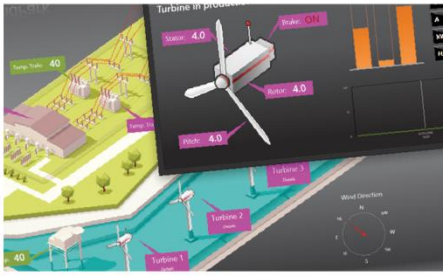
INDUSTRIAL AUTOMATION SOFTWARE

progea

Movicon.next™

Automation Platform.next generation



Movicon NExT

2.0 Platform NExT

Ver.3.4.268

Table of Contents

1. Platform.NExT Projects

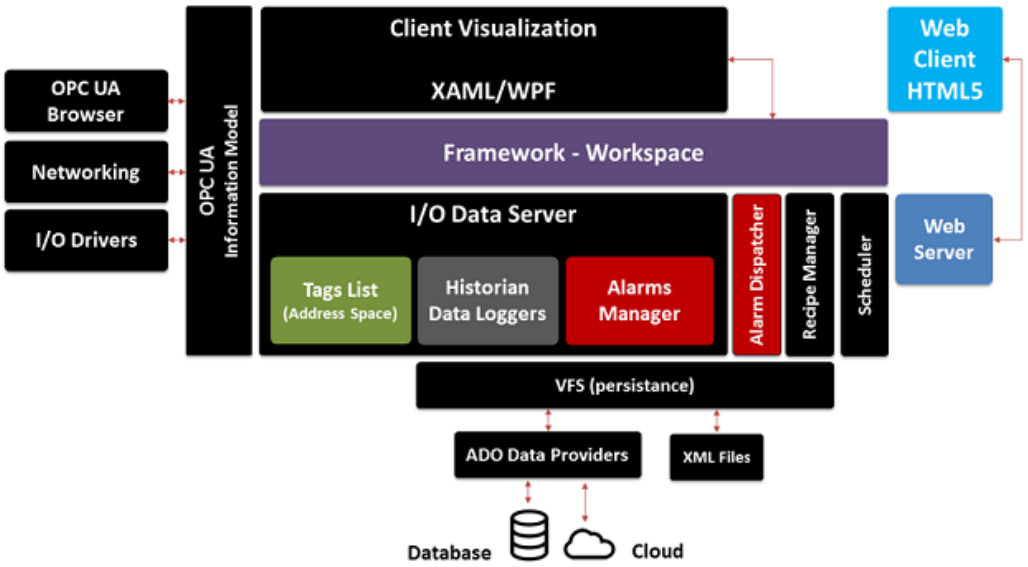
1.1. Project Architecture

Automation Platform.NExT uses a innovative and scalable architecture concept based on the most modern software technologies around today.
The architecture used a Client - Server model based on a .NET framework created specifically for industrial systems by Progea.
The projects are principally built one part Server (for managing communications, data, alarms and historical logging) and one part Client (for managing the Movicon.NExT graphical displays). The platform's information model is based on the OPC UA technology and totally transparent to users. Therefore a Server project provides its Variables for full disposal by means of publishing them as OPC UA tags in the Address Space enabling the Client project to access the list of these Variables according to its assigned access rights. The Platform.NExT modularity enables Movicon Clients to connect to third party OPC UA Servers without using Server side, or vice-versa enabling third party Client connection to the Platform.NEXT Server application.

The Platform NExT technology added value can be seen in the following areas:

- Configuration ambient completely based on WPF
- Data Server modular with OPC UA information-based model
- HMI graphical interface based totally on WPF with indispensable XAML graphics quality
- .NET technology with openness to third party integration
- Built-in Web Server with HTML5 technology
- Powerful and performing Historian based on SQL Server and open to DB Providers
- Alarm Management and Downtime Analysis
- Great variety of built-in visualization, analysis and control tools.

Below is a simplified schemata of the Movicon project architecture:



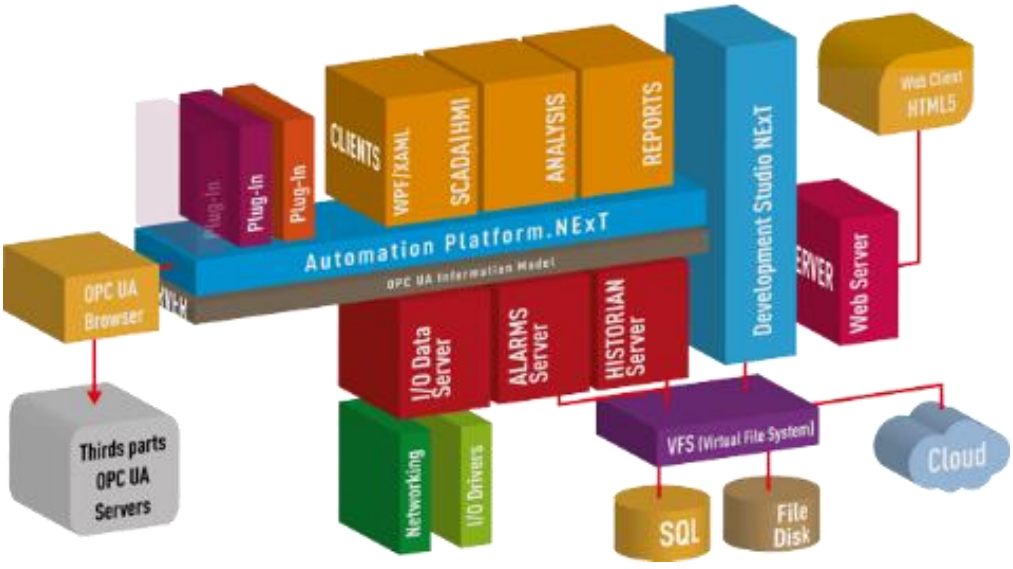
Simplified Movicon project architecture.

Platform.NExT uses the most innovative technologies such as "OPC UA", "WPF", "WCF".

Attention! As from the 3.2 version each saved project will not be retro-compatible with the previous versions. Therefore is will not be possible, for example, to open a project with the 3.1 version if previously created and saved with the program's 3.2 version.

1.2. The Platform.NExT Modules

Progea's Automation Platform.NExT is the most modern software technology available on the automation market. Its consists of an industrial framework developed by Progea that provides a suite of functional modules with ""plug-in" technology to give your application modularity and flexibility.
These platform's modules are available when installed with the setup procedures. If they are not installed, the platform will display only available modules within its context.
This technology enables platform expandability to integration of future function modules developed by Progea and third party modules as well.



This diagram shows the platform with its maoin functional modules and their collocation within the system's architecture.

Platform.NExT is a modular and scalable platform. The modules can be made available in the platform if installed using the normal installation procedures. This tutorial

presumes that all these function modules have been installed as necessary for supervision applications. These modules are:

- **Data Server,** to communicate with the field and Tag definitions
- **Movicon.NExT**, for graphical interface
- **Alarm Manage**r, for managing Alarms and Event Historicals
- **Historian,** for recording data.

1.3. Project Run Time Commands

The project can be launched using the "Start Runtime" button. This command will startup the Server side automatically starting up the Client side.
When activating the "Stop Runtime" command, both Cliend and Server will automatically be aborted by returning back to Editor mode.

If you wish to start up the Server side only without Client graphics, select the "I/O Data Server" resource and press the "Start Server" button. Once the Server starts up you will be able to execute a screen "Run Test". To start up the screen in run mode right click on the screen and then activate the "Run Test" command, or press the "F5" shortcut key with screen in focus.

Warning! When activating a screen "Run Test" command, the Command List will not be enabled and as a consequence it will not be possible to use the Change Page, Variable setting and other commands. However, it will be possible to monitor the Server variable values.

For further information on starting projects up using command line please refer to the chapter on "Start Runtime"

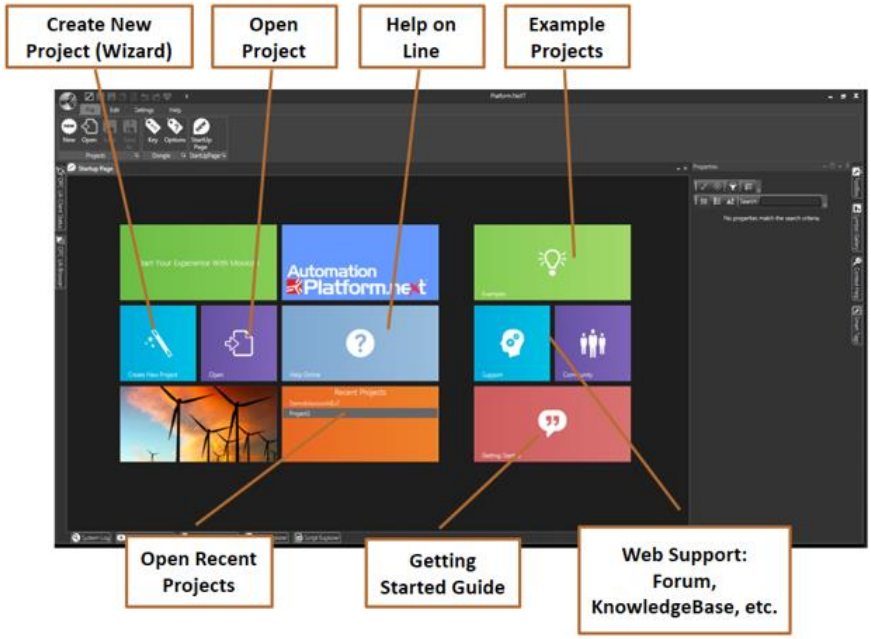
1.4. Starting as a Windows service

The Platform:NExT Data Server application fully supports the Windows Service. Therefore a Server project can be installed in the Services and started up as one according to the indications reported in the topic relating to "Starting up Platform.NExT projects".

2. Working with Projects

2.1. Startup Page

When not using the startup in runtime commands, the platform will startup showing the "startup Page" through which the user can open the project desired or used generic commands to access platform information.
The image below shows the possible operations provided by the Startup Page's Tile interface. These include access commands for opening or creating projects together with access to important and useful information with examples and Tutorial to help the design engineer in their work.



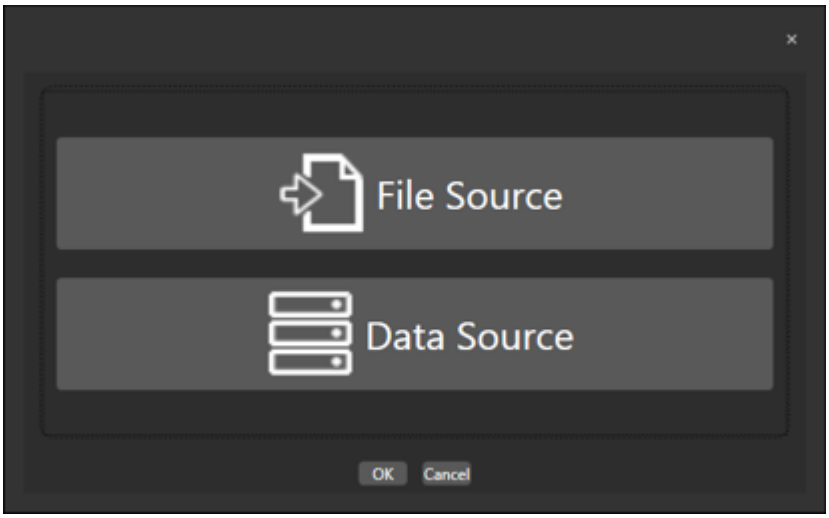
The Startup window appears at the Platform.NExT. platform startup

New projects can be created or previously created ones opened from the Startup Page. In addition to the Tile icons the "New" and "Open" commands are also available from the "File" menu. A list of the last project opened is also available in the "Startup Page" through which the recently used project can be opened directley with a simple mouse click.

Opening a Project

The "Open" command is used for opening an existing project. The projects can be based on XML files or Database and for this reason the system will request the existing project's data source by selecting one of the following:

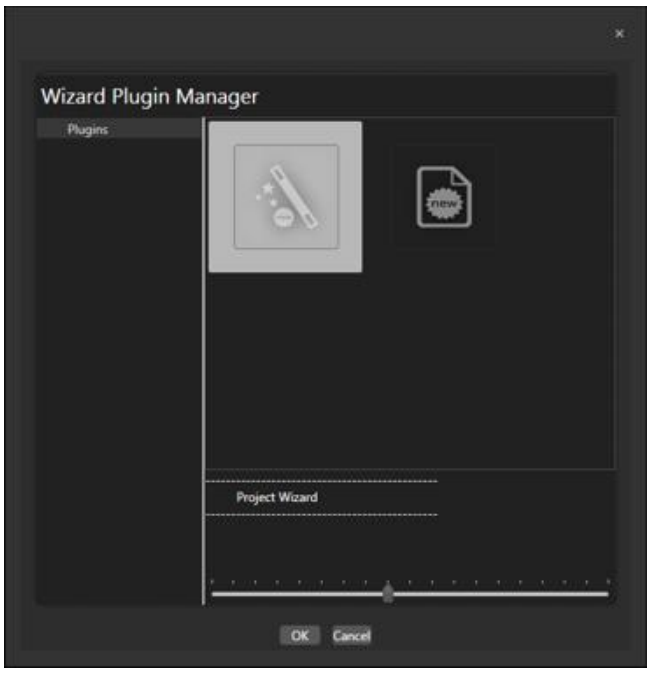
- File Source: Selection of normal XML project files by browsing the files in the PC folder
- Data Source: Selecton of a project residing on database by accessing the database using the desired DB connection string.



The project data will be saved encoded in a table structure. When choosing to save the project on file in xml format, a project folder will be created by Movicon within which the project information will be saved on files and sub-folders. The sub-folders will be displayed in a tree structure similar to the one used for project resources displayed in the Movicon "Project window". Once confirmed with 'OK' the project will open in the workspace.

Creating a new Project

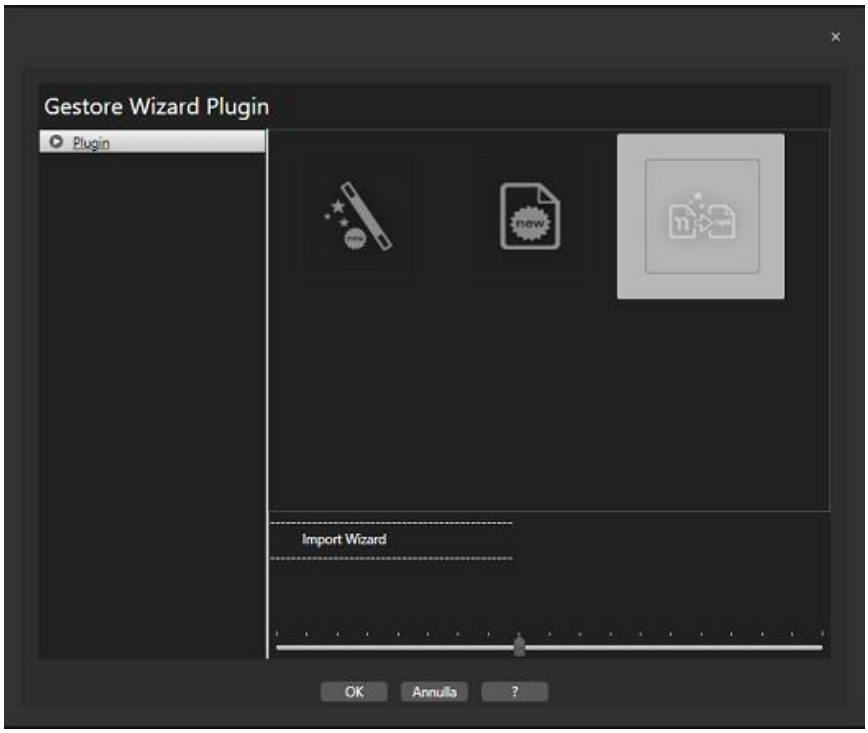
The "Create New" command is used for creating new projects using a system wizard to guide the user with a brief and simple step-by-step procedure. The first wizard step-by-step procedure starts by prompting user to select the project "model" they wish to create, for example user can choose from a simple empty project or a guided procedure to create a project containing Screens, I/O Data, Alarms and Historicals. Any project created can always be modified by adding, changing or deleting its features.
However and above all you must specify the name and path of the new project being created and whether is it to be saved as XML files or on database.



The Tutorial included with this guide will teach you how to use the wizard for creating a new project.

Import Wizard

The 'Import Wizard' command is used to import a Movicon 11 project by means of using a system wizard that guides the user in a few simple step-by-steps.
In order to import a project, you will need to select and open the 'Create New Project' window from startup page and then select the Import Wizard icon which is the last one displayed.

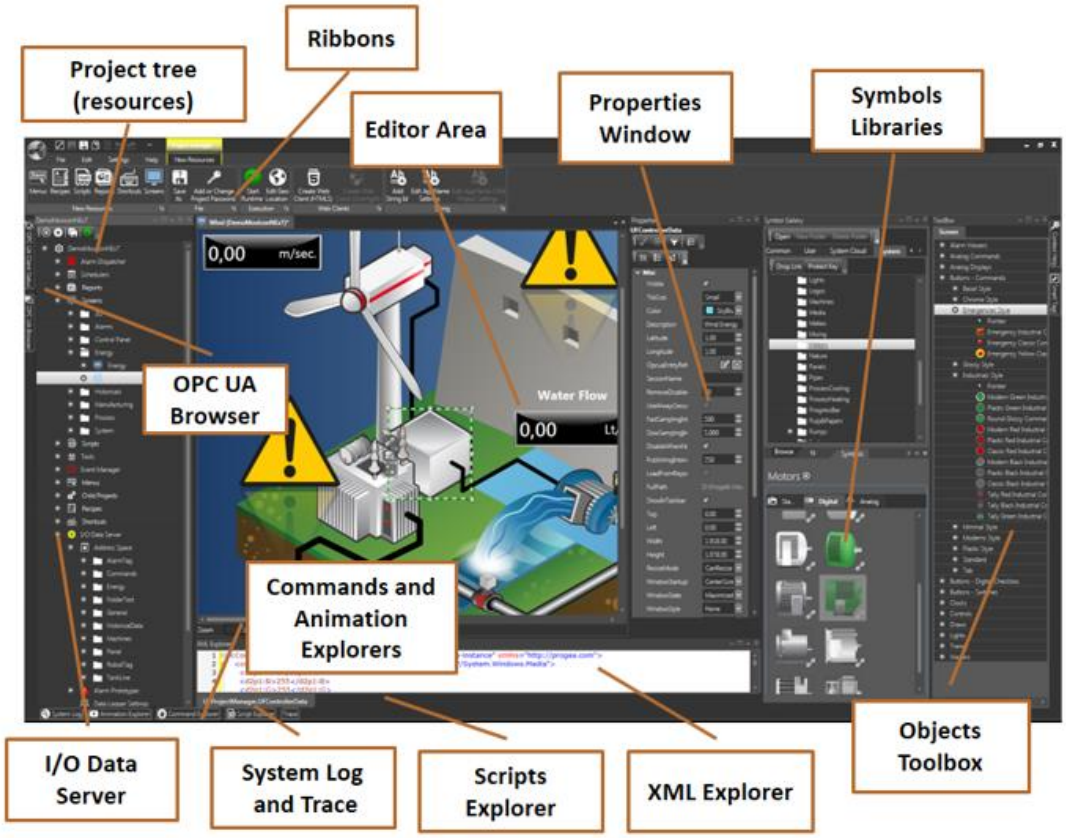


When confirming another screen will open through which you can select the project to be imported and its destination folder. Click 'Next' to proceed to the next screen which enables you to exclude those resources you don't need imported.

It is also possible to choose whether or not to enable log with 'Trace all successfully imported resources' or 'Trace all resources not imported successfully'. The log files are located in the project folder created by the wizard and report all information on what is or what not is supported. The log files are: "ImportedProjectLog" and "NotImportedProjectLog"

2.2. Workspace

The Automation Platform.NExT workspace has been designed according to the most modern ergonomics concepts and is completely based on WPF graphics technology. The Project Explorer Window is main point of the workspace and shows all the resources and module that are available in the platform in a tree structure as well as the Property Windows and a great variety of tools as described below.



Panoramic view of the Platform Next Workspace.

The user interface provides the design engineer the full use of Ribbon, to rapidly access commands which are also available when right clicking in the relevant context.
The "Docking View" techniques are used to dock or hide floating windows within the workspace. These can be done using these three modes:

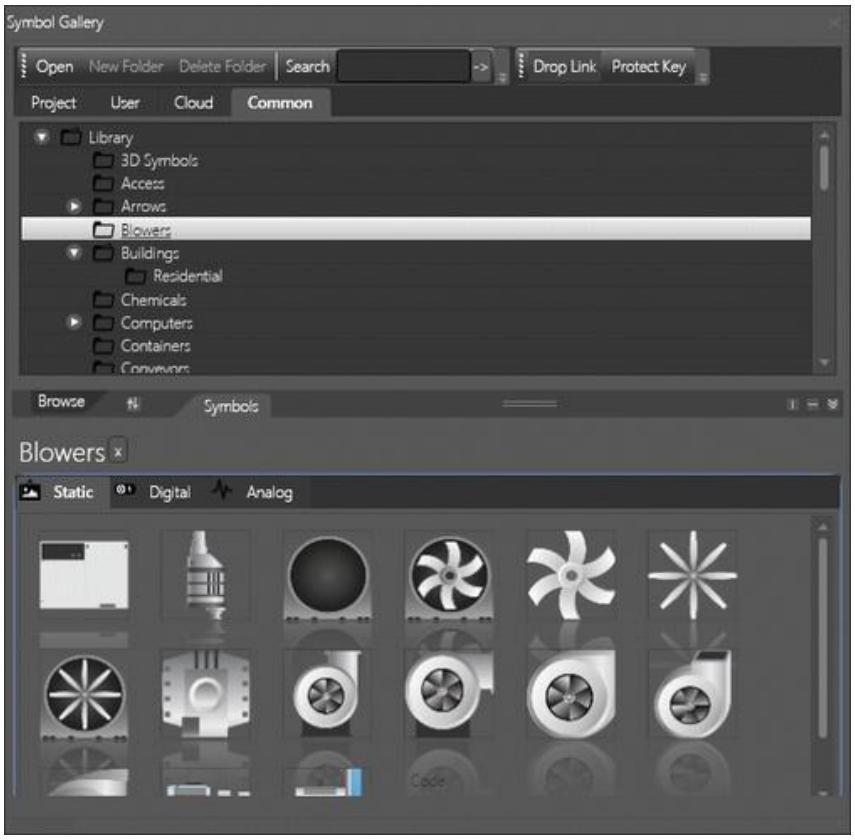
- **Floating** : this is set for default and represents the window as a Tab on the border that when clicked will activate the window.
- **Always visible** : using the window's dot button will make it permanently visible in the workspace.
- **Pop Up** : when dragging the window within the workspace will unanchor it from the border to be managed by the system as a model and pop-up window. To re-anchor a "pop-up" window simple double click its borders.

Reset Workspace

The workspace configurations are persistent which means that each system display window setting will be retained for the next startup.
The entire workspace configuration can also be reset back to its default settings. This is done by restarting Platform.NExT keeping the CTRL pressed throughout the whole duration of the startup procedure. This will reset all the workspace's configurations by returning all the windows back to their default position settings.

Symbol Gallery Window (Symbol Library)

The Movicon "Symbol Gallery Window" contains a great selection of rather complex symbols created with the "WPF" technology. This library, in addition to containing the symbols provided by Progea, can be enriched with the programmer's own customized symbols as well as adding completely new ones.
The "Symbol Gallery Window" is divided into two panes: the pane at the top shows the list of symbols by category and group displayed in a tree structure, the pane at the bottom shows a preview of the symbols from the group selected.

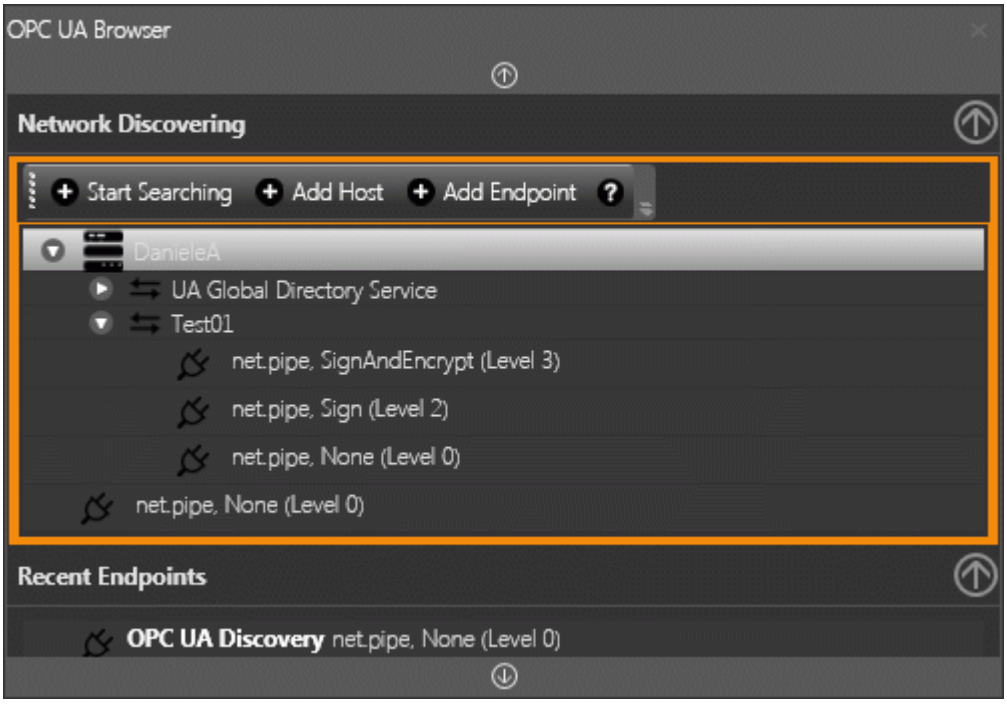


The "Symbol Gallery" window (Symbol Library).

At the top of the Symbol Gallery/Library there is a set of Tabs that identify the different Categories in which the Symbols are grouped. Each Category contains a tree structure listing all the symbols divided into different groups and sub-groups.

OPC UA Browser Window

The "OPC UA Browser" window is used for displaying the list of OPC UAs available.



The OPC UA Browser window.

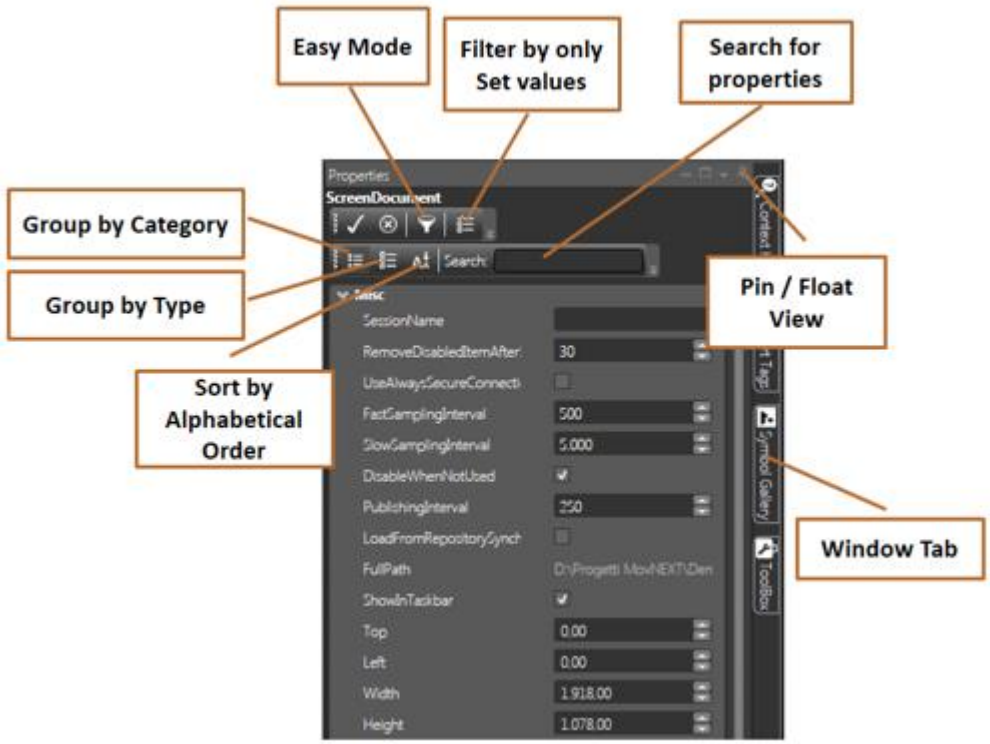
Movicon Ribbons

Movicon uses "Ribbons" within the developer user interface ambient instead of the usual Menus and ToolBars. Ribbons are panels that host buttons and icons, organized with a series of tabs to expose different sets of commands. Ribbons have been designed to make application functionality easier, more traceable and accessible using less mouse clicks compared to those interfaces based on menus. Some tabs which have been defined as "Contextual" appear only when certain objects or resources have been selected. These tabs expose only those features and properties belonging to the object currently selected. For example, when opening a screen within the workspace, the "Screen" tab will appear in the Movicon title bar enabling the Ribbon so that its screen and object contents can be configured.

Ribbon for grouping or ungrouping symbols.

2.3. Properties Window

The "Property Window" is essential to all projects for setting the characteristics of each PlatformNext resource and component.
Movicon, by virtu of its extreme simplicity, always uses the Properties Window for configuring its resources. The "Properties Window" automatically refreshes each time a project resource or object is selected, by displaying the selected object's inherent properties.

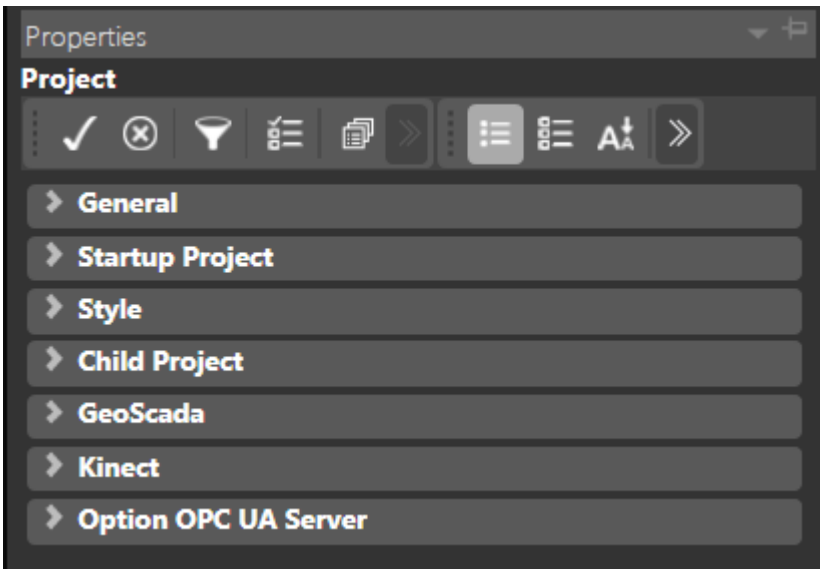


The Properties Window.

The Properties Window is opened in "Easy" mode at Movicon startup. This means that only some of the selected object or resource will display. These properties are generally those most commonly used. In order to display the complete list of properties, simply toggle to "Full" mode by clicking on the command represented by the toggle switch icon.

2.4. General Project Properties

The properties described below are not just specific to one individual resource but are common to all the project resources. Part of these properties, such as the Geo-Localization, are also displayed in other resources such as screens.



General

Below are listed the general project properties.

Project File
Indicates the complete project path (not editable).

Project Folder Path
Indicates the project folder path (not editable) .

Autoload Screen List
Used to set the list of screens to be automatically loaded at project startup.

Description
This is used to enter a brief project description.

Enable Backup
When enabling this property, a backup copy will be automatically saved with using the "Save All" command.

⚠ This property is not supported for projects saved on DataBase.

Max. Backup File
Sets the maximum number of backup files to be saved. Once this number has been reached, the project will start overwritting the oldest ones.

Startup Project

Below are described the properties which are used during the project startup.

Startup Page Type
Select the type of page to use at the project startup (Main Screen, Tile Page, Geo Page, Gallery Page).

Autoload Screen LIst
This is used to set the list screen that will automatically load at project startup.

Attention! This property has no effect on pop-up screens.

Startup Logics List
This is used to set a list of logic to be executed automatically at project startup.

Startup Scripts List
This is used to set a list of scripts to be executed automatically at project startup.

Startup Screen Name
This is used to set which screen is to open at project startup and requires that the 'Startup Page Type' be set to 'Main Screen'.

Style

The Style properties are those which determine the project's general aesthetic characteristics.

Project Theme
This property is used to set the project's internal theme graphics for designing the various window and toolbar styles.

Touch Control Type
This is used to set the touch technology type (if supported) to be used in the project.

Background Color
This property is used to set the background color of the "Main Page". This property is only used if the "Startup Page Type" has been set with "Main Page".

Child Project

This property group contains the general settings for Child projects.

Child Visible
Enabling this property will show the child project screen in the "Main Page".

Child Tile Size
Sets the size of the tile where the child project screens will be displayed in the "Main Page".

GeoScada

This group of properties concern the Geo-Localization feature.

GeoMap Type
This property is used to set the geographical map type to be used. This property can only be used when the "Startup Page Type" has been set with the "Geo Page" option.

Voice

This property group contains some general settings for the voice controls.

Enable Speech Command
This property is used to enable the Speech Command function.

Speech Confidence

Increases or decreases Speech Command recognition sensibility of the voice Controller.

OPC UA Server Option

These properties determine some of the project's data connection modes.

Remove Item Delay
Sets the delay time in milliseconds with which inactive OPC UA Item subscriptions will be removed.

Remove Item Num
This property is used to set the number of inactive OPC Items to be removed at each time interval.

Only Secure Connections
When enabling this, only server connections declared secure by specific certificates will be used.

Fast Sampling Interval
This property is used to set the update frequency of existing OPC Items in use. This parameter is passed to the Serve, and any existing driver, when loading the project and the variables go into use.

Slow Sampling Interval
This property is used to define the update frequency of existing OPC ITEM that are going out of use. This parameter is passed to the Server, and any existing driver, when the project is unloaded.

Publishing Interval
This value is calculated on the "Remove Item Delay" setting to indicate after how many seconds the project is to be unloaded and its variables put out of use.

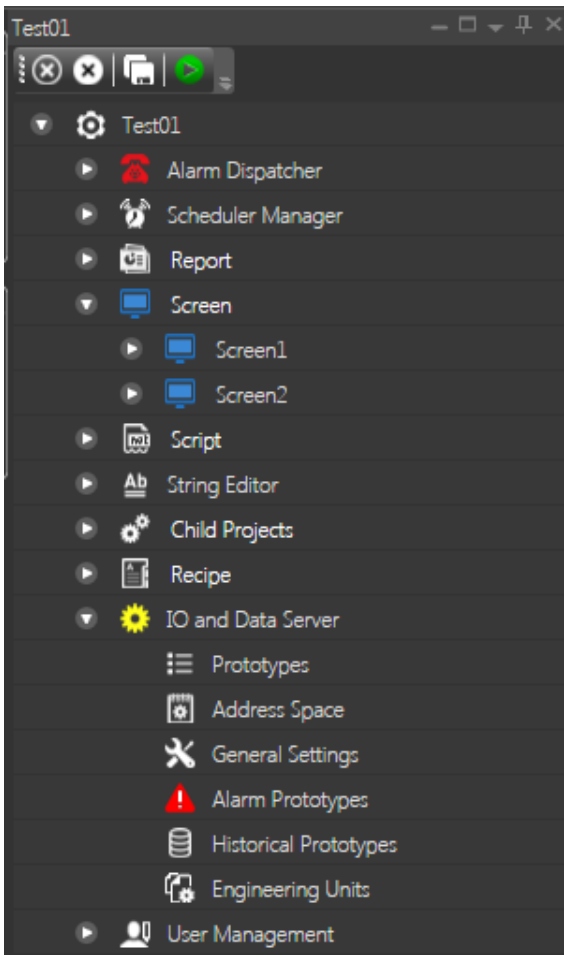
Disable Items When Not Used
When enabled sets the "Inactive Item" status when tags are not used in the project.

2.5. Project Structure

The data structure on disk will be differ according to the choice made between XML file or Database for creating a project.

- If you have chosen to use a project on Database, the project's data will be saved coded in a table structure for the database type chosen (eg. Ms SQL Server).
- If you have chosen to creat the project on XML file, the system will create a project folder and a series of sub-folders within which the various files containing project information will be saved.

In both cases, however the project's structure will be managed within the workspace area using the main project explorer window which presents the project's modules and resources in a tree structure as shown below.
Each individual module or resource is described in this guide.



Movicon Project Window

The Project Window (or Project Explorer Window) is where all the project resources are listed in a tree structure. This window is used for displaying all information relating to the resources which form the essence of the project itself.
The 'Project Explorer' window gathers all the Resource Groups into one tree structure. When selecting each single Resource Group or a group sub-component, the properties of the object in question will activate and show in the "Property Window" for modifying and setting as necessary. The commands found at the top of the Project Explorer window are described below:

- **Close**: closes the opened project. When more than one project is opened, the one that is currently active will be closed
- **Close All**: closes all opened projects
- **Save All**: saves all opened projects
- **Start Runtime**: starts opened project in Runtime. When more than one project is opened, the one that is currently active will be started up in runtime

2.6. Project BackUp

When enabling the project's "Enable Backup" property, a backup will automatically be made of the project every time it is saved. The Backup file will be saved in the project folder with a prefix indicating data and time of project save. The number of backup files

to be kept can be specified in the “MaxBackupCount” property. Once this number has been reached the oldest file will be deleted every time a new one is created.

Warning! Backup files will only be executed when the "Save all" command for saving all project files is used. A Backup will not be done when the "Save" command is used due to the fact that it only saves the resource being focuses at time of save.

The backup functionality is not managed when the project is saved on Database. In this case the functions provided by the DBMS can be used.

2.7. Files and Project Folders

The various information on Platform.NExT installation, folder and file usages are saved in the following paths:

Movicon installation folder:
C:\Program Files\Progea\Movicon.NExTx.x\

Toolbox Symbol Folder
C:\ProgramData\Progea\Movicon.NExTx.x\Toolbox\

Symbol Gallery (System)Folder:
C:\ProgramData\Progea\Movicon.NExT\Symbols\

Symbol Gallery (User) Folder:
C:\Users\UserName\AppData\Roaming\Progea\Movicon.NExT\Symbols\

Symbol Gallery (Project) Folder:
..\ProjectName\ProjectName\Symbols\

Symbol Gallery (Common) Folder:
C:\ProgramData\Progea\Movicon.NExT\CommonSymbols\

Saved Editor settings Folder:
C:\Users\UserName\AppData\Local\IsolatedStorage\

Saved License file Folder:
C:\ProgramData\Progea\Movicon.NExT\

Save Help file Folder:
C:\ProgramData\Progea\Movicon.NExT\HelpOnLine\

Saved Symbol Localization file Folder:
C:\ProgramData\Progea\Movicon.NExT\Cultures\

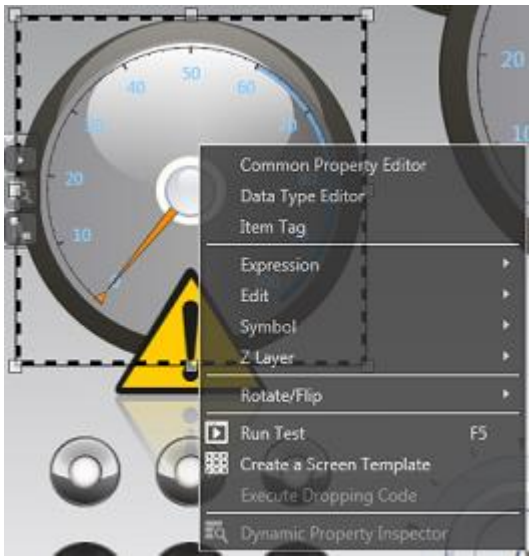
Save Screen Template file Folder:
C:\ProgramData\Progea\Movicon.NExT\NewScreenTypes\

Conventions:
<C:\> = name of disk where product has been installed
<UserName> = name of Windows user
<ProjectName> = Name assigned to the specific Platform.NexT project

2.8. Tag Browser Window

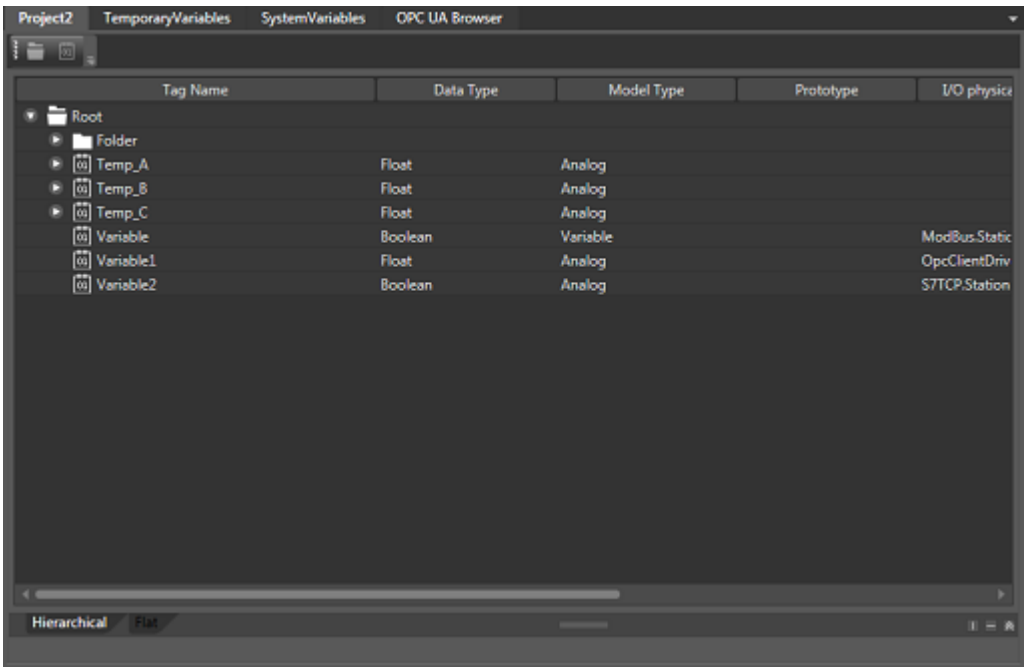
The "Tag Browser Window" is used to associate variables to objects or project resources. The "Tag Browser Window" can be opened in different modes when associating variables to objects:

- Double click on the object with the SHIFT key pressed down.
- The "Item Tag" from the contextual menu which appears when right clicking on the object.
- The "Item Tag" from the contextual menu which appears when clicking on the first button at the top of the object's adorner menu.
- The appropriate property in the object's Property Window.



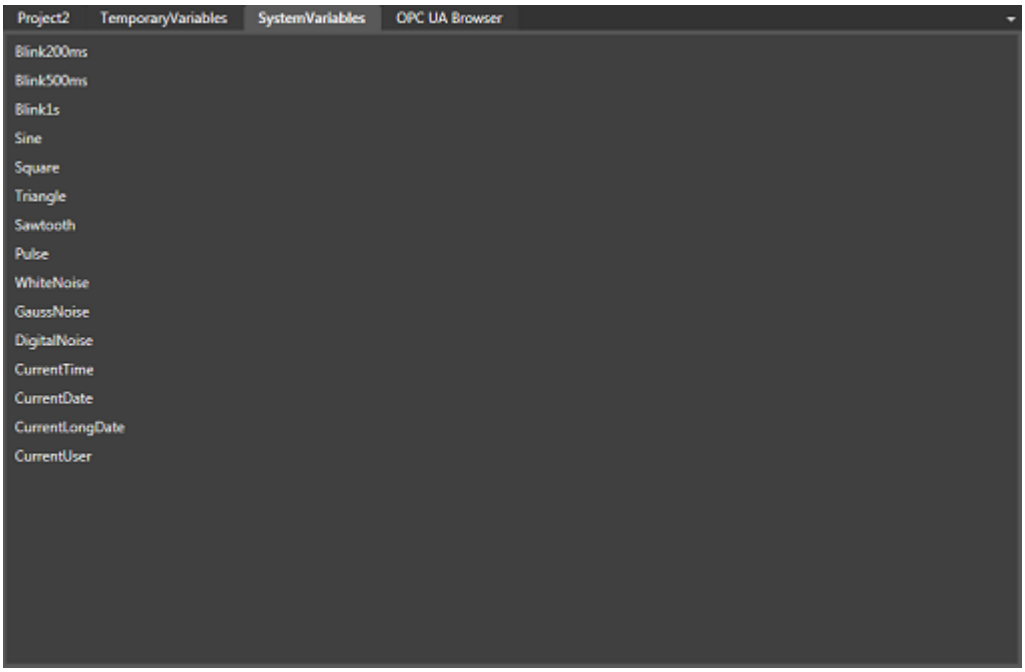
Tags are assigned to object by double clicking on or selecting the tag desired and then clicking OK in the window to confirm.
The "Tag Browser Window" opens with four Tabs to associate tags to the object from. Each Tab lists a different tag type:

- **Project:** The first tab shows the name of the project and the list of the Tags defined in the Server's address space and is used to insert the project's Server variables.



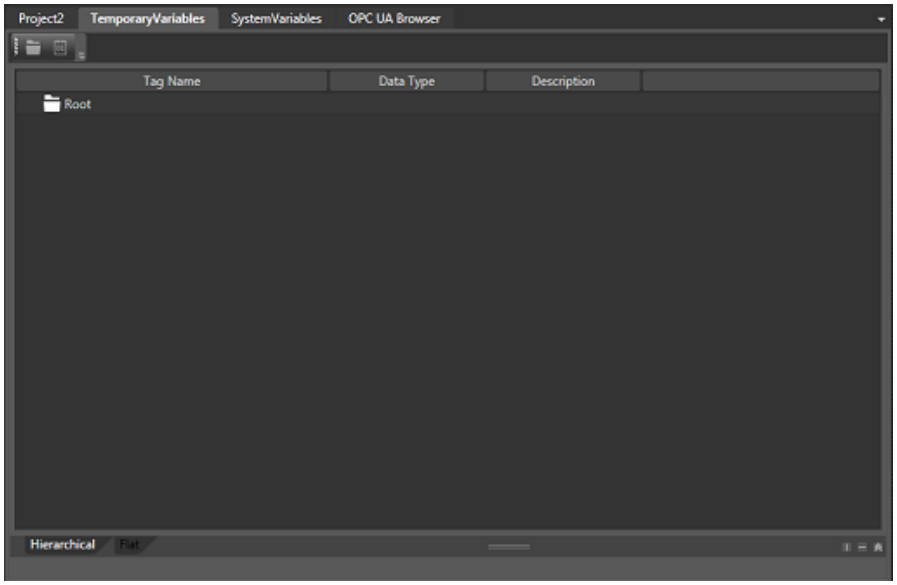
It is also possible to perform Tag editing operations within this context that include adding new tags or folders and modify, delete or move existing tags.

- **System Variables:** The second tab is dedicated to the System Variables and lists a series of predefined variables that are used to obtain system or simulation curve information (see "System Variables and Local Variables").



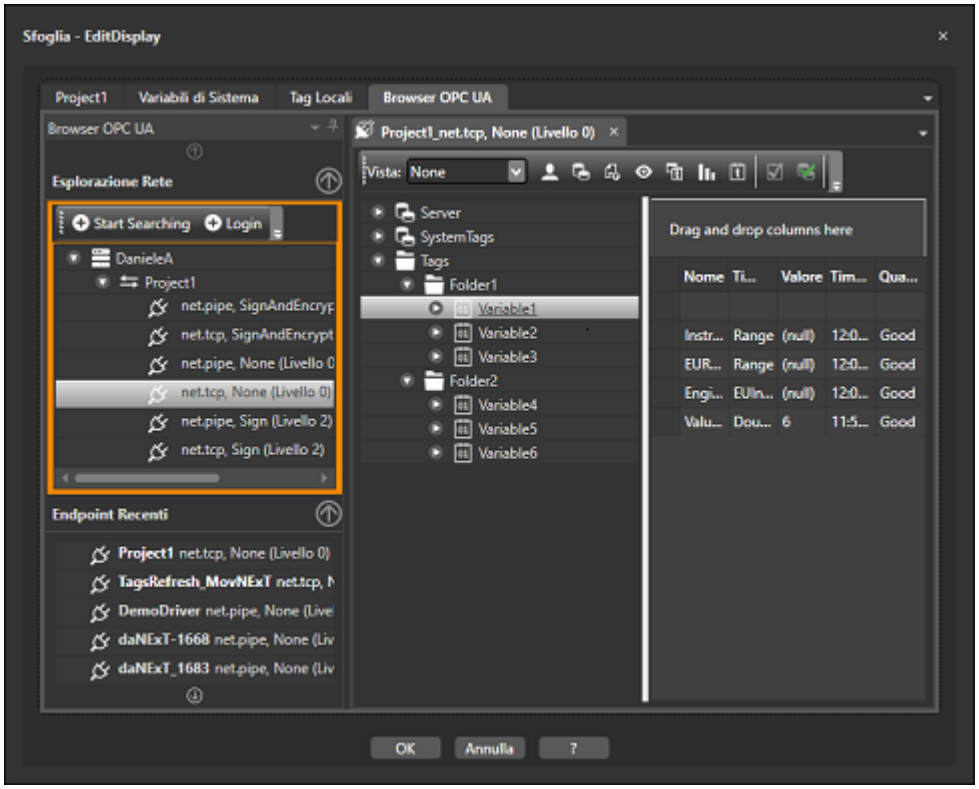
This tab can only be used for selecting tags and not for editing them.

- **Local Variables**: The third tab dedicated to Local Variables list the local variables defined in the project (see "System Variables and Local Variables). This tab is used to insert Local Variables in the project.



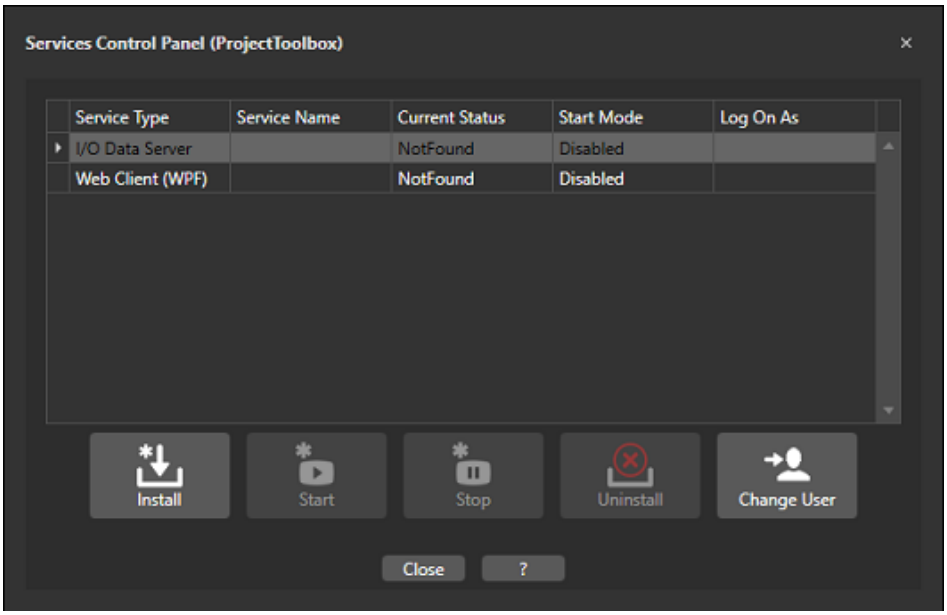
It is also possible to perform Tag editing operations within this context that include adding new tags or folders and modify, delete or move existing tags.

- **OPC UA Browser:** The third tab is dedicated to the "OPC UA Browser" and shows a list of all the Servers (including those of Movicon) which are running on the machine and in net. Once you have browsed the Server of interest, you can then select a tag from those being deployed by the Server in its address space (see "I/O Data Server").



2.9. Services Control Panel

In 'Services Control Panel' command is located in the 'New Resource - Execution' Ribbon and opens a window that is used to install and uninstall the project's services.



The Movicon processes that can be started up as Windows Service are:

- I/O Data Server: Project's IO Server
- Alarm Dispatcher: process to send alarm notifications by email, sms and other.
- Scheduler: process for executing time or event scheduled commands.
- Logic: process for executing the project's logic resource.
- Script: process for executing the project's Basic Script resources.
- Recipe: process for executing the project's Recipe resources.
- Web Client: process for executing the WebClient Server to connect using apps.

This window shows a series of status information (service name, status, etc.) for each service and commands for installing and running the various services.

Those services that don't need to be installed because not used by the project (eg. projects without recipes would not need to be installed with the recipe service for instance), are not displayed in the window shown above. Only those services that correspond to resources that have been defined in the project will be listed.

Each service can be installed using a specfic user. When a user is not specified, using the 'Change User' command before installing the service, the servcie will be installed with the machines "LocalSystem" default user.

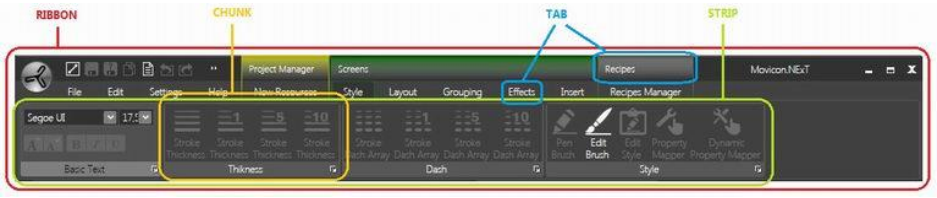
In order to use another user that is not the "LocalSystem" default user, they must have access rights to 'Access as Service' in the operating system. This setting can be

configured in the operating system by accessing the 'Control Panel - Administrative tools - Local Security Policy'. In the window that opens, select 'Local Policies - User Rights Assignment' and go to the 'Access as Service' item on the right and add the user desired to startup the process as service.

All the above listed services, except for the Recipe service and the Web Client (WPF) service, are dependent on the "I/O Data Server" service. This means that in order to startup these services, you will need to startup the "I/O Data Server" service as well.

2.10. Ribbons

The user interface of the development interface uses Ribbons instead of the classic Menus and toolbars.
The Ribbons expose controls relevant to the resource selected. These controls are displayed on panels using buttons and icons long with contextual tabs or cards that organize commands grouped according to functionality type.
These Ribbons are designed to make applications userfriendlier and more accessible and intuitive that requires less mouse clicks in respect to user interfaces based on menus.
Each tab contains a series of chunks. A chunk contains a set of commands that are related by functionality within a tab. Some tabs are contextual and appear only when the relating object or resource is selected. These tabs in fact expose functionalities and specific properties of the currently selected object. For instance, when you open a screen within the workspace, the 'Screen' tab will also display in the Platform.NExT title bar enabling the other Ribbon features to configure the screen or the objects it contains. The Screen ribbon tab will also contain another series of tabs containing chunks. Each chunk groups commands which are similar by functionality type.



Some of the Platform.NExT Ribbon Tabs are permanently available in the user interface, while others activate only when the resource of interest is opened within the workspace.

Hidden Ribbons

You can gain space within the workspace by using the "Minimize Ribbon" command, that is activated from the system menu's command icon list at the stop, and display the ribbon tabs and chunks only when needed when clicking on the Tab desired.

3. Unit Converter

3.1. Unit Converter

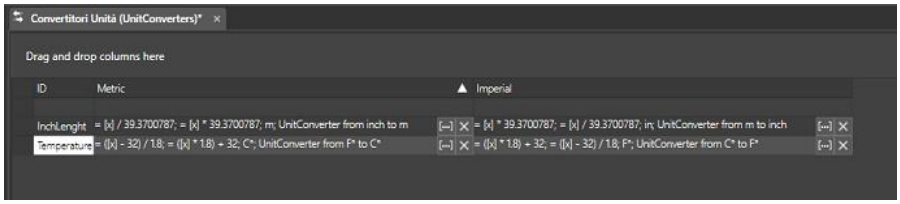
This resource permits you to convert numeric values from one unit of measure to another by applying the corresponding conversion factor.
Some of the most common conversions are: 1 meter = 39.3701 inches (39,3701 conversion factor) or 1 °C (Celsius) = 33.8 °F (Fahrenheit) (*1.8+32 conversion factor) and so forth.

To insert one Unit Converter in the project you must first populate the relevant conversion table which can be opened from the Unit Converter resource found in the Next resource tree.

First you must add a 'New Conversion System' and then add all the convertors you wish to use in it.

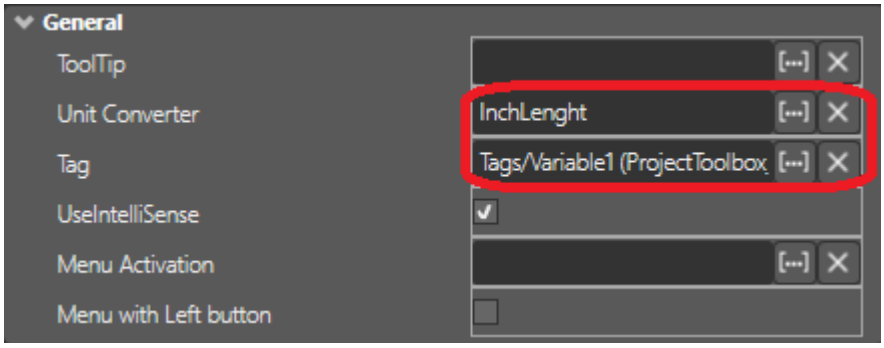
In the screenshot below you will see 2 converters (InchLength and Temperature) and 2 Conversion Systems (Metric and Imperial).

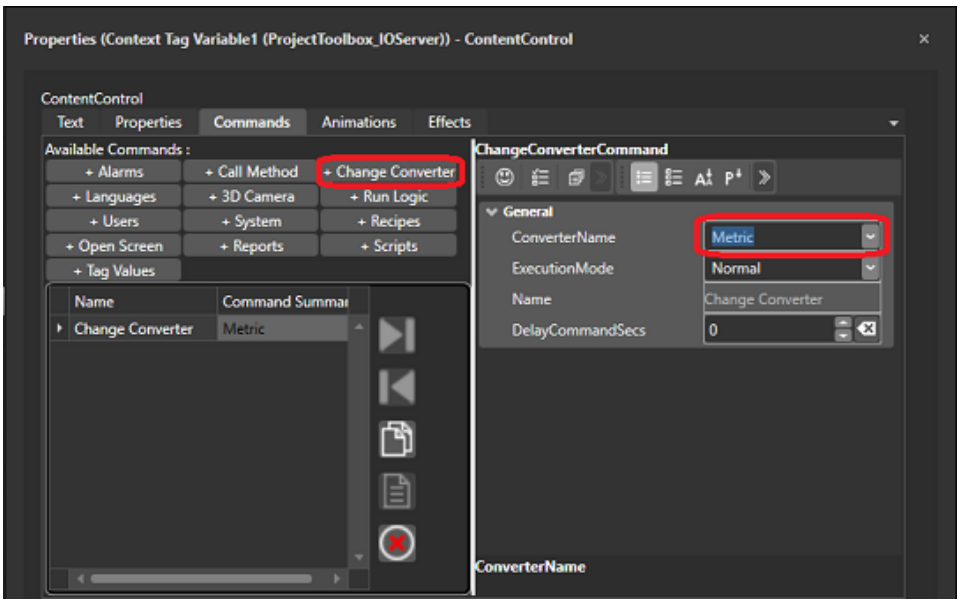
The [x] syntax used in the conversion expression refers to the tag associated to the graphical object to which the converter is applied.



Next you will need to:

- Insert the convertor in the properties of the graphical object that must apply the conversion to display numeric values (such as the 'display' object for example).
- Predispose a button to associate the Change Converter command to and which will be used to activate the desired Conversion System.

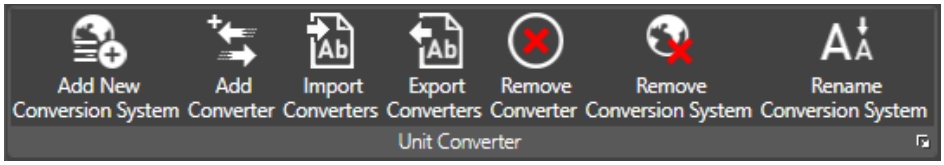




This screenshot shows how to predispose a 'Changer Converter' button

When the Unit Converter Resource is activated for the first time (or when the relative table is empty), a PopUp will appear requesting asking the user if they want to import a preconfigured list of converters which is installed with the product and found in the folder: C:\ProgramData\Progea\Movicon.NExT.3.1\Converters\UnitConverters.cvs .

Once this has been done, it will always possible to get new convertors simply by using the Import Converters procedure.



The presence of a Converter in a graphical object may be concomitant with an Expression: when the expression is displayed, make sure that it is evaluated after the converter has been applied.

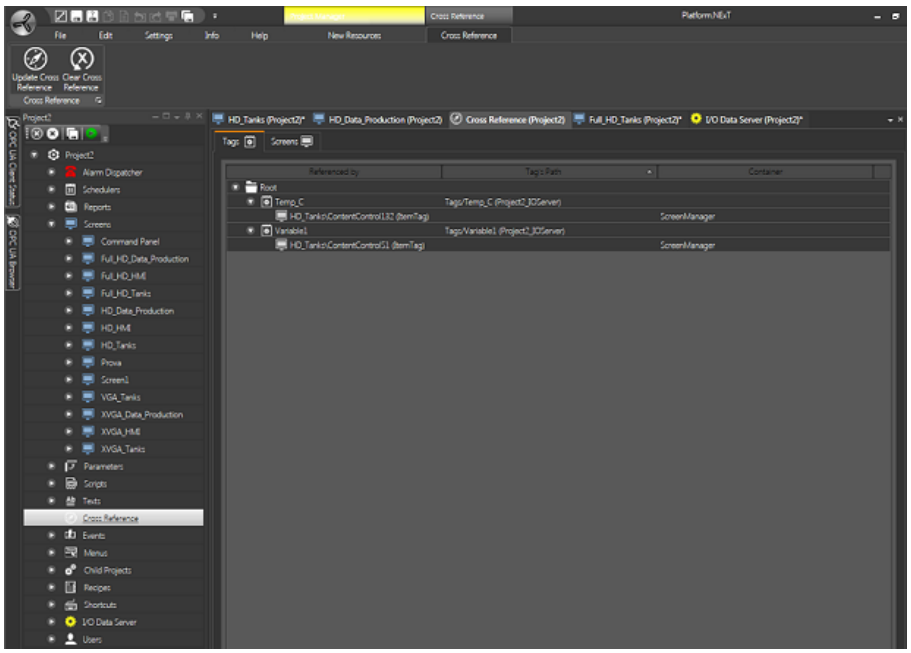
4. Cross Reference

The Cross Reference is an important Platform.NExT tool that is used for analysing the use of Tags in projects. This analysis is performed in the project and reports the exact object or resource each Tag is used in.

The Cross Reference appears as a project resource together with the other resources listed and displayed in the project's tree structure. When you access this resource with a double click, a Ribbon will display with two commands that are used to:

- Update Cross Reference
- Clean Cross Reference

This image shows an example to the Cross Reference resource:



Update Cross Reference

The Update Cross Reference ribbon command launches a project analysis to find out where the Tags are being used in the project. It then indicates the exact element, resource or object where each listed tag is used. When double clicking on the tag user element, the system will open the corresponding resource selecting the element or object to simplify things for the designer engineer.
For example, the Tag may be used in various resources:

- in objects within screens
- events (both for the tag that invokes the command or the tag in the associated command)
- script
- etc...

All the points where each individual tag is used in the project will be indicated for each tag.

Keep in mind that the command performs an analysis on the project's current situation. Therefore any modifications made afterwards will not be updated in the Cross Reference until the "Update Cross Reference" command is used again.

Variables which are detected in scripts will only be those used with direct notation. Other variable types, such as those retrieved with the GetVariable and Set Variable functions, will not be detected.

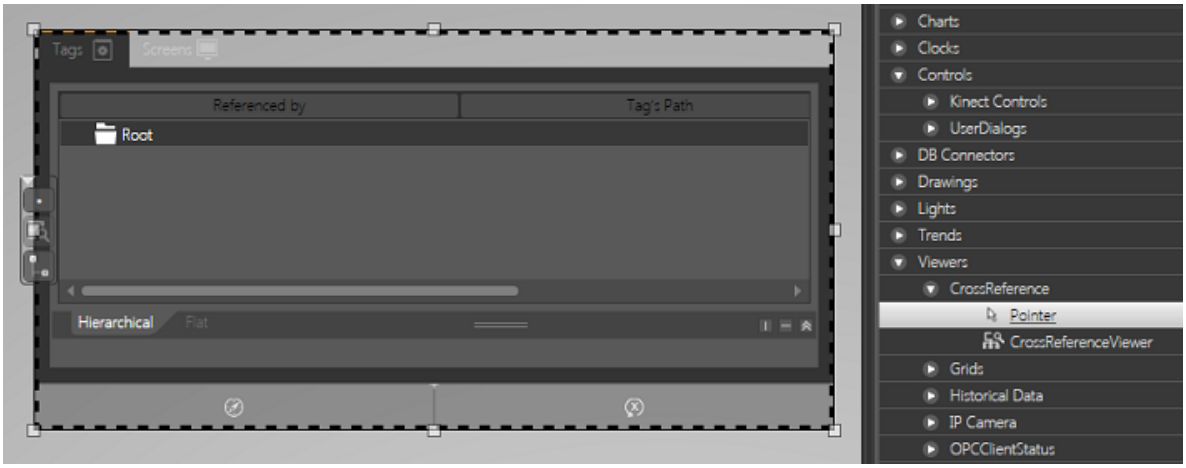
Clean Cross Reference

The 'Clean Cross Reference' ribbon command activates the reset and cancels the list shown in the Cross Reference.
Be reminded that the Cross Reference does not update automatically after every modification.

Cross Reference Screens

The Cross Reference has a 'Screens' tab that highlights how the project screen navigation has been set.
This tab shows the list of screens whereby each one shows the complete path and its associated controls through which the screen in question can be reached.

A library object has also been provided within the Symbol Library to display this information as well.



A window containing the "Cross Reference" object can be called in runtime using the "Shift+F3" combo keys.

Navigating variables or screens within the list's tree structure is not possible when using the control in the web client. Click directly on the node to expand it and not on the triangle on its left. It is also not possible to change the "Hierarchical"-"Flat" visualization with two cards at the bottom of the object.

5. Child Projects

Platform.NExT is used for structuring projects by decentralizing the resources to other projects (child) with dynamic relationships to enable project distribution. Platform.NExT offers a powerful and innovative feature to confront the new challenges in design engineering supervision systems. Child projects are normal Platform.NExT projects that although designed to function autonomously can be linked to parent projects to establish "Parent-Child" relationships due to which enables the Parent project to provide all Child project resources as if they were its own.

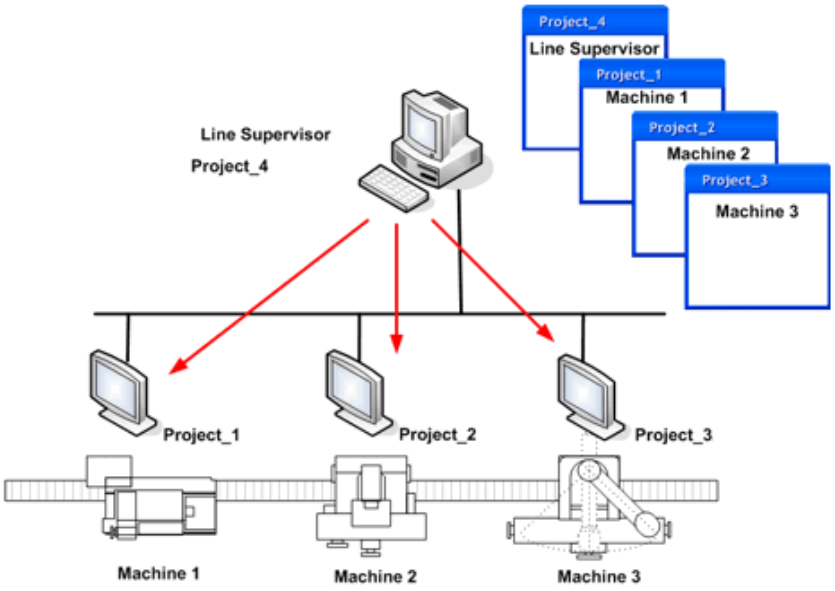
A project can be associated with several Child Projects. Likewise the Child project can also be a Parent project to several other Child projects thus creating a drop-down tree structure of projects.

This possibility opens the way towards many types of advantages. The main ones are:

Distributed Project Design Engineering

Project structured with Parent-Child relationships propose many advantages to those companies that rely on team work. In respect to those traditional technologies where several people work on the same project at once sharing out tasks, Platform.NExT instead offers the possibility to distribute such tasks over diverse and independent projects. This allows the Team Leader to be in charge of team members through his Parent project by using their Child projects which at the same time are completely independent.
The relationship between projects is established through Dynamic Linking and not englobing and therefore the Parent Project will be provided with all automatically updated Child project resources to enable team members to continue updating their projects autonomously.
The Parent Project will be able to contain all the Child project resources without having to distinguish between resource names or duplicate resource names because the name difference will be given by the child project path. Therefore, a VAR0001 for example can coexist in the Father project as well as in a Child project by being identified by their name and project path.

Note: This architecture provides the option to automatically startup child projects in runtime when the Parent project is started up in runtime. This option is called "Child Project Setting Options" property and is found in the Project properties.



Distributed Execution

The use of Parent-Child project relations is very handy for modular plant systems or machines where the plant is divided into different areas that are also independent from each other. In this case diverse projects can be created, one for each area, and then integrated into one unique Parent Project from which pages and variables of Child Projects can be accessed.

Example:
An automated production line composed of 3 independent machines: Each machine has its own project run locally on the machine's PLC. The machines are then integrated in the production line and connected to the General Supervisory PC.
The great advantages offered by Platform.NExT's capability to drastically save development time is to realize general supervision projects such as a "Parent Project" with the three machines' "Child Projects" residing in the Local PCs.
In this way, the Parent project can automatically get all the individual variables of the various child projects, to produce general summary layout screens. Screens from each of machines can then be accessed from the general layout screens residing in the Parent Project by simply opening the child project screens residing in the machines' local PCs. Not only does this offer the advantage of reducing project development time but also the advantage of having all machine modifications automatically available in the general supervisor as well.

Note: In the example using this architecture, child projects do not need to be preset to startup automatically in runtime when the Parent Project is started up in runtime (being autonomous projects) by enabling the option provided in the Project's "Child Project Setting Options" properties. However the child projects must be defined with the Server (supervisor) project's IP address in these settings.

5.1.1. Child Project Paths

The **"Child Projects"** can be retrieved by using the PC's local path where the "Parent Project" resides and a network path.
In cases where the project is local, it would be best to insert the folder containing the "Child Project" in the "Parent Project" folder, despite the fact that any path can be defined. However by doing this the "Parent Project's" search path will always relate to the "Parent Project's" path and problems finding absolute paths can be avoided when moving the entire "Parent Project" to a different path or another PC.
If the "Child Project" resides in the "My Documents" folder, it will always be searched for in relation to the "My Documents" folder of the user logged on to the operating system at that current moment.
If the "Child Project" is in another folder or in a network computer, the search path will be a physical search path. Therefore in this case it must be made explicit that the network PC folder must be shared.

It is always more convenient to insert the "Child Project" in the "Parent Project's" folder or subfolder whenever possible to avoid using absolute search paths.

5.1.2. Accessing Child Project Resources

The great advantage of using "Child Projects" is that the resources of both "Parent Project" and "Child Project" are easily accessible to both projects. For example, "Parent Projects" are able to call "Child Project" screens and vice-versa.
In fact during Runtime mode, screen pages from both projects can be displayed without it being obvious that they come from different projects.
Parent and Child Project access is established by simply using a slightly different syntax during project design mode. When using the browser window to select resources on the Parent Project side, it is normally possible see the Child Project resources as well. This will facilitate the selection of the resource desired. However, this is not possible on the "Child Project" side and, therefore, requires that the name of the desired resource be entered manually.

The syntax used for accessing the "Child Project" resources from "Parent Project" side is:

ChildProjectName\ResourceName

for example:

ChildProject\Screen 1
ChildProject\VAR00001
ChildProject\Basic Script 1
ChildProject\Menu 1

The syntax used for accessing the "Parent Project" resources from "Child Project" side is:

..\ResourceName

for example:

..\Screen 1
..\VAR00001

..\Basic Script 1
..\Menu 1

The different levels of nesting must also be taken into consideration. For example, in situations where the "Parent Project" has two child projects: "Child Project1" and "Child Project2", the syntax below must be used to access the "Child Project2" resources from the "Child Project1" side:

..\ProjectName\ResourceName

for example:

..\Child Project2\Screen 1
..\Child Project2\VAR00001
..\Child Project2\Basic Script 1
..\Child Project2\Menù 1

Another situation would be that a "Parent Project" has a "Child Project" which has its own "Child Project". Therefore, this would result as having "Parent Project", a "Child Project" and a "Child Project2". The syntax to use for accessing the "Child Project2" resources from the "Child Project" in this case will be:

ChildProjectName\ChildProject2Name\ResourceName

for example:

ChildProject\ChildProject2\Screen 1
ChildProject\ChildProject2\VAR00001
ChildProject\ChildProject2\Basic Script 1
ChildProject\ChildProject2\Menu 1

The syntax to use for accessing the "Parent Project" resources from the "Child Project2" side is:

..\..\ResourceName

for example:

..\..\Screen 1
..\..\VAR00001
..\..\Basic Script 1
..\..\Menu 1

Caution! If local screen variables have been defined for the screen being used, the syntax to use for accessing the parent project variables may be different. If the parent project variable has the same name of the local variable, the following syntax should be used:

..\..\<variable name>

The "..\" suffix is used for returning back from pointing to a hierarchy of variables. The hierarchy is as follows:

Local Screen Variables -> project Variables -> parent project variables

It is also possible to use the variables between Parent and Child in the objects' basic expressions. However. it is important to take into consideration those local variables that are not supported in basic scripts. In this case, when you insert a basic expression in an object in the Child Project, the syntax to use is:

[VAR00001] + [VAR00002] -> the Child project variables are loaded
[..\VAR00001] + [..\VAR00002] -> the Parent project variables are loaded
[..\..\VAR00001] + [..\..\VAR00002] -> no variables are loaded due to reference error

INDUSTRIAL AUTOMATION SOFTWARE

progea

Progea Srl
Via D'Annunzio, 295
I-41123 Modena
info@progea.com
Tel +39 059 451060

Progea International SA
via Sottobisio, 28
6828 Balerna (CH)
international@progea.com
Tel +41 91 96 76 610

Progea Deutschland GmbH
Marie-Curie Str., 12
D-78048 VS Villingen
info@progea.de
Tel +49 (0)7721 99838 0

Progea North America Corp.
2380 State Road 44, Suite C
Oshkosh, WI 54904
info@progea.us
Tel. +1 (888) 305-2999