

Menu Items

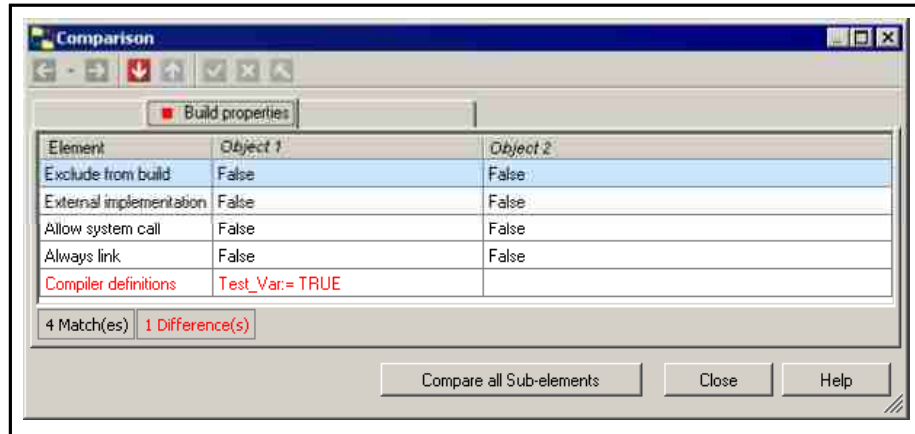


Fig.3-89: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: Network Variable Lists, Receiver End

The comparison applies to global variable lists used as network variables on the receiver end that were created as independent objects of an application.

These lists are created automatically and remain current within the project.

In this way, the comparison is only executed as a check for differences.

When comparing network variable lists, the content display is XML based. Modification (merging) is not possible.

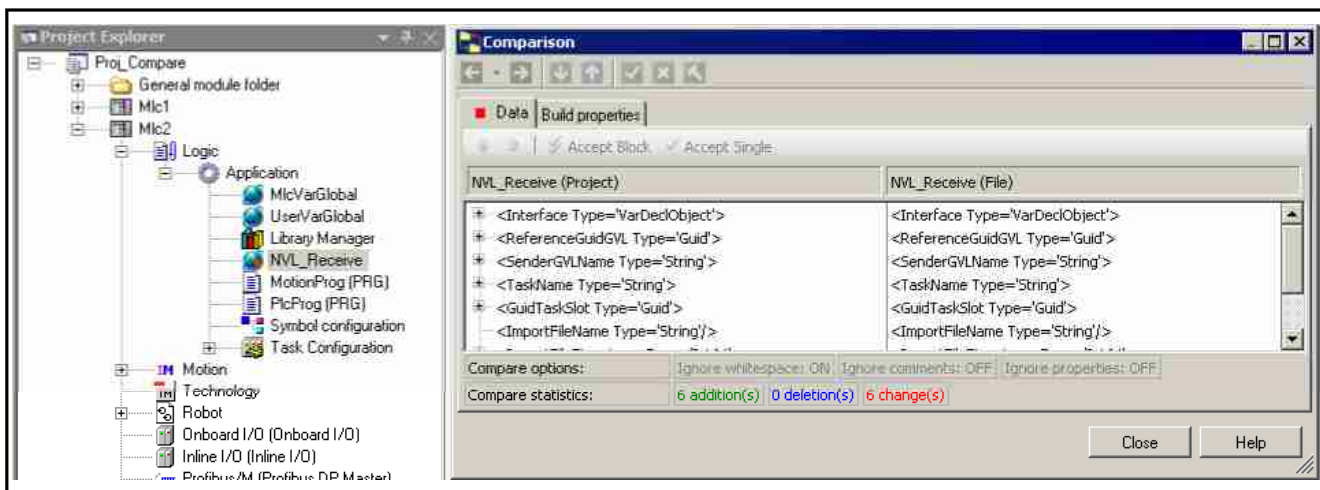




Fig.3-90: Comparison: Network variable lists on the receiver end

The next/previous difference is selected with  or .

The next tab shows the build properties.

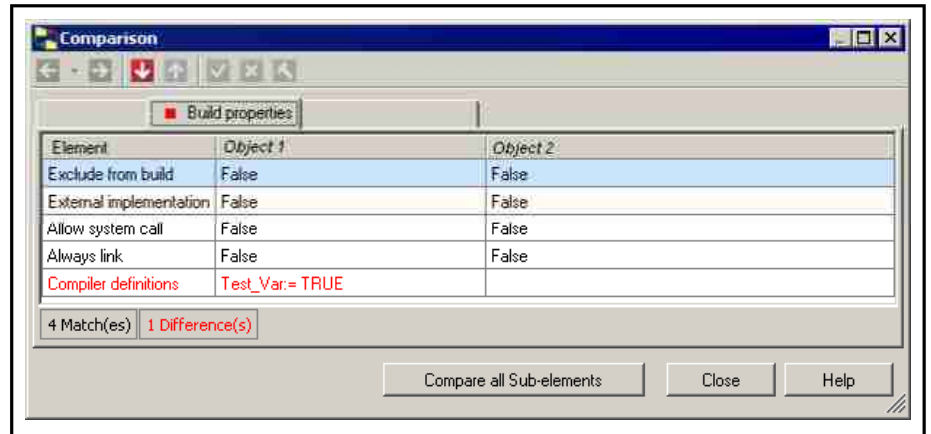






Fig.3-91: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs,  can be used to select the next difference ( for the previous difference).

If the checkmark  appears, it can be used to prepare the acceptance (merging):

Select the hammer  to merge.

Select the cross  to prevent (undo) merging.

Comparison: Programs, Function Blocks, Functions...

The comparison applies to the following objects:

- PROGRAM, FUNCTION_BLOCK, FUNCTION
- ACTION, TRANSITION
- METHOD, PROPERTY with GET and SET

The objects named can be declared and implemented in the IEC programming language editors available in the system.

Comparison: ST Comparisons of the declaration and objects which have been implemented in structured text are text based.

Menu Items

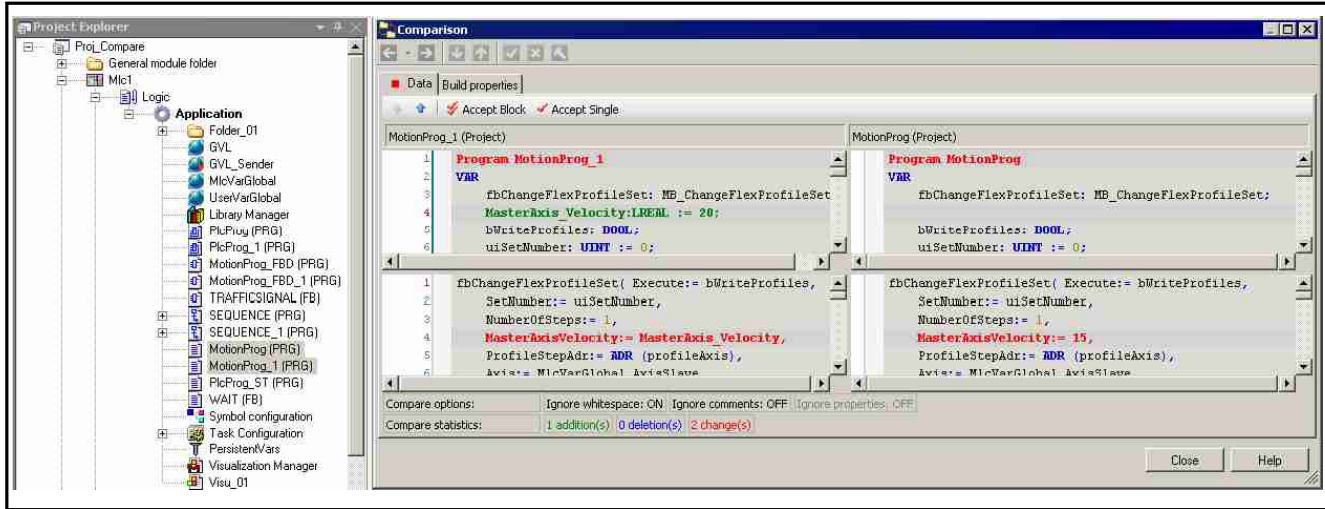


Fig.3-92: Comparison: PROGRAM in ST, declaration and implementation
Merging is possible in the declaration and in the implementation.

The next/previous difference is selected with or .

Accept Block can be used to accept the complete highlighted text block.

Accept Single can be used to accept text line by line. In this case, only the possible result of the merge is displayed first. Clicking the button again resets the display.

Select the hammer to merge.

Select the cross to prevent (undo) merging.

The next tab shows the build properties.

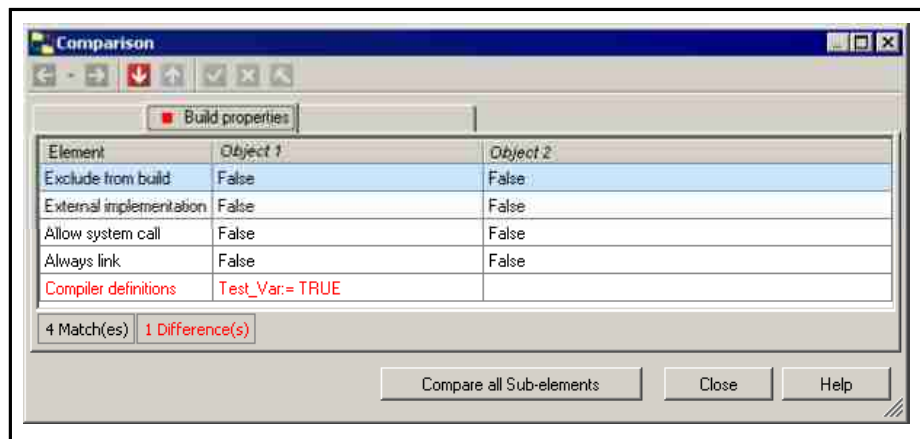


Fig.3-93: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Menu Items

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: FBD

Objects that have been implemented as function block diagram are compared graphically.

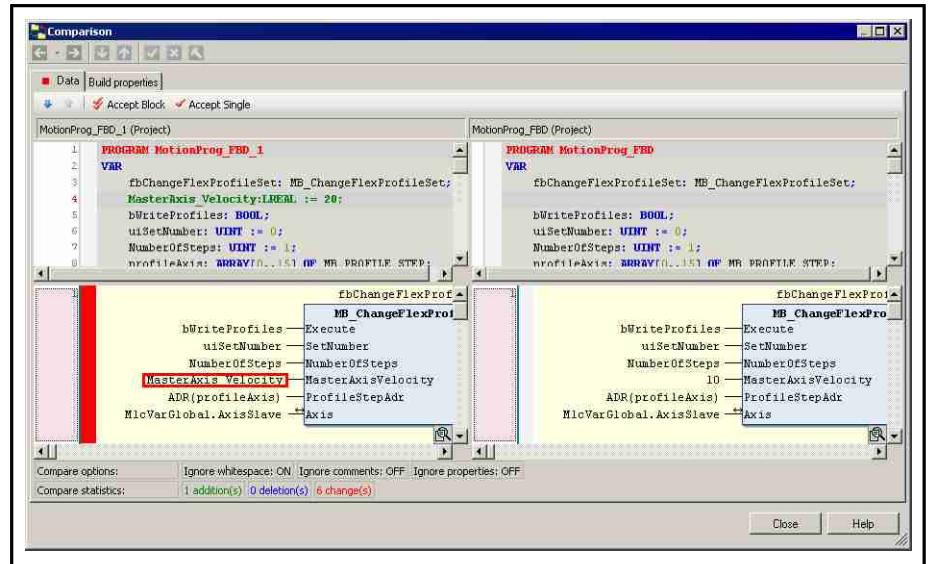


Fig.3-94: Comparison: PROGRAM in FBD, declaration and implementation

Merging is possible in the declaration and in the implementation.

The next tab shows the build properties.

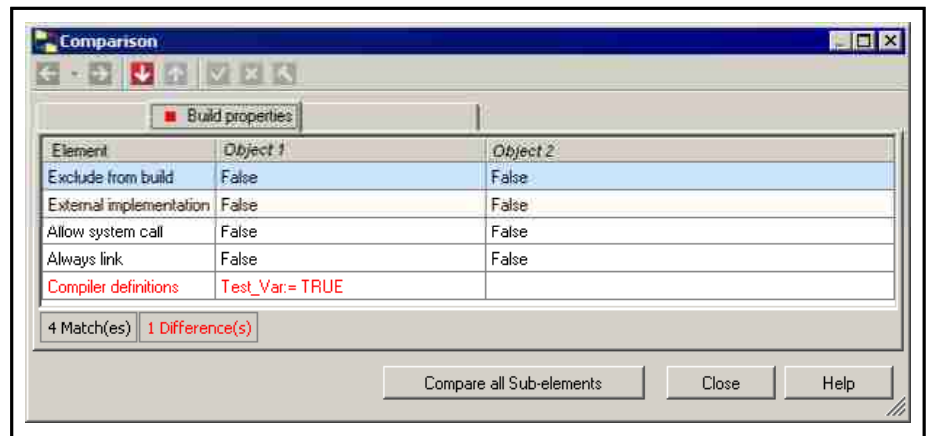


Fig.3-95: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Menu Items

Comparison: IL POUs implemented in the instruction list are compared as XML. Merging is not possible here.

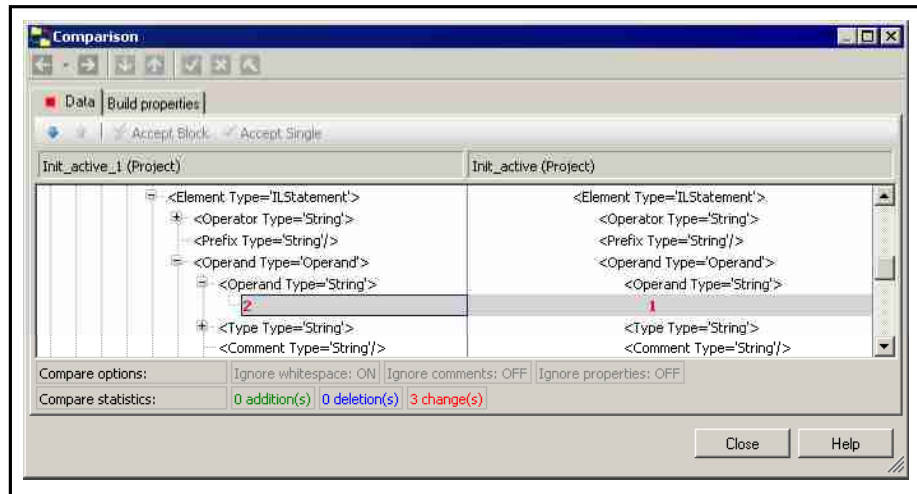


Fig.3-96: Comparison: Action in instruction list, implementation

The next tab shows the build properties.

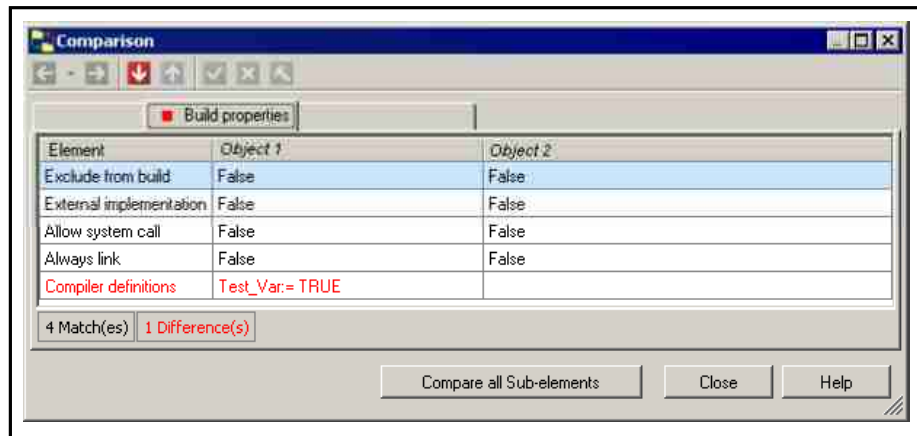


Fig.3-97: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: SFC Objects that have been structured by means of SFC (sequential function chart) are compared graphically. Merging is only possible in the declaration.

Menu Items

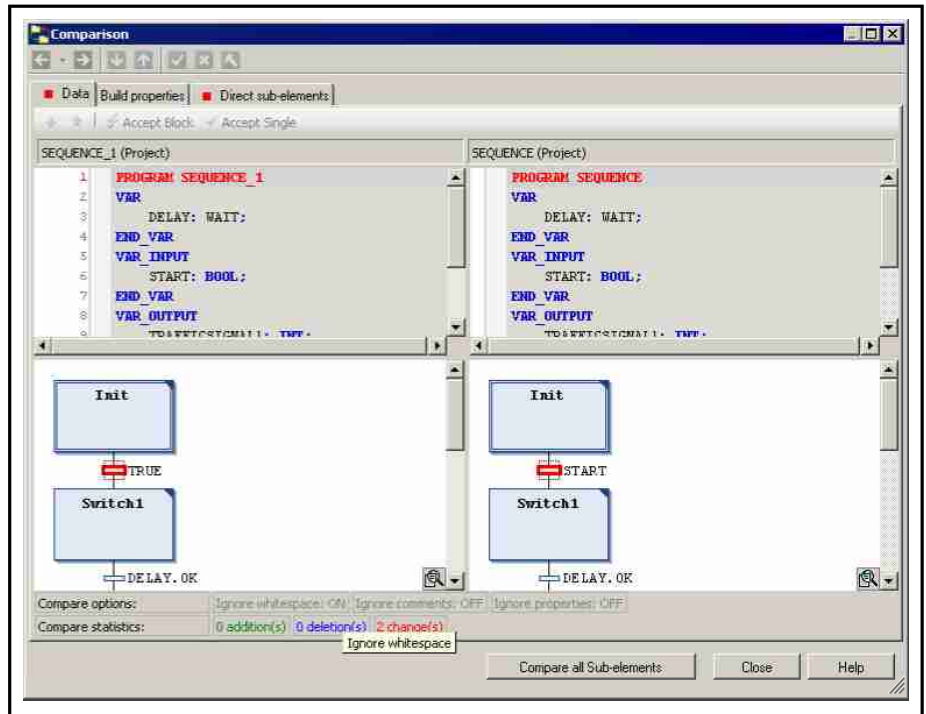


Fig.3-98: Comparison: FUNCTION_BLOCK with SFC structure, declaration and implementation

The next tab shows the build properties.

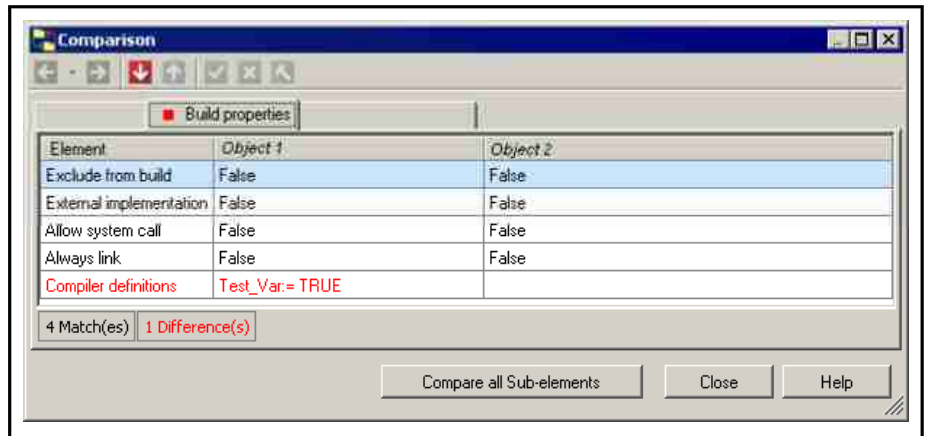


Fig.3-99: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Menu Items

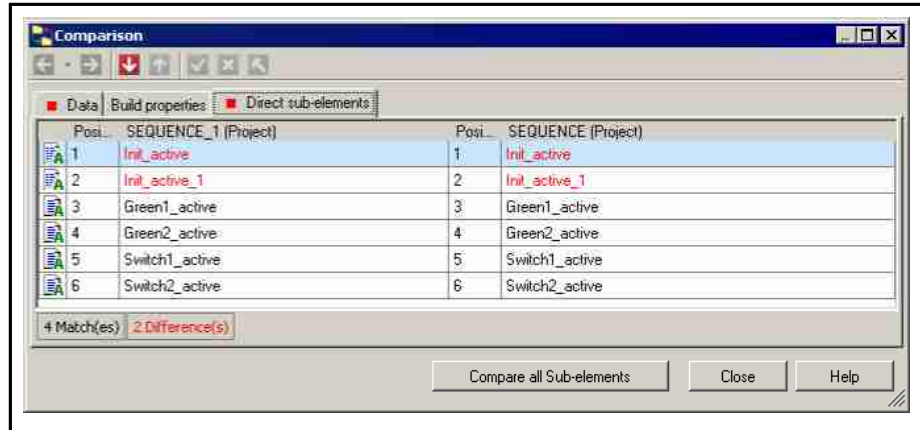


Fig.3-100: Comparison: POU "Direct subelements"

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

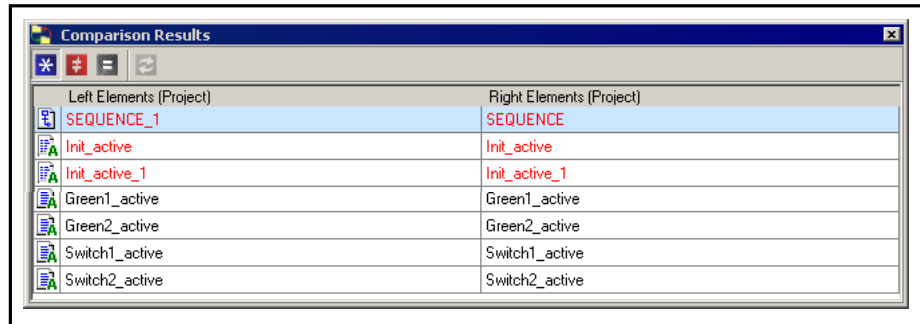


Fig.3-101: Comparison: POU with all subelements with the filter option "All elements", "Different elements", "Identical elements"

Comparison: CFC

POUs implemented in the instruction list are compared as XML. Merging is not possible here.

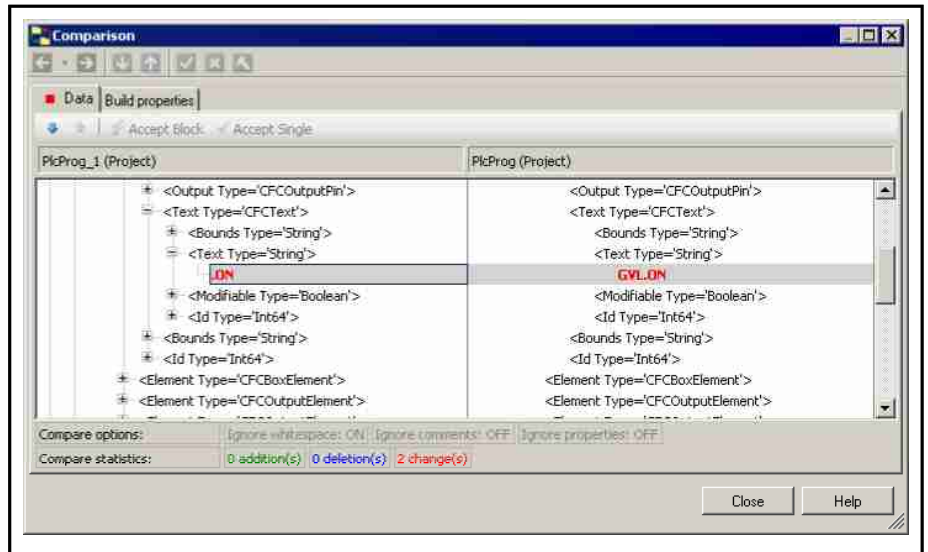


Fig.3-102: Comparison: PROGRAM in CFC, declaration and implementation
The next tab shows the build properties.

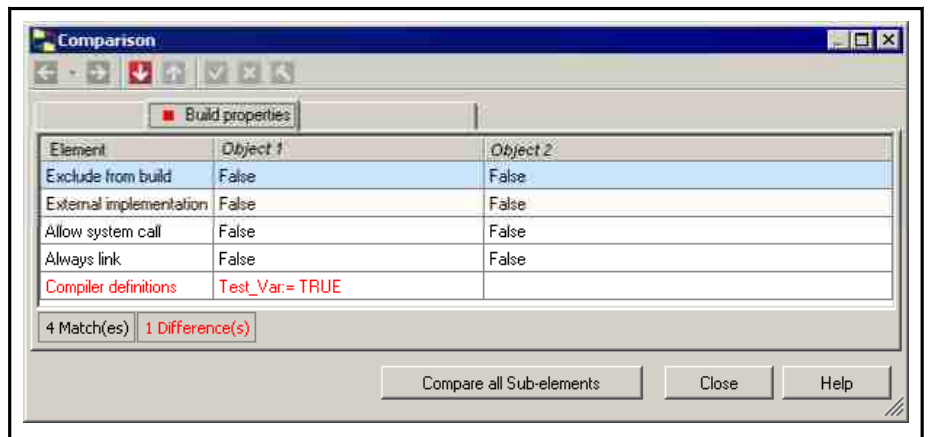


Fig.3-103: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: Interfaces, Interface Methods, Interface Properties

When comparing interfaces, interface methods or interface properties, the content display is XML based.

Modification (merging) is not possible.

The next/previous difference is selected with or .

Menu Items

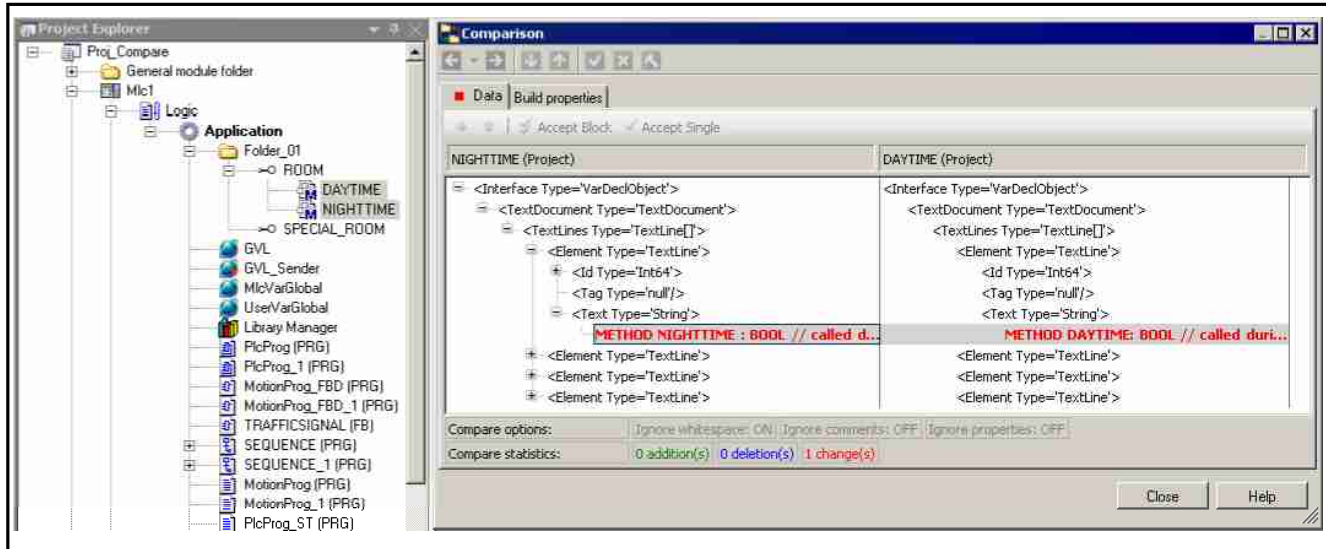


Fig.3-104: Comparison: Interfaces, interface methods, interface properties

The next tab shows the build properties.

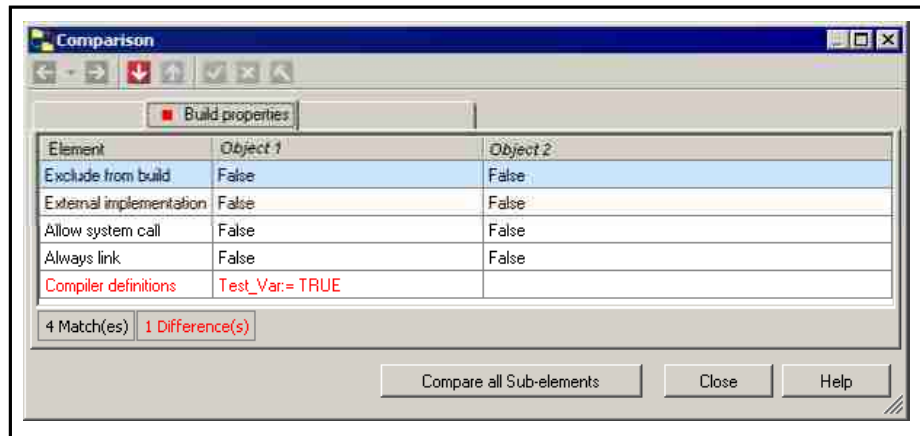


Fig.3-105: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: Library Manager

When comparing Library Managers, the content is listed in tabular form. Acceptance is possible line-by-line.

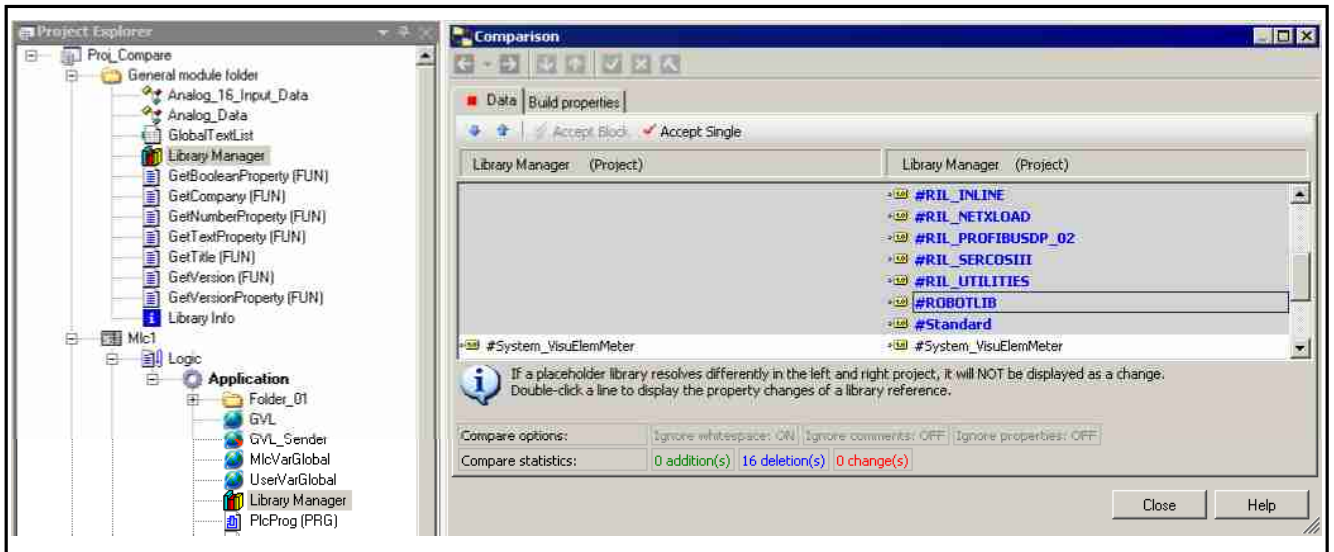


Fig.3-106: Comparison: Library Manager, here ML_Robot added into the project

The next/previous difference is selected with or .

Accept Block can be used to accept the complete highlighted text block.

Accept Single can be used to accept text line by line. In this case, only the possible result of the merge is displayed first. Clicking the button again resets the display.

Select the hammer to merge.

Select the cross to prevent (undo) merging.

The next tab shows the build properties.

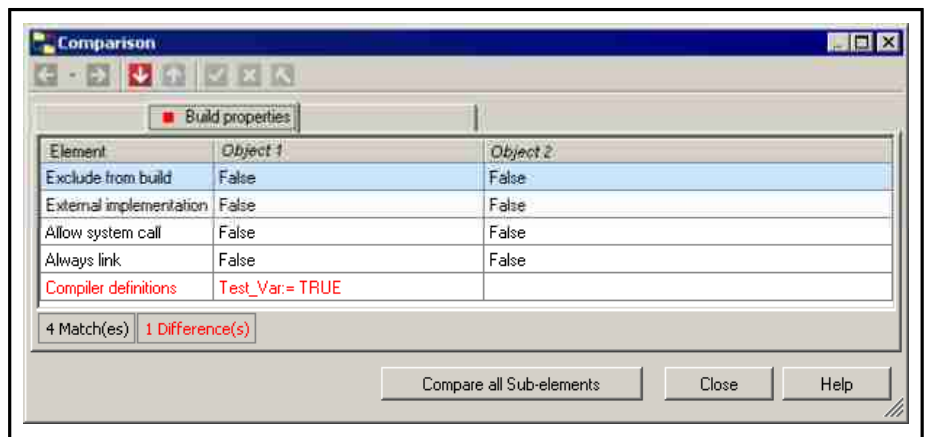


Fig.3-107: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Menu Items

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: Library Info

When comparing "library info", the content display is XML based. Modification (merging) is not possible.

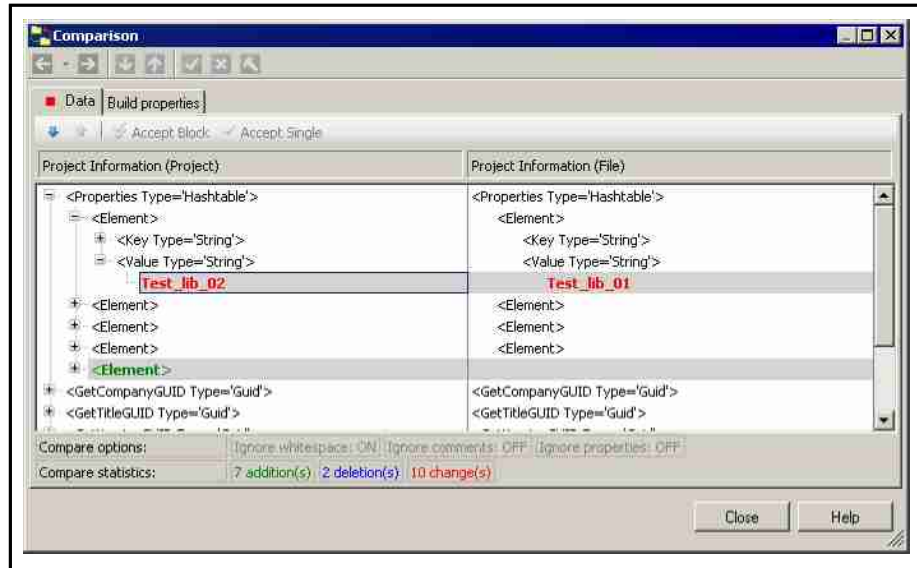


Fig.3-108: Comparison: Library Info; library enabled with version number 10.06.0.1



The terms "Project Information" and "Library Info" designate the same element.

The next tab shows the build properties.

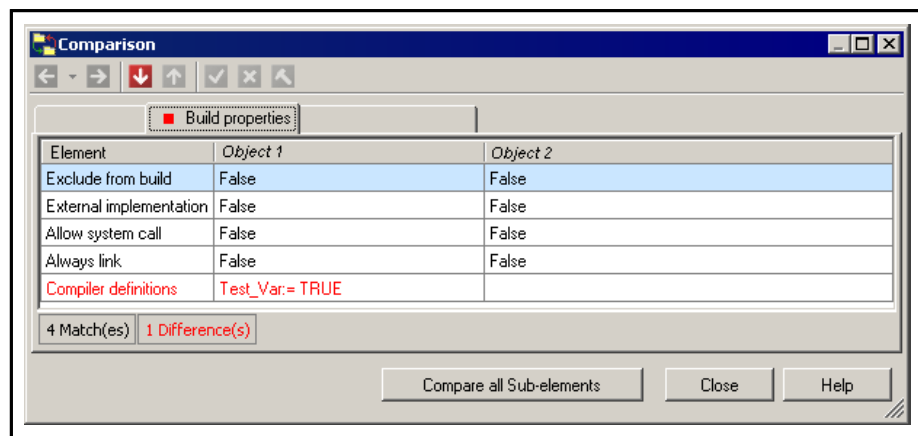


Fig.3-109: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

Menu Items

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: Data Server

When comparing data servers, the content display is XML based. Modification (merging) is not possible.

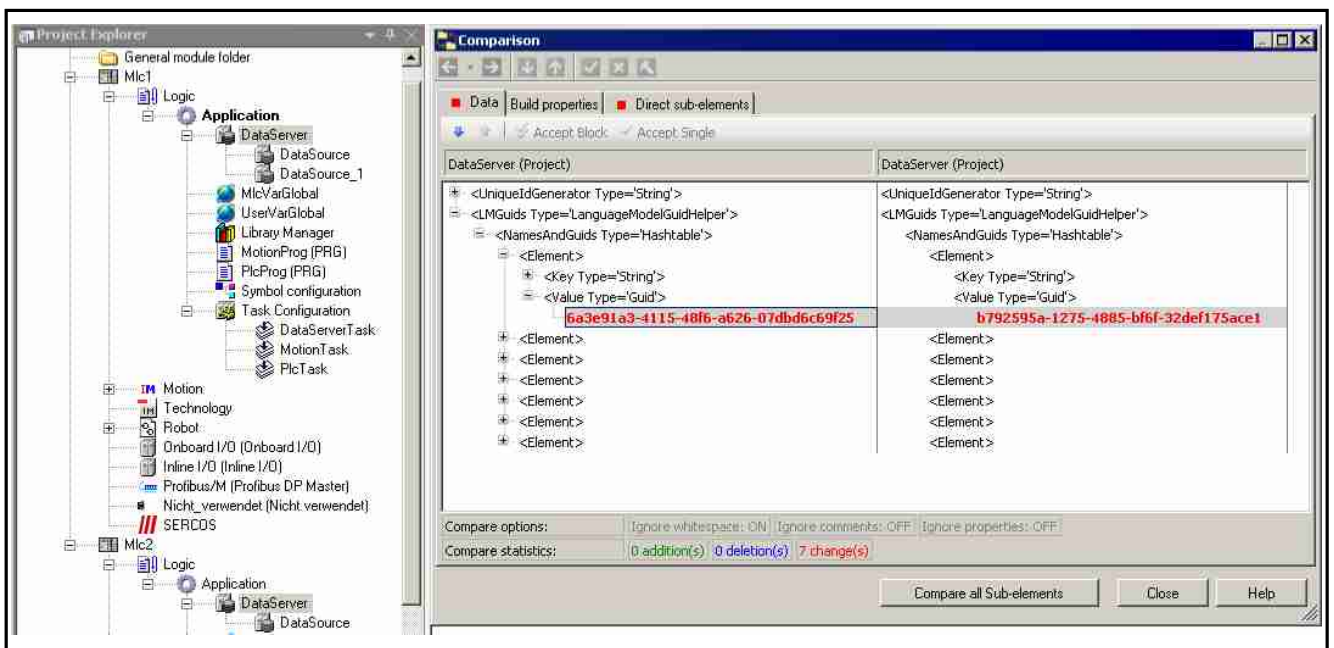


Fig.3-110: Comparison: Data server

The next/previous difference is selected with or .

The next tab shows the build properties.

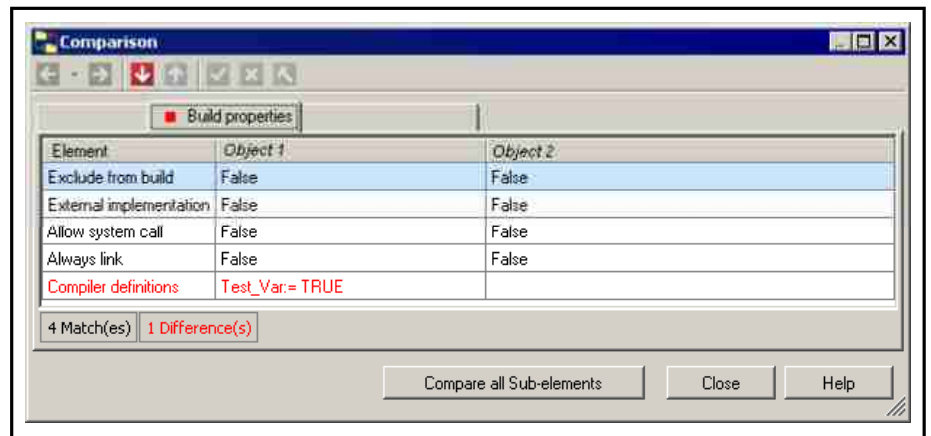


Fig.3-111: Comparison: Build properties object 1/object 2, one difference identified and marked in red

Menu Items

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

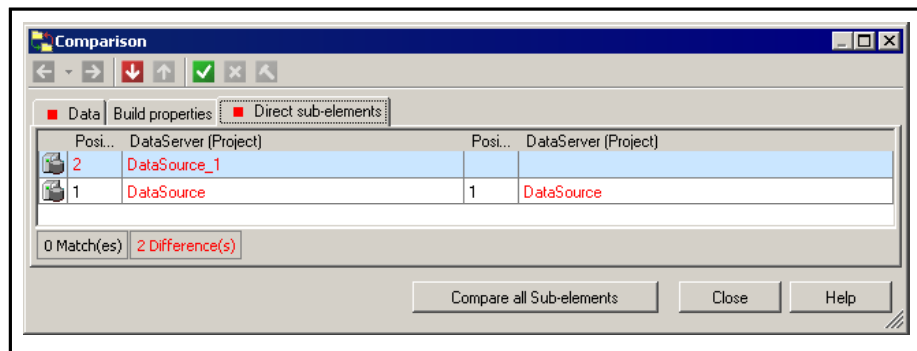


Fig.3-112: Comparison: Data server, Direct subelements

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

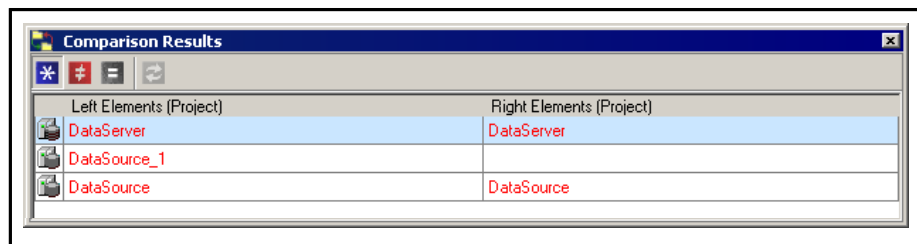


Fig.3-113: Comparison results: Data server

Comparison: Data Source (Data Server)

When comparing data sources, the content display is XML based. Modification (merging) is not possible.

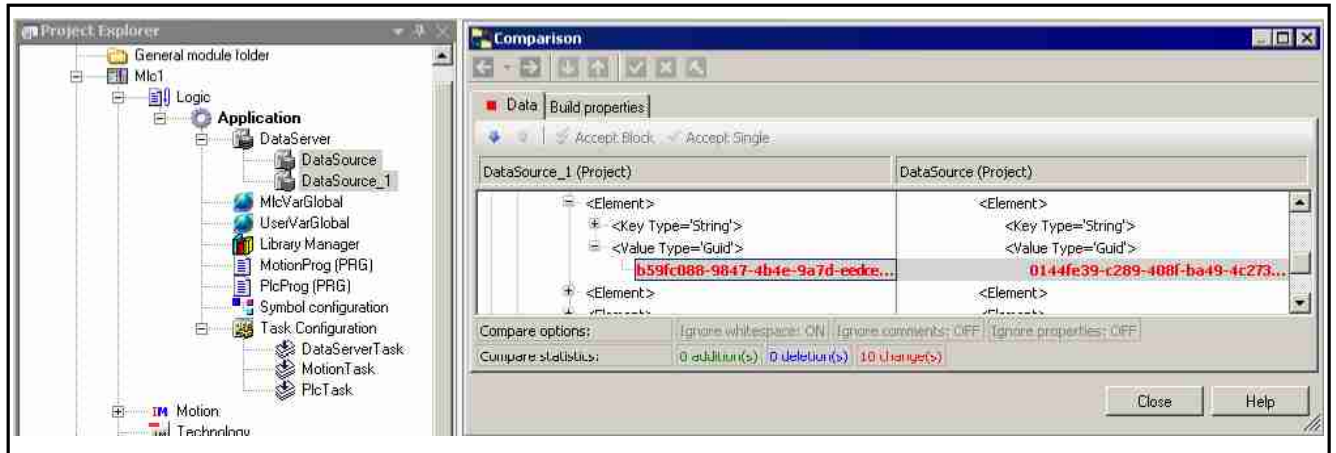


Fig.3-114: Comparison: DataSources

The next/previous difference is selected with or .

The next tab shows the build properties.

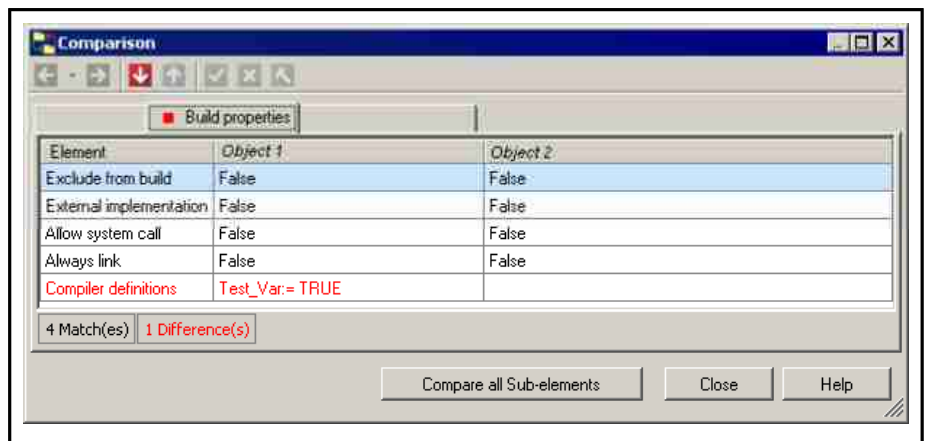


Fig.3-115: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: Symbol Configuration

When comparing symbol configurations, the content display is XML based. Modification (merging) is not possible.

Comparison: Visualization Manager

When comparing visualization managers, the content display is XML based. Modification (merging) is not possible.

The next tab shows the build properties.

Menu Items

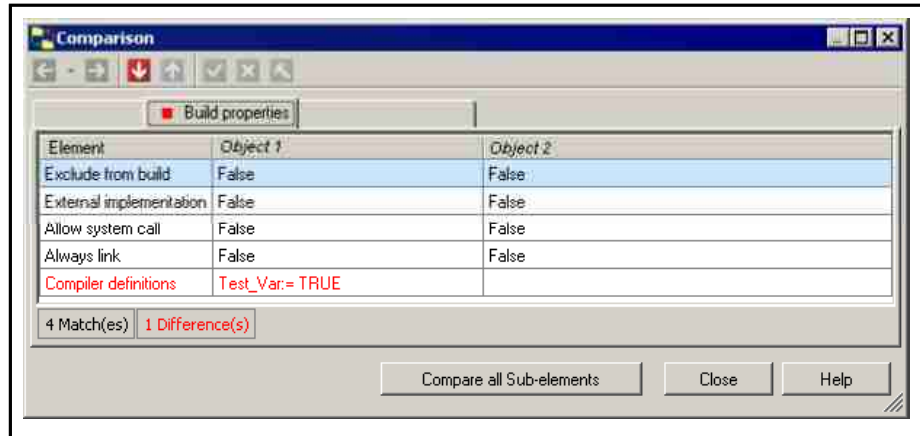


Fig.3-116: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: Visualization

When comparing visualizations, the content display is XML based. Modification (merging) is not possible.

The next tab shows the build properties.

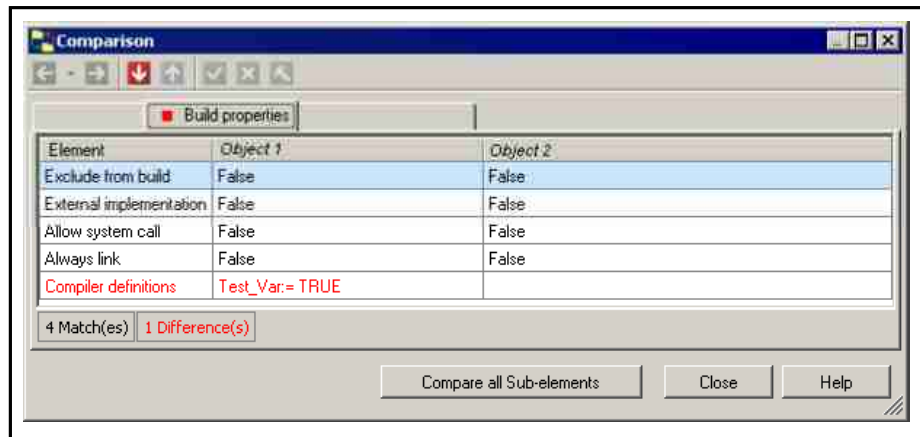



Fig.3-117: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer  to merge.

Select the cross  to prevent (undo) merging.

Comparison: Text List

When comparing text lists, the content display is XML based. Modification (merging) is not possible.

The next tab shows the build properties.

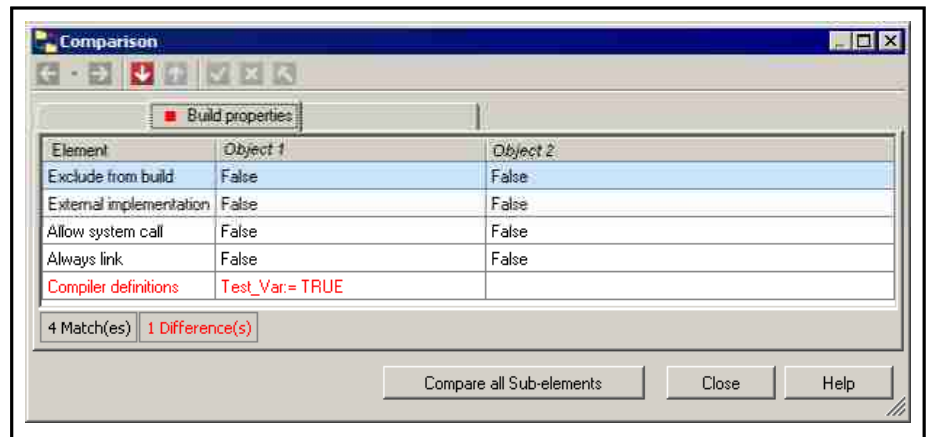






Fig.3-118: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs,  can be used to select the next difference ( for the previous difference).

If the checkmark  appears, it can be used to prepare the acceptance (merging):

Select the hammer  to merge.

Select the cross  to prevent (undo) merging.

Comparison: Image Pool

When comparing image pools, the content display is XML based. Modification (merging) is not possible.

Comparison: Recipe Manager

When comparing recipe managers, the content display is XML based. Modification (merging) is not possible.

Menu Items

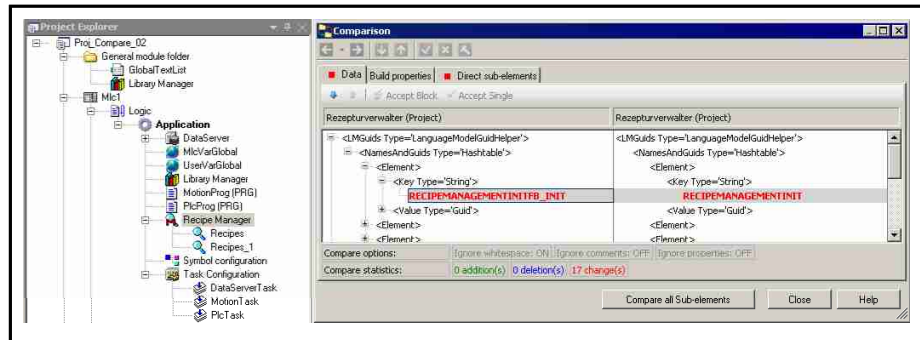


Fig.3-119: Comparison: Recipe Manager

The next/previous difference is selected with or .

The next tab shows the build properties.

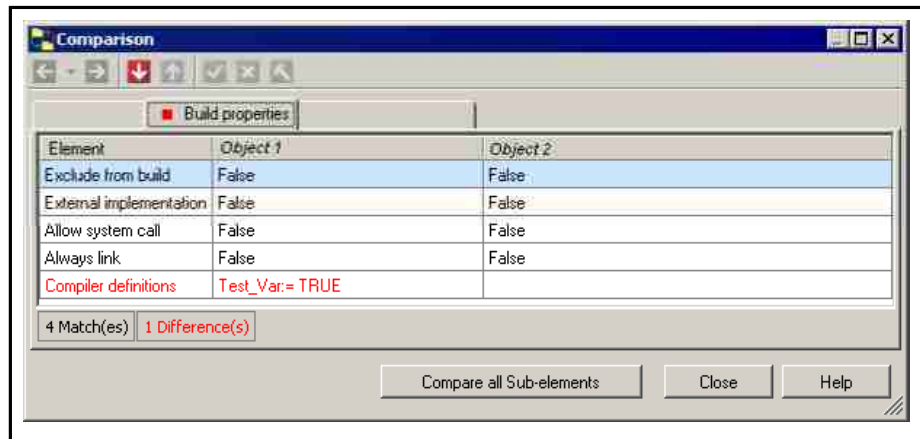


Fig.3-120: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

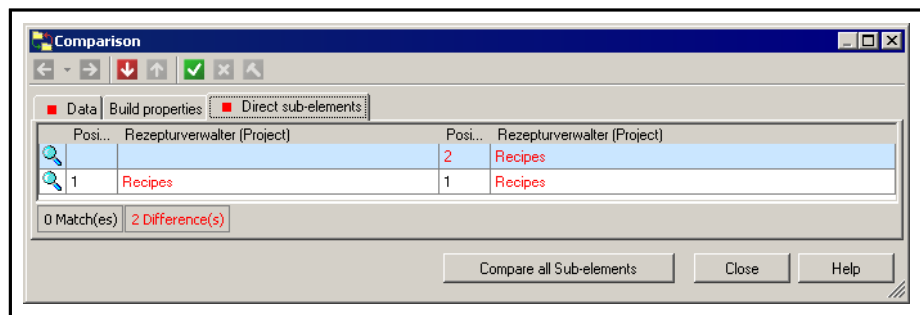


Fig.3-121: Comparison: Recipe manager, Direct subelements

If the assignment in the columns of the two objects differs, Menu Items can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

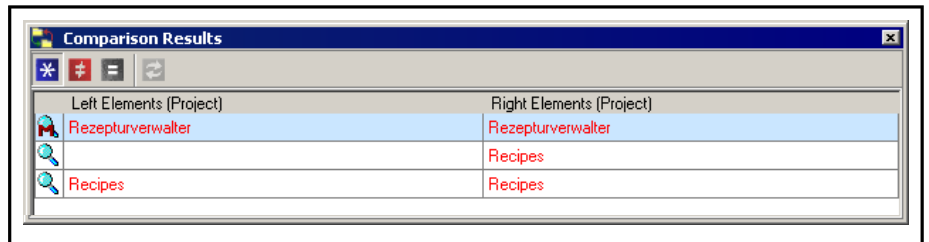


Fig.3-122: Comparison results: Recipe Manager

Comparison: Recipe Definition

When comparing recipe managers, the content display is XML based. Modification (merging) is not possible.

The next tab shows the build properties.

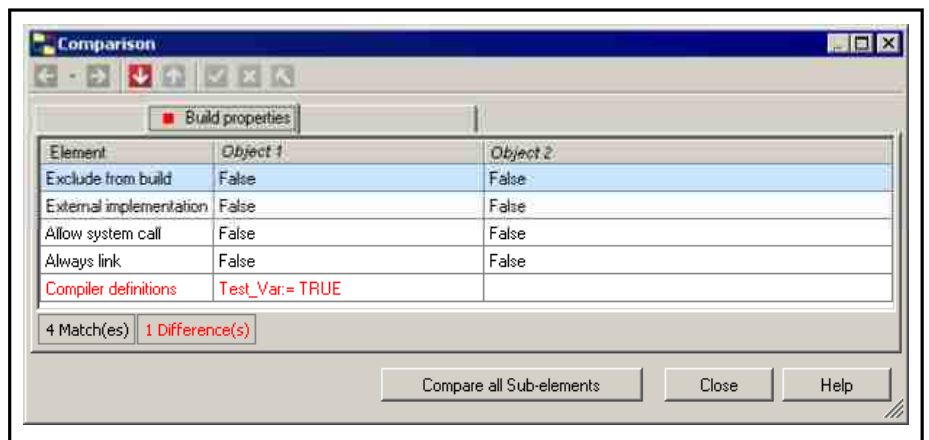


Fig.3-123: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

Comparison: Devices (Onboard I/O, Inline I/O, Field Busses, ...)

The comparison applies to the devices that are associated with the respective control (onboard I/O, inline I/O, field busses). Since these differ from con-

Menu Items

ontrol type to control type, the following comparison results have to be seen as examples.

In principle, it is tried to provide a familiar environment at comparison with regard to the input and to enable merging based on the comparison.

Onboard I/O:

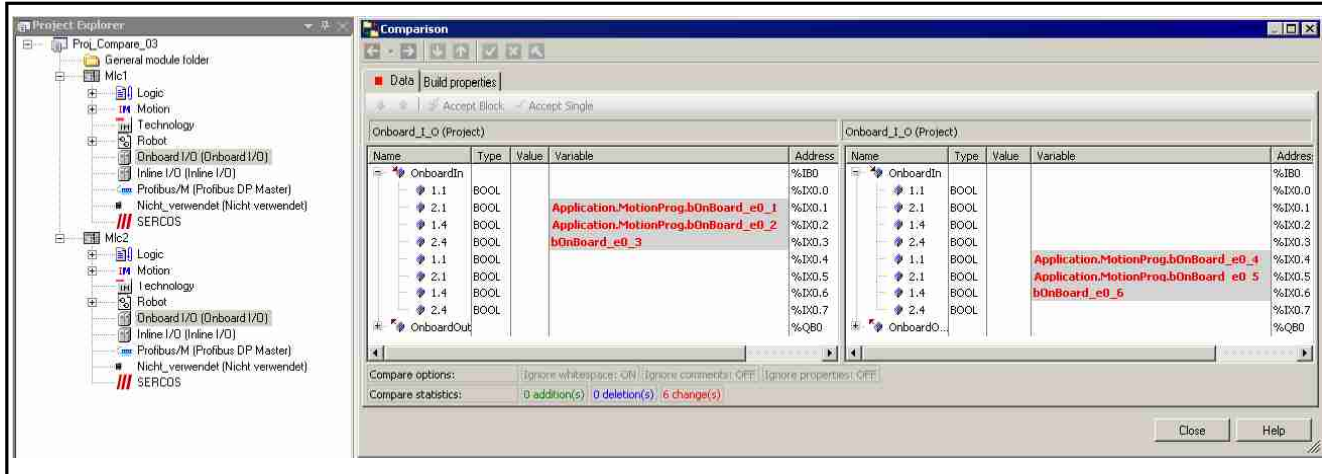


Fig.3-124: Comparison: Onboard I/O:

The next/previous difference is selected with or .

Accept Block can be used to accept the complete highlighted text block.

Accept Single can be used to accept text line by line. In this case, only the possible result of the merge is displayed first. Clicking the button again resets the display.

Select the hammer to merge.

Select the cross to prevent (undo) merging.

The next tab shows the build properties.

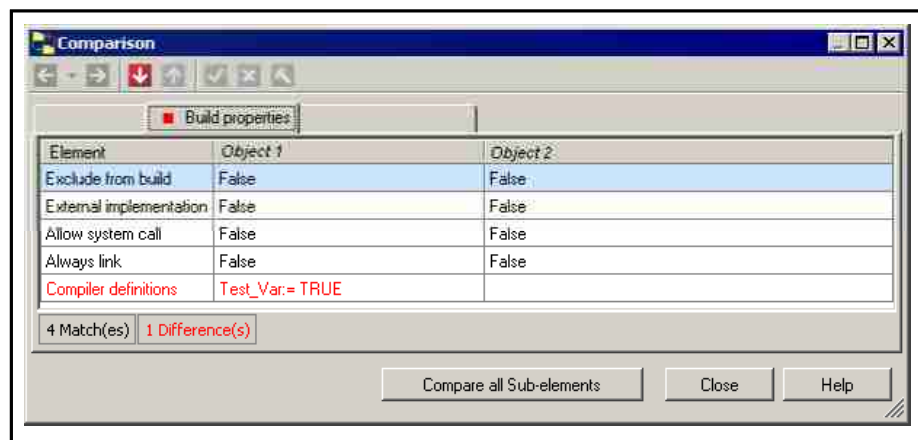




Fig.3-125: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

Menu Items

If the checkmark  appears, it can be used to prepare the acceptance (merging):

Select the hammer  to merge.

Select the cross  to prevent (undo) merging.

Inline I/O objects: The comparison shows the Inline cycle counters and diagnostic data tabs belonging to the Inline I/O objects. Merging is not intended.

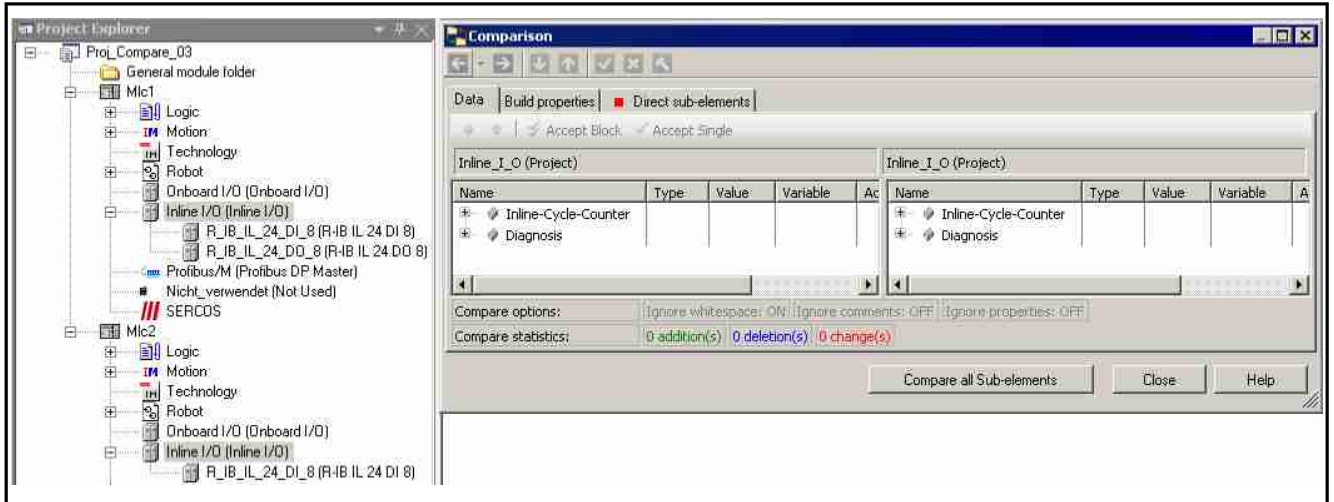


Fig.3-126: Comparison: Inline I/O: Inline cycle counters and diagnostic data tabs
The next tab shows the build properties.

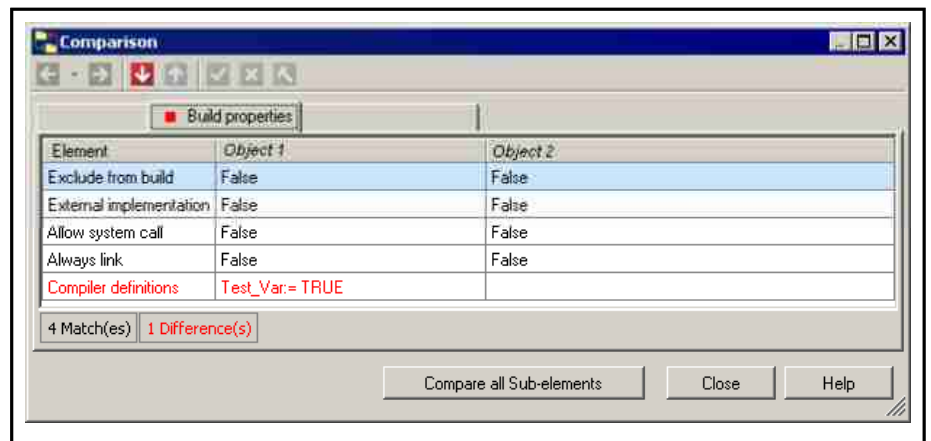






Fig.3-127: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs,  can be used to select the next difference ( for the previous difference).

If the checkmark  appears, it can be used to prepare the acceptance (merging):

Select the hammer  to merge.

Select the cross  to prevent (undo) merging.

Menu Items

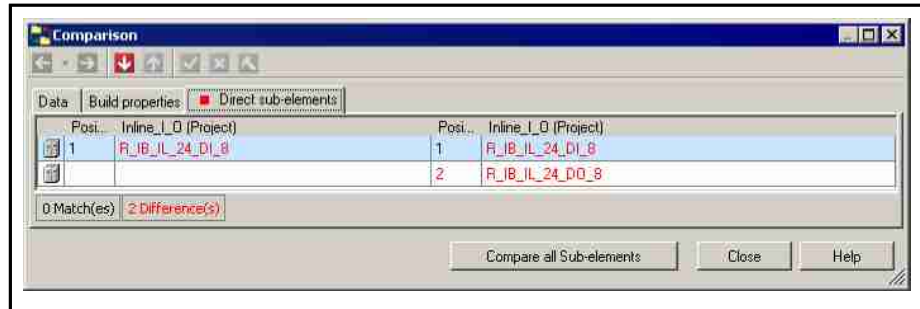


Fig.3-128: Comparison: Inline I/O: Direct subelements

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

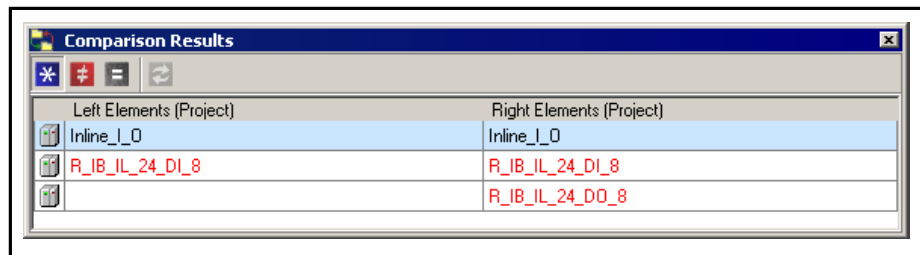


Fig.3-129: Comparison results: Inline I/O: All subelements

Inline I/O modules

If the Inline I/O modules under "Direct subelements" and "Comparison results" have identical names and are marked in red, this is an indication of different assignments:

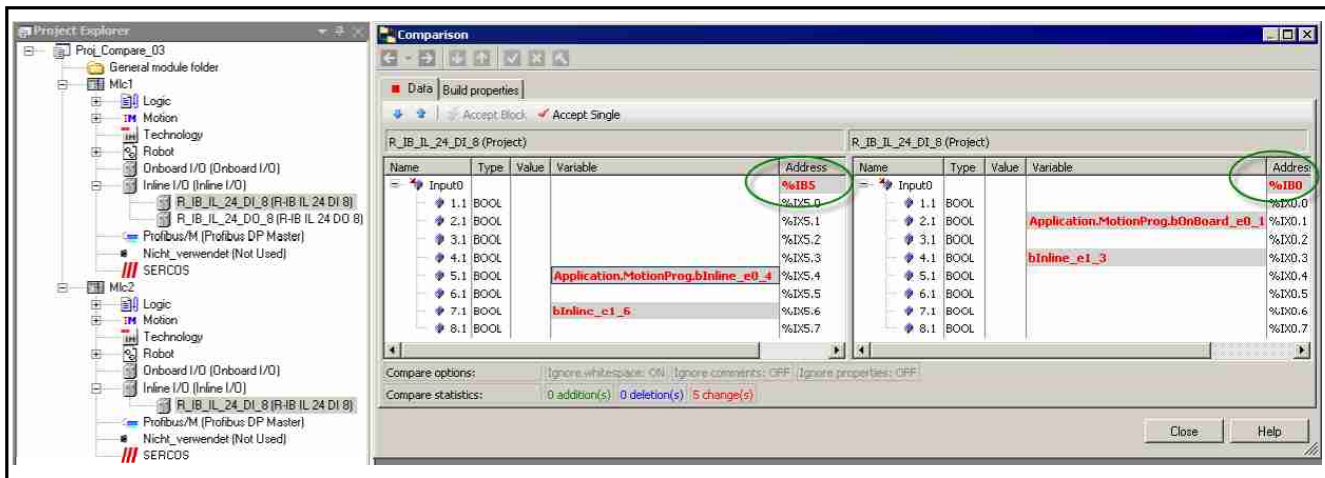


Fig.3-130: Comparison: Inline I/O modules

The different addresses with identical location of the modules in the Project Explorer indicate an address shift.

The next/previous difference is selected with or .

Menu Items

Accept Block can be used to accept the complete highlighted text block.

Accept Single can be used to accept text line by line. In this case, only the possible result of the merge is displayed first. Clicking the button again resets the display.

Select the hammer to merge.

Select the cross to prevent (undo) merging.

The next tab shows the build properties.

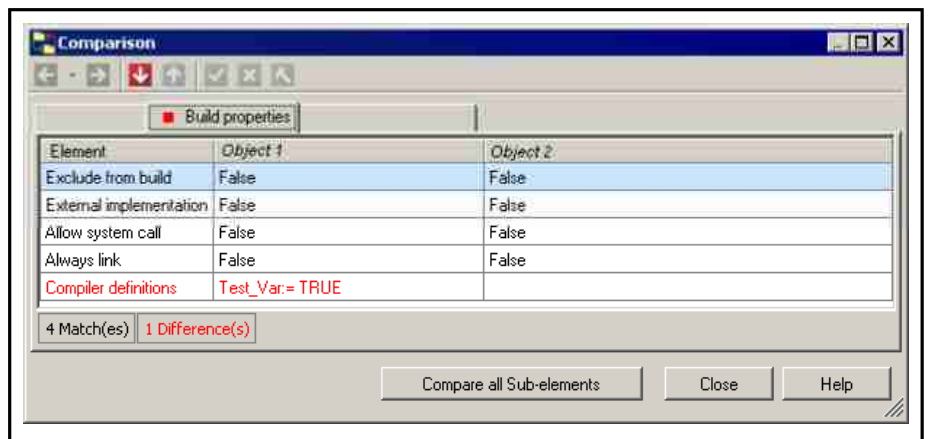


Fig.3-131: Comparison: Build properties object 1/object 2, one difference identified and marked in red

If the assignment in the columns of the two objects differs, can be used to select the next difference (for the previous difference).

If the checkmark appears, it can be used to prepare the acceptance (merging):

Select the hammer to merge.

Select the cross to prevent (undo) merging.

3.6.2 Simulation

The command is available in the context menu of the relevant control and is used to switch on and off the simulation mode of the programming system.

In the simulation mode, the active application can be started and debugged on a "simulated target device". As such a simulated device is always integrated into the programming system, no real target device is needed to test the online behavior of an application. If the command is called from the context menu if an application is selected in the Project Explorer, login is carried out with this selected application even if it is not set as "active application".

Menu Items

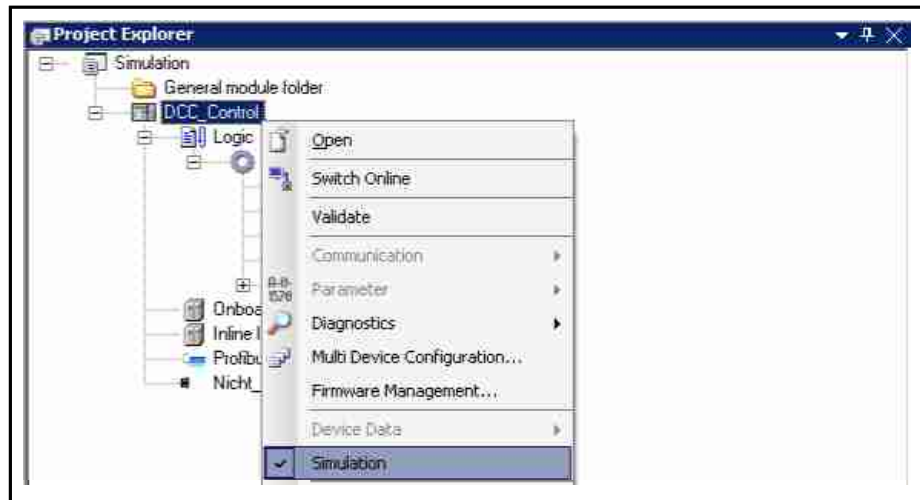


Fig.3-132: Context menu, simulation switch-on

No communication settings need to be entered for the simulated device.

Following a successful login, the corresponding online commands can be used to test the application.

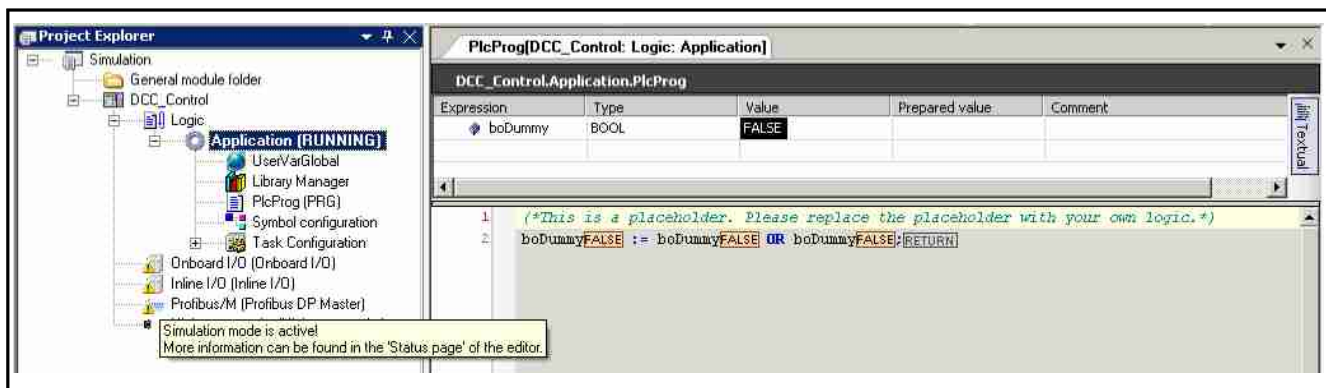


Fig.3-133: Control in the simulation mode

To switch off the simulation mode, first log off from the control and then call the "Simulation" command again. The tick in front of the command disappears, the entry of the target device in the Project Explorer is no longer shown in italics and it can be thus logged into the real target device.

3.6.3 IndraLogic Project Versions

From IndraWorks 12VRS, a PLC project contains a project version. Using this version code, compatible PLC projects can be generated for all known releases (V02, V04, V06, ...) of an IndraWorks version.

If an older project version is identified while a project is opened, the following dialog appears:



Fig.3-134: Dialog: Project in old project version detected

Menu Items

Using the dialog, an accidental conversion of the PLC project can be prevented.

If the project is available in a 12VRS release version, it can be edited in all later IndraWorks releases of this version without conversion and saved in a release-compatible form. The project can be explicitly saved in a later release version. For this purpose, the "IndraLogic project version" command is used which is by default executed under **Tools ▶ IndraLogic project version** :

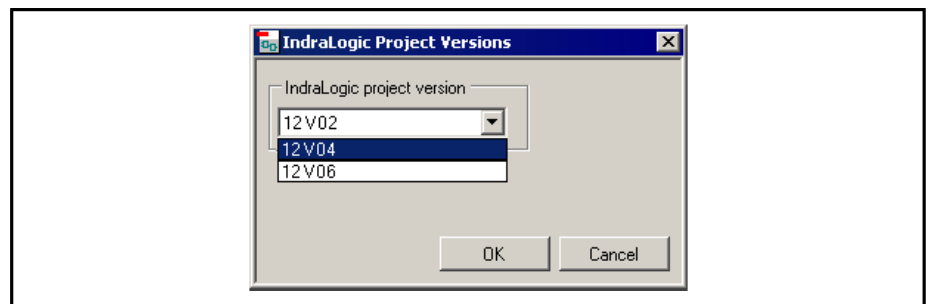


Fig.3-135: Dialog: IndraLogic project version

Using the dialog, the project can be converted in a later release version. Conversion into an older format is not possible as here, data would be lost.

A newly created project is automatically always generated in the latest release version. As long as no device has been inserted into this project, the project can be converted to an older project release using the <IndraLogic Project Version> command described above.



By using new features of the current Indraworks version that are not available in the project version, the project version can be updated automatically.

These automatic changes can be corrected using the "Undo" command.

3.6.4 Library Repository - Commands

Library Repository - Commands, General

"Library management" provides access to the "[Library Repository ...](#)" command for opening the dialog with the same name.

By default, the command is located in the "Tools" menu. If required, the menu structure can be changed using the dialog in **IndraWorks ▶ Tools ▶ Customize ▶ Commands ▶ Library Manager**.

Further information

- [Using libraries, see page 83.](#)

Library Repository...

Icon: 

This command opens the **Library Repository** dialog, which can also be opened from the [Library manager editor window, page 367,](#).

A library repository is a storage location for libraries that were installed on the local system in order to be included in IndraLogic projects.

Menu Items



A library project "*.library" located in a library repository cannot be opened from the repository for editing or for viewing in the programming system.

To achieve this, the library has to be added to the library manager.

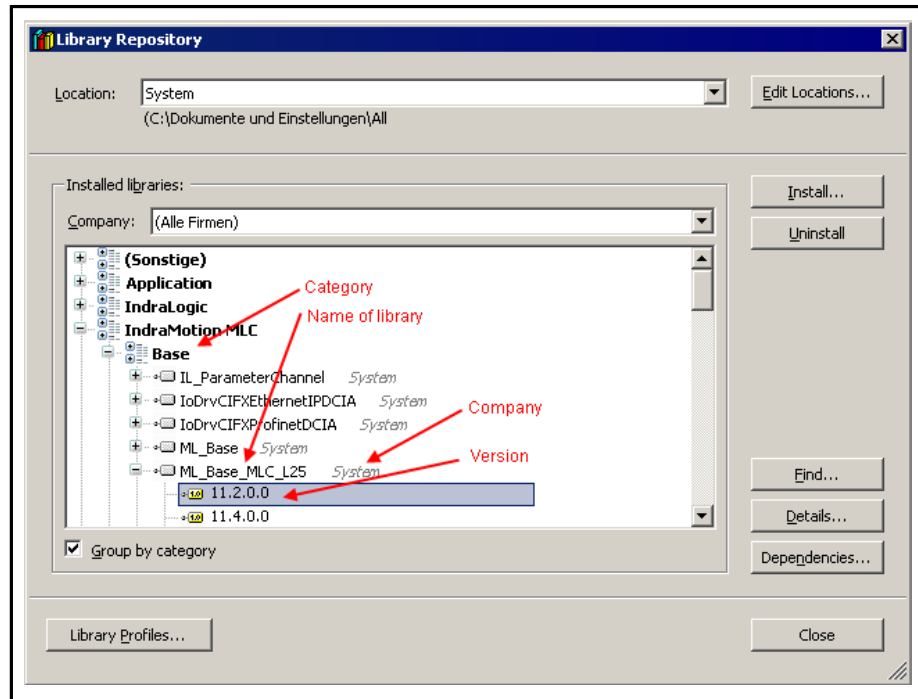


Fig.3-136: Main dialog "Library Repository"

The dialog displays the currently defined storage locations (repositories) and the libraries installed there.

This dialog can be used to add new repositories, to modify or remove them and libraries can be installed or uninstalled here.

A list of all of the currently **installed libraries** for the currently selected **storage location**, a directory on the local system in which the library files are located, and for the **company** that is currently set, is displayed. This list displays the respective name (title), version and company for a library as entered in the library information in the library file.

If the **Group according to category** option is selected, the list is grouped according to library categories. The category names appear as nodes that can be expanded or reduced via click. The respective libraries appear below the categories and the respective versions appear below the libraries. If grouping is not performed by category, the libraries are listed alphabetically according to company.

The categories are defined by external description files (*.libcat.xml).

The individual buttons are described in the following:

- **Edit:** [Library Repositories, page 187](#)
- **Install, Uninstall:** [Installing and uninstalling libraries, page 188](#)
- **Details:** [Further information on libraries, page 189,](#)
- **Dependencies:** [Display of the libraries used, page 189,](#)
- **Library profiles:** [Display of the library profiles, page 190,](#) ordered according to placeholder name or compiler version.

Menu Items

Library repositories One or more repositories can be used for managing libraries. All of the currently defined repositories are in the selection list next to **Storage location**: By default the "System" repository is always created during the installation of IndraLogic.

To change the path or name of a repository, use the **Edit...** buttons to open the **Edit repositories** dialog:

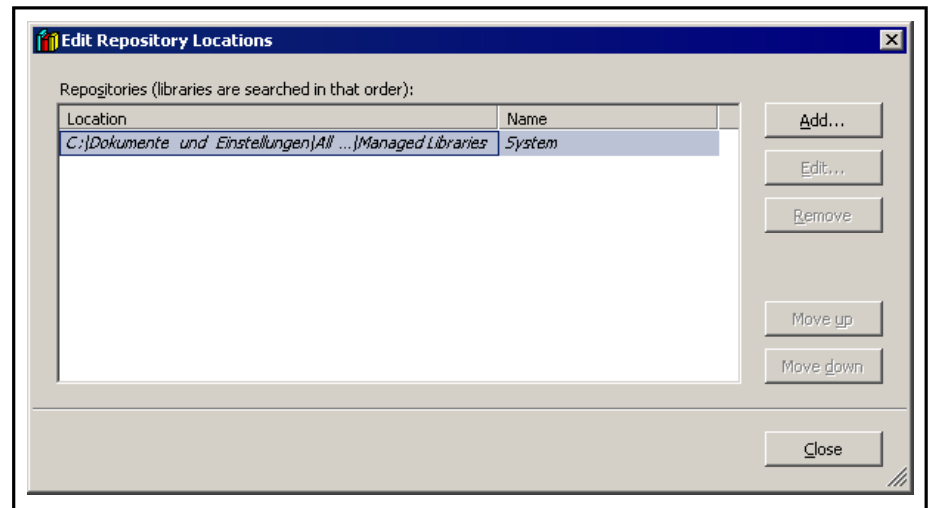


Fig.3-137: "Edit Repository" dialog

The currently defined storage locations are listed in the repository window. Later, a search for a library is made in the sequence listed here. To change the sequence, use the **Move up** and **Move down** buttons to move the currently selected entry.

Note that the storage location **<All companies>** lists the libraries in all of the currently defined repositories. No installation or uninstallation can be performed in this display.

Creating a new repository To create a new repository, use the **Add** button. The **Repository Location** dialog opens, where the path of the new repository has to be entered in the **Location** field. Use the **...** button to search for a suitable directory or create one.



Attention: The selected directory **has to be empty!**

In the **Name** field enter a representative name for the repository, e.g. Libraries for system xy.

To edit an existing repository, select the entry in the Repositories dialog and use the **Edit** button to open the **Repository Locations** dialog, where in this case, the current information for location and name are already entered and can be modified.

Menu Items



Fig.3-138: Set storage location and name for repository dialog



1. Only empty directories can be used for a new repository.
2. The "System" repository cannot be edited, which is indicated by the italic font used for this entry.

Deleting a repository

If a repository in the list is selected and <Remove> is clicked, a query appears asking if only the entry is to be removed from the list of repositories or if the directory with the library files is also to be deleted from the file system.

Installing and uninstalling libraries

Only one library that is installed on the local system in a library repository can be included in a project using the library manager.

A prerequisite for the installation is that the library file has to contain a title and version information in its [library information, page 371](#); information on category and company is optional.

To install a library, select the repository in the "Library Repository" dialog to which the library is to be added and click the <Install...> button.

The "Select Library" dialog opens, where it can be searched for the library file.

- By default, the filter is set for "libraries" that contain the appropriate library information and have the default extension *.library.
- The filter can also be set for "compiled libraries" that contain the appropriate library information and have the default extension *.compiled-library.
- The filter can also be set to "IndraLogic libraries (before 2.x)" to find libraries with old formats and with the extension *.lib or to
- "All files".

Select the desired library file and close the dialog. The library is added to the list of installed libraries in the library repository dialog.

If you try to install a library that cannot be installed, since it does not have the required library information (title, version), the following error message appears:

The library entered is not a managed library. (Reason: No library information could be found.)

To uninstall a library, select it from the list of installed libraries in the library repository dialog and press <Uninstall>.

Search in library

Dialog to search libraries in the specified storage location. This search is used to find function blocks and the related libraries. Apart from the search string for the function block, the placeholders "*" and "?" can be used.

Menu Items

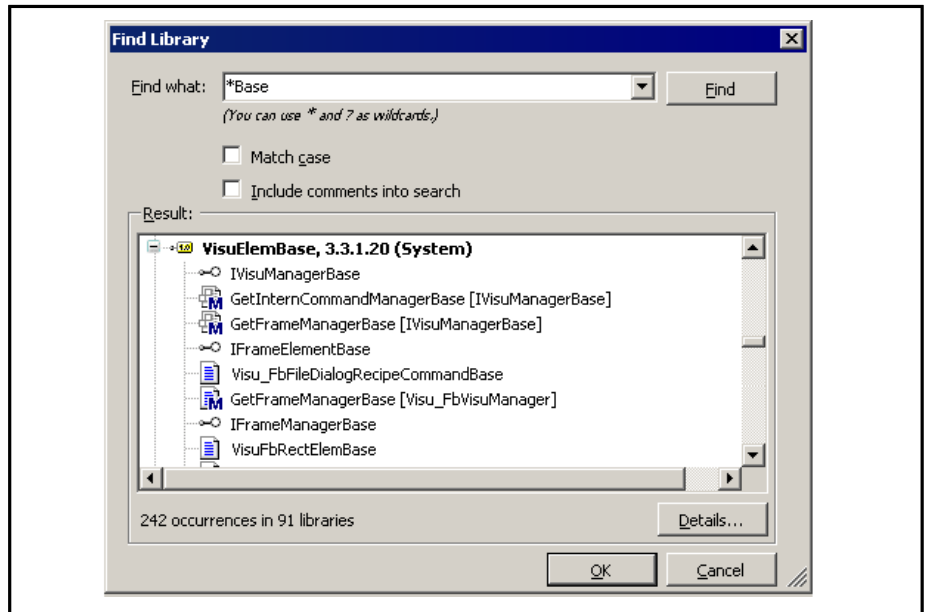


Fig.3-139: "Find library" dialog

Pay attention to upper and lower cases: This search takes the case of the search string into account

Integrating comments into the search: Not only the name of the function block but also the comment is searched

Details...

When a library is selected from the list of installed libraries - ensure that the line selected contains the version - it button opens a dialog that displays the following details: Title, version, company, size, created (date), changed (date), edited (date of most recent access), properties.

This information originates from the [library information, page 371](#), in the library file.

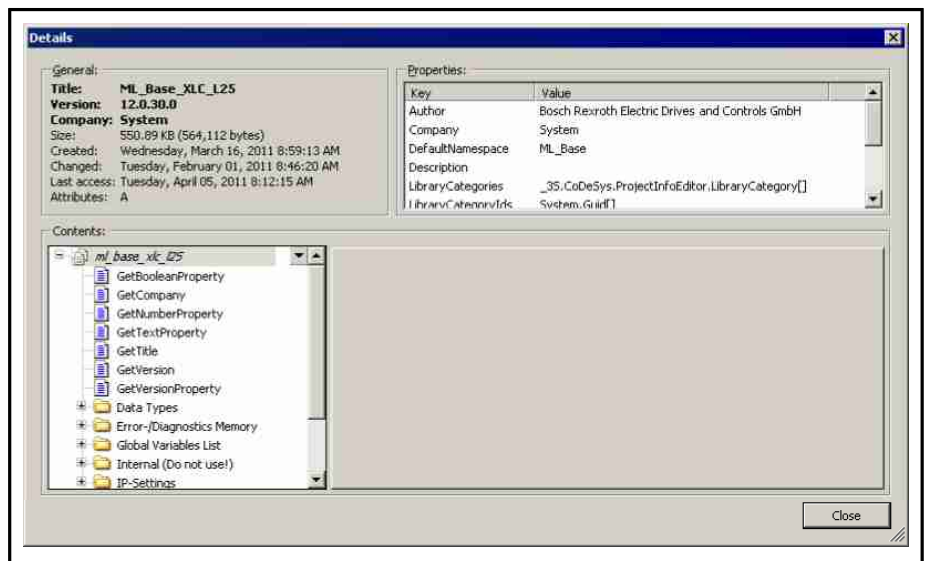


Fig.3-140: "Details" dialog

Dependencies...

For the currently selected library, this button opens a dialog that displays the dependencies of other libraries, i.e. the integrated libraries are shown.

The title, version and company are displayed for each library.

Menu Items

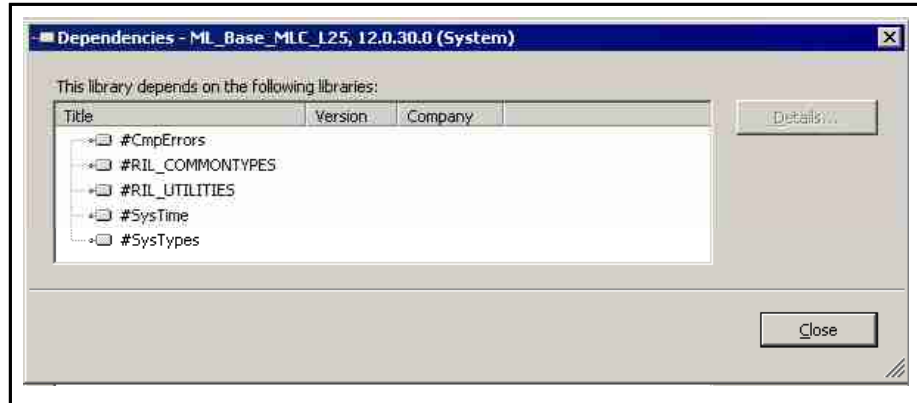


Fig.3-141: "Dependencies" dialog

Library profiles

For the current library repository, the placeholders/compiler versions are displayed.

The list can be ordered according to placeholders or compiler versions. The following two figures show the possibilities using an example.

The dialog simultaneously serves as export/import interface for library profiles.

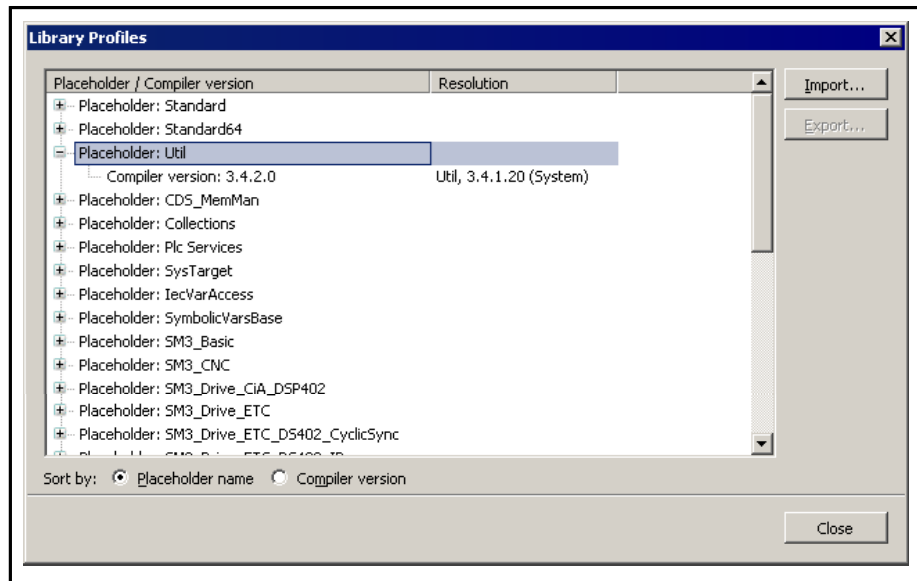


Fig.3-142: "Library Profiles" dialog, ordered according to placeholder names

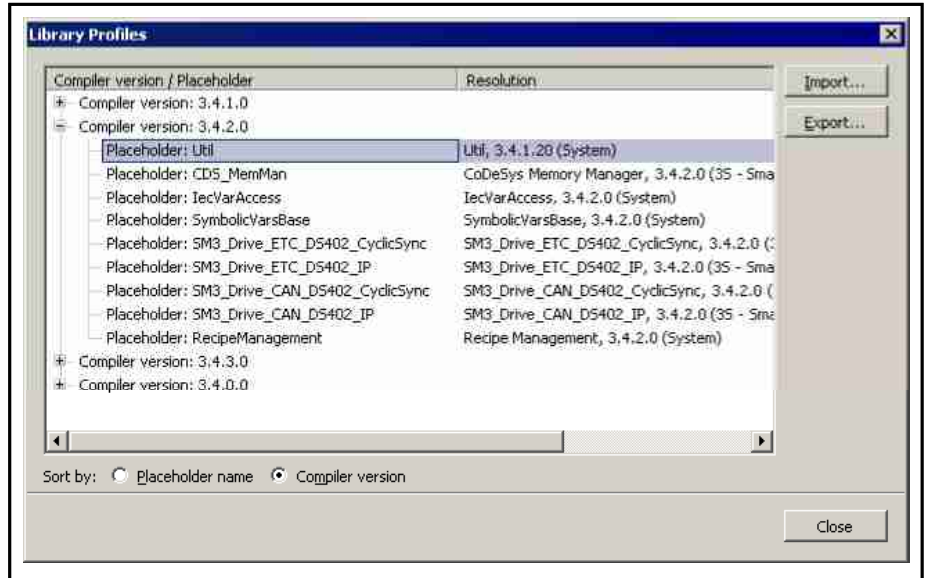


Fig.3-143: "Library Profiles" dialog, ordered according to compiler versions used

3.6.5 Generating a Library in IndraWorks

The libraries of the IndraLogic PLC programming system contain function blocks, functions, data types, variable lists and constants provided by the control manufacturer. The libraries, page 83, are either directly installed during the installation of "IndraWorks Engineering" and are visible in the library manager, page 367, of the relevant application or they can be added from the library repository, page 185,.

This section describes how the user can create a library or compiled library from function blocks, functions, data types, etc. which is equal to the manufacturer's libraries.

For that purpose, the function blocks, functions, data types, etc. are to be arranged in the "General module" folder in a structured form.

In the context menu of the "General module" or in the main menu under "Tools", three menu items are available

- "Generate library..."
- "Generate compiled library..."
- "Generate and install compiled library..."



Fig.3-144: Menu items for generating a library or compiled library (context menu)

Menu Items

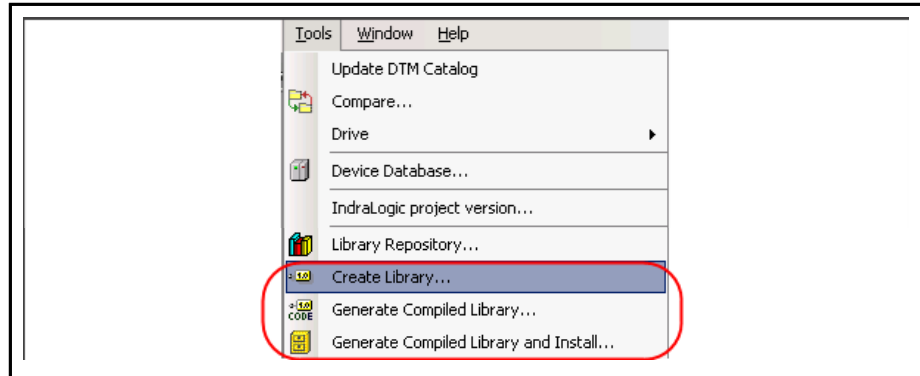


Fig.3-145: Menu items for generating a library or compiled library (main menu)

All menu items can only be used if the opened project contains a "Library info" object, page 371,. If no such object is available and the user selects one of these menu items, the following message is output:

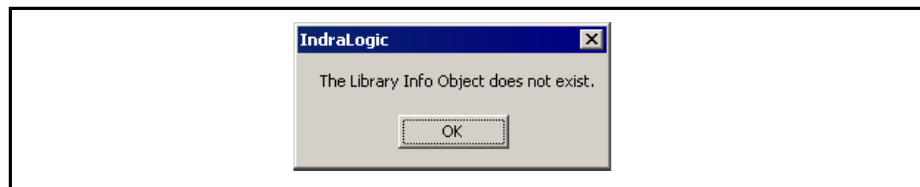


Fig.3-146: Error message, library info object not available (yet)

In this case, add such object by means of the context menu (or from the library):

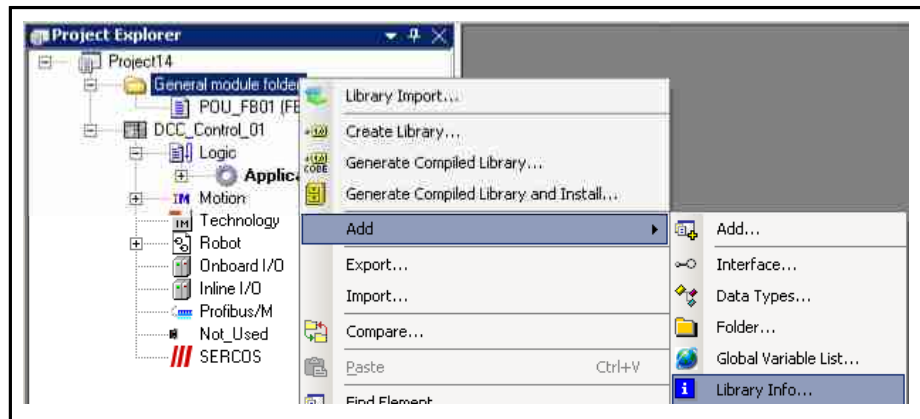


Fig.3-147: Adding a library info object to the planned library

In this object, at least complete the mandatory fields "Company", "Title" and "Version".

Menu Items

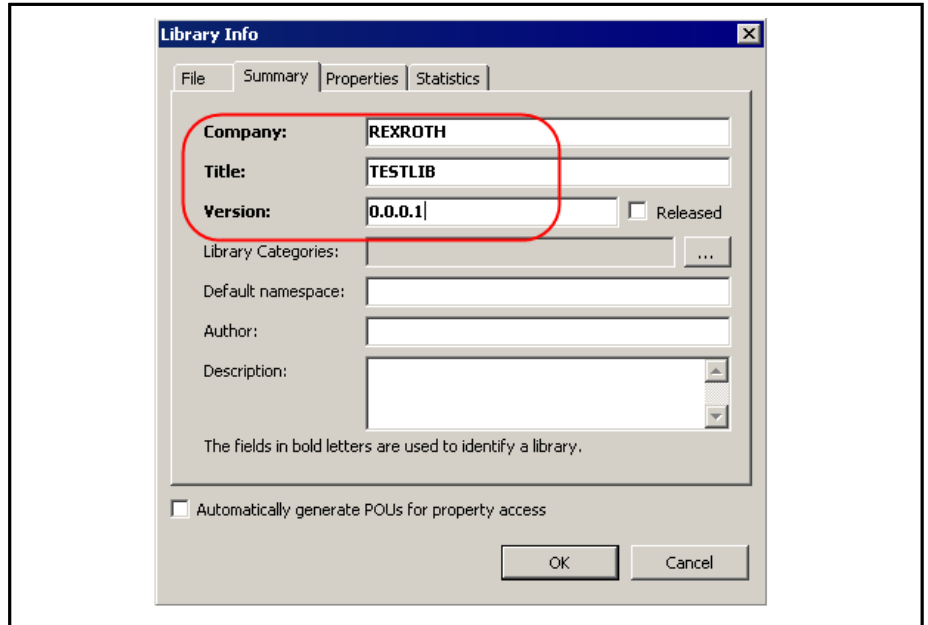


Fig.3-148: Dialog: Library info object

If this was not performed or not performed correctly, the following message is output.



Fig.3-149: Message: Mandatory parts of the library info object are missing



This mandatory data is later used by the library in the library repository.

The name of a library or compiled library is generally displayed in the details of the library repository (normally, title and name should match):



Menu Items

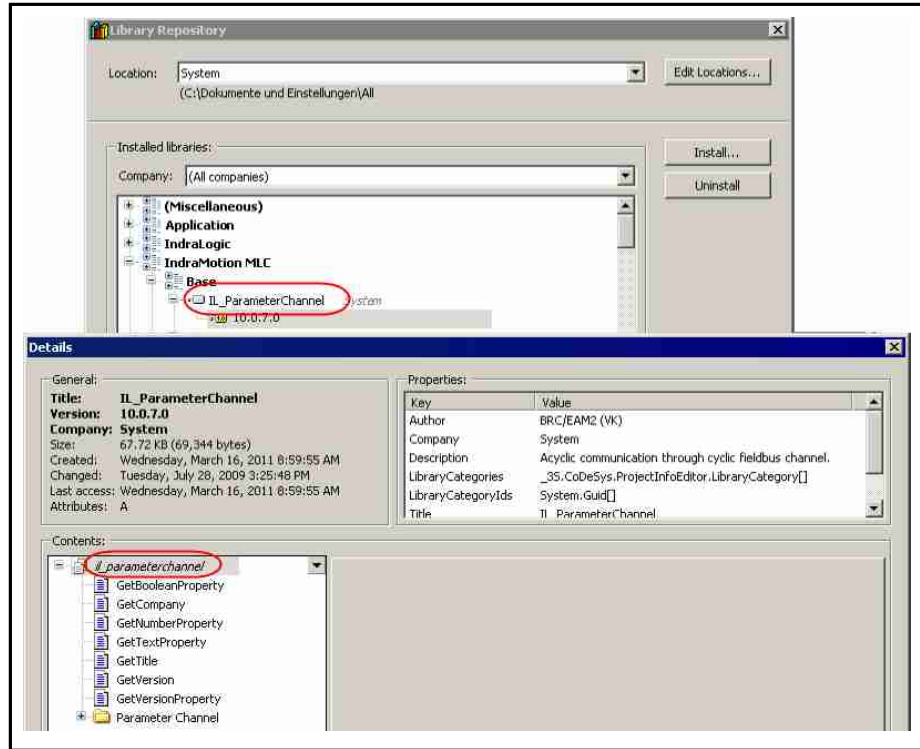


Fig.3-150: Library in the library repository

From now, the process and the result of the three menu items differ:

Generate library...

The user first of all selects a directory (and can change the name of the library file) in which the library is to be stored. It has to differ from the IndraLogic directory of the currently opened project. If this is not the case, the user is requested to select a different directory.

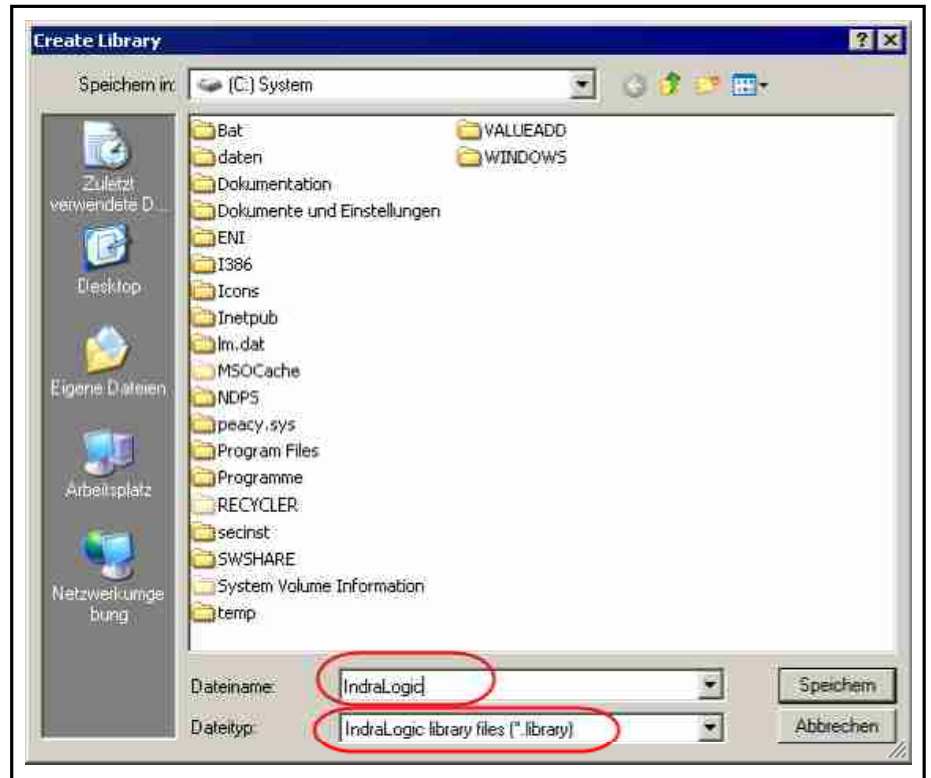


Fig.3-151: Dialog: Selection of the storage location and the name of the library to be generated

After the user has confirmed this dialog, the opened project is first of all saved and then, the library file is generated in the selected directory (in the preceding example in the path "C:\\" the file "indralogic.library").



The "storage" name of the library only serves the finding and management outside the library repository. It does not influence the title in the "Library Info".

Generate compiled library...

The user first of all selects a directory (and can change the name of the library file) in which the compiled library is to be stored. It has to differ from the "IndraLogic" directory of the currently opened project. If this is not the case, the user is requested to select a different directory.

Menu Items

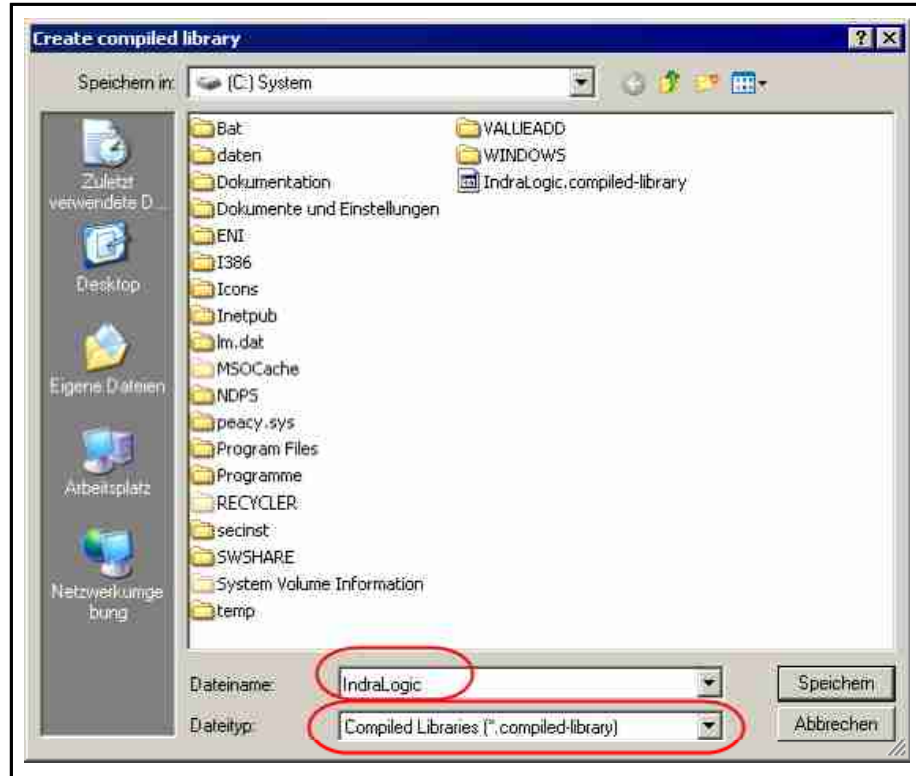


Fig.3-152: Dialog: Selection of the storage location and the name of the "compiled library" to be generated

After the user has confirmed this dialog, the opened project is first of all saved and then, the file with the compiled library is generated in the selected directory (in the preceding example in the path "C:" the file "indralogic.compiled-library").



The "storage" name of the "compiled library" only serves the finding and management outside the library repository. It does not influence the title in the "Library Info".

Generate and install compiled library...

If the user selects this menu item, the "indralogic.compiled-library" file is generated in the "Indralogic" directory of the opened project and immediately installed in the library repository.

The data is taken from the mandatory fields "Company", "Title" and "Version" of the library information.

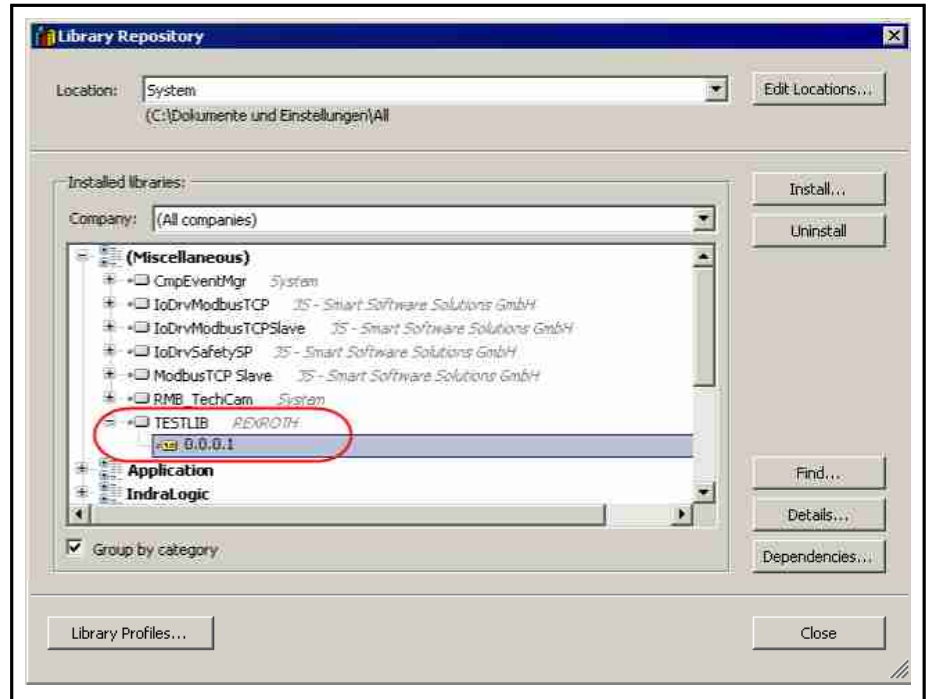


Fig.3-153: Dialog: "Library Repository", new library TESTLIB

3.6.6 Options Options, General Information

The "Options" dialogs allow to configure the behavior and appearance of the IndraLogic user interface. The default settings are determined by the current profile.

The dialogs can be accessed in **Tools ► Options**.

The current settings in the options dialog are saved on the local system as default settings and they overwrite the settings made when installing IndraLogic as delivered.

IndraLogic 2 G options

- [Options, general settings, page 198](#)
- [Export options, page 198](#)
- [Visualization options, page 199](#)
- [Options, declaration editor, page 201,](#)
- [Options, sequential function chart, page 202,](#)
- [Options, smart coding, page 205,](#)
- [Options, CFC editor, page 207](#)
- [Options, FBD, LD and IL editors, page 207](#)
- [Options, device editor, page 211](#)
- [Options, text editor, page 212](#)
- [Options, syntax highlighting, page 219,](#)
- [Options, sequential function chart editor, page 219,](#)
- [Options, libraries, page 222,](#)
- [Options, converter for IndraLogic 1.x projects, page 224](#)

Menu Items

Options, IndraLogic 1.x

- [Options, IndraLogic 1.x settings, page 225](#)

Options, General Settings

Menu: **Tools** ▶ **Options** ▶ **IndraLogic 2G** ▶ **General settings.**

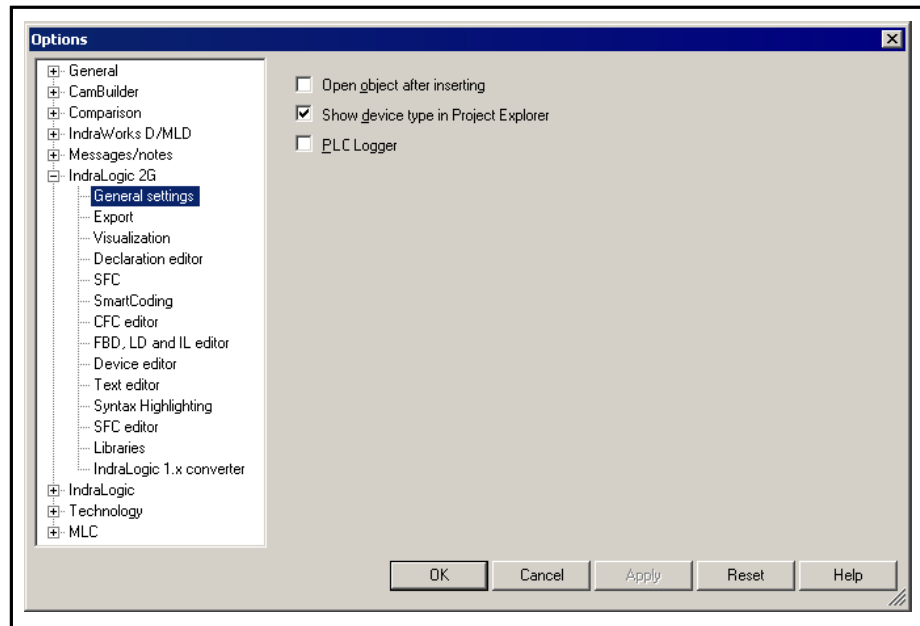


Fig.3-154: "General settings" option

- **Open document when inserting:**
When this option is selected, new objects that are inserted into the Project Explorer are opened immediately.
- **Show device type in the Project Explorer**
The name of the object is extended with its device type.
- **Activate PLC logger**
Using this button, the "Log" tab of the device dialog of every control can be activated.

Options, Export

Menu: **Tools** ▶ **Options** ▶ **IndraLogic 2G** ▶ **Export.**

The format of the export files generated by IndraWorks can be set on this option page:

Menu Items

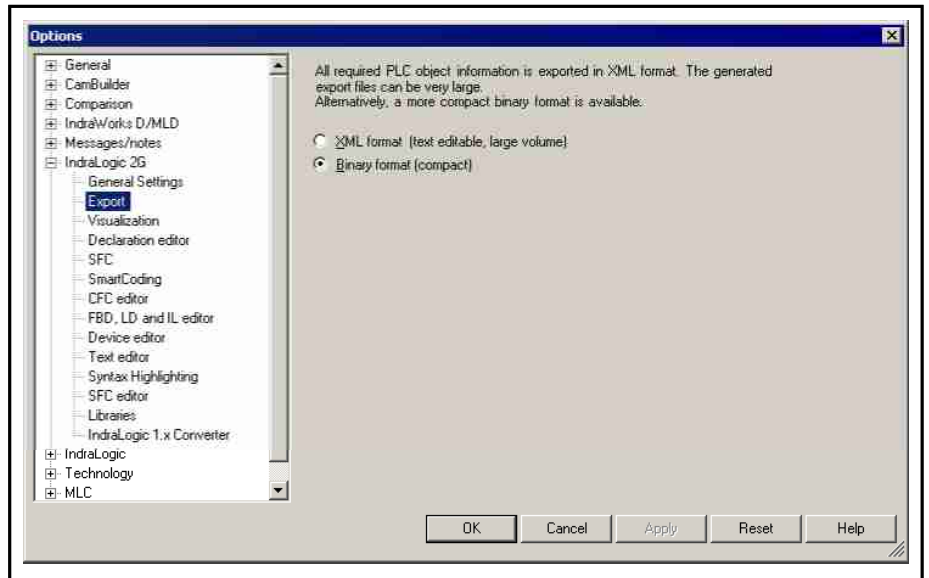


Fig.3-155: Options, "Export"

Export format *Format selection*

- With the "Binary format (compact)" setting, the objects selected by the user for export are exported in a compact format (as far as possible). The information associated with the PLC objects is integrated into the export file in binary format.
- When the "XML format" setting is applied, all of the objects selected by the user for export are exported in XML format. This could cause very large files.

Options, Visualizations

Menu: **Tools** ▶ **Options** ▶ **IndraLogic 2G** ▶ **Visualizations**.

General settings for working with visualizations in IndraLogic 2G are made in this dialog.

General tab

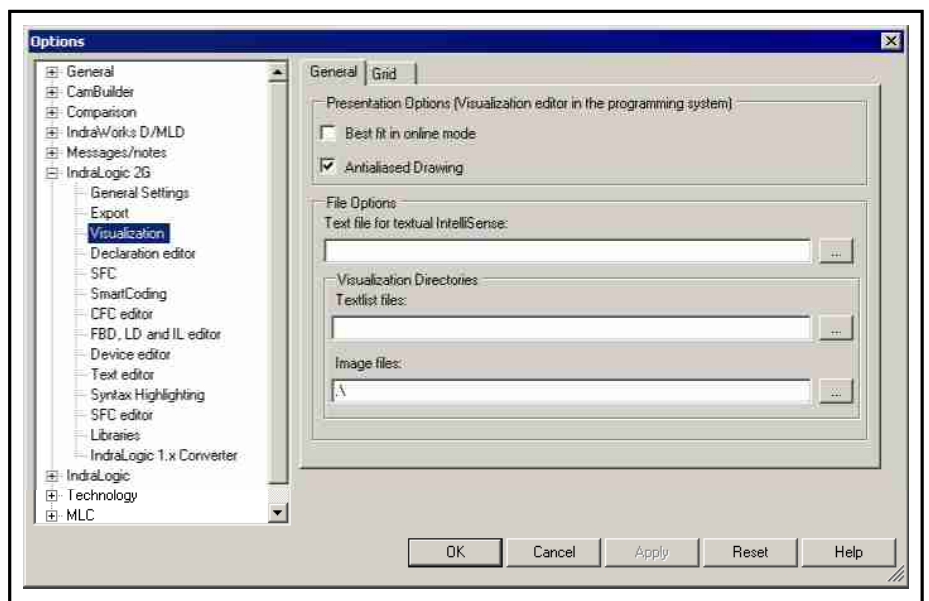


Fig.3-156: Dialog for visualization options, general information

Menu Items

Display options (visualization editor in the programming system)

- **Optimum size in online mode**

This option has to be enabled the visualization is to be adjusted to the current editor window size in online mode so that all elements are always visible.

- **Antialiased drawing**

in preparation

File options


- **Text file for textual "List components":**

In the input field located below, the name (including the path) of a file (of .csv format) can be entered. It can then be used as a source for the suggestions contained in the "List components" function during editing of the 'Texts/Text' field within the properties of a visualization element.

This text file can come from the [export of a global text list, page 288](#),. It has a tabular structure which has to correspond to the structure of text lists which is described on the [help page for text lists, page 55](#),.

- *Visualization directories*


- **Text list files**

Use the  button to open the dialog for defining an existing or a new directory for the language files that are to be used in the project for configuring text and language in the visualization.

The files found in this directory are made available when a text list is to be imported, e. g. after it has been processed externally by a translator. Only one directory can be entered here.

For text and language configuration, see also the description in [Text lists, page 55](#).

- **Image files**

Use the  button to open the dialog for defining an existing or a new directory for the image files that are to be accessible in the project for visualizations. Enter several directories here in a list with the items separated by semicolons.

Regarding image files, see also the description in [Image pools, page 61](#).

Grid tab

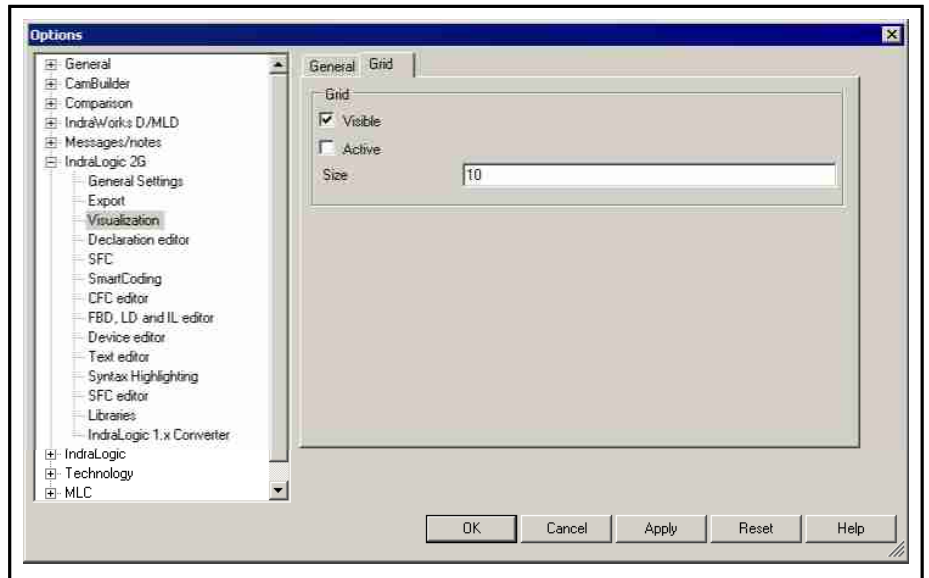


Fig.3-157: Dialog for visualization options, Grid
Grid

- **Visible:**
when this option is enabled, a grid based on the size defined below is shown in the visualization editor for orientation purposes.
- **Active:**
when this option is enabled, visualization elements can only be positioned on the points in the grid "size" defined below, no matter if the grid is set as "visible" or only "active".

When inserting or moving an element this means that the center of the element can only be positioned on one grid point.

When changing an element, this means that the small rectangle on the element edge that is moved in order to change the size or form of the element can only be moved to one grid point.
- **Size:**
vertical and horizontal distance between the grid points in pixels.

Options, Declaration Editor

Menu: **Tools** ▶ **Options** ▶ **IndraLogic 2G** ▶ **Declaration Editor**.

This dialog enables general settings to be made for working in the [declaration editor](#), page 326.

Menu Items

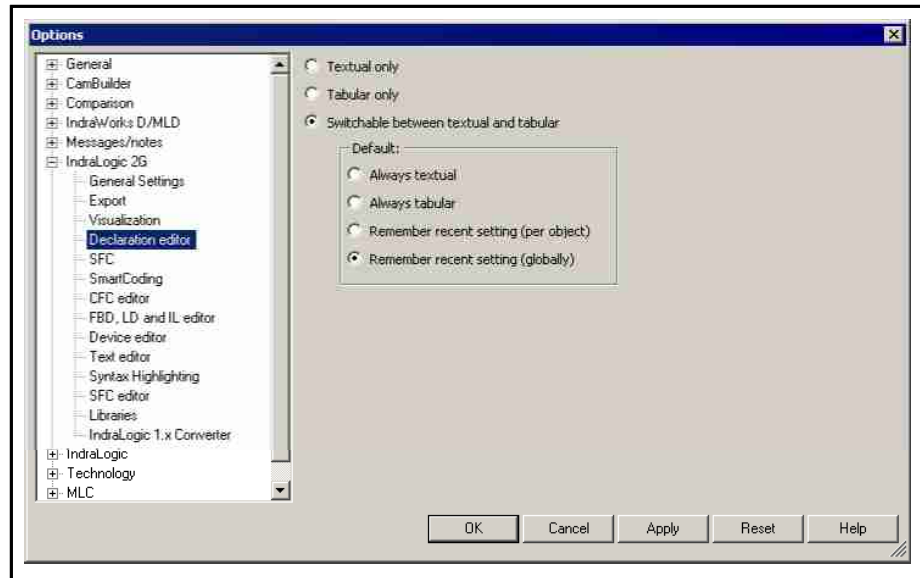


Fig.3-158: "Declaration editor options" dialog

The display for the declarations can be specified:

Only textual

Only tabular

Switchable between textual and tabular: If this setting is activated, there are two buttons in the declaration editor window by means of which it can be switched between the textual and the tabular form.

In this case, one of the following settings defines which of the two views is used by default upon opening of an object in the editor:

- **Always textual**
- **Always tabular**
- **Save latest setting (per object):** If an object is re-opened, the declaration editor appears in the same view as during the last editing process.
- **Save latest setting (global):** If an object is opened, the declaration editor appears in the same view as during its last use, regardless of the object.

Options, SFC

Menu: **Tools** ► **Options** ► **IndraLogic 2G** ► **SFC**.

The **default settings for SFC objects** can be defined in this dialog.

Each **new** SFC object is automatically then provided with these settings in its properties.



The changes to the settings made here have to be transferred explicitly to **existing** SFC objects.

See [Modifying the properties of existing POU objects, page 205](#).

Note that the [basic editor options, page 219](#), i.e. settings for layout and display are managed in a separate options dialog ("SFC editor").

Compilation

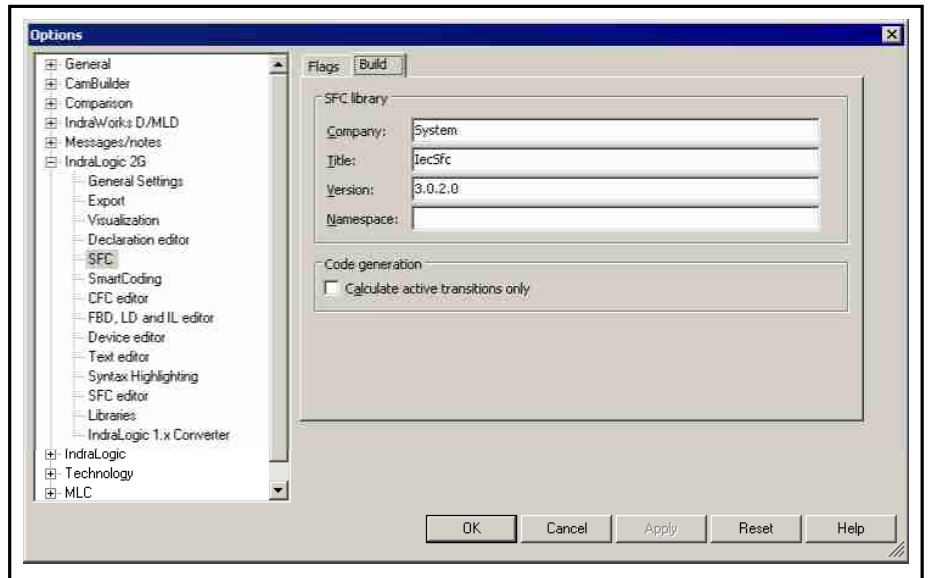


Fig.3-159: SFC options, compilation

SFC library:

The SFC library defined here is used by default for new SFC objects. **Company**, **Title** and **Version** are entered as they are in the project settings for the library as delivered. A **namespace** can be entered as well in order to uniquely address a library. This could be necessary if several versions of the library are available on the system.

When entering a namespace, however, always ensure that the space entered here is the same as that defined in the [Library manager, page 368](#), for the library!

By default this dialog contains settings for the **Sflec.library** library, which is delivered along with the standard profile.

Code generation:

Calculate active transitions only: When this option is selected, only the currently active transitions are calculated.

Menu Items

Variables

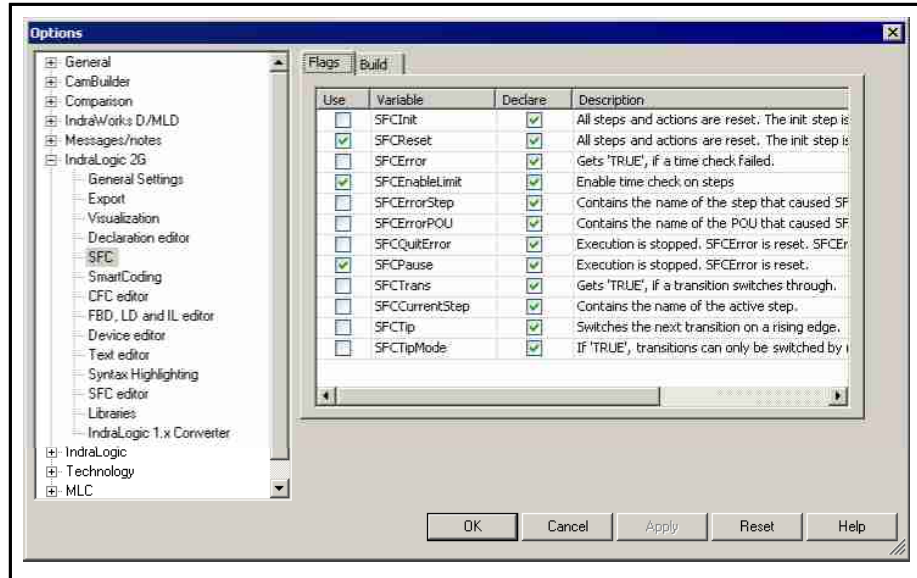


Fig.3-160: SFC options, variables

All possible "flags", i.e. implicitly generated variables for monitoring and controlling the processing in an SFC chart are listed here. A short description is included in the dialog.

Detailed information on the flags, see page 414.

Clicking on the checkbox for a flag variable enables automatic declaration (**Declare**) and use (**Use**). These settings are then accepted as default settings for new SFC objects.

If "Declare" is selected but "Use" is not selected, the variable is declared, but the flag has no function during processing.



Note that a flag variable with automatic declaration is visible in the declarations of the SFC editor **only in online mode!**

Menu Items

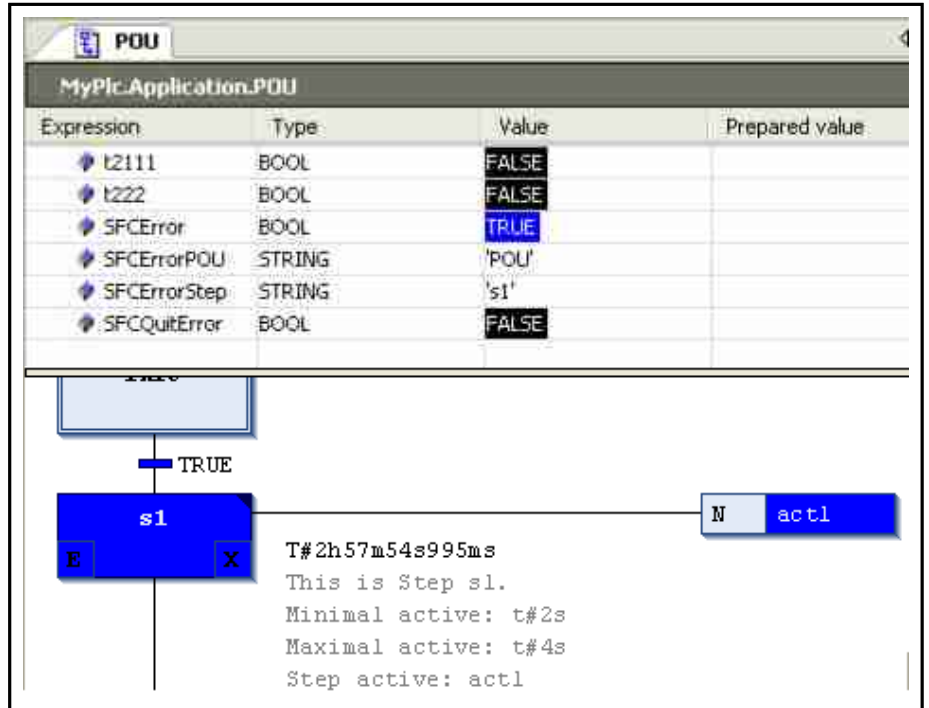


Fig.3-161: Example of an SFC error flag in the online mode of the editor
A timeout is indicated in step "s1" in the SFC object "POU" by the SFCError flag.

Modifying the properties of existing POU objects

Since modified options are only effective for newly created POU objects, existing POU objects have to be modified later on (right click, "Properties"):

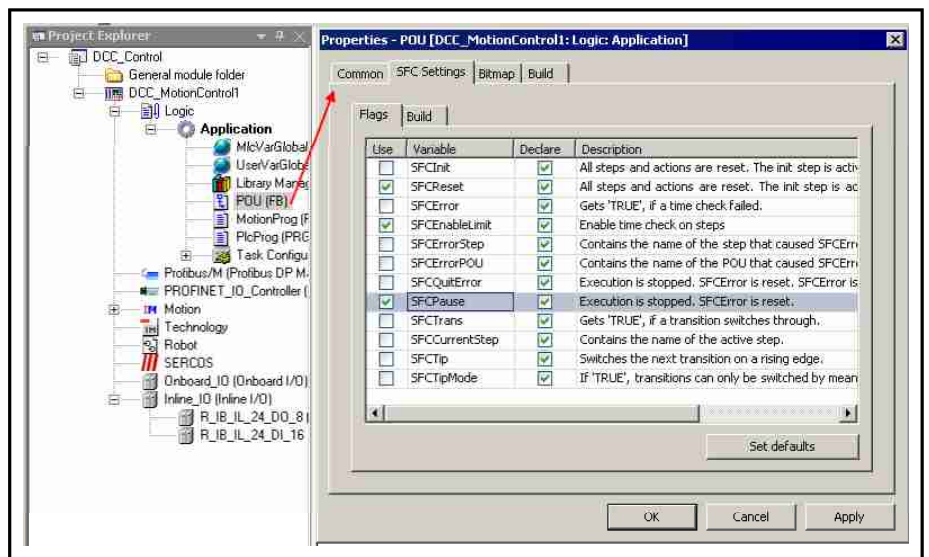


Fig.3-162: Modifying the SFC settings for an individual POU object

Options, Smart Coding

Menu: **Tools** ► **Options** ► **IndraLogic 2G** ► **Smart coding**.

This dialog contains settings that makes entering code easier.

Menu Items

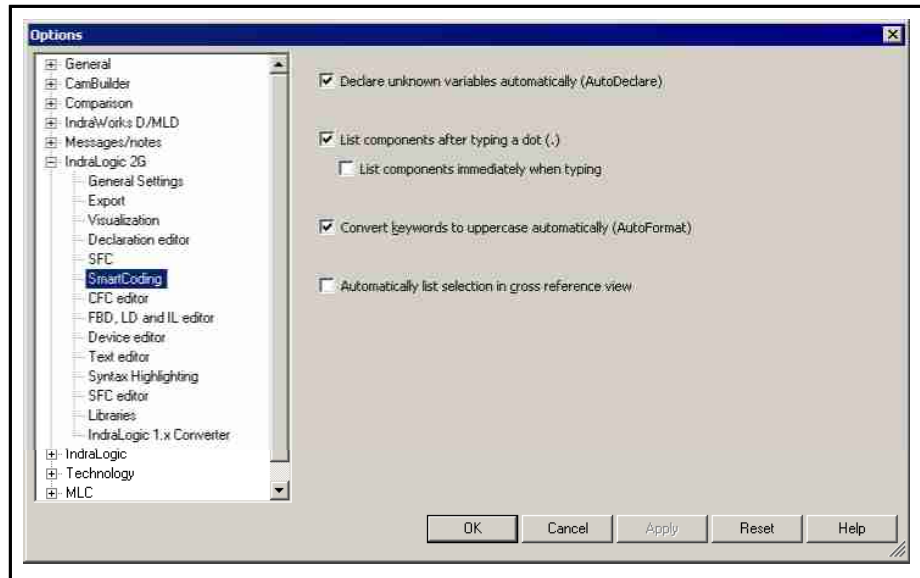


Fig.3-163: Options, smart coding

- | | |
|---|--|
| <p>Declare unknown variables automatically (AutoDeclare)</p> | <p>When this option is selected, the Declare variable, page 100, dialog opens automatically as soon as an identifier that is not yet declared is entered in a language editor and the user moves beyond the input line.</p> |
| <p>List components after typing a dot (.):</p> | <p>This option enables the List components function, page 103,
That means that if a dot (.) is entered in an editor in a position in which an identifier is expected, a selection list appears with the possible inputs at this position.</p> |
| <p>List components immediately when typing:</p> | <p>If this option is selected, after any character string is entered in an editor, a selection list appears with the available identifiers and operators. This is also a type of "List components" functionality.
The first entry in the selection list that starts with the same character string as that which was entered is automatically highlighted as the selection and can be accepted at the cursor position by pressing <Enter>.
Otherwise, another entry can be selected from the list.</p> |
| <p>Convert keywords to upper case automatically (AutoFormat):</p> | <p>If this option is selected, all of the keywords are automatically written in upper case letters.
Example:
When
<code>bVar:bool;</code>
is input, it is converted to
<code>bVar: BOOL;</code>
.</p> |
| <p>Automatically update cross references when changing selection:</p> | <p>If this option is enabled, the cross reference list, page 107, automatically displays a list of the references of the variables that are currently in focus in the editor.</p> |

Differences from IndraLogic 1.x with regards to the cross reference list:

- The cross reference list can be displayed in a window (view) next to the editor window.
- The project does not have to be compiled in order to display the cross references.
- Component accesses, enumeration constants and user-defined data types are also displayed.
- A dynamic display of the cross references for the respective variable in focus can be enabled.

Options, CFC Editor

Menu: **Tools** ► **Options** ► **IndraLogic 2G** ► **CFC editor**.

This dialog enables general settings to be made for working in the [CFC editor](#), page 309.

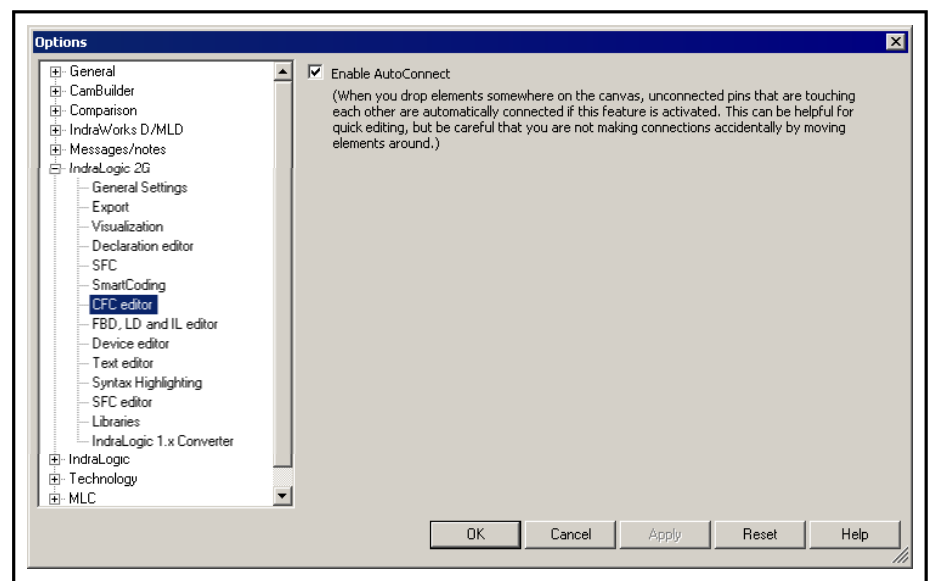


Fig.3-164: "CFC Editor Options" dialog

Enable AutoConnect

If this option is active, the following applies:

If a CFC element is dragged and dropped somewhere on the editor working area, unconnected pins that are "touching" each other are automatically connected. This might support faster editing; however ensure not to generate unwanted links accidentally when shifting elements!

Options, FBD, LD and IL

Menu: **Tools** ► **Options** ► **IndraLogic 2G** ► **FBD, LD and IL**

The FBD/LD/IL editor is configured in this dialog:

Menu Items

General

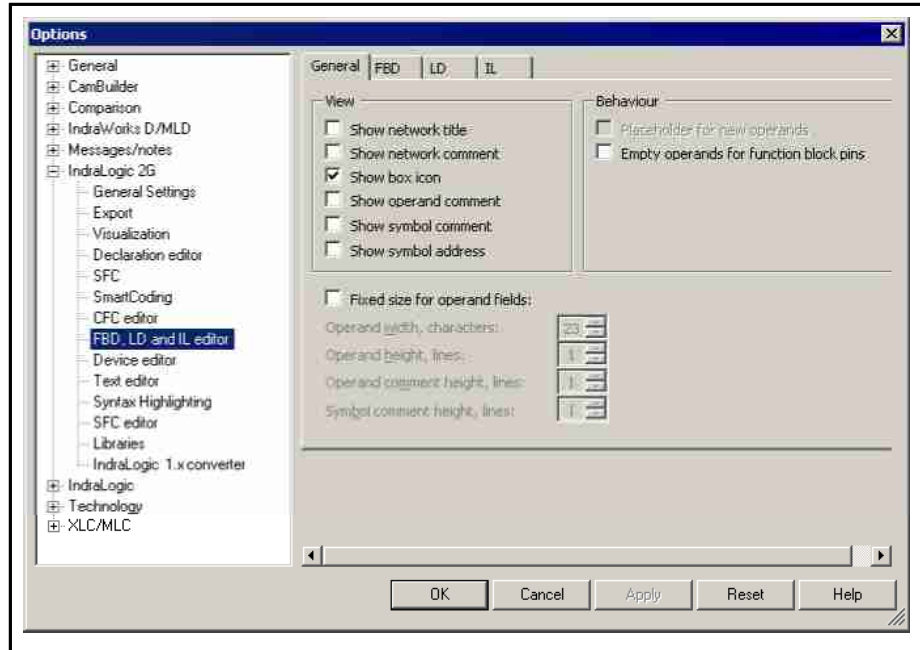


Fig.3-165: Options for displaying the FBD/LD/IL editor

View

- Show network title: The network title - if defined - is displayed in the upper left corner of the network.
- Show network comment: The network comment - if defined - is displayed in the upper left corner of the network. If the network title is also displayed, the comment appears in the line below.
- Display function block symbol: If a function block or a function has been provided with a symbol (bitmap) via a library or via the object properties, it is displayed in the function block element in the FBD and LD editor. The standard operators are provided with symbols, as well

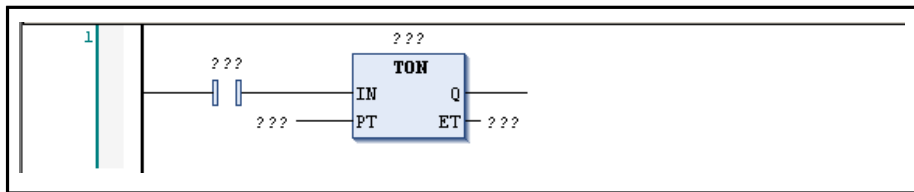


Fig.3-166: "Show function block symbol" option not activated

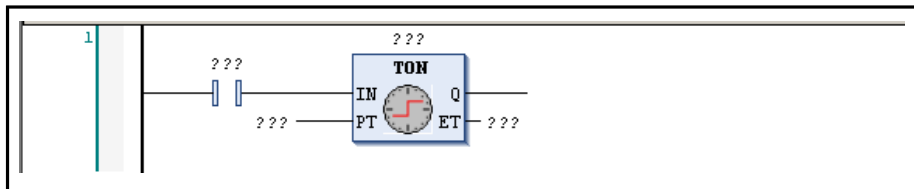


Fig.3-167: "Show function block symbol" option activated

- Show operand comment: The comment that might have been added to a variable in the implementation section of the editor is displayed. The operand comment refers only to the local use position of the variable, in contrast to the "symbol comment" which is indicated in the declaration of the variable.



Menu Items

- Show symbol comment For each symbol (variable) which has been provided with a comment in its declaration, this comment is displayed above the variable name. Note that in addition to or instead of this comment, a local "operand comment" can be assigned, as well.
- Show symbol address For each symbol (variable), the address that might have been assigned is displayed above the variable name.

Behavior

- Placeholder for new operands: ### in preparation ###
- Empty operands for function block pins ### in preparation ###

Fixed size for operand field

- The FBD, LD and IL editor automatically adjusts the size of the operand comment fields.
- If this option is selected, the following parameters define the display size of the information fields for an operand:
- Width of operand field Maximum number of characters that can be shown for operand name.
 - Height of operand field Maximum number of lines that are used for the display of the operand name.
 - Height of operand comment field Maximum number of lines that are used for the display of the operand comment.
 - Height of operand symbol field Maximum number of lines that are used for the display of the operand symbol.

FBD

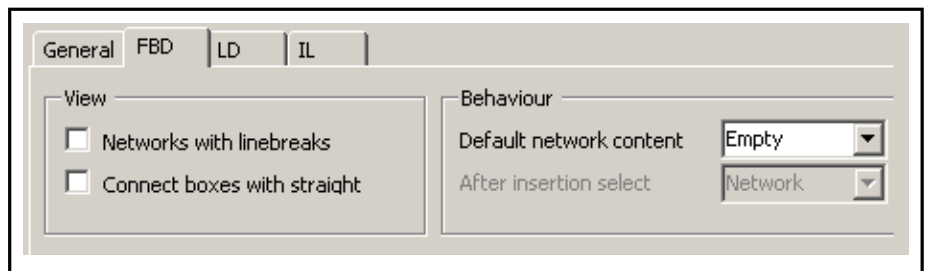


Fig.3-168: Options for displaying the FBD editor

Menu Items

View

Network with line breaks

If this option is selected, the networks are displayed with line breaks so that the largest number of function blocks possible can be displayed in the current width of the editor window. Consequently, the networks might require more height.

If the editor window is not wide enough, the networks are not wrapped.

Connect function blocks with straight lines

If this option is selected, the elements are arranged in a network so that the lines between them receive a fixed, short length and the width required for displaying the networks is reduced to the largest possible extent. Consequently, the function block boxes might be extended vertically in order to create enough space for the input and output elements.

If this option is not selected, the elements maintain their standard size and the connection lines are modified to fit within the space required.

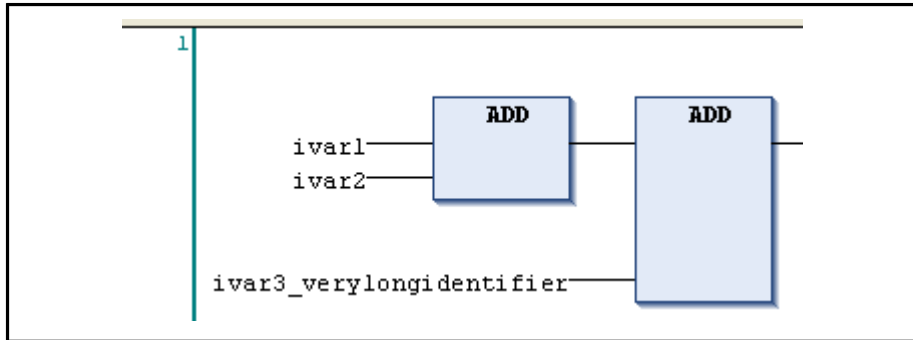


Fig.3-169: Option disabled:

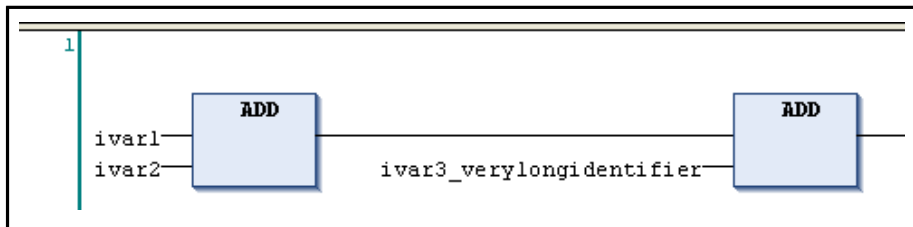


Fig.3-170: Option enabled:

Behavior

Network content

This selection describes which content is displayed after the insertion of a new network in the editor window.

(Empty, assignment, empty block).

Selection after insertion

This selection describes which element is selected after insertion of a new network.

(Element, network).

LD

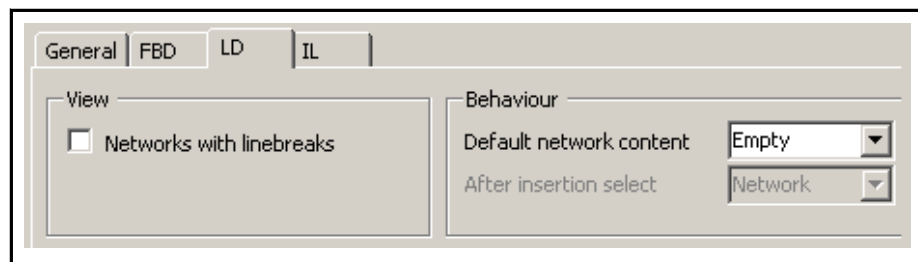


Fig.3-171: Options for displaying the LD editor

Menu Items

View

Network with line breaks If this option is selected, the networks are displayed with line breaks so that the largest number of function blocks possible can be displayed in the current width of the editor window. Consequently, the networks might require more height. If the editor window is not wide enough, the networks are not wrapped.

Behavior

Network content This selection describes which content is displayed after the insertion of a new network in the editor window.
(Empty, contact and coil, empty block).

Selection after insertion This selection describes which element is selected after insertion of a new network.
(Element, network).

IL

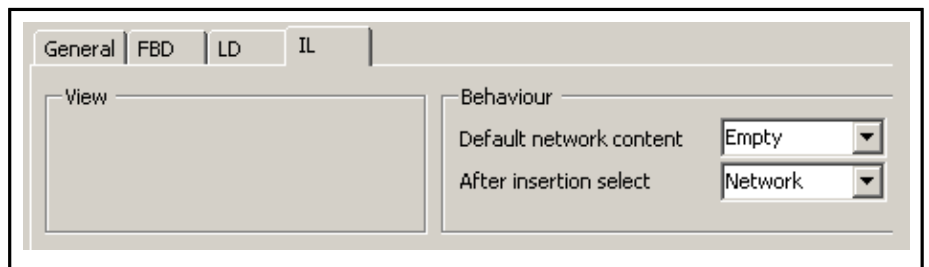


Fig.3-172: Options for displaying the IL editor

Behavior

Network content This selection describes which content is displayed after the insertion of a new network in the editor window.
(Empty, LD and ST, CAL).

Selection after insertion This selection describes which element is selected after insertion of a new network.
(Text, network).

Options, Device Editor

Menu: **Tools** ▶ **Options** ▶ **IndraLogic 2G** ▶ **Device Editor**.

This dialog enables general settings to be made for working in the [device editor, page 331](#).

Menu Items

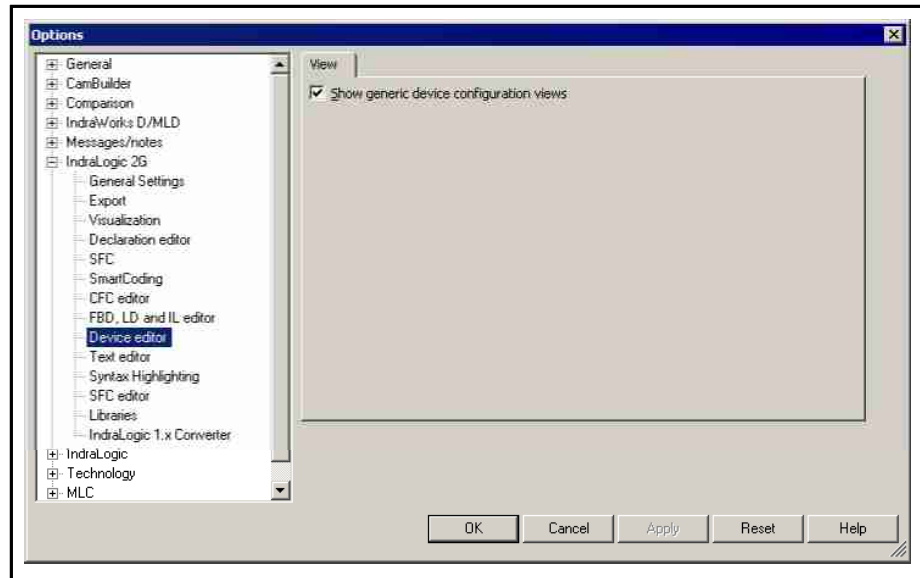


Fig.3-173: "Device editor options" dialog

The "View" subdialog contains the **Show generic device configuration editors** setting.



This option enables

1. the "Field Bus Diagnostics" tab in the object properties for PROFIBUS DP master and PROFINET IO controllers.
2. For devices that can be parameterized (field bus objects), the configuration dialog is available.

Options, Text Editor

Options, Text Editor, General Information

Menu: **Tools** ► **Options** ► **IndraLogic 2G** ► **Text editor**.

This dialog contains settings for working in one of the text editors:

Tabs

- **Editing:** Defining Undo steps, tabs, indentation, breaks, etc., page 212
- **Text area:** Color and font definitions for the text area, page 215
- **Margin area:** Definitions of colors, font, mouse activity in the margin of the editor, left of the text area, page 217
- **Monitoring:** Enabling/disabling Inline monitoring, display of monitoring fields, page 218

Options, text editor, editing

Defining Undo steps, tabs, indentation, breaks, etc.

Menu Items

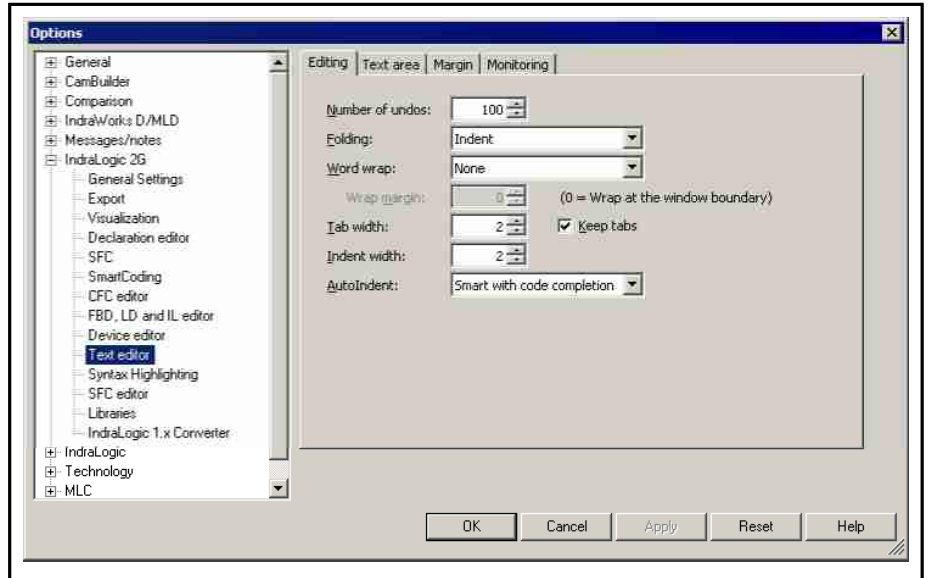


Fig.3-174: Options, text editor, editing

- Number of Undo steps The number of editing steps entered here is saved and can be undone with the "Undo" command.

- Fold (outline form): The selected option determines if the code is to be structured using indents. If yes, indented sections or sections that are identified by a special comment can be hidden or shown by using plus and minus signs placed before the first line of the respective section. The following options are possible:
 - None:** No indent.
 - Indent:** All lines that are indented relative to the previous lines are collected into an "indented unit" with a sign placed before the preceding line.

Examples: The lines between VAR and END VAR or between IF and END_IF are indented, Thus, they are collected in an indented unit marked with a minus sign in front of "VAR" when open and a plus sign in front of "VAR" when closed.

Open and closed indented units:

Menu Items

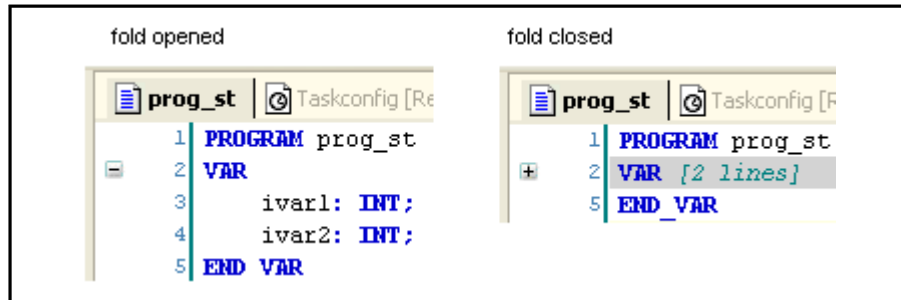


Fig.3-175: Opened and closed indented units (declaration)

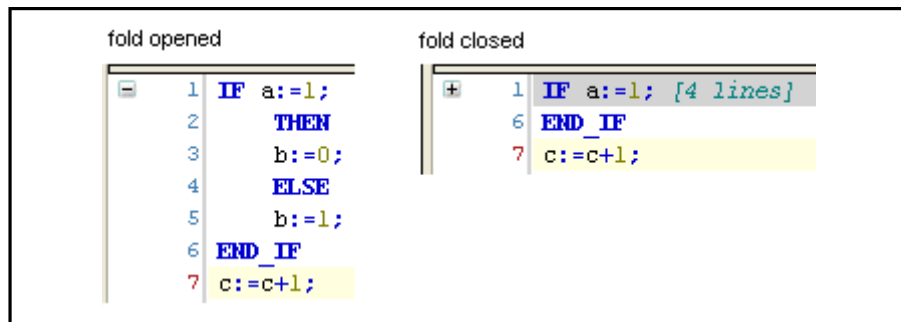


Fig.3-176: Opened and closed indented units (ST)

Explicit: If this option is selected, the code section which includes an indented section is identified explicitly with comments: A comment that includes three opening curly brackets "<<<" has to be located in front the section; another comment that follows the section has to include three closing curly brackets ">>>". The comments can include additional text.

Example of the definition of an indented section by using special comments:

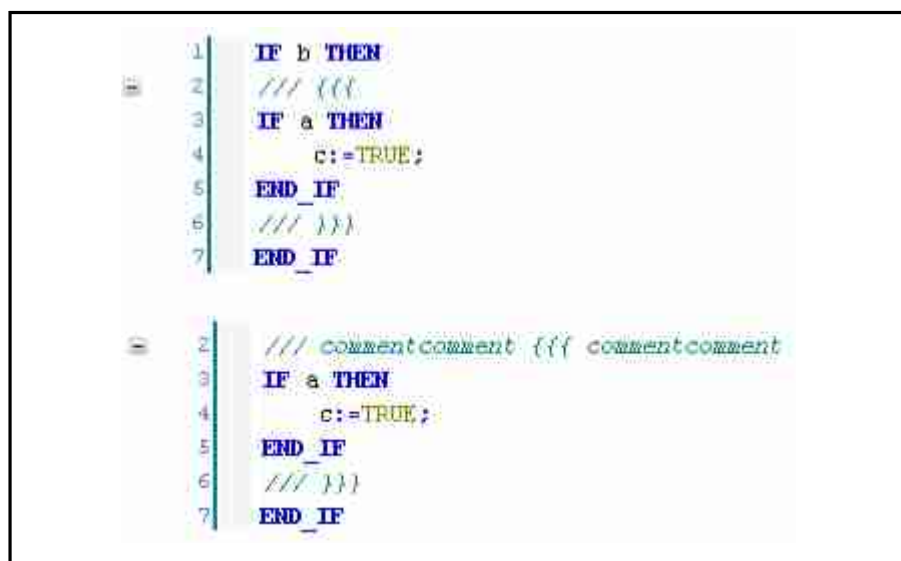


Fig.3-177: Indented section explicitly defined by special comments:

Word wrap:	<p>None: The lines are wrapped endlessly.</p> <p>Soft: The lines are broken at the margin of the editor window if a "0" is entered at the break position.</p> <p>Hard: The lines are broken based on the number of characters entered at the break position.</p>	Menu Items
Tab width:	Tab width in number of characters.	
Indent width:	If the Auto Indent option (see below) is enabled, an indentation to the right is made here based on the number of characters entered, i.e. a corresponding number of spaces are inserted at the beginning of the line.	
Auto indent:	<p>Never: No automatic indentations are made during editing. The character strings entered always start at the left margin of the text area.</p> <p>Block:</p> <p>Smart: The lines following a line that contains a keyword (e.g. VAR or IF) are automatically indented based on the Indent width listed above.</p> <p>Smart with code completion: The lines following a line that contains a keyword (e.g. VAR or IF) are automatically indented based on the Indent width listed above and in addition, the corresponding closing keyword is inserted (e.g. END_VAR or END_IF).</p>	
Keep tabs	If this option is selected, an empty space in a line inserted with the <Tab> key, based on the defined Tab width (see above), is not broken down into individual single spaces. This means that if the tab width is changed, the empty spaces created with the <tab> key are modified as well.	

Options, text editor, text area

Color and font definitions for the text area

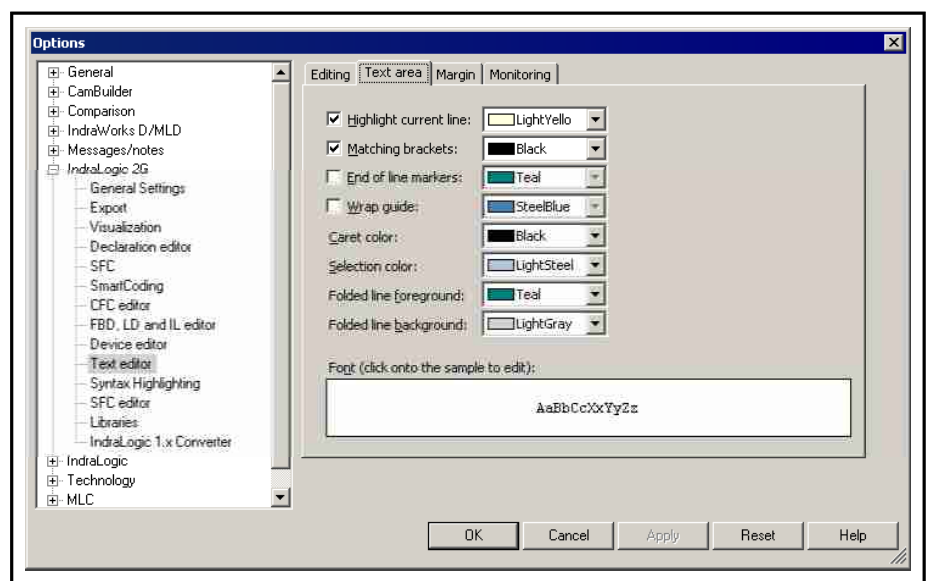


Fig.3-178: Options, text editor, text area

The following options can be enabled for better optical support during editing:

Menu Items

Highlight current line:	The line that contains the cursor is highlighted with the set color.
Matching brackets:	When the cursor is positioned before or after parentheses "(" or ")" in a code line, the matching closing or opening parenthesis is marked with a frame in the set color. Likewise, if the cursor is in the first or last line of a parentheses-enclosed expression ¹⁾ the corresponding last or first line of this area is shown with a square frame in this color.
End of line markers:	The end of each editing line is marked by a small dash in the set color after the last character in the line (including spaces).
Line break position:	If a soft or hard line break is enabled (see above, 'Editing' dialog), the defined line break position is shown with a vertical line in the selected color.
Caret color:	Color of the cursor
Selection color:	The currently selected text area is highlighted with the selected color.
Folded line foreground:	The header of a closed, indented section in the code is displayed in the selected color.
Folded line background:	The header of a closed, indented section in the code is highlighted in the selected color.
Fonts:	The font set here is used in the text editor. Click on the sample field to open the standard dialog for configuring the font.

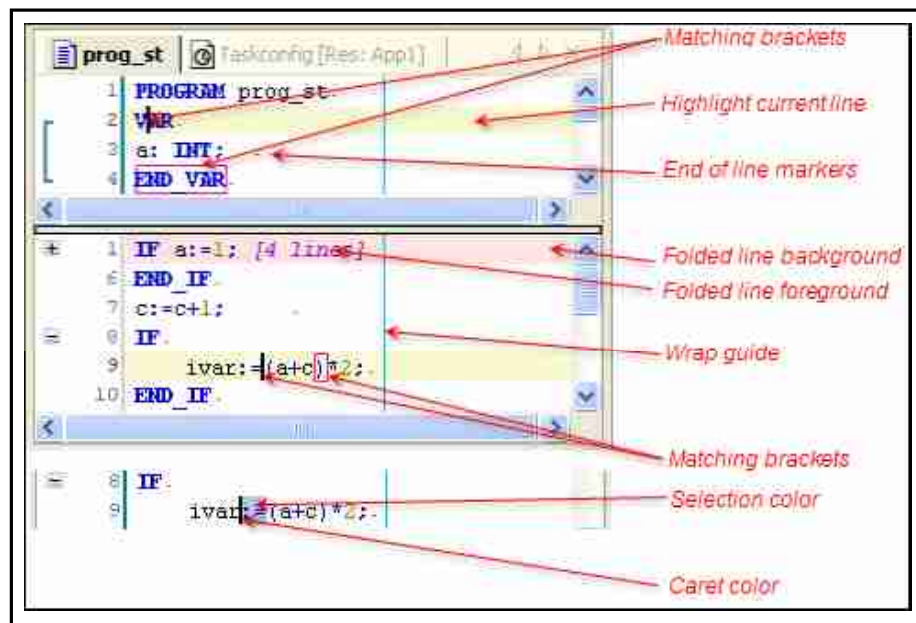


Fig.3-179: Example settings for the text area

1) A parentheses-enclosed expression includes all of the lines between the starting and end keyword of a construct (e.g. all of the lines between IF and END_IF). See also the "Margin area" subdialog.

Options, text editor, margin area

Definitions of colors, font, mouse activity in the margin of the editor, left of the text area

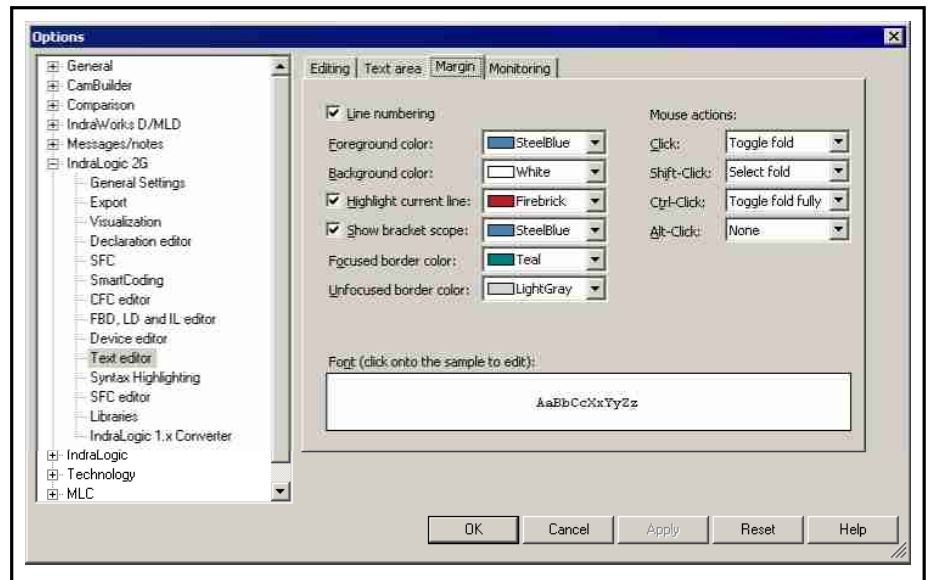


Fig.3-180: Options, text editor, margin area

The following settings and options refer to the left margin area in the text editor window separated from the input area by a vertical line.

Line numbering:	Line numbers are displayed in the declarations and in the implementation section of the text editors, always starting with 1 in the first line.
Foreground color:	Color of the line numbers
Background color:	Background color of the margin area
Current line:	Color of the line number for the line in which the cursor is currently positioned
Parentheses-enclosed expression:	A parentheses-enclosed expression includes the lines between the keywords that open and close a construct, e.g. between IF and END_IF. The bracket scope function can be enabled using the 'Matching brackets' option in the 'Text area' dialog. If the cursor is positioned in front of, after or within one of the corresponding keywords, the bracket area is displayed with a bracket in the margin.
Focused border color:	Color of the vertical separating line between the margin area and the input area in the currently active section of the text editor window.

Menu Items

- Unfocused border color: Color of the vertical separating line between the margin area and the input area in the section of the text editor window that is not currently active.
- Mouse actions: One of the following actions can be assigned to each of the specified mouse movements or mouse shortcuts (click, <Shift> + click, <Ctrl> + click, <Alt> + click).
The action is executed if the mouse movement is executed on the plus or minus sign positioned in front of the header of a parentheses-enclosed area:
 - None:** No action.
 - Select fold:** All of the lines in the parentheses-enclosed area are selected.
 - Toggle fold:** The parentheses-enclosed area, or in the case of nested enclosed areas, the first level of the parentheses-enclosed area, is opened or closed.
 - Toggle fold fully:** All levels of a nested parentheses-enclosed area are opened or closed.

Options, text editor, monitoring

Enabling/disabling Inline monitoring, display of monitoring fields

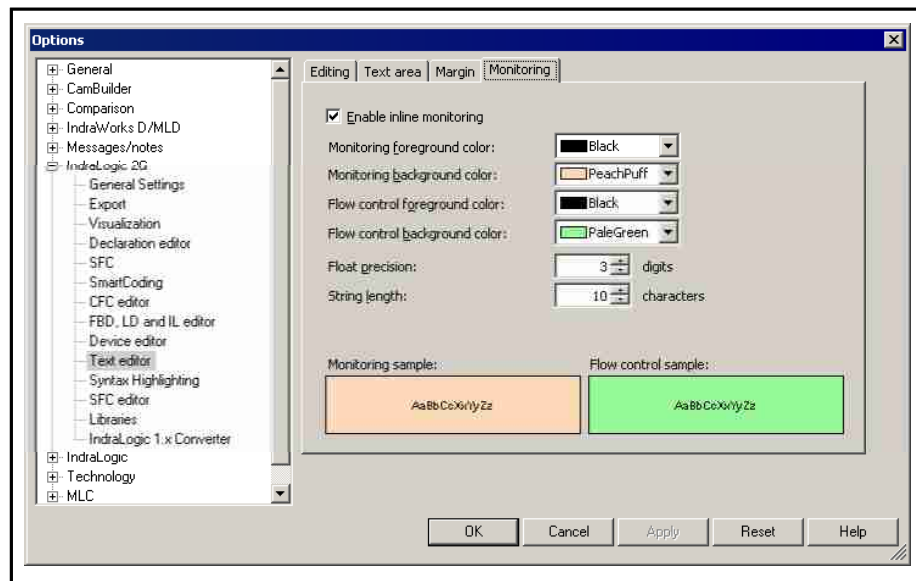


Fig.3-181: Options, text editor, monitoring

- Enable Inline monitoring: The display of monitoring fields after variables in a text editor in online mode can be enabled and disabled.
- Foreground color: The value in the monitoring field is displayed in the selected color.
- Background color: The background in the monitoring field is displayed in the selected color.
- Sequence control foreground color: The value in the monitoring fields at the sequence control positions is displayed in the selected color.
- Sequence control background color: The background of the monitoring fields at the sequence control positions is displayed in the selected color.

Menu Items

- Floating point precision: Numbers with decimal points in the monitoring field are displayed to the number of places set here.
- String length: String variable values are displayed in the monitoring field to the maximum length entered here (number of characters)
- .. Preview: Preview of the values currently set for a monitoring field

Preview of the values currently set for a monitoring field

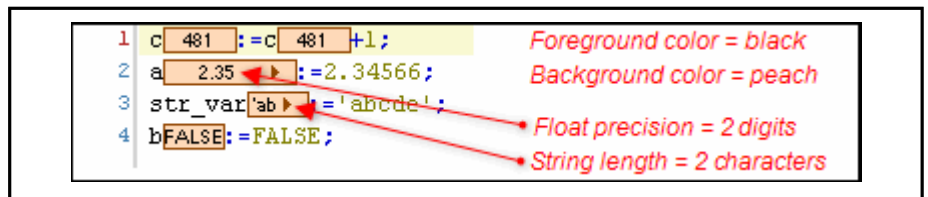


Fig.3-182: Example of Inline monitoring in the ST editor

For a preview with the values just set for the sequence control refer to [Sequence control in the different language editors](#), page 148.

Options, Syntax Highlighting

Menu: **Tools** ► **Options** ► **IndraLogic 2G** ► **Syntax Highlighting**.

This dialog contains color and font settings for the various text elements in an editor (e.g. operands, Pragmas, comments, etc.).

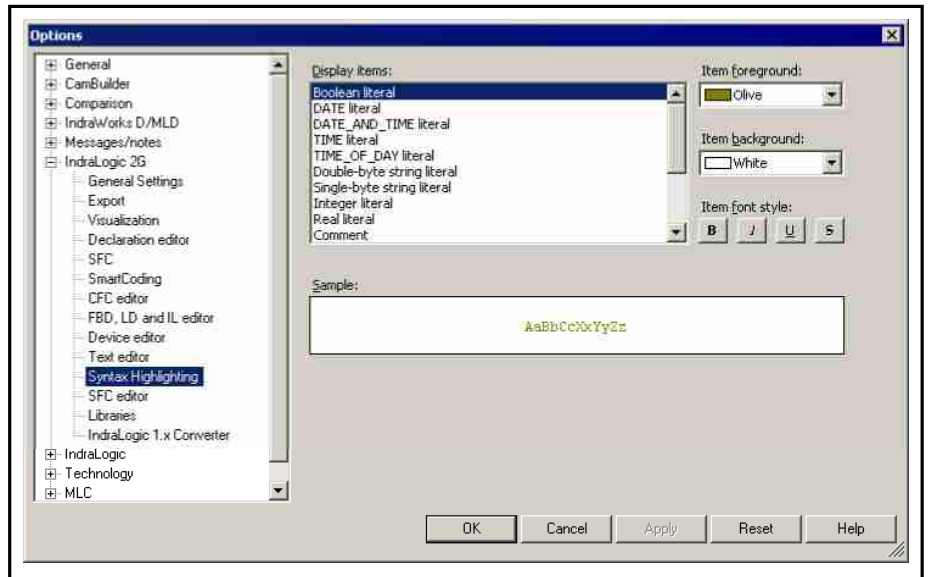


Fig.3-183: "Options" dialog, syntax highlighting category

Both color and font for the text editor display can be set for the text elements listed in **Display items**. A selection list can be used to set the color for the **item foreground** and the **item background**. The **item font style** is defined by clicking the desired symbol for "bold" (**B**), "italic", "underline" and "strike-through".

An example of the current settings can be seen in the **Sample (preview)** window.

Options, SFC Editor

Menu: **Tools** ► **Options** ► **SFC (sequential function chart) editor**.

Menu Items

The dialog contains tabs in which the settings for the [SFC editor, page 399](#), can be made.

Layout

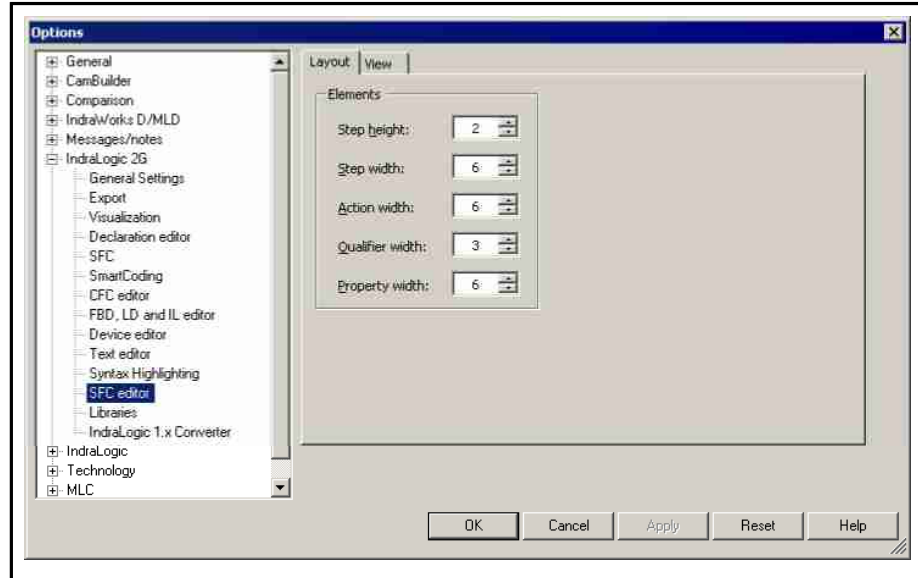


Fig.3-184: "SFC editor options" dialog, layout

In the "Layout" tab, the values are entered in "grid units".

1 grid unit = font size currently set in the text editor options (Text area/Font).

Step height: Height of a step element. Possible values: 1-100.

Step width: Width of a step element. Possible values: 2-100.

Action width: Width of an action element. Possible values: 2-100.

Qualifier width: Width of the qualifier. Possible values: 2-100.

Property width: Width of the display area for the step properties. Possible values: 2-100.

The settings are always effective immediately in all currently open SFC editor windows.

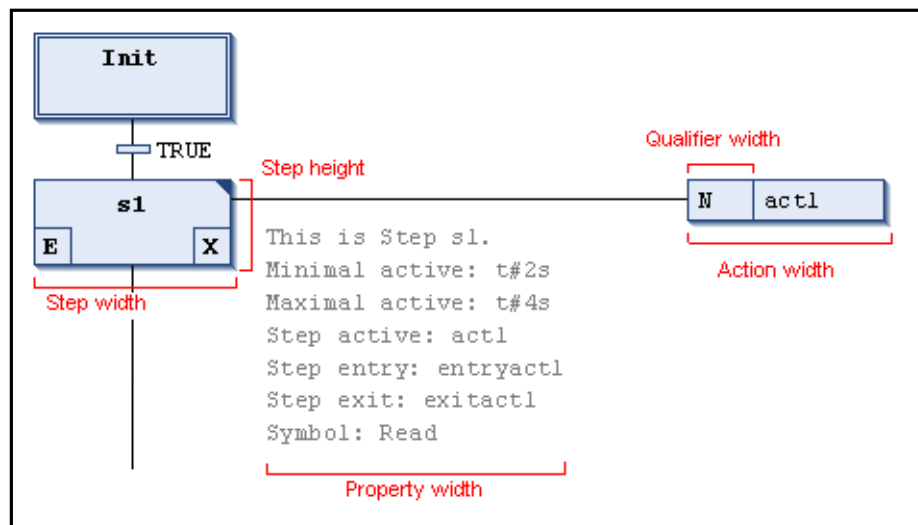


Fig.3-185: Setting parameters for the layout in SFC

View The "View" tab is described in the following:

Menu Items

Property	Value	With name
Step		
Common		
Comment	<input type="checkbox"/>	<input type="checkbox"/>
Symbol	<input type="checkbox"/>	<input type="checkbox"/>
Specific		
Times		
Minimal active	<input type="checkbox"/>	<input type="checkbox"/>
Maximal active	<input type="checkbox"/>	<input type="checkbox"/>
Actions		
Step active	<input type="checkbox"/>	<input type="checkbox"/>
Step entry	<input type="checkbox"/>	<input type="checkbox"/>
Step exit	<input type="checkbox"/>	<input type="checkbox"/>
Transition		
Common		
Comment	<input type="checkbox"/>	<input type="checkbox"/>
Symbol	<input type="checkbox"/>	<input type="checkbox"/>
Jump		
Common		
Comment	<input type="checkbox"/>	<input type="checkbox"/>
Macro		
Common		
Comment	<input type="checkbox"/>	<input type="checkbox"/>
Branch		
Common		
Comment	<input type="checkbox"/>	<input type="checkbox"/>
Action association		
Common		
Comment	<input type="checkbox"/>	<input type="checkbox"/>
Symbol	<input type="checkbox"/>	<input type="checkbox"/>

Fig.3-186: Visibility of the properties, SFC editor options, view

Visibility of properties:

Define here which element properties are to be displayed next to the element in the SFC chart. For each property, checkmarks can be placed in the "Value" column and in the "With name" column. If the "Value" column is selected, the value is shown in the chart. If the "With name" column is selected, the property name is shown.

Menu Items

Example:

The screenshot displays the 'Element properties' window on the left and the 'Editor View' on the right. The 'Element properties' window shows a table of properties for step 's1':

Property	Value
Common	
Name	s1
Comment	This is Step s1.
Symbol	Read
Specific	
Initial step	<input type="checkbox"/>
Times	
Minimal ac...	t#2s
Maximal ac...	t#4s
Actions	
Step active	act1
Step entry	entryact1
Step exit	exitact1

The 'Editor View' shows a ladder logic diagram with an 'Init' step leading to step 's1'. To the right of step 's1', the following properties are displayed:

```

This is Step s1.
Symbol: Read
Minimal active: t#2s
Maximal active: t#4s
Step active: act1
Step entry: entryact1
Step exit: exitact1
  
```

Fig.3-187: Element properties

Online: If the **Show step time** option is selected, in online mode the current step time is displayed to the right of all of the steps for which time properties have been defined.

The screenshot shows the 'Editor View' in online mode. The 'Init' step is shown with a time of 'T#0ms'. The 's1' step is shown with a time of 'T#1s403ms'. The properties for step 's1' are displayed to its right:

```

T#1s403ms
This is Step s1.
Minimal active: t#
  
```

Fig.3-188: Example display of a step time in online mode

Options, Libraries

Menu: **Tools** ▶ **Options** ▶ **IndraLogic 2G** ▶ **Libraries**.

Menu Items

The "**Mappings**" of **library references** that are used when converting an "old" project are managed in this dialog. Until a "mapping" for a specific library is saved here, define it every time an old project is opened that includes this library.

See information on [opening and converting projects in the current format, page 116](#).

See information on [library management, page 83](#).

A "mapping" defines how a library reference should appear after the project is converted into the current format. There are three possibilities:

- The reference is kept, i.e. the library is also converted into the current format (*.library) and installed in the local library repository.
- The reference is replaced by another, i.e. one of the installed libraries replaces the one that was previously included.
- The reference is deleted, i.e. the library is no longer included in the converted project.

If the "mapping" is saved, it is added to the option dialog in the 'Libraries' category:

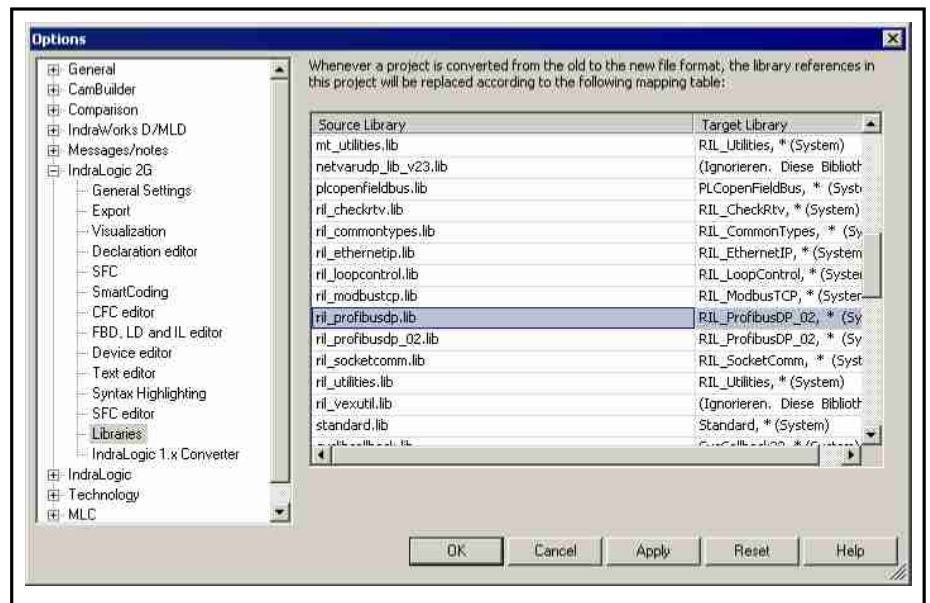


Fig.3-189: Options for libraries

All mappings listed here are applied to the library references of the next old project that is converted. Thus, the mapping definition does not have to be repeated if the same library is included again in a project to be converted.

Each "mapping" is displayed on one line:

Source library: Library path included in the project before conversion

Target library: Name and storage location of the library that is to be included in the project after conversion. If the reference is to be removed, this is shown in the text "(Ignore. This library reference will not be converted.)".

The mappings list can be edited: To do this, double-click on the respective field or select a field and press <Enter>.

When editing a source library entry, the button opens the standard browse dialog in which it can be searched for a library in the "old" format.

Menu Items

When editing a target library entry, the 'Set target library' dialog opens.

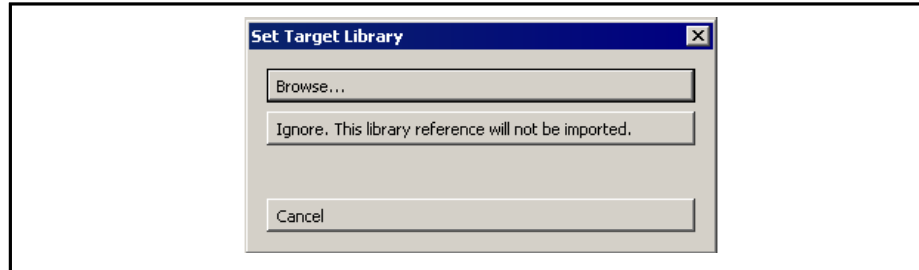


Fig.3-190: 'Set Target Library' dialog

To open a library in the current format, the **Search** button can be used to open the **Select Library** dialog where one of the libraries installed in the local library repository can be selected.

It can be sorted according to company and category, as in the [library repository dialog, page 185](#).

Use the <Ignore. ...> button to specify that the present source library is to be removed from the project forever when the project is converted.

A **new entry can also be added** to the list directly from this dialog by editing the last line, "(Insert new mapping here.)".

Options, Converter for IndraLogic 1.x Projects

This dialog can be accessed using **Tools ▶ Options ▶ Converter for IndraLogic 1.x projects**.

The "mapping" for converting IndraLogic 1.x projects is defined in this dialog, i.e. it is determined how the components of a project are handled when they are transferred into the current project format.

In the 'Device' tab the conversion of the device references (which target system is used) is described:

Until a "mapping" is defined for a device in this dialog, you are prompted to define one if an old project that used this device is opened.

See information on

- [opening and converting projects in the current format, page 116](#).
- [handling devices in the Project Manager, page 63](#).

A "device conversion mapping" defines how the reference should appear on a target system after the project is converted, i.e. a "**target device**" (device in the new project) is assigned a "**source device**" (original device in the old project). There are two possibilities:

- In the device tree for the converted project, the source device is replaced by the target device that is currently available on the system, i.e. is installed.
- The source device is not converted (is ignored), i.e. it will not appear in the device tree for the new project. In this case, even application-specific objects such as the task configuration are not transferred into the new project.

A device mapping can be defined and saved ...

- ... while converting an old project that uses the device (see the description of the [Data transfer, page 116](#) command)
- ... or by direct entry here in the converter option dialog on the 'Devices' tab:

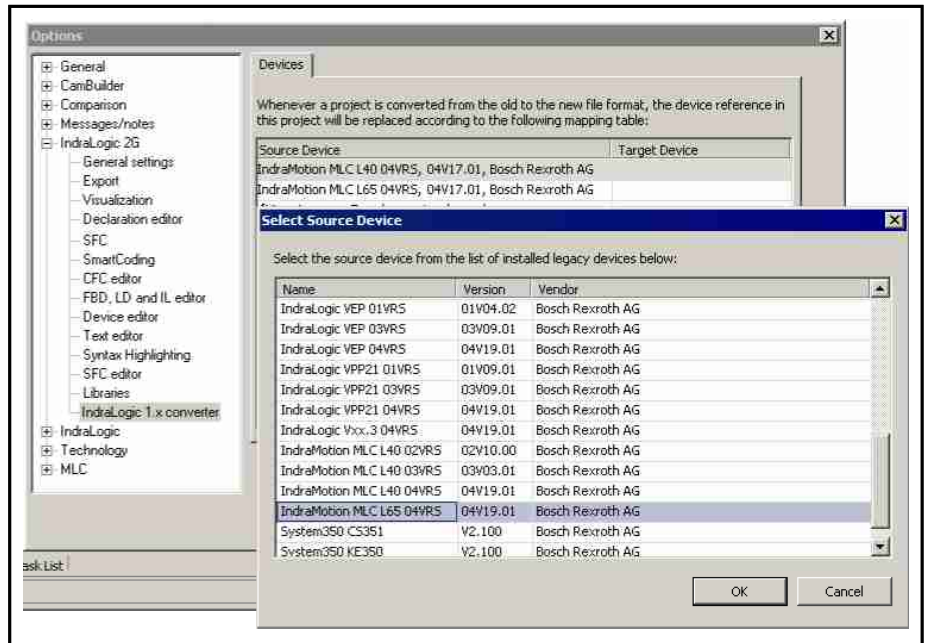


Fig.3-191: IndraLogic converter options

To edit a field in the mapping table, double-click on the field or select it and press <Enter>. Assign each "source device" a "target device":

For entries in the "source device" column, the **'Select original device'** dialog opens, where all devices that the converter can handle are listed. Select the desired device to insert in into the field.

For entries in the 'target device' column, the **'Select target system'** dialog opens, where one of the currently installed devices can be selected. The dialog is managed like the Device database dialog.



If **'None'** is selected, the source device is **not be available in the new project**, which also means that application-specific objects such as the task configuration are not transferred!

All device mappings listed in the Options dialog are applied the next time an old project is converted. In this way, the mapping definition does not have to be repeated if the same device is used in several projects.

Options, IndraLogic 1.x, Settings

This dialog can be accessed using **Tools ▶ Options ▶ IndraLogic (1.x), settings**.



These options are only in effect when working with IndraLogic 1.x!

Menu Items

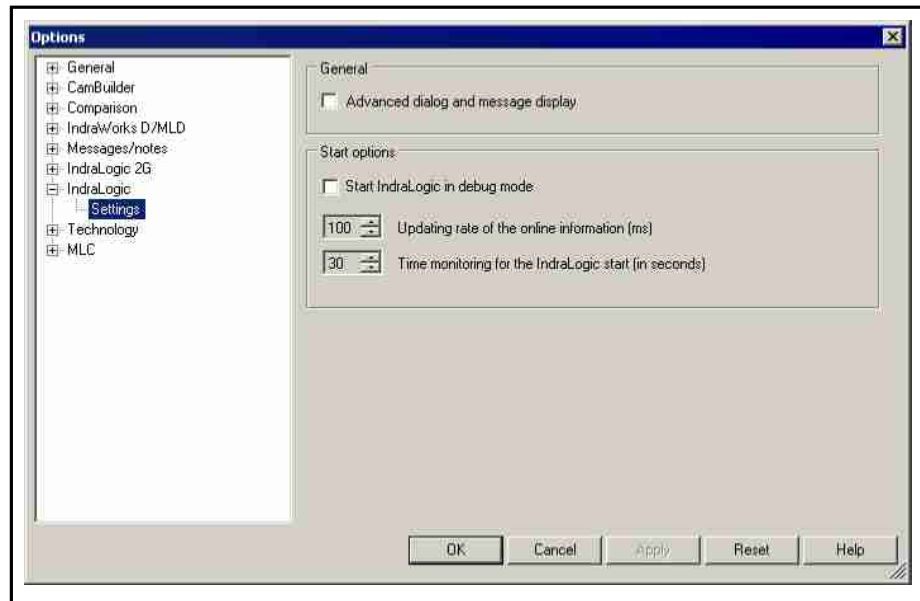


Fig.3-192: Options, IndraLogic 1.x settings

General

- Extended dialog and message display
in preparation

Start options

- Start IndraLogic in debug mode
If a project was created in IndraWorks with a 1.x control, double-click the "Logic" node (or its objects), IndraLogic 1.x opens. When it opens, information is displayed in the message window.
If the checkmark is placed next to "Start IndraLogic in debug mode", these messages are also recorded. The "ILStartingMessages.txt" protocol is stored in the IndraWorks installation directory in the IndraLogic folder.
- Updating rate of the online information
in preparation
- Time monitoring for the IndraLogic start
When IndraLogic 1.x is opened, a monitoring period begins. If IndraLogic does not start within this time period, an error message is output.

3.7 Library Manager - Commands

3.7.1 Library Manager - Commands, General Information

The commands for the [Library manager, page 367](#) can be called from its editor window. Furthermore, the "Add library" command can be accessed in the menu bar as soon as the editor is activated.

If required, the menu configuration can be changed using the dialog in **IndraWorks ► Tools ► Customize**.

- [Add library..., page 227](#)
- [Properties..., page 230](#)
- [Reload library, page 231](#).

Menu Items

For general information on the [IndraLogic 2G library management](#), see page 83.

3.7.2 Add Library...

By default, this command is available in the [Library menu](#), page 371, and in the [Library Manager window](#), page 368,.

Use the command to add a library to the project using the currently active library manager.

Only libraries installed in a local [repository](#), page 185, can be added. Several versions of a library can be used simultaneously in the project using one or more Library managers.

The command opens the 'Add library' dialog.

To insert a library to your project by default only consider the **Library** tab dialog.

The **Placeholder** tab dialog is only required if a library project is currently created in which a target specific library has to be referenced and this library should not be specified yet.

'Library' tab dialog

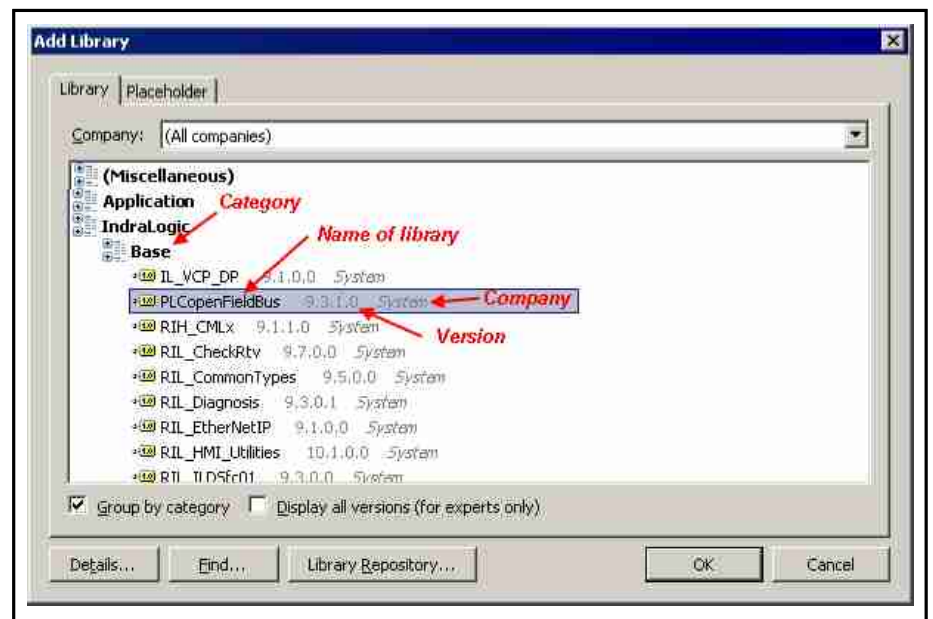


Fig.3-193: 'Add library' dialog, 'Library' tab. Displays only the most recent version.

Menu Items

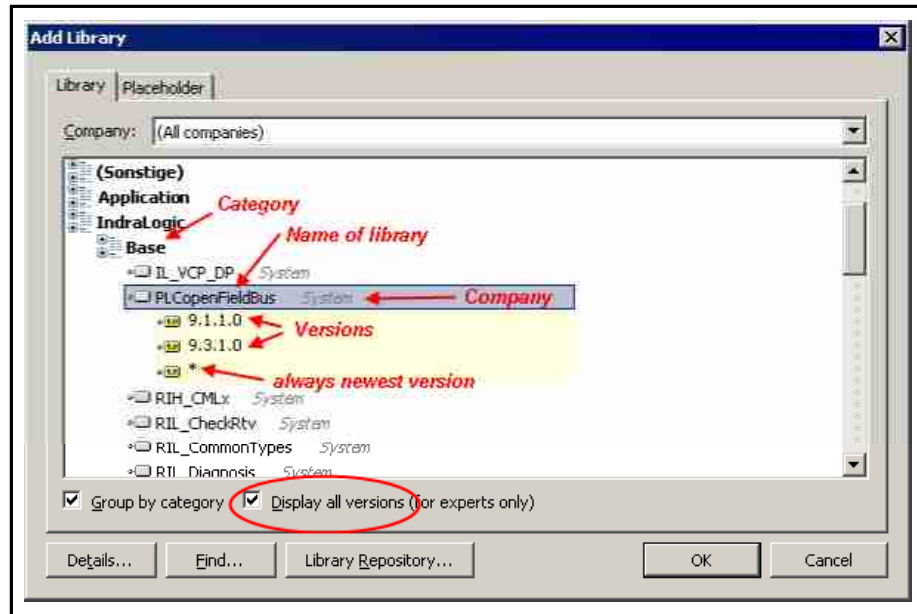


Fig.3-194: 'Add library' dialog, 'Library' tab. Displays all versions.

All installed libraries are shown here. The list can be filtered according to **company**. "(All companies)" displays all available libraries).

If the **Group by category** option is selected, the libraries for the company that is currently set are grouped according to the defined categories. The category designations appear as nodes. A single click on a node opens the list of the related libraries (**Library names**) or subcategories. A single click on a library name opens the list of the available **versions**. If the libraries are not grouped according to categories, the list of all of the libraries is sorted alphabetically.

Select the desired version of the library. A "*" appears in the list in addition to the currently installed versions of a library. If the asterisk is selected instead of a concrete version of the library, the **latest version** available in the repository is always used in the project.

Select the desired library. If **Display all versions (for experts only)** is selected, all of the currently installed versions of a library are listed. A "*" also appears in the list; it stands for "Newest available version in the repository". In this case, you can also select among the various versions. However, this option is not selected by default. Thus, only the most recent version is displayed. In this case, **several libraries** can be selected: To do this, press and hold the <Shift> key down while selecting the desired libraries.

After confirming the selection with OK, the library version is added to the library manager.

The **Find...** button is used to find libraries containing the requested function blocks. A search dialog opens to enter the name of the function block as well as placeholders like "*" and "?".

To add a library that is not yet installed on the local system, use the **Library Repository** button to open the dialog of the same name in order to carry out the installation.

'Placeholder' tab dialog

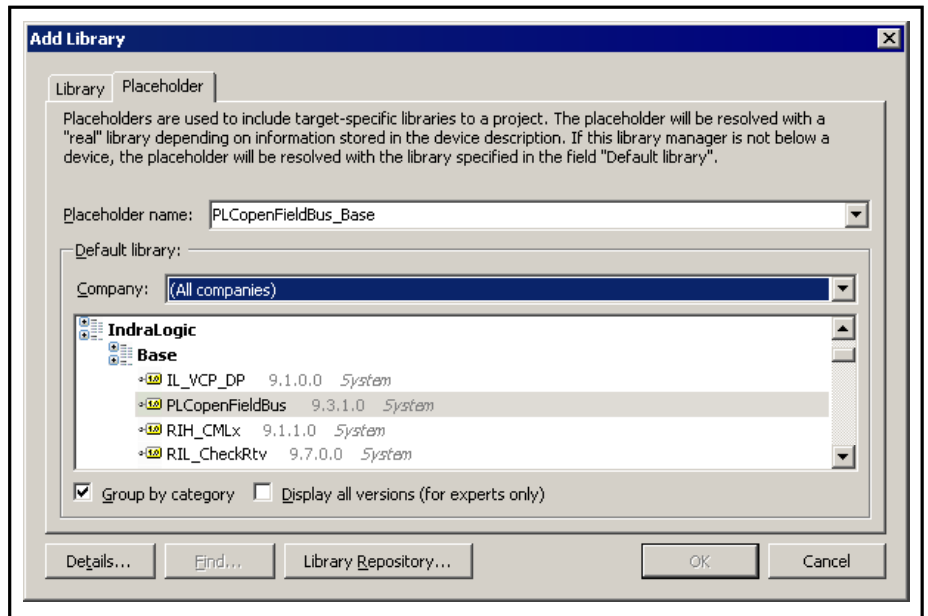


Fig.3-195: 'Add library' dialog, 'Placeholder'

Placeholder with a project

If any project is to be compatible with several exchangeable target devices, the device-specific libraries have to be integrated into the project library manager via placeholders.

As soon as a target device is specified, the placeholders are filled according to the related device description. But even if there is currently no device description available, the placeholders already allow for a syntactic project test.

To add a library to the library manager using placeholders, it has to be selected first from the list provided in the **Default library** section. The selection can be limited by specifying a **company**.

Apart from that, the **placeholder name** has to be entered in the corresponding field. In order to simplify correct entry, all placeholder names currently defined in device descriptions are provided in the selection list.

Placeholder within a library project

If a library project is based on other libraries which depend on the target device, these libraries have to be integrated into the library project by means of placeholders.

That means, a concrete library is not integrated **here**, but instead, just a placeholder. Later, when <library_xy> is used for a specific device in a project, this placeholder is replaced by the name of the device-specific library. However, this name has to be defined in the associated **device description file**. An entry has to be present there that assigns the name of an existing "real" library to the placeholder name.

If, for whatever reason, no device-specific library is available, the placeholder is replaced with the **Default library** entered here in this dialog. (This enables the library project to be compiled without errors, for example, even if there is no existing suitable device description at this point in time.)

In the **Placeholder name** field enter a character string as a name for the placeholder.

In addition, select one of the currently installed libraries as the **Default library**.

Menu Items

This procedure is like adding a library as in the 'Library' subdialog. As in that dialog, select the "Display all versions" (for experts only) option to display a list of all versions of a library currently installed on the system.

After closing the dialog with **OK**, the "Placeholder library" is added to the "Library Manager" tree.

If the [Properties dialog, page 230](#), is opened for this entry, information on the assigned default library can be accessed via the "Default" field.

The placeholder has not yet been replaced in the following example; later, when the library manager is located below a device with the corresponding device description, the name of the device-specific library appears in the 'Name' field instead of the placeholder.

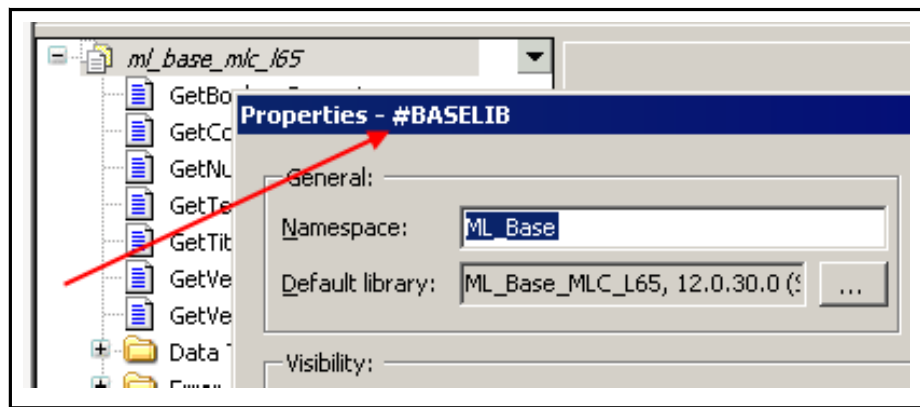


Fig.3-196: Example of a library placeholder in the library manager

3.7.3 Properties...

This command is available in the [Library manager, page 368](#), editor window if a library entry is currently selected.

It opens the **Properties** dialog for the selected library and allows the following settings with regard to namespace, version management and behavior to be made in case the library is referenced in another library and is included in a project via the other library:

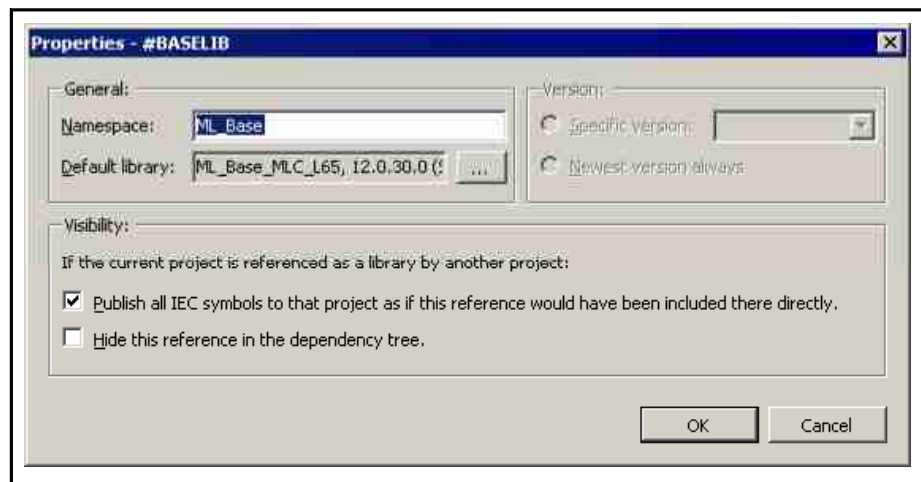


Fig.3-197: "Properties" dialog for a library

General information:

Namespace: The current namespace for the library is displayed. By default, it is identical with the library name, unless a different default namespace was explicitly defined in the project information when the library was created. The

Menu Items

namespace can be modified for the local project in this "Properties" dialog at any time.

For further information on the [namespace for libraries](#), see page 368.

Default: If a library placeholder is currently selected in the library manager, this field contains the name of the library that is to replace the placeholder when the device-specific library is available.

For further information on [Library placeholders](#), see page 229,.

Version: Determine here which version of the library is to be used:

Specific version: Exactly this version is used. Select the desired version from the list.

Newest version always: The newest version of the library found in the library repository is always used. This means that the library modules actually used can change, since a newer version of the library is available.

Visibility: These settings are useful when the library is included in another library, i.e. is referenced there. By default, these settings are not selected.

Publish all IEC symbols to that (referencing) project...: As long as this option is not selected if library function blocks are later included in another library, they can be uniquely addressed by using the corresponding namespace path. This is a combination of the namespace of the "father" library and the its own namespace and is appended to the front of the function block name (prefix).



If, for example, the library "RIL_Uilities" references the library "Util" and "Util" contains the function block "BLINK", an instance "bk1" of "BLINK" has to be declared as follows:

```
bk1: RIL_UTILITIES.UTIL.BLINK;
```

For further information about [using libraries](#), see page 83.



The option **should only be selected** if you wish to create a so-called "container library", i.e. a library that does not define its own blocks, but instead, only references other libraries to create a type of library bundle. This can be useful for including several libraries in a project simultaneously, by simply including the container library. In this case, however, it is desirable to position the individual libraries in the package at the top level in the project's library manager in order to shorten the access path for the library blocks. And that is exactly what can be achieved by selecting these options ahead of time.

Hide this reference in the dependency tree: If this option is selected, the library is later included in another library and this library is then added to a project, it is not be displayed in the Library Manager.

This allows **"hidden" libraries** to be added, but it has to be carried out consciously, since errors occur during compilation that are related to errors in the library. There might be problems finding the cause.

3.7.4 Reload Library

This command is available in the [Library manager](#), page 368, editor window if a library is selected that was not loaded correctly when the project was opened.


If, for whatever reason, a library is not available at the defined repository path when a project is opened, a corresponding error message is output.

Menu Items

After having corrected the error and the library is properly available again, select the command 'Reload library' for this library. This way, reload the library without exiting the project.

3.8 Image Pool - Commands

3.8.1 Image Pool - Commands, General Information

Icon: 

An [image pool, page 61](#), is an object for managing images.

The "image pool" object is available in the "VI logic objects" folder in the "PLC objects" library. Use drag&drop to insert it below the "Application" node or in the global "General module" folder.

Alternatively, also use **Add ► Image pool** in the respective context menu; see the following figure.

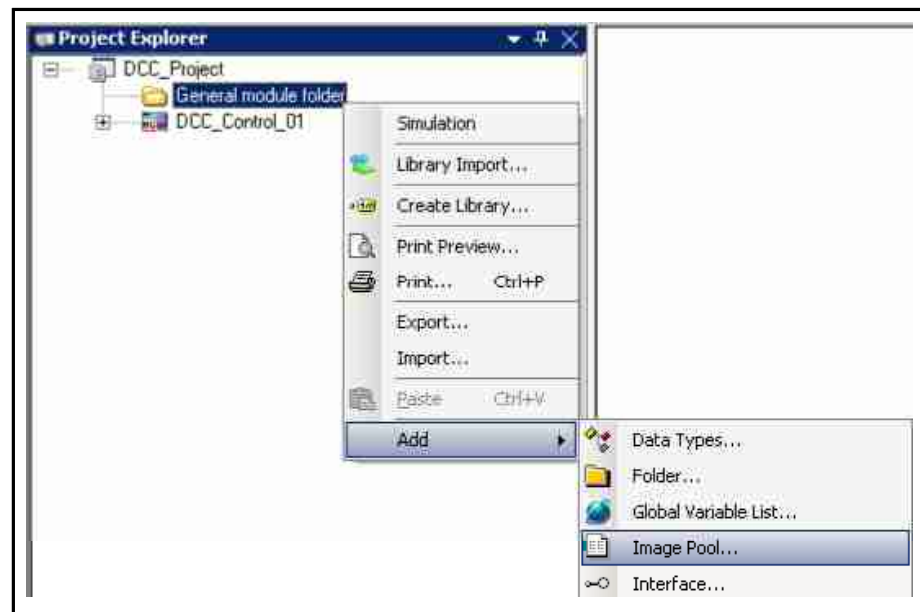


Fig.3-198: Adding an image pool into the project

The menu structure can be reconfigured using the Customize dialog.

Commands:

- [Generate global image pool, page 232](#)
- [Generate image pool, page 233](#)

3.8.2 Generating a Global Image Pool



Note that "GlobalImagePool" is automatically generated in the "General module" folder when the first use of static image files is defined in a project.

In the main menu click on **VI Logic Visualization ► Generate global image pool** to create the global image pool "GlobalImagePool" manually.

Alternatively, **Generate global image pool** is also available in the context menu when the mouse is moved to the working area of a visualization.

[image pool, page 61](#), contains more information on "image pools".

"GlobalImagePool" is a pool of static images that are used in visualizations.

3.8.3 Generating an Image Pool

Menu Items

in preparation

3.9 Recipe Manager - Commands

3.9.1 Recipe Manager - Commands, General Information

A **recipe manager**, an object assigned to an application, can be assigned 1..n recipe definitions (see [recipe manager](#), page 382).

Each of these **recipe definitions** is a table, the columns of which are "**recipes**" and the lines of which are "**variables**".

If a new recipe definition is assigned to the recipe manager, its initial set of columns comes from the recipe manager.

The following recipe manager commands are used to insert additional columns or delete columns or to insert new variables or delete variables.

The commands are available in the context menu when a recipe definition is being processed in the editor. In the basic configuration they are also available in the **VI Logic recipe definition** main menu.

If necessary, the menu structure can be modified in the dialog under **IndraWorks ▶ Tools ▶ Customize ▶ Commands ▶ Recipe Manager**.

Commands:

- [Add recipe](#), page 233
- [Add variable](#), page 233
- [Remove variable](#), page 234
- [Remove recipe](#), page 234

3.9.2 Add Recipe

Icon: 

This command is used to add a new column for a [recipe](#), page 384 to the recipe definition currently being processed.

Click on **VI Logic recipe definition ▶ Add recipe** to add a new recipe.

Alternatively, the command is also available in the context menu.

The "New Recipe" dialog opens to enter a name for the recipe; see the following figure.

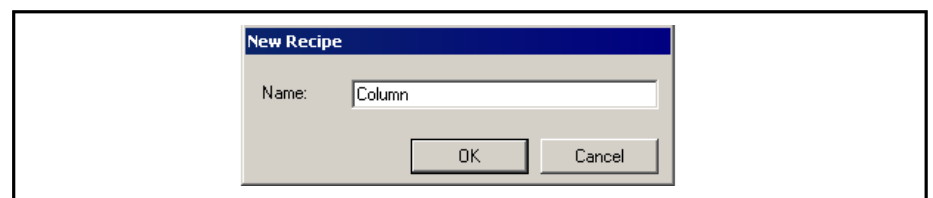



Fig. 3-199: New recipe dialog



A new recipe can also be created in online mode using a correspondingly configured visualization element; see "[Visualization elements - Properties](#)", page 451.

3.9.3 Add Variable

Icon: 

234/697

Bosch Rexroth AG

DOK-IWORKS-IL2GPRO*V12-AP01-EN-P

Rexroth IndraWorks 12VRS IndraLogic 2G PLC Programming System


Menu Items

Use "Add variable" to add a new line for a variable entry at the end of the recipe definition that is currently open.

Enter the desired variable in the Variable column; see also [recipe definition, page 384](#).

Click on **VI Logic recipe definition ► Add variable** to add a new variable. Alternatively, the command is also available in the context menu.

3.9.4 Remove Recipe

Icon: 


"Remove recipe" is used to delete a recipe from the currently open recipe definition, see also ["recipe definition", page 384](#).

Select one of the fields in the recipe column and click on **VI Logic recipe definition ► Remove recipe** to remove a recipe. Alternatively, the command is also available in the context menu. The column is removed.



Recipes can also be deleted in online mode using a correspondingly configured visualization element; see ["Visualization elements - Properties", page 451](#).

3.9.5 Remove Variables

Icon: 

Use "Remove variable" to delete one or more variables from the currently open recipe definition, see also ["recipe definition", page 384](#).

Highlight one of the fields in the line that describes the variable or highlight several lines while holding down the <Ctrl> key and click on **VI Logic recipe definition ► Remove variable** to remove variables.

Alternatively, the command is also available in the context menu. All selected lines are removed.

3.10 Object Management - Commands

3.10.1 Object Management - Commands, General Information

The commands for managing objects in a project are described in this section. By default, these commands are included in the context menu of the relevant object, suitable for the object.

The commands:

- [Add object, page 234](#)
- [Edit object, page 236](#)
- [Set active application, page 237](#).

3.10.2 Adding an Object

This command provides access to the existing object types.

The only object types listed are those that can be inserted at the current position in the Project Explorer.

For example, a data server object can only be inserted if "Application" is selected in the Project Explorer or an action can only be inserted if a function block or program is currently selected.

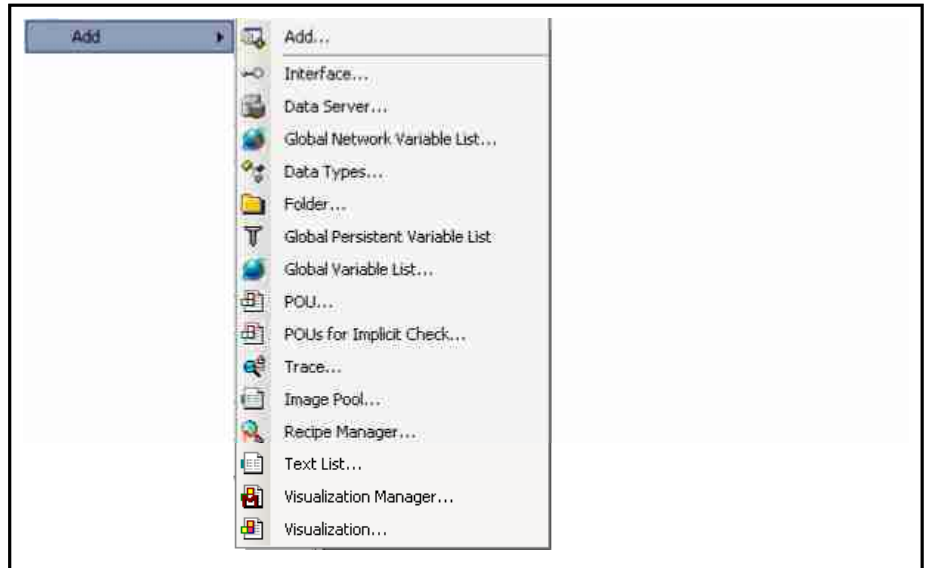


Fig.3-200: Adding an object

Select the desired object type using the respective context menu and in the following "Add object" dialog, define the **Name** and the respective settings for the object.

Please note the [Recommendations for names, page 505](#) to achieve a certain amount of consistency.

The settings that need to be made depend on the object type.

For more information, see the help pages for object type or the corresponding editor:

- [Action, page 51](#)
- [Application, page 66](#)
- [Library manager, page 367](#)
- [Image pool, page 61](#)
- [Data source, page 318](#)
- [Data server, page 71,](#)
- [DUT, page 43](#)
- [Property, page 46](#)
- [Device, page 63](#)
- [Global variables list, page 367](#)
- [Method, page 45](#)
- [Persistent variables, page 54](#)
- [POU, page 28](#)
- [POUs for implicit checks, page 63,](#)
- [Recipe definition, page 384](#)
- [Recipe manager, page 382](#)
- [Interface, page 49](#)
- [Step, page 405,](#)
- [Symbol configuration, page 306](#)
- [Task configuration, page 423](#)
- [Task, page 428](#)