object and linked library object. Parameter links display the direction of the flow of information.

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

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add new parameter link</td>
<td>Displays the <strong>Parameter Browser</strong> dialog.</td>
</tr>
<tr>
<td>Delete selected</td>
<td>Deletes the selected parameter.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Control</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ellipsis (...) button</td>
<td>Displays the <strong>Parameter Browser</strong> dialog.</td>
</tr>
<tr>
<td>Data Flow arrow (-&gt;) button</td>
<td>Click to toggle the data flow direction between the linked parameters.</td>
</tr>
</tbody>
</table>

See also

- [Change the columns displayed in a tab](page 53)
- [Use find to limit the display based on a text string](page 53)
- [Add a linked library](page 95)
- [Add parameter links](page 96)

**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 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.

See also

Add New / Edit Linked Library window on page 97

Add parameter links on page 96

Add parameter links

Parameter links are used to read or write parameter value 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 on an empty row and click **Add**.

The **Parameters Browser** window opens.

4. Select a parameter and click **OK**.

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

**See also**

- [Add a linked library](#) on page 95
- [Parameters browser window](#) on page 98

**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 windows 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Select the library you want to link to:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Filter</td>
<td>Text Entry</td>
<td>Apply a filter to find the desired selected library object.</td>
</tr>
<tr>
<td>Connect to Database</td>
<td>Button</td>
<td>Connect to a different database to access additional libraries.</td>
</tr>
<tr>
<td>Libraries in this ACD file</td>
<td>List</td>
<td>Displays selected library objects in the same ACD as the chosen object. Click to select a library object.</td>
</tr>
<tr>
<td>Server *****</td>
<td>List</td>
<td>Displays selected library objects from other databases or locations. Click to select a library object.</td>
</tr>
</tbody>
</table>

**01 Link Details**

<table>
<thead>
<tr>
<th>Link Name</th>
<th>Text Entry</th>
<th>Mandatory field. Enter the name of the link to be displayed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mandatory</td>
<td>Pull-down</td>
<td>True/False option. Set this field to True if the linked library must be resolved before creating, updating, or importing an object.</td>
</tr>
<tr>
<td>Name</td>
<td>Field Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Include Condition</td>
<td>Expression</td>
<td>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 to write, validate, and test the expression. Only editable when Mandatory is set to False. Default value is Always.</td>
</tr>
</tbody>
</table>

### 02 Library Details

<table>
<thead>
<tr>
<th>Catalog Number</th>
<th>Text Entry</th>
<th>Editable field that displays the catalog number of the selected library object. Field auto-populates when a library object is selected. Use an underscore (_) as a wildcard to for a single character, or a percentage sign (%) as a wildcard for multiple characters.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Family</td>
<td>Text Entry</td>
<td>Editable field that displays the family of the selected library object. Field auto-populates when a value is assigned in the selected library object. Use an underscore (_) as a wildcard to for a single character, or a percentage sign (%) as a wildcard for multiple characters.</td>
</tr>
<tr>
<td>Solution</td>
<td>Text Entry</td>
<td>Editable field that displays the solution of the selected library. Field auto-populates when a value is assigned in the selected library object. Use an underscore (_) as a wildcard to for a single character, or a percentage sign (%) as a wildcard for multiple characters.</td>
</tr>
<tr>
<td>Library Type</td>
<td>Text Entry</td>
<td>Editable field that displays the library type of the selected library. Field auto-populates when a value is assigned in the selected library object. Use an underscore (_) as a wildcard to for a single character, or a percentage sign (%) as a wildcard for multiple characters.</td>
</tr>
<tr>
<td>Category</td>
<td>Text Entry</td>
<td>Editable field that displays the category of the selected library. Field auto-populates when a value is assigned in the selected library object. Use an underscore (_) as a wildcard to for a single character, or a percentage sign (%) as a wildcard for multiple characters.</td>
</tr>
<tr>
<td>Revision</td>
<td>Auto-populate</td>
<td>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. Example: ( \geq 2 ) requires that the linked library version be equal to 2 or greater.</td>
</tr>
</tbody>
</table>

### 03 Instance Details

<table>
<thead>
<tr>
<th>Instance Name</th>
<th>Text Entry</th>
<th>Enter an instance name for the linked library.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Description</td>
<td>Text Entry</td>
<td>Enter a description of the instance.</td>
</tr>
</tbody>
</table>

See also

- [Add a linked library](#) on page 95
- [Expression Builder](#) on page 56

**Parameters browser window**

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 on 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:

<table>
<thead>
<tr>
<th>Table Heading</th>
<th>Table Heading</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library Catalog Number</td>
<td>The library object name, which appears when the library object is registered in the Application Code Manager application.</td>
</tr>
<tr>
<td>Library Location</td>
<td>Where the library is located.</td>
</tr>
</tbody>
</table>
| Find                  | Searches the library for parameters that include the text string typed in the adjacent box. Two method of search are provided:  
                         | • FindPrefix. Searches the beginning of each field in the object for the string.  
                         | • FindPartial. Searches each field for any instances of the specified 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. |

See also

Use find to limit the display based on a text string on page 53

Decorator Panel - Linked Libraries tab on page 94

Decorator Panel - Interface 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 Interface 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 on the icons or right-clicking the object and selecting the command.
### Name | Action
--- | ---
Add New | Displays the Add New Interface dialog.
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.

See also

- [Use interfaces to standardize data structures](#) on page 100
- [Add an interface](#) on page 101
- [Delete or edit interfaces](#) on page 102
- [Add interface members](#) on page 103

#### Use interfaces to 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 diagram below illustrates this example.

See also

- [Decorator Panel - Interface tab](#) on page 99
Add an interface on page 101

Add interface members on page 103

Add an interface

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

To add a new interface

1. Select the Interfaces tab.

2. In the Interfaces tab, select Add New. The Add New Interface 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 an available interface members from the Interface Members grid in the bottom 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:
a. **Exact Match.** 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.

b. **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 datatype can be selected.

7. In **Key Id** enter an identifier which 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.

**See also**

- Use interfaces to standardize data structures on page 100
- Delete or edit interfaces on page 102
- Add interface members on page 103
- Add New Interface window on page 103
- Tags window on page 106

**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.

See also

Add interface members on page 103

Add an interface on page 101

Add interface members

Add interface members as necessary.

To add interface members

1. To add interface members, select an interface, then either:

   a. Click the Add New 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.

See also

Add an interface on page 101

Delete or edit interfaces on page 102

Add New/Edit Interface window

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

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 same 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.
### 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 windows 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.**

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Categorized</td>
<td>Button</td>
<td>Click to change the parameter display to categorize parameters by function.</td>
</tr>
<tr>
<td>Alphabetical</td>
<td>Button</td>
<td>Click to change the parameter display to parameters an alphabetized order.</td>
</tr>
<tr>
<td>Property Pages</td>
<td>Button</td>
<td>Click to display the property pages when they are hidden.</td>
</tr>
</tbody>
</table>

### 01 Interface Detail

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Text entry</td>
<td>The name of the interface. This field is required</td>
</tr>
<tr>
<td>Usage</td>
<td>Drop-down</td>
<td>Choose whether the interface in an input interface consumed by the library or output interface produced by the library.</td>
</tr>
<tr>
<td>Interface Description</td>
<td>Text entry</td>
<td>(optional) Describe the usage of the interface</td>
</tr>
</tbody>
</table>

### 02 Interface Connection

<table>
<thead>
<tr>
<th>Key Id</th>
<th>Text entry</th>
<th>The identifier for the key assigned to this interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revision</td>
<td>Text entry</td>
<td>The revision number of this interface.</td>
</tr>
</tbody>
</table>

### Interface Members

<table>
<thead>
<tr>
<th>Add Member</th>
<th>Button</th>
<th>Click to open the Tags window and choose an I/O Configuration tag to assign to this interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit</td>
<td>Button</td>
<td>When an interface member is selected, click to open the Interface Member window and modify its parameters.</td>
</tr>
<tr>
<td>Delete Member</td>
<td>Button</td>
<td>When an interface member is selected, click to delete the interface member from the list.</td>
</tr>
<tr>
<td>Import Members</td>
<td>Button</td>
<td>Not available.</td>
</tr>
<tr>
<td>List</td>
<td>Grid</td>
<td>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</td>
</tr>
</tbody>
</table>

See also

- [Add interface members](#) on page 103
- [Add an interface](#) on page 101
- [Delete or edit interfaces](#) on page 102
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 an alphabetized order.
Property Pages Button Click to display the property pages when they are hidden.

01 Interface Detail

<table>
<thead>
<tr>
<th>Name</th>
<th>Text entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Text entry</td>
<td>The name of the interface. This field is required</td>
</tr>
<tr>
<td>Usage</td>
<td>Drop-down</td>
<td>Choose whether the interface is an input interface consumed by the library or output interface produced by the library.</td>
</tr>
<tr>
<td>Interface Description</td>
<td>Text entry</td>
<td>(optional) Describe the usage of the interface</td>
</tr>
</tbody>
</table>

02 Interface Connection

<table>
<thead>
<tr>
<th>Key Id</th>
<th>Text entry</th>
<th>The identifier for the key assigned to this interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revision</td>
<td>Text entry</td>
<td>The revision number of this interface.</td>
</tr>
</tbody>
</table>

Interface Members

<table>
<thead>
<tr>
<th>Add Member</th>
<th>Button</th>
<th>Click to open the Tags window and choose an I/O Configuration tag to assign to this interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit</td>
<td>Button</td>
<td>When an interface member is selected, click to open the Interface Member window and modify its parameters</td>
</tr>
<tr>
<td>Delete Member</td>
<td>Button</td>
<td>When an interface member is selected, click to delete the interface member from the list.</td>
</tr>
<tr>
<td>Import Members</td>
<td>Button</td>
<td>Not available.</td>
</tr>
<tr>
<td>List</td>
<td>Grid</td>
<td>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</td>
</tr>
</tbody>
</table>

See also

Add interface members on page 103
Add an interface on page 101
Delete or edit interfaces on page 102

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:
   a. Click the interface member, then on the Interface Members toolbar click Edit.
   b. Right-click the interface member and then click Edit.
   c. 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.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the interface member.</td>
</tr>
<tr>
<td>Data Type</td>
<td>The data type of the interface member. This field is read only.</td>
</tr>
<tr>
<td>Value</td>
<td>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 <strong>Expression Builder</strong> window.</td>
</tr>
<tr>
<td>Description</td>
<td>A description of the interface member.</td>
</tr>
</tbody>
</table>

**Tags window**

How do I open the Tags window?

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

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

The **Tags** window includes these items:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag tree</td>
<td>Navigate the tree control to find the controller tag, task tag, or input/output tag to assign as an interface member.</td>
</tr>
<tr>
<td>Tag list</td>
<td>List displays to tags selected by name and data type. The listed tags are added to the interface member by clicking <strong>OK</strong>.</td>
</tr>
</tbody>
</table>

**See also**

- Decorator Panel - Interface tab on page 99
- Add interface members on page 103
- Expression Builder window on page 56

**Decorator Panel - 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


- 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:

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Supported item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Text entry</td>
<td>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, Module</td>
<td>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.</td>
</tr>
<tr>
<td>Description</td>
<td>Text entry</td>
<td>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</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Name Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Supported item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description Language</td>
<td>Pull-down menu</td>
<td>Tag, Task, Program, Routine, Function Block sheet, Sequential function chart, Structured text chart, Motion group, Add-On Instruction, Data type</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td></td>
<td>The language used for the description. The default is English. This field cannot be edited for data type items.</td>
</tr>
<tr>
<td>Library object</td>
<td>Read only</td>
<td>Tag, Task, Program, Routine, Function Block sheet, Sequential function chart, Structured text chart, Motion group, Add-On Instruction, Data type, Module</td>
</tr>
<tr>
<td><strong>Library object</strong></td>
<td></td>
<td>The library object that contains the item.</td>
</tr>
<tr>
<td>Logix Path</td>
<td>Read only link</td>
<td>Tag, Task, Program, Routine, Function Block sheet, Sequential function chart, Structured text chart, Motion group, Add-On Instruction, Data type</td>
</tr>
<tr>
<td><strong>Logix Path</strong></td>
<td></td>
<td>A link to the applicable item for the element in the Studio 5000 Logix Designer application. Click the link to open the screen. Close the Library Designer to access the Studio 5000 Logix Designer application.</td>
</tr>
<tr>
<td>Configure Instantiation Rules: Condition</td>
<td>Text entry</td>
<td>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</td>
</tr>
<tr>
<td><strong>Configure Instantiation Rules: Condition</strong></td>
<td></td>
<td>Sets the condition under which the current item is instantiated. Default is <strong>Always</strong>. A condition can be entered manually or generated by an expression.</td>
</tr>
<tr>
<td>Name</td>
<td>Field Type</td>
<td>Supported item</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Configure Instantiation Rules: Usage | Pull-down 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. Default value is "One per Object". |
| Exclude Base Library Substitutions | Check box           | 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 base library object for the current library object to be overridden.                                                                                      |
| Exclude Library Substitutions | Check box           | 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.                                                                                                                       |
### Decorator panel

<table>
<thead>
<tr>
<th>Name</th>
<th>Field Type</th>
<th>Supported item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Mode</td>
<td>Check box</td>
<td>Task</td>
<td>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.</td>
</tr>
<tr>
<td>Catalog No</td>
<td>Read only</td>
<td>Module</td>
<td>Catalog number of the item.</td>
</tr>
<tr>
<td>Major Version</td>
<td>Read only</td>
<td>Module</td>
<td>The major version for the item.</td>
</tr>
<tr>
<td>Minor Version</td>
<td>Read only</td>
<td>Module</td>
<td>The minor version for the item.</td>
</tr>
<tr>
<td>Parent Module</td>
<td>Text entry</td>
<td>Module</td>
<td>The name that is applied to the module when the library object is instantiated. A predefined parameter is assigned by default.</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Substitutions</td>
<td>Opens the Substitution Builder.</td>
</tr>
<tr>
<td>Select By &gt; Included in Library</td>
<td>Selects all elements that are included in a library.</td>
</tr>
<tr>
<td>Select By &gt; Excluded from Library</td>
<td>Selects all elements that are excluded from libraries.</td>
</tr>
<tr>
<td>Select By &gt; Usage &gt; One per Object</td>
<td>Selects all elements where the Usage field has been set to &quot;Include Once per Object&quot;.</td>
</tr>
<tr>
<td>Select By &gt; Usage &gt; One per SubObject</td>
<td>Selects all elements where the Usage field has been set to &quot;Include Once per SubObject&quot;.</td>
</tr>
<tr>
<td>Select By &gt; Usage &gt; Include Once</td>
<td>Selects all elements where the Usage field has been set to &quot;Include Once&quot;.</td>
</tr>
<tr>
<td>Select By &gt; Include Condition</td>
<td>Selects all elements where the Condition field matches the option selected in the submenu. Submenu displays all conditional inclusions rules for the current diagram.</td>
</tr>
<tr>
<td>Select Same &gt; Usage</td>
<td>Selects additional elements that match the Usage field for the currently selected element.</td>
</tr>
<tr>
<td>Select Same &gt; Include Condition</td>
<td>Selects additional elements that match the Condition field for the currently selected element.</td>
</tr>
<tr>
<td>Select Same &gt; Both of the Above</td>
<td>Selects additional elements that match both the Usage field and the Condition field for the currently selected element.</td>
</tr>
<tr>
<td>Select Same &gt; Tag</td>
<td>Selects additional elements that reference the same Tag as the currently selected element.</td>
</tr>
<tr>
<td>Select All</td>
<td>Selects all elements.</td>
</tr>
<tr>
<td>Remove</td>
<td>Deactivates the selected elements.</td>
</tr>
<tr>
<td>Add</td>
<td>Not Available</td>
</tr>
<tr>
<td>Undo</td>
<td>Undoes the last action.</td>
</tr>
<tr>
<td>Redo</td>
<td>Redoes the last action.</td>
</tr>
<tr>
<td>Extended Properties</td>
<td>Displays any extended properties set in Logix Designer that can be modified. Select the ellipsis button (...) to open the Expression Builder.</td>
</tr>
</tbody>
</table>

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

**See also**

[Apply decoration to tag values](#) on page 113
Change the name or description

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

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 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 Substitution 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.

**See also**

*Substitution Builder window* on page 62

**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 / Edit Parameter** 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.

**See also**

*Add a new parameter* on page 66

*Add New / Edit Parameter window* on page 73
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.

See also

Add an external reference on page 67
References window on page 62

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.

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 window opens.
4. Edit the expression using the **Expression Builder**.

5. Click **OK**.

**See also**

- **Expression Builder window on page 56**

**Apply decoration to 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 which has an alarm.

2. In **Decorator Panel - Library Object Properties**, select **Alarms**.

3. Double-click an alarm in the list, edit the properties, then select **OK**.

**See also**

- **Alarms on page 119**
- **Edit Alarm on page 121**

**Parameterize tag extended 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**.

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 Decorator Panel - Library Object Properties, select the Description Language to parameterize an extended property for a specific language.

3. In Decorator Panel - Library Object Properties, select Tags, then select a tag from the list.


5. In Extended Properties, select an extended property, then select the ellipsis button (...) to open the Expression Builder.

6. Use Expression Builder to write, validate, and test the expression, then click OK.

See also

Expression Builder on page 56

Extended Properties on page 119

Decorator Panel - Library Object Properties on page 106

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**.

See also

- Set a rule for instantiation on page 116
- Change the name or description on page 111

**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.

3. In Usage, 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 field.

See also

Add a tag as a parameter on page 112

Expression Builder window on page 56

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
**Tips:** Rung names 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 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 Substitution.

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

2. Review the substitutions listed in the Element Substitution 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.

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 window.

8. Click OK to exit the Substitution Builder.

9. (optional) If supported by the element, check the Exclude Base Library Substitutions or Exclude Library Substitutions checkbox so the new substitution takes precedence.

See also

Substitution Builder window on page 62

**Activate a new element**

When modifications are made in the Studio 5000 Logix Designer application to an element which has been added to a library object in the Library Designer, the element must be updated in the 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.

**See also**

[Apply decoration to library object elements](#) on page 106

### Extended Properties

**How do I open 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 Logix Designer can be parameterized. Overridden pass-through values are displayed in **Extended Properties**.

**See also**

[Parameterize tag extended properties](#) on page 114

### Alarms

**How do I open Alarms?**

1. In **Library Objects**, select a controller or local tag.

2. In **Decorator Panel - Library Object Properties**, select **Alarms**.

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

#### 01 General
- **Name**
  - The alarm definition name. Can contain up to 40 characters.
  - Read only
- **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 which 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 type is **DEV_HI** or **DEV_LO**.

#### 03 Data
- **On Delay**
  - Duration in milliseconds from the time the alarm condition occurs until the alarm notification is set.
  - Allows substitution
- **Off Delay**
  - Duration in milliseconds from the time 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.
    - 501...750 are **High** priority.
    - 751...1000 are **Urgent** priority.
  - Allows substitution
- **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

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
<th>Read Only / Allows Substitution</th>
</tr>
</thead>
<tbody>
<tr>
<td>FactoryTalk View Command</td>
<td>A FactoryTalk View command that can be run on the operator station when the alarm notification appears.</td>
<td>Allows substitution if the alarm does not have an Alarm Definition attribute.</td>
</tr>
<tr>
<td>Latched</td>
<td>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.</td>
<td>Allows substitution.</td>
</tr>
<tr>
<td>Acknowledgment Required</td>
<td>Sets if acknowledgment is required for the alarm.</td>
<td>Allows substitution.</td>
</tr>
<tr>
<td>Include in Alarm Set roll-up count</td>
<td>Sets if this alarm is included in the rollup counters of an alarm set containing this alarm.</td>
<td>Allows substitution.</td>
</tr>
<tr>
<td>Include in Alarm Set operations</td>
<td>Sets if this alarm is included when an operation is performed on an alarm set containing this alarm.</td>
<td>Allows substitution.</td>
</tr>
</tbody>
</table>

#### See also

- [Apply decoration to a tag alarm condition](#) on page 114
- [Edit Alarm](#) on page 121

### Edit Alarm

How do I open Edit Alarm?

1. In **Library Objects**, select a controller or local tag which has an alarm.
2. In **Decorator Panel - Library Object Properties**, 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** or **Substitution Builder** for the property.

#### See also

- [Apply decoration to a tag alarm condition](#) on page 114
- [Expression Builder](#) on page 56
- [Substitution Builder](#) on page 89
Chapter 5

Library Object Manager

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 SE/ME) and Historian (FactoryTalk Historian SE) 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 Historian Tags and FactoryTalk Alarms and Events (FTAE) Digital Alarms.
- Create repositories for storing and distributing library object files.

See also

Navigate the Library Object Manager interface on page 124
Navigate the Library Object Manager 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.

<table>
<thead>
<tr>
<th>Interface element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Menu bar</td>
<td>Use to configure settings for library attributes and view log information.</td>
</tr>
</tbody>
</table>
| 2 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 base 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 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. 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, the Library Object Manager application 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.  
    Click the + icon to the left of an item in the tree to display elements that are contained within it. Click the - icon to collapse the display element.  
<p>| 3 Library Content         |                                                                                              |
| 4 Properties              |                                                                                              |</p>
<table>
<thead>
<tr>
<th>Interface element</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 the 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 Alarms and Events: A FactoryTalk Alarms and Events 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 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 drop down 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 or the Tag Browser. Click the Apply button to accept any changes made to the properties settings. |

See also

- Set the default options for library object classification on page 125
- Add FactoryTalk View SE/FactoryTalk View Machine Edition content to a library object on page 158
- Add FactoryTalk Alarms and Events content to a library object on page 166
- Add FactoryTalk Historian content to a library object on page 177
- Expression Builder on page 157

Set the default options for library object classification

Use the Settings dialog box to set default options for the Solutions, Library Type, Category and Family items in the Properties panel.

To set the default options for library object classification

1. Click Settings > Library Attributes.
   
   The Settings dialog box opens displaying the Solutions, Library Type, Category and Family 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.

See also

Settings dialog box reference on page 126
Create a list of options on page 126

Create a list of options

How do I open the List Editor?

1. Click Settings > Library Attributes.

2. Click the ellipsis (...) button next 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. 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.

See also

Settings dialog box reference on page 126

Set the default options for library object classification on page 125

Settings dialog box reference

Use the Settings dialog box to set default options for the Solutions, Library Type, Category and Family 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.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solutions</td>
<td>The library object repository for a set of library objects.</td>
</tr>
<tr>
<td>Library Types</td>
<td>The general classification for a library object based on its function. Example: module, value, or motor.</td>
</tr>
<tr>
<td>Categories</td>
<td>A more specific classification for a library object, based on its function. Example: digital, analog, communication, or controller type</td>
</tr>
<tr>
<td>Families</td>
<td>The specific identifier for the base catalog number of the item. Example: 1734, 1738, Logix, or Project</td>
</tr>
</tbody>
</table>

See also

- [Set the default options for library object classification](#) on page 125
- [Create a list of options](#) on page 126
Chapter 6

Library Repositories

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 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.

See also

- Add the Application Code Manager database as a repository on page 135
- Add a folder repository on page 134
- Add and mount an ACD repository on page 129
- Create and copy library objects on page 141

**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 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.

See also

- Unmount an ACD repository on page 130
- Move between the Library Object Manager application and the Studio 5000 Logix Designer application on page 131
- Move between the Library Object Manager application and the Library Designer on page 131
- Export an ACD repository to a L5X format on page 132
- Remove an ACD repository from the Library Object Manager application on page 132

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.
See also

Move between the Library Object Manager application and the Studio 5000 Logix Designer application on page 131

Move between the Library Object Manager application and the Library Designer on page 131

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 129
- Unmount an ACD repository on page 130

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.

See also

Move between the Library Object Manager application and the Library Designer on page 131

Create and copy library objects on page 141

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 129

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.

See also

Export an ACD repository to a L5X format on page 132

Create and copy library objects on page 141

Export an ACD repository to a L5X format

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 repository and then select Export to L5X.

2. The Save As window opens. Navigate to the folder location to save the file and click Save.

See also

Remove an ACD repository from the Library Object Manager application on page 132

Remove an ACD repository from the Library Object Manager application

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.

See also

- Add a folder repository on page 134
- Add the ACM database as a repository on page 135
- Create and copy library objects on page 141

**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.

See also

- Rename a group on page 133
- Delete a group on page 133

**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**.

See also

- Add a group on page 133
- Delete a group on page 133

**Delete 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.

See also

- Add a group on page 133
- Rename a group on page 133

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 an existing folder or create a new one.

   a. To select an existing folder, navigate to the folder and double-click the listing. Click **OK**.

   b. To create a new folder, 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**. The Folder opens.

See also

- Remove a folder repository from the Library Object Manager application on page 135
- Create and copy library objects on page 141
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.

See also

*Add the Application Code Manager database as a repository* on page 135

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 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.
   a. To select the default ACM Database, click **OK**.
   b. To select a different data server, select from the pull-down 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**.
   c. To select a different database from the current server, select from the **Select or enter a database name** pull-down menu. Then click **OK**.
   d. 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 box 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.

   See also
   
   Remove the Application Code Manager database from the Library Object Manager application on page 136
   
   Connection Properties settings on page 136
   
   Advanced Connection Properties settings on page 137

Remove the Application Code Manager database from the Library Object Manager application

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 column, expand Repositories.

2. Right-click the ACM Database folder and select Remove. The ACM Database is removed from the Library Repositories column.

   See also
   
   Library Repositories on page 129
   
   Add the ACM database as a repository on page 135
   
   Add a folder repository on page 134
   
   Add and mount an ACD repository on page 129

Connection Properties settings

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.
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data source:</td>
<td>Database type. Always select Microsoft SQL Server (SqlClient)</td>
</tr>
<tr>
<td>Server name:</td>
<td>Selects a computer name and SQL Server instance from the list or type a computer name and SQL Server instance in the following format:</td>
</tr>
<tr>
<td></td>
<td>&lt;Computer Name&gt; &lt;SQL Server Instance&gt;</td>
</tr>
<tr>
<td>Log on to the server</td>
<td></td>
</tr>
<tr>
<td>Use Windows Authentication</td>
<td>Allows SQL Server log on using Windows authentication. When selected the logged on Windows user account credential will be sent to SQL Server to authenticate the session.</td>
</tr>
<tr>
<td>Use SQL Server Authentication</td>
<td>Allows SQL Server log on using SQL Server authentication. When selected the user name and password must be provided for authentication by the SQL Server.</td>
</tr>
<tr>
<td></td>
<td>• <strong>User name</strong>. The SQL Server user name, “sa” by default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Password</strong>. The SQL Server password associated with the user name specified, “ApplicationAdm1n” by default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Save my password</strong>. When selected, saves the SQL Server password specified so that it can be used in the next session.</td>
</tr>
<tr>
<td>Connect to a database</td>
<td></td>
</tr>
<tr>
<td>Select or enter a database name:</td>
<td>Select a database name from the list or enter a database name.</td>
</tr>
<tr>
<td>Attach a database file:</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>• Type a database file name or use the <strong>Browse</strong> button to use the <strong>Open</strong> dialog to locate the database file by clicking through the file system.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Logical name</strong>. Type the logical name of the database.</td>
</tr>
<tr>
<td>Advanced</td>
<td>Select to configure advanced database properties.</td>
</tr>
<tr>
<td>Test Connection</td>
<td>Tests the connection to the database. If a “Test connection succeeded.” message is not returned, check that the following settings are correct:</td>
</tr>
<tr>
<td></td>
<td>• Computer name</td>
</tr>
<tr>
<td></td>
<td>• SQL Server authentication</td>
</tr>
<tr>
<td></td>
<td>• Network access (remote SQL Server)</td>
</tr>
</tbody>
</table>

**See also**

- Add the Application Code Manager database as a repository on page 135
- Advanced Connection Properties settings on page 137

**ACM Database - Advanced Connection Properties settings**

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.

**Note:** Applying the recommended settings will improve ACM performance especially for network connections.
### Area

<table>
<thead>
<tr>
<th>Setting</th>
<th>Possible Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Advanced</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MultipleActiveResultSets</td>
<td>True, False (default)</td>
<td>When True, multiple result sets can be returned and read from one connection.</td>
</tr>
<tr>
<td><strong>Network Library</strong></td>
<td>blank (required if local) Named Pipes (DBNMPNTW) Shared Memory (DBMSLPCN) TCP/IP (DBMSSOCN) (recommended if networked) VIA (DBMSGNET)</td>
<td>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.</td>
</tr>
<tr>
<td><strong>Packet Size</strong></td>
<td>8000 (recommended)</td>
<td>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 and 32767 bytes.</td>
</tr>
<tr>
<td><strong>Transaction Binding</strong></td>
<td>Implicit Unbind (default) Explicit Unbind</td>
<td>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.</td>
</tr>
<tr>
<td><strong>Type System Version</strong></td>
<td>Latest (default) SQL Server 2012 SQL Server 2008 SQL Server 2005</td>
<td>Indicates which server type system the provider will expose through the DataReader.</td>
</tr>
<tr>
<td><strong>Connection Resiliency</strong></td>
<td>ConnectRetryCount 2 (recommended)</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>ConnectRetryInterval 5 (recommended)</td>
<td>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.</td>
</tr>
<tr>
<td><strong>Context</strong></td>
<td>Application Name .Net SqlClient Data Provider</td>
<td>The name of the application.</td>
</tr>
<tr>
<td></td>
<td>Workstation ID</td>
<td>The name of the workstation connecting to SQL Server.</td>
</tr>
<tr>
<td><strong>Initialization</strong></td>
<td>ApplicationIntent ReadWrite (default) ReadOnly</td>
<td>Declares the application workload type when connecting to a server.</td>
</tr>
<tr>
<td></td>
<td>Connect Timeout 30 (recommended)</td>
<td>The length of time in seconds to wait for a connection to the server before terminating the attempt and generating an error. A value of 0 indicates no limit, and should be avoided in aConnectionString because an attempt to connect waits indefinitely.</td>
</tr>
<tr>
<td></td>
<td>Current Language</td>
<td>The SQL Server Language record name.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Area</td>
<td>Setting</td>
<td>Possible Values</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>Pooling</td>
<td>Enlist</td>
<td>True (default) False</td>
</tr>
<tr>
<td></td>
<td>Load Balance Timeout</td>
<td>0 (default)</td>
</tr>
<tr>
<td></td>
<td>Max Pool Size</td>
<td>1000 (recommended)</td>
</tr>
<tr>
<td></td>
<td>Min Pool Size</td>
<td>0 (default)</td>
</tr>
<tr>
<td></td>
<td>PoolBlockingPeriod</td>
<td>Auto AlwaysBlock NeverBlock (recommended)</td>
</tr>
<tr>
<td></td>
<td>Pooling</td>
<td>True (recommended) False</td>
</tr>
<tr>
<td>Replication</td>
<td>Replication</td>
<td>False (default) True</td>
</tr>
<tr>
<td></td>
<td>Column Encryption Setting</td>
<td>Enabled Disabled (default)</td>
</tr>
<tr>
<td></td>
<td>Encrypt</td>
<td>True (default) False</td>
</tr>
<tr>
<td></td>
<td>Integrated Security</td>
<td>True False (default)</td>
</tr>
<tr>
<td></td>
<td>Password</td>
<td>*******</td>
</tr>
</tbody>
</table>
### Area Setting Possible Values Description

<table>
<thead>
<tr>
<th>Area</th>
<th>Setting</th>
<th>Possible Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persist Security Info</td>
<td>True</td>
<td>False (default)</td>
<td>When <strong>False</strong>, 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.</td>
</tr>
<tr>
<td></td>
<td>TrustServerCertificate</td>
<td>True (recommended)</td>
<td>When <strong>True</strong> (and <strong>Encrypt</strong> is set to <strong>True</strong>), SQL Server uses SSL encryption for all data sent between the client and server without validating the server certificate. If <strong>TrustServerCertificate</strong> is set to <strong>True</strong> and <strong>Encrypt</strong> is set to <strong>False</strong>, the channel is not encrypted.</td>
</tr>
<tr>
<td></td>
<td>User ID</td>
<td>sa</td>
<td>Indicates the user ID to be used when connecting to the data source.</td>
</tr>
<tr>
<td>Source</td>
<td>AttachedDbFilename</td>
<td></td>
<td>The name of the primary file, including the full path name, of an attachable database.</td>
</tr>
<tr>
<td></td>
<td>Context Connection</td>
<td>True (default)</td>
<td>When <strong>True</strong>, indicates the connection should be from the SQL Server context. Available only when running in the SQL Server process.</td>
</tr>
<tr>
<td></td>
<td>Data Source</td>
<td>localhost\SQLACM (default)</td>
<td>Indicates the name of the data source to connect to.</td>
</tr>
<tr>
<td></td>
<td>Failover Partner</td>
<td></td>
<td>The name or network address of the instance of SQL Server that acts as a failover partner.</td>
</tr>
<tr>
<td></td>
<td>Initial Catalog</td>
<td>Initial Database Name</td>
<td>The name of the initial catalog or database in the data source.</td>
</tr>
<tr>
<td></td>
<td>MultiSubnetFailover</td>
<td>True (default)</td>
<td>If your application is connecting to a high-availability, disaster recovery (AlwaysOn) availability group (AG) on different subnets, setting this value to <strong>True</strong> configures SqlConnection to provide faster detection of and connection to the (currently) active server.</td>
</tr>
<tr>
<td></td>
<td>TransparentNetworkIPResolution</td>
<td>True (default)</td>
<td>If your application connects to different networks, setting this value to <strong>True</strong> configures SqlConnection to provide transparent connection resolution to the currently active server, independently of the network IP topology. When set to <strong>True</strong>, 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 others IP addresses in parallel. When the first IP address answers, the application will establish the connection with the respondent IP address. If <strong>MultiSubnetFailover</strong> is set to <strong>True</strong>, this setting is ignored. If <strong>Failover Partner</strong> is specified, this setting is ignored. The default setting is <strong>False</strong> if <strong>Authentication</strong> is set to either <strong>Active Directory Password</strong> or <strong>Active Directory Integrated</strong>, otherwise the default setting is <strong>True</strong>.</td>
</tr>
<tr>
<td></td>
<td>User Instance</td>
<td>True (default)</td>
<td>Indicates whether the connection will be re-directed to connect to an instance of SQL Server running under the user’s account.</td>
</tr>
</tbody>
</table>

**See also**

- [Connection Properties settings on page 136](#)
- [Add the Application Code Manager database as a repository on page 135](#)
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.

See also

Create a library object on page 141

Copy a library object from the ACD repository to a folder repository or the ACM database on page 141

Copy multiple library objects from the ACD repository to a folder repository or the ACM database on page 142

Copy multiple library objects between folder repositories and the ACM database on page 143

Library Content on page 155

Create a new 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 folder or database icon and then select New Library. The New Library window opens.

2. Enter the information for the new library object. Red dots appear next to required items. After all of the required information is completed, the Apply button becomes available.

3. Click Apply to create the library object.

See also

Copy a library object from the ACD repository to a folder repository or the Application Code Manager database on page 141

New Library settings on page 144

Publish 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 copy 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** dialog box opens.

   **Important:** When copying 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 ACM database. Information that originates from the ACM database cannot be edited in the **Library Import Configuration** dialog box.

   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.

2. Enter the information for the library object.

3. Click **Apply** to add the library object to the repository.

**See also**

- Copy multiple library objects from the ACD repository to a folder repository or the ACM database on page 142
- New Library settings on page 144
- Library Import Configuration window on page 145

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

To copy multiple library objects from the ACD Repository

1. Select multiple library objects.

   a. Hold down the SHIFT key to select a contiguous block of library objects.

   b. Hold down the CONTROL 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** window opens.

**Important:** When copying 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 ACM database. Information that originates from the ACM database cannot be edited in the **Library Import Configuration** dialog box.

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.

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 **Apply** 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 check box for that library object to omit it from the import list.

See also

- [Create and copy library objects](#)
- [Copy multiple library objects between folder repositories and the ACM database](#)
- [New Library settings](#)
- [Library Import Configuration window](#)

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.

**Copy multiple library objects between folder repositories and the ACM database**

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 CONTROL 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 Apply 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.

**See also**

Create and copy library objects on page 141

**New Library settings**

The New Library dialog box has these settings.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>01 Library Status</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>Pull-down menu</td>
<td>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.</td>
</tr>
<tr>
<td><strong>02 Revision History</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Revision Description</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td><strong>03 Library Details</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CatalogNumber</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Description</td>
<td>Text entry</td>
<td>The description of the library object. This is a required setting.</td>
</tr>
<tr>
<td>Family</td>
<td>Pull-down menu or text entry</td>
<td>The family for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Solution</td>
<td>Pull-down menu or text entry</td>
<td>The solution for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Library Type</td>
<td>Pull-down menu or text entry</td>
<td>The library type for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Category</td>
<td>Pull-down menu or text entry</td>
<td>The category for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Content Type</td>
<td>Text</td>
<td>Displays the type of the object. Example: task, program, or routine.</td>
</tr>
<tr>
<td>Owner</td>
<td>Text entry</td>
<td>The user or entity that originally published the library object. This is a required setting.</td>
</tr>
</tbody>
</table>
### 04 Library Usage Rules

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Revision</td>
<td>Text entry (integer)</td>
<td>The major revision number for the library object. For a new library object, this defaults to 1.</td>
</tr>
<tr>
<td>Minor Revision</td>
<td>Text entry (integer)</td>
<td>The minor revision number for the library object. For a new library object, this defaults to 0.</td>
</tr>
</tbody>
</table>

#### 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. 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. Rule is entered manually as a logical expression.

### See also

- [Create a library object](#) on page 141
- [Create and copy library objects](#) on page 141

### Library Import Configuration window

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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01 Library Status</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>Pull-down menu</td>
<td>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.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>03 Library Details</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CatalogNumber</td>
<td>Text entry</td>
<td>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. If you are copying into the ACM Database, the setting value is generated by the database and cannot be edited.</td>
</tr>
<tr>
<td>Description</td>
<td>Text entry</td>
<td>The description of the library object. This is a required setting. If you are copying into the ACM Database, the setting value is generated by the database and cannot be edited.</td>
</tr>
<tr>
<td>Family</td>
<td>Pull-down menu</td>
<td>The Family of the library object. This is a required setting. If you are copying into the ACM Database, the setting value is generated by the database and cannot be edited.</td>
</tr>
<tr>
<td>Solution</td>
<td>Pull-down menu</td>
<td>The Solution for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Library Type</td>
<td>Pull-down menu</td>
<td>The Library Type for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Category</td>
<td>Pull-down menu</td>
<td>The Category for the library object. This is a required setting.</td>
</tr>
<tr>
<td>Content Type</td>
<td>Text</td>
<td>Displays the type of the object, Task, Program, or Routine.</td>
</tr>
<tr>
<td>Owner</td>
<td>Text entry</td>
<td>The user or entity that originally published the library object. This is a required setting. If you are copying into the ACM Database, the setting value is generated by the database and cannot be edited.</td>
</tr>
<tr>
<td>Major Revision</td>
<td>Text entry (integer)</td>
<td>The major revision number for the library object. For a new library object, this defaults to 1. If you are copying into the ACM Database, the database generates a default entry based on existing library objects with the same CatalogNumber.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Minor Revision</td>
<td>Text entry (integer)</td>
<td>The minor revision number for the library object. For a new library object, this defaults to 0. If you are copying into the ACM Database, the database generates a default entry based on existing library objects with the same CatalogNumber.</td>
</tr>
</tbody>
</table>

**04 Library Usage Rules**

<table>
<thead>
<tr>
<th>Upstream Keys</th>
<th>Text entry</th>
<th>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. Rule is entered manually as a logical expression.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downstream Keys</td>
<td>Text entry</td>
<td>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. Rule is entered manually as a logical expression.</td>
</tr>
</tbody>
</table>

**See also**

- [Copy a library object from the ACD repository to a folder repository or the Application Code Manager database](#) on page 141

- [Create and copy library objects](#) on page 141
Chapter 7

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 detail with the library, such as specifications or usage documentation.

See also

Add an attachment on page 147
Assign an attachment reference to a library on page 148
Import an attachment from an .HZ1 file on page 150
Export an attachment to an .HZ1 file on page 151

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, 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.

See also

Assign an attachment reference to a library on page 148
Attachment on page 152
Assign an attachment to a library

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.

See also

Add an attachment on page 147
Delete an attachment reference from a library on page 150

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, change the properties of the attachment, then click OK.

See also

Attachment on page 152
Change the include condition of an attachment reference

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.
4. In Expression Builder, modify, test, and save the include condition, then select OK.
5. Click Apply.

See also

- Extract an attachment on page 150
- Expression Builder on page 157
- Edit an attachment on page 148

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.
3. Select one or more attachments from the list.
4. Click Delete.

See also

- Delete an attachment reference from a library on page 150
- Attachments on page 151
Delete an attachment reference from a library

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.

See also

Assign an attachment reference to a library on page 148
Delete an attachment on page 149

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.

See also

Attachment on page 152

Import an attachment from an .HZ1 file

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.

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.

See also

- Export an attachment to an .HZ1 file on page 151
- Attachments on page 151

### Export an attachment to an .HZ1 file

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**.
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**.

See also

- Import an attachment from an .HZ1 file on page 150
- Attachments on page 151

### 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.

<table>
<thead>
<tr>
<th>Button or Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add New Attachment</td>
<td>Opens <strong>Attachment</strong> to add a new attachment file to the database.</td>
</tr>
<tr>
<td>Edit</td>
<td>Opens <strong>Attachment</strong> to edit the properties of an existing attachment file.</td>
</tr>
<tr>
<td>Delete Attachments</td>
<td>Removes the selected attachment files from the database.</td>
</tr>
</tbody>
</table>
### Chapter 7  
Library Attachments

<table>
<thead>
<tr>
<th><strong>Button or Column</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Files</td>
<td>Extracts the selected attachment file(s) to a folder location. The output files are extracted to their original filenames and formats.</td>
</tr>
<tr>
<td>Export to .HZ1 Attachment Files</td>
<td>Exports the selected attachment file(s) to a folder location. The output files are exported with filenames matching the file ID in the .HZ1 format.</td>
</tr>
<tr>
<td>Import from .HZ1 Attachment Files</td>
<td>Imports one or more .HZ1 format attachment files to from a folder location.</td>
</tr>
</tbody>
</table>
| 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.                                                                                           |

### See also

- Add an attachment on page 147
- Edit an attachment on page 148
- Extract an attachment on page 150
- Import an attachment from an .HZ1 file on page 150
- Export an attachment to an .HZ1 file on page 151

### 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.

<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>File ID</td>
<td>Displays the unique file identifier of the attachment.</td>
</tr>
<tr>
<td>Name</td>
<td>Sets the name of the attachment.</td>
</tr>
<tr>
<td>File</td>
<td>Sets the filename of or path to the attached file.</td>
</tr>
<tr>
<td>Description</td>
<td>Sets the description of the attachment.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revision Description</td>
<td>Sets the description of the last change or revision to the attachment.</td>
</tr>
<tr>
<td>Last Updated</td>
<td>Displays the last date and time the attachment was revised.</td>
</tr>
<tr>
<td>Last User Updating File</td>
<td>Displays the last user that revised the attachment.</td>
</tr>
</tbody>
</table>

**See also**

- Add an attachment on page 147
- Edit an attachment on page 148
Chapter 8

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 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 HSL4 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
Chapter 8  Library Content

- Symbol Builder
- Tag Editor
- Message Editor

See also

- Review the decoration settings of a library object on page 156
- View the Logix code of a library object on page 157
- Add FactoryTalk View SE/FactoryTalk View Machine Edition content to a library object on page 158
- Add FactoryTalk Alarms and Events content to a library object on page 166
- Add FactoryTalk Historian content to a library object on page 177

Review the decoration settings of a library object

The **Library Content** 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.

See also

- Library Content on page 155
- Move between the Library Object Manager application and the Studio 5000 Logix Designer application on page 131
- Move between the Library Object Manager application and the Library Designer on page 131
View the Logix code of a library object

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 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.

See also

Library Content on page 155

Move between the Library Object Manager application and the Studio 5000 Logix Designer application on page 131

Move between the Library Object Manager application and the Library Designer on page 131

Add FactoryTalk View SE/FactoryTalk View Machine Edition content to a library object on page 158

Expression Builder

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.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The <strong>Add New</strong> button is only active on the <strong>Parameter</strong> tab and opens the <strong>Add New Parameter</strong> window.</td>
<td></td>
</tr>
</tbody>
</table>
| The **Move Up/Move Down** buttons are not active in the **Expression Builder** window. | The find drop-down 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 drop down 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.  

Decorative element tabs  
The decorative elements available to the current expression. These include all elements added to the current library object and all elements added to 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.  
Expression 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.

**See also**

- **Library Content** on page 155
- **Review the decoration settings of a library object** on page 156
- **View the Logix code of a library object** on page 157

**Add FactoryTalk View SE/FactoryTalk View Machine Edition content to a library object**

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** (Site Edition) and **ME Symbols** (Machine Edition), 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. Click the **SE Symbols** or **ME Symbols** folder to select it.

b. Right-click the folder and select **Add**. The **Symbol Builder** dialog box opens.

c. In **Select Display Support File**, enter the file location for the exported FactoryTalk View Symbol file and then press ENTER or click the ellipsis (...) button to open Windows Explorer and navigate to the file and click **Open**. The symbols contained in the file are added to the **Select Root Node of Symbol** list.

d. In the **Select Root Node of Symbol** list select the symbol or group to import.

e. (optional) In the **Apply Substitutions** 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 **Apply Substitutions** list.

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

**See also**

*Symbol builder dialog box* on page 165

---

**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 that needs 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.

See also

Expression Builder on page 157

Apply positioning to a symbol on page 162

Configure the object tag and path parameters for a symbol

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.

   \{{\text{AreaPath}\}{\text{ObjectName}}} \n
   **Important:** The value entered must include the second pair of curly brackets enclosing the \(\text{AreaPath}\) and \(\text{ObjectName}\) tokens. Note that the Path Parameter is also a part of the Object Tag Parameter.

3. Enter the following value in the Path Parameter.

   \{{\text{AreaPath}}\} \n
   **Important:** The value entered must include the second pair of curly brackets enclosing the \(\text{AreaPath}\) 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. If for some reason they need to be changed, 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 a value is configured in the AreaPath parameter for the controller.

**See also**

- [Apply positioning to a symbol on page 162](#)
- [Delete a symbol from a library object on page 163](#)
- [Adding FactoryTalk Alarms and Events (FTAE) Content to a library object on page 166](#)

**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 top left corner, using absolute or relative positioning.

In absolute positioning, the symbol is positioned relative to the top 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 top 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 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.

See also

- Add FactoryTalk View SE/FactoryTalk View Machine Edition content to a library object on page 158

- Delete a symbol from a library object on page 163

Delete a symbol from a library object

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.
See also

Adding FactoryTalk Alarms and Events (FTAE) Content to a library object on page 166

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 dialog box opens.

3. Select Display Support 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 Windows Explorer and navigate to the file and the click Open. The Select Root Node of Symbol list updates to show the symbols contained in the file.

4. In the Select Root Node 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.

See also

Apply positioning to a symbol on page 162

Delete a symbol from a library object on page 163

Symbol properties on page 164

Symbol builder dialog box on page 165

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

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbol Name</td>
<td>Text entry</td>
<td>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 <strong>Symbol Name</strong> setting.</td>
</tr>
</tbody>
</table>
| 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 added to an ACM project. With FactoryTalk View Symbols, there may be several options which are appropriate for different applications. In this case, add a parameter token to the **Include Condition** property which 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 (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.

**See also**

- [Expression Builder](#) on page 157
- [View the Logix code of a library object](#) on page 157

### 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.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Display Export File</td>
<td>Text entry or browse</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Setting Type Description

<table>
<thead>
<tr>
<th>Setting</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Root Node of Symbol</td>
<td>List</td>
<td>A list of all symbols in the file. Symbols can be selected individually or as a group. However, only one symbol or group can be selected for each import.</td>
</tr>
<tr>
<td>Apply Substitutions</td>
<td>Checkboxes</td>
<td>When selected, apply substitutions for symbol attributes. There are two types of substitutions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Predefined.</strong> The default substitutions that are applied by Library Object Manager when a symbol is imported.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Saved.</strong> Custom user-defined substitutions developed in response to a specific implementation requirement.</td>
</tr>
</tbody>
</table>

### Add FactoryTalk Alarms and Events content to a library object

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.

### See also

- Add a digital alarm to a library object on page 166
- Add a message to a digital alarm on page 167
- Change the tag update rate for a digital alarm on page 171

### Add a digital 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.

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.

See also

Add FactoryTalk Alarms and Events content to a library object on page 166
Digital alarm properties on page 172

Add a new message to a digital alarm

Alarm messages describe alarm conditions. Each alarm can be associated with only one message.

To add a new 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 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 a new 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.

See also

- Add an existing message to a digital alarm on page 170
- Create a message using the Message Editor on page 169
- Message Editor dialog box on page 176

Create a message using the Message Editor

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** 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 a new 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 **SaveExpression** 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.

See also

Message Editor dialog box on page 176

**Add an existing message to a digital alarm**

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.

**See also**

- [Create a message using the Message Editor](#) on page 169

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.

**See also**

- [Change the tag update rate for a digital alarm](#) on page 171

- [Add a new message to a digital alarm](#) on page 167

- [Create a message using the Message Editor](#) on page 169

**Delete message content from a library object**

**Change the tag update rate for a digital alarm**

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.

See also

Delete message content from a library object on page 171

Digital alarm properties - 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.
<table>
<thead>
<tr>
<th>Name</th>
<th>Setting Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Text entry or Expression</td>
<td>The name that will be applied to the digital alarm when the library object is instantiated. This name will appear in 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.</td>
</tr>
<tr>
<td>Include Condition</td>
<td>Text entry or Expression</td>
<td>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.</td>
</tr>
<tr>
<td>Input Tag</td>
<td>Text entry or Tag</td>
<td>The ItemID that will be applied to the Digital Alarm 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 browse to an alarm trigger tag, click the ellipsis (...) button to open the Tag Browser.</td>
</tr>
<tr>
<td>Condition</td>
<td>Pull-down menu</td>
<td>The condition that triggers the digital alarm: • Source &lt;&gt; 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.</td>
</tr>
<tr>
<td>Latched</td>
<td>Checkbox</td>
<td>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 check box.</td>
</tr>
<tr>
<td>Severity</td>
<td>Text entry or Tag</td>
<td>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 (example: when the connection to a controller is lost, or if a data server is rebooted). To browse to a severity tag, click the ellipsis (...) button to open the Tag Browser.</td>
</tr>
<tr>
<td>Acknowledge required</td>
<td>Checkbox</td>
<td>Determines whether an acknowledgment is required to turn off the alarm. If the alarm does not require acknowledgment by an operator, clear the check box. 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.</td>
</tr>
<tr>
<td>Minimum duration</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Show Alarm as Tag</td>
<td>Checkbox</td>
<td>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 check box to enable tags for this alarm. Clear the check box to disable tags for this alarm.</td>
</tr>
<tr>
<td>Message</td>
<td>Text entry, Message, or Expression</td>
<td>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 new 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.</td>
</tr>
<tr>
<td>ID</td>
<td>Read only</td>
<td>The unique numeric ID for the message in the Message setting.</td>
</tr>
</tbody>
</table>
### Name Setting Type Description

<table>
<thead>
<tr>
<th>Name</th>
<th>Setting Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td>Button</td>
<td>Opens the Message Editor to create a new message.</td>
</tr>
<tr>
<td>Edit</td>
<td>Button</td>
<td>Opens the Message Editor to edit an existing message.</td>
</tr>
<tr>
<td>Browse</td>
<td>Button</td>
<td>Opens the Message Browser.</td>
</tr>
<tr>
<td>Associated Tags</td>
<td>Text entry or Tag</td>
<td>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.</td>
</tr>
<tr>
<td>Alarm Class</td>
<td>Text entry or Expression</td>
<td>The class for the digital alarm. Type a classification string for the alarm that allows filtering of alarms at run time. 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.</td>
</tr>
<tr>
<td>FactoryTalk View Command</td>
<td>Text entry or Expression</td>
<td>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.</td>
</tr>
<tr>
<td>Alarm Group</td>
<td>Text entry or Expression</td>
<td>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 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.</td>
</tr>
</tbody>
</table>

### See also

*Digital alarm properties - Status Tags tab* on page 174

*Digital alarm properties - Control Tags tab* on page 175

*Expression Builder* on page 157

### Digital alarm properties - 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Setting Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disabled Tag</td>
<td>Text entry or Tag</td>
<td>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.</td>
</tr>
</tbody>
</table>
### 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 include **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** check box 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Setting Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disable Tag</td>
<td>Text entry or Tag</td>
<td>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.</td>
</tr>
</tbody>
</table>

---

### See also

- [Add a digital alarm to a library object](#) on page 166
- [Digital alarm properties - Digital tab](#) on page 172
### 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Setting Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message</td>
<td>Text entry</td>
<td>The message appears here as it is compiled. Text can be entered directly in this setting. Variable tags are added when Add is clicked.</td>
</tr>
</tbody>
</table>

**Add Variable:**

- **Variable** [Pull-down 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** [Pull-down 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.*

---

**See also**

- Add a digital alarm to a library object on page 166
- Digital alarm properties - Digital tab on page 172
- Digital alarm properties - Status Tags tab on page 174
### Name Setting Type Description

<table>
<thead>
<tr>
<th>Name</th>
<th>Setting Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Fixed Width</td>
<td>Checkbox</td>
<td>Limits the amount of information displayed for the tag value within the message to the value entered in the Number of Characters setting.</td>
</tr>
<tr>
<td>Number of Characters</td>
<td>Text entry</td>
<td>Sets the maximum number of characters to display for the tag value within the message.</td>
</tr>
<tr>
<td>Number of Digits</td>
<td>Text entry</td>
<td>Sets the number of digits to display for the tag value within the message.</td>
</tr>
<tr>
<td>Decimal Places</td>
<td>Text entry</td>
<td>Sets the number of decimal places to display for the tag value within the message.</td>
</tr>
<tr>
<td>Left Fill</td>
<td>Pull-down menu</td>
<td>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.</td>
</tr>
</tbody>
</table>

These settings appear if Numeric is selected in the Data Type setting.

Add Button | Text entry | The unique numeric ID for the message. |

**Add FactoryTalk Historian content to a library object**

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.

See also

Configure a Historian tag on page 179

Delete a Historian tag from a library object

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.

See also

Add a Historian tag to a library object on page 177
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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Setting Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag Name</td>
<td>Text entry or expression</td>
<td>The name that will be applied to the Historian Tag when the library object is instantiated. This name will appear in 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.</td>
</tr>
<tr>
<td>Include Condition</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Scan Class</td>
<td>Text entry</td>
<td>The scan class for the Historian Tag. The scan class consists of a period of time 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.</td>
</tr>
<tr>
<td>Descriptor</td>
<td>Text entry</td>
<td>A description 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.</td>
</tr>
<tr>
<td>Engineering Units</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Instrument Tag</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>FTLD Interface Number</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Point Type</td>
<td>Pull-down menu</td>
<td>The data type for the Historian point. Options are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Digital</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Float16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Float32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Float64</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Int16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Int32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Timestamp</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Blob</td>
</tr>
<tr>
<td>Typical Value</td>
<td>Text entry</td>
<td>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.</td>
</tr>
<tr>
<td>Zero</td>
<td>Text entry</td>
<td>The zero value for the Historian Tag. This setting is only active if a Float or Int option is selected for Point Type.</td>
</tr>
<tr>
<td>Span</td>
<td>Read only</td>
<td>The span for the Historian tag. This setting is only active if a Float or Int option is selected for Point Type.</td>
</tr>
</tbody>
</table>
See also

Delete a Historian tag from a library object on page 178

Expression Builder on page 157

Add Custom Collections content to a library object

You can create custom collections which can be used for documentation generation.

Follow these steps to add FactoryTalk Historian 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.

See also

Add an item to Custom Collections on page 180

Item properties on page 181

Add a new attribute on page 181

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 panel.

2. Right-click NewCollection, and then select Add.

   An item namedNewItem is added and you can configure its properties in the Properties panel.

See also

Add Custom Collections content to a library object on page 180

Item properties on page 181

Add a new attribute on page 181
The following table displays the options in the **Properties** panel of an item object.

<table>
<thead>
<tr>
<th>Options</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item Name</td>
<td>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 <strong>Expression Builder</strong>.</td>
</tr>
<tr>
<td>Include Condition</td>
<td>Displays the Include Condition of the item. The default is <strong>Always</strong>. A condition can be entered manually or generated by an expression. To create an Expression, click the ellipsis(...) button to open the <strong>Expression Builder</strong>.</td>
</tr>
<tr>
<td>Add New Attribute</td>
<td>Opens <strong>Attribute Builder</strong>.</td>
</tr>
<tr>
<td>Edit Attribute</td>
<td>Opens <strong>Attribute Builder</strong>.</td>
</tr>
<tr>
<td>Delete Attribute</td>
<td>Deletes an attribute.</td>
</tr>
<tr>
<td>Filter</td>
<td>Enter the key words or the name of an attribute to search it quickly.</td>
</tr>
<tr>
<td>Name</td>
<td>Displays the name of an attribute.</td>
</tr>
<tr>
<td>Value</td>
<td>Displays the value of an attribute.</td>
</tr>
</tbody>
</table>

**See also**

- [Add an item to Custom Collections](#) on page 180
- [Add a new attribute](#) on page 181

### Add a new attribute

Use the **Attribute Builder** dialog box to add an attribute to an item.

**To add a new 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 expression.
3. Select **OK**.

**See also**

- [Item properties](#) on page 181

### 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.

<table>
<thead>
<tr>
<th>Button or Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add New Attachment</td>
<td>Opens <strong>Attachment</strong> to add a new attachment file to the database.</td>
</tr>
<tr>
<td>Button or Column</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Edit</td>
<td>Opens Attachment to edit the properties of an existing attachment file.</td>
</tr>
<tr>
<td>Extract Files</td>
<td>Extracts the selected attachment file(s) to a folder location. The output files are extracted to their original filenames and formats.</td>
</tr>
<tr>
<td>Export to .HZ1 Attachment Files</td>
<td>Exports the selected attachment file(s) to a folder location. The output files are exported with filenames matching the file ID in the .HZ1 format.</td>
</tr>
<tr>
<td>Import from .HZ1 Attachment Files</td>
<td>Imports one or more .HZ1 format attachment files to from a folder location.</td>
</tr>
<tr>
<td>Filter</td>
<td>Limits the attachment list to attachments with properties matching the characters entered in the Filter field.</td>
</tr>
<tr>
<td></td>
<td>The filter function uses exact character matching and is case insensitive. Wildcard characters are not supported.</td>
</tr>
<tr>
<td>Select</td>
<td>Toggles whether a row is selected before deleting or exporting the attachments.</td>
</tr>
<tr>
<td>Name</td>
<td>Displays the name of the attachment.</td>
</tr>
<tr>
<td>Description</td>
<td>Displays the description of the attachment.</td>
</tr>
<tr>
<td>File Name</td>
<td>Displays the original file name of the attachment.</td>
</tr>
<tr>
<td>File ID</td>
<td>Displays the unique file identification number of the attachment.</td>
</tr>
<tr>
<td>Include Condition</td>
<td>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.</td>
</tr>
<tr>
<td>Extraction Path</td>
<td>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.</td>
</tr>
<tr>
<td>Revision Description</td>
<td>Displays the latest revision description of the attachment.</td>
</tr>
<tr>
<td>Modified Date</td>
<td>Displays the last modified date of the attachment.</td>
</tr>
<tr>
<td>Modified By</td>
<td>Displays the last user that modified the attachment.</td>
</tr>
</tbody>
</table>

See also

Attachments on page 151
Copyright notice

Copyright © 2019 Rockwell Automation Technologies, Inc. All Rights Reserved. Printed in USA.

This document and any accompanying Rockwell Software products are copyrighted by Rockwell Automation Technologies, Inc. Any reproduction and/or distribution without prior written consent from Rockwell Automation Technologies, Inc. is strictly prohibited. Please refer to the license agreement for details.

End User License Agreement (EULA)

You can view the Rockwell Automation End-User License Agreement ("EULA") by opening the License.rtf file located in your product’s install folder on your hard drive.

The software included in this product contains copyrighted software that is licensed under one or more open source licenses. Copies of those licenses are included with the software. Corresponding Source code for open source packages included in this product are located at their respective web site(s).

Alternately, obtain complete Corresponding Source code by contacting Rockwell Automation via the Contact form on the Rockwell Automation website: http://www.rockwellautomation.com/global/about-us/contact/contact.page
Please include "Open Source" as part of the request text.

A full list of all open source software used in this product and their corresponding licenses can be found in the OPENSOURCE folder included with these Release Notes. The default installed location of these licenses is C:\Program Files (x86)\Common Files\Rockwell\Help\<Product>\ReleaseNotes\OPENSOURCE\index.htm.

Trademark Notices

Allen-Bradley, Rockwell Automation, and Rockwell Software are trademarks of Rockwell Automation, Inc.
Any Rockwell Automation software or hardware not mentioned here is also a trademark, registered or otherwise, of Rockwell Automation, Inc.

Other Trademarks

All other trademarks are the property of their respective holders and are hereby acknowledged.

Trademarks not belonging to Rockwell Automation are property of their respective companies

Warranty

This product is warranted in accordance with the product license. The product’s performance may be affected by system configuration, the application being performed, operator control, maintenance, and other related factors. Rockwell Automation is not responsible for these intervening factors. The instructions in this document do not cover all the details or variations in the equipment, procedure, or process described, nor do they provide directions for meeting every possible contingency during installation, operation, or maintenance. This product’s implementation may vary among users.

This document is current as of the time of release of the product; however, the accompanying software may have changed since the release. Rockwell Automation, Inc. reserves the right to change any information contained in this document or the software at any time without prior notice. It is your responsibility to obtain the most current information available from Rockwell when installing or using this product.

Contact Rockwell Automation

Customer Support Telephone — 1.888.382.1583

Online Support — http://www.rockwellautomation.com/support/
## Rockwell Automation support

Use the following resources to access support information.

<table>
<thead>
<tr>
<th>Technical Support Center</th>
<th>Knowledgebase Articles, How-to Videos, FAQs, Chat, User Forums, and Product Notification Updates</th>
<th><a href="https://rockwellautomation.custhelp.com">https://rockwellautomation.custhelp.com</a></th>
</tr>
</thead>
</table>

## Documentation feedback