API documentation

Contents

Introduction

Bentley’s ProStructures CONNECT Edition is a very complex and flexible software that allows user to do almost anything they want. However, in certain situations, automating certain tasks or creating a simple extension can drastically increase the user’s productivity. Fortunately, this is possible with ProStructures. Any user with basic experience in programming can create their own plugin to suit his or her needs. This document outlines the necessary steps that need to be completed by the user to use the ProStructures API to create his custom plugin. A sample plugin will also be described as an example.

Creating a plugin

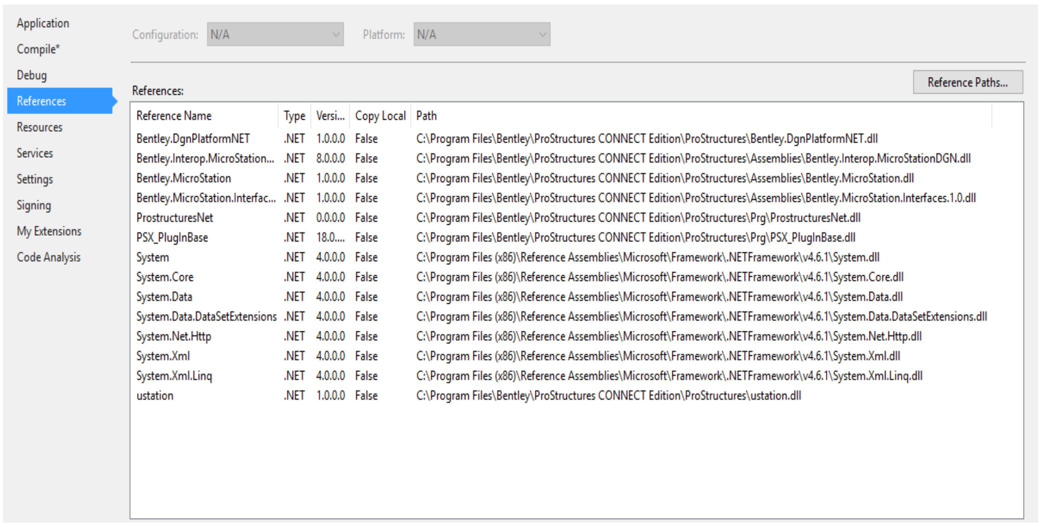
Initial setup

Before anything else, the user needs to create a project. In ProStructures, there is an API for ProConcrete written in C# and one for ProSteel written in vb.net. The following instructions are for the ProSteel API and assume that the user has Visual Studio 2015 installed on his computer.

1) Create a new Visual Basic Class Library project. Your project will be built as a DLL.

2) Set the properties for your project. It is important that those properties are set correctly for your project to work.

   - **Application tab:** Write a name for your assembly. Choose a name that is short and simple if possible as this will make it easier to write the command for the plugin. Once this is done, set the target to .NET framework 4.6.1 to make sure that your project uses the same version as the API. Finally, before going to the next tab, make sure that the application type is set to Class library.
   - **Compile tab:** Set the target CPU to x64.
   - **Debug tab:** If you want to debug your project, check the “Start external program” radio button and set the path to your ProStructures CONNECT Edition application file. By default, this is located under C:\Program Files\Bentley\ProStructures CONNECT Edition\ProStructures.
   - **References tab:** Add the following references to your project (next page):



Configuration: N/A Platform: N/A

References:

Reference Name	Type	Versi...	Copy Local	Path
Bentley.DgnPlatformNET	.NET	1.0.0.0	False	C:\Program Files\Bentley\ProStructures CONNECT Edition\ProStructures\Bentley.DgnPlatformNET.dll
Bentley.Interop.MicroStation...	.NET	8.0.0.0	False	C:\Program Files\Bentley\ProStructures CONNECT Edition\ProStructures\Assemblies\Bentley.Interop.MicroStationDGN.dll
Bentley.MicroStation	.NET	1.0.0.0	False	C:\Program Files\Bentley\ProStructures CONNECT Edition\ProStructures\Assemblies\Bentley.MicroStation.dll
Bentley.MicroStation.Interfac...	.NET	1.0.0.0	False	C:\Program Files\Bentley\ProStructures CONNECT Edition\ProStructures\Assemblies\Bentley.MicroStation.Interfaces.1.0.dll
ProstructuresNet	.NET	0.0.0.0	False	C:\Program Files\Bentley\ProStructures CONNECT Edition\ProStructures\Prg\ProstructuresNet.dll
PSX_PlugInBase	.NET	18.0....	False	C:\Program Files\Bentley\ProStructures CONNECT Edition\ProStructures\Prg\PSX_PlugInBase.dll
System	.NET	4.0.0.0	False	C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.6.1\System.dll
System.Core	.NET	4.0.0.0	False	C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.6.1\System.Core.dll
System.Data	.NET	4.0.0.0	False	C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.6.1\System.Data.dll
System.Data.DataSetExtensions	.NET	4.0.0.0	False	C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.6.1\System.Data.DataSetExtensions.dll
System.Net.Http	.NET	4.0.0.0	False	C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.6.1\System.Net.Http.dll
System.Xml	.NET	4.0.0.0	False	C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.6.1\System.Xml.dll
System.Xml.Linq	.NET	4.0.0.0	False	C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.6.1\System.Xml.Linq.dll
ustation	.NET	1.0.0.0	False	C:\Program Files\Bentley\ProStructures CONNECT Edition\ProStructures\ustation.dll

Those libraries are needed for your plugin.

After the initial setup

Once the initial setup is completed, you will need to create three new files in your project.

Xml file

First, you will need to create a xml file. This xml file is useful for mapping the command to a function that you will create in your solution. If this is done properly, this function will be called when you type the command in the ProStructures application.

AddInMain.vb

You will also need to create a vb.net class file that implements the Bentley.MstnPlatformNET. You will provide an id and this id will be used to load your plugin in the application.

UserConnection.vb

Finally, you will need to create a vb.net class file with this exact name. Once this is done, define the four following functions in the UserConnection class:

Public sub Build(ByRef handle as String)

Public sub Clean(ByRef handle as String)

Public sub Draw(ByRef handle as String)

Public sub Edit(ByRef handle as String)

where handle is the connection's id in the form of a String. Those four functions will be called by the API when a user changes the connection or changes the properties of an existing connection in the PsProperties.

.vbproj file

You will need to add the following instructions to the .vbproj file of your solution to make your xml command file work:

```
<ItemGroup>
   <EmbeddedResource Include="commands.xml">
     <LogicalName>CommandTable.xml</LogicalName>
     <SubType>Designer</SubType>
   </EmbeddedResource>
 </ItemGroup>
```

Add those instructions in the <ItemGroup> tag of the file.

Those are all the necessary steps that need to be followed for your plugin to work. You can structure the rest of your solution in any way that suits you.

Note: The steps described above can be confusing. We suggest looking at the sample project provided if some questions arise. This sample is a basic project that can be loaded in the application. You could simply start from this project if you want.

Stiffener Distribution

In this section, we will look at a sample plugin that was created using the ProStructures API from a technical standpoint. The functions that directly use the API will be explained.

Technical overview

Commands.xml

This xml file maps the command to a function in the solution. For this project, the only command is CREATE.

KeyInCommands.vb

Contains a friend class with a single shared sub. This shared sub is mapped to the xml file and is called when the user inputs the create command in the application.

UserConnection.vb

This class acts as a controller for the solution. It encapsulates any form of communication between the form and the connection parameters. It also contains the four most important functions for the plugin: Build, Clean, Draw and Edit. As stated earlier, these four functions are called by the API when the user modifies an existing connection.

Build: The build function is called when the user changes the properties of a connection in the PsProperties. The build function receives the id of this connection as a parameter and calls the BuildI function. The BuildI function is the function responsible for building the connection. It reads the data from the existing connection, retrieves the support shape's info and updates the stiffeners' insertion points. Once this is done, the old stiffeners are removed from the connection. This is important because otherwise, the new plates would be inserted on top of the old stiffeners. The new stiffeners are then appended to the connection and the connection's parameters are written to the physical connection. It is important to append your created entities to your connection. If you forget this, your entities will not move properly with the support shape if the support shape moves.

Draw: Responsible for drawing entities in the application. Usually, a basic implementation is required.

Edit: This function is called when the user right clicks on an existing connection and clicks the "Change Connection" option. The form for the plugin will open and the user will be able to change some parameters for the connection.

Clean: Called when the user right clicks on an existing connection and clicks the "Delete Connection" option. It receives the id of this connection and calls the CleanI function. The CleanI function removes every entity that was previously appended to the connection, sets the isBuilt flag of the connection to false and erases the connection from the application.

AddInMain.vb

Implements the base AddIn class. This class is identical for every plugin except for the id.

TemplateHandler.vb

This shared class is responsible for writing to sve, wndcfg and tpl files and reading from them. Contains multiple API function calls related to sve files, wndcfg files and tpl files. The sve and wndcfg files are saved under:

C:\ProgramData\Bentley\ProStructures CONNECT Edition\Configuration\ProStructures\Localised\USA_Canada\Temp

and the tpl files are saved under:

C:\ProgramData\Bentley\ProStructures CONNECT Edition\Configuration\ProStructures\Localised\USA_Canada\Varia

ConnectionHandler

This class is responsible for writing and reading to/from a connection. The parameters related to a connection need to be saved in the application because otherwise, it would not be possible to modify it. When the user decides to change a connection, it calls the Edit or Build function of our plugin depending on how he changed it. Before rebuilding the connection, our plugin needs to read the data from the existing connection.

InTransaction.vb

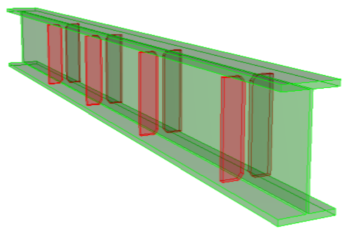
This is a helper class for writing and reading to/from a connection. When you want to access or modify the data of a connection, you need to open it in read or write mode, otherwise the application crashes. This class was useful for this but it is not necessary.

Note: We strongly suggest taking a detailed look at the Stiffener Distribution plugin supplied with this document. It contains detailed comments and the code related to the API is clearly explained and highlighted. The functions or blocks of codes that directly use the API are clearly indicated in the solution with a "------------------- API ------------------" tag.

We also suggest looking at the ProStructuresNET.chm file. This is the documentation for the API. It is located under: C:\Program Files\Bentley\ProStructures CONNECT Edition\ProStructures\Prg.

Feature Description

In this section, we will cover the stiffener distribution plugin's purpose along with its limitations. Some procedures will also be outlined to give the user an idea on how to use it.



Feature summary/GUI

The stiffener distribution tool is a very flexible feature that allows users to place multiple stiffeners on a certain steel shapes. The shapes supported by this feature are the ones with the following section types: He, I, U.

Options Tab

Origin of the distribution: The stiffeners can be distributed from four different points: from the start of the shape, from the end of the shape, from the middle of the shape and from a user selected point (this can be any point on the shape).

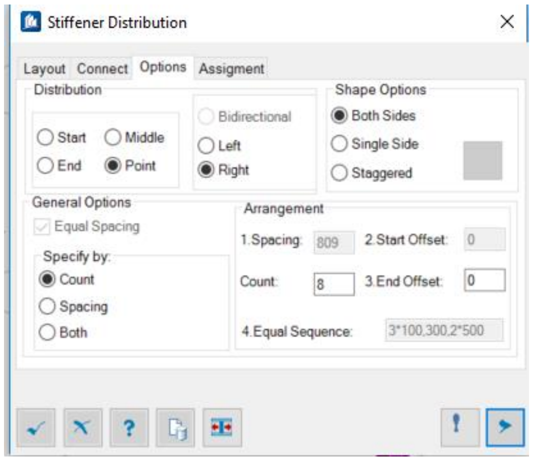
Direction: If the stiffeners are distributed from the start or from the end, they will always be distributed towards the opposite end of the shape. However, if they are distributed from the middle or from a user selected point, they can be distributed towards the left, the right or in both directions. If the user chooses to distribute from the middle in both directions, the stiffeners will be equally distributed in both directions. However, if the user chooses to distribute them from a point that is not centered, the stiffeners will not be equally distributed in both directions. The program will distribute will distribute the most stiffeners that it can towards the shorter side and the rest of the stiffeners will be distributed in the other direction.

Placement options: There are three different placement options that the user can choose from: distribution on both sides of the shape, distribution on one side of the shape and distribution in a staggered manner. If the user chooses to distribute on one side only, a button is enabled which allows him to switch the side on which the stiffeners will be placed. Similarly, if the user chooses to distribute in a staggered manner, the button allows the user to choose the side where the first stiffener will be inserted.

Offsets: The user can specify a start and end offset for his distribution. No stiffeners will be placed in the areas located between an offset and a shape extremity.

Distribution method: The user can choose to distribute the stiffeners by specifying a count, a spacing or both. If he chooses the specification by count, the program will calculate the exact spacing that is required between each stiffener to fit this exact count and the first and last stiffeners will be placed on an offset (limits). If the user chooses the specification by spacing, the program will calculate the number of stiffeners that can fit on the support shape. The first and last stiffeners will not necessarily be placed on an offset. Finally, if the user chooses the specification by both, he will need to enter a custom sequence or a series of custom sequences depending on whether the equal spacing checkbox is checked. This option allows the user to distribute his stiffeners in a more flexible way.

Restrictions: There are certain restrictions related to the distribution. First, if the user chooses to distribute stiffeners by specifying a count and selects a point as the origin of the distribution, he will not be able to distribute the stiffeners in both directions. Also, if the user chooses to distribute the stiffeners by both spacing and count and he unchecks the equal spacing checkbox, he will not be able to distribute his stiffeners in both directions. Finally, if the user chooses to distribute his stiffeners on a support shape with a cross section of type U, he will not be able to specify a placement option. These shapes can only have stiffeners on one side.



Distribution: Start, [✓ Point], Middle, End

Bidirectional, Left, [✓ Right]

Shape Options: [✓ Both Sides], Single Side, Staggered

General Options: [✓ Equal Spacing]

Specify by: [✓ Count], Spacing, Both

Arrangement

1.Spacing: 809

2.Start Offset: 0

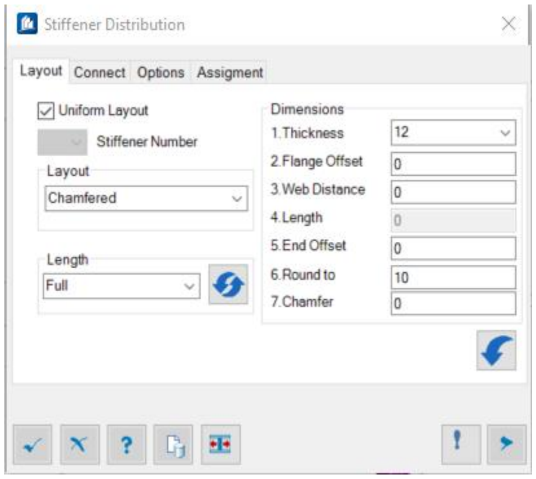
Count: 8

3.End Offset: 0

4.Equal Sequence: 3*100,300,2*500

Layout tab

In the layout tab, the user can specify multiple parameters for his stiffeners: thickness, flange offset, web offset, length, end offset, round to and chamfer. The user can also specify the layout type and the length for his stiffeners. Those parameters alter the stiffener's appearance or where they are inserted relative to the support shape. These parameters can be different for every stiffener as there is a combo box that allows the user to select which stiffener is currently selected. They can also be the same for every stiffener if the uniform layout checkbox is checked. In that case, the parameters will be applied to every stiffener. Finally, there is a reset button in the bottom right of the tab. If this button is clicked the parameters are reset to their default values for the current configuration.



Stiffener Distribution

Layout | Connect | Options | Assigment

☑ Uniform Layout

Stiffener Number

Layout: Chamfered

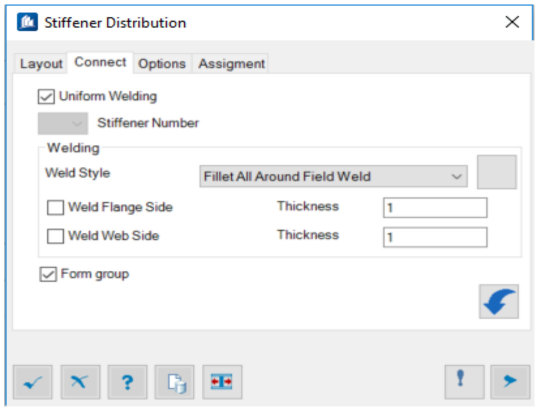
Length: Full

Dimensions

1.Thickness	12
2.Flange Offset	0
3.Web Distance	0
4.Length	0
5.End Offset	0
6.Round to	10
7.Chamfer	0

Connect tab

In the connect tab, the user can add welding to his stiffeners. There is an option to weld the side of the stiffener attached to the flange of the shape and an option to weld the sides of the stiffener attached to the webs of the shape. The user can also specify a thickness for his welds in a textbox. Like in the layout tab, the user can choose different welding parameters his stiffeners or he can apply the same parameters for all of them. There is also a reset button at the bottom right corner.



Stiffener Distribution

Layout | Connect | Options | Assigment

☑ Uniform Welding

Stiffener Number

Welding

Weld Style: Fillet All Around Field Weld

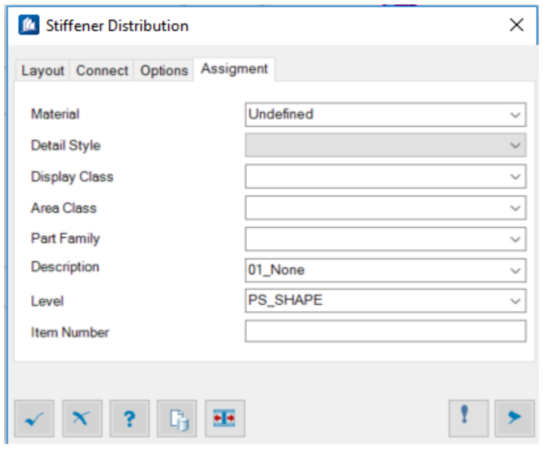
☐ Weld Flange Side — Thickness: 1

☐ Weld Web Side — Thickness: 1

☑ Form group

Assignment tab

In the assignment tab, the user can specify a material for his stiffeners. He can also assign his stiffeners to a display class, area class, part family, a level and he can assign a detail style and a description.



Stiffener Distribution

Layout | Connect | Options | Assigment

Material	Undefined
Detail Style	
Display Class	
Area Class	
Part Family	
Description	01_None
Level	PS_SHAPE
Item Number	

Global buttons

There are multiple buttons located at the bottom of the form. Most of those are buttons that are common to every connection. However, one of the button is responsible for slightly shifting the insertion point of the stiffeners on the shape. There is also a button responsible for building a connection with the parameters on the form. The connection will not be rebuilt until this button is pressed.

Usage and workflow

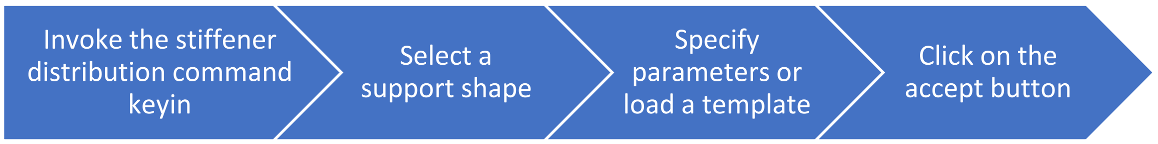
Load the plugin: *mdl load PSN_StiffenerDistribution*

Call the plugin command (once loaded): *PSN_StiffenerDistribution create*

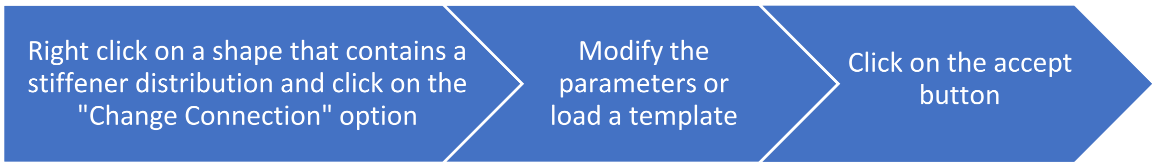
Call the plugin command (without having to load it first):

mdl keyin PSN_StiffenerDistribution, STIFFENERDISTRIBUTION PSN_StiffenerDistribution create

Creation procedure:

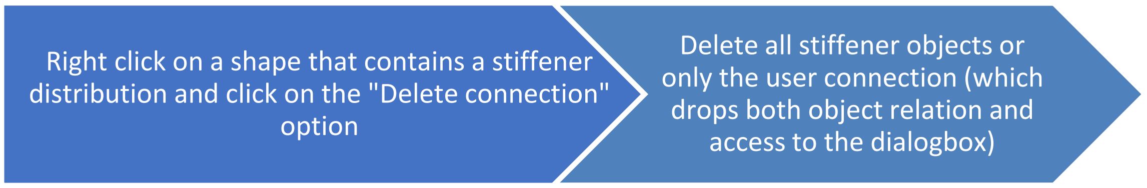

Invoke the stiffener distribution command keyin
Select a support shape
Specify parameters or load a template
Click on the accept button

Editing procedure:


Right click on a shape that contains a stiffener distribution and click on the "Change Connection" option
Modify the parameters or load a template
Click on the accept button

To edit a stiffener distribution when more than one connection is associated with the support shape, users must travel to the desired connection via the PsProperties>Modifications>Loglinks.

Deletion procedure:


Right click on a shape that contains a stiffener distribution and click on the "Delete connection" option
Delete all stiffener objects or only the user connection (which drops both object relation and access to the dialogbox)

To delete a distribution when more than one connection is associated with the support shape, users must travel to the desired connection via the PsProperties>Modifications>Loglinks.