Mastering SOLIDWORKS API

DEPLOY

YOUR ADD-IN

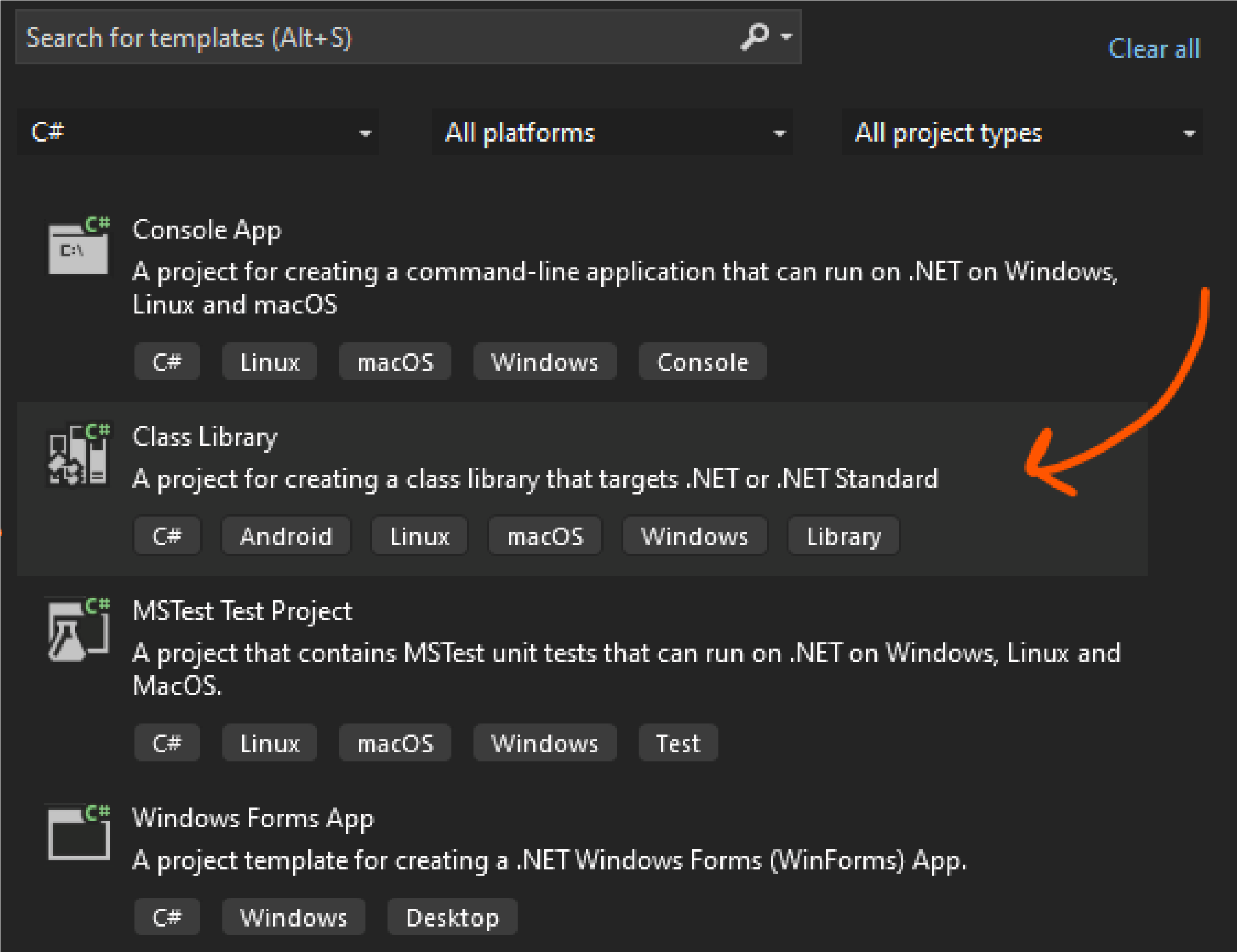
TO THE

TASK PANE

# #01 PREPARE THE PROJECT

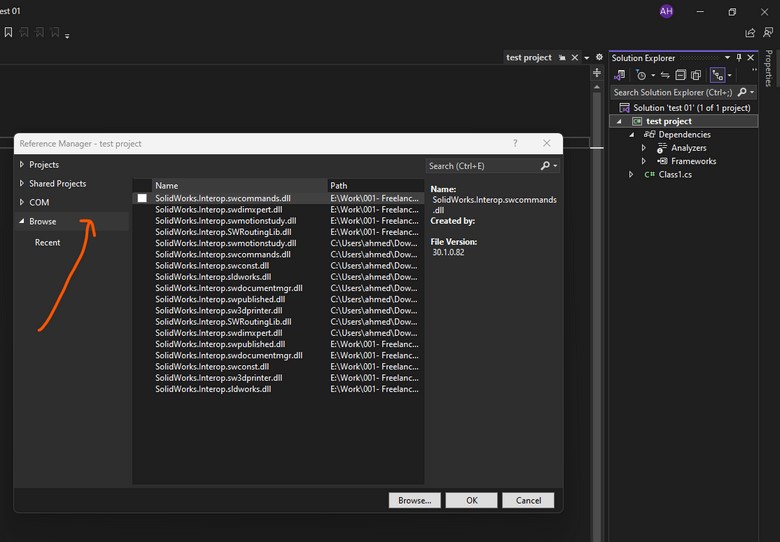
In this tutorial we will use C# language.

Open a new visual studio project, and make sure you select the “class library”. give your project a suitable name. let’s assume it will be “test project”.



# #02 ADD RESOURCES

After creating the project, in the left properties pan, press right click on the project name and select Add > Project Reference.



From the appeared window press “Browse” and head to SOLIDWORKS location in the program file directory.

Choose the following files and press Ok:

SolidWorks.Interop.sldworks.dll

SolidWorks.Interop.swcommands.dll

SolidWorks.Interop.swconst.dll

SolidWorks.Interop.swdimxpert.dll

SolidWorks.Interop.swdocumentmgr.dll

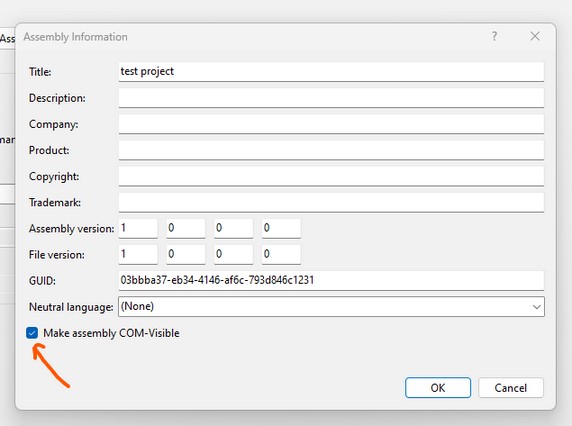
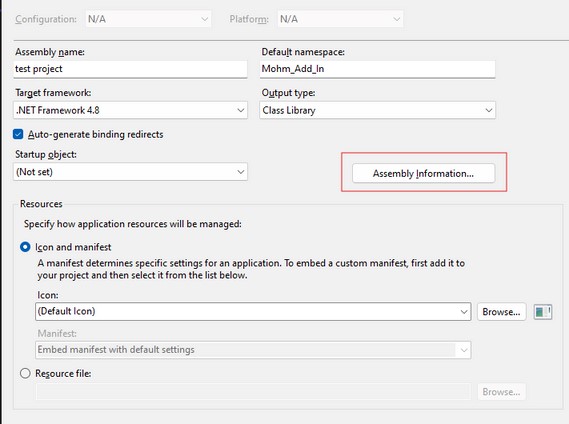
SolidWorks.Interop.swmotionstudy.dll

SolidWorks.Interop.swpublished.dll

# #03 PREPARE PROJECT (1)

There is some steps you need to take in order to make your project run the code smoothly without any problems.

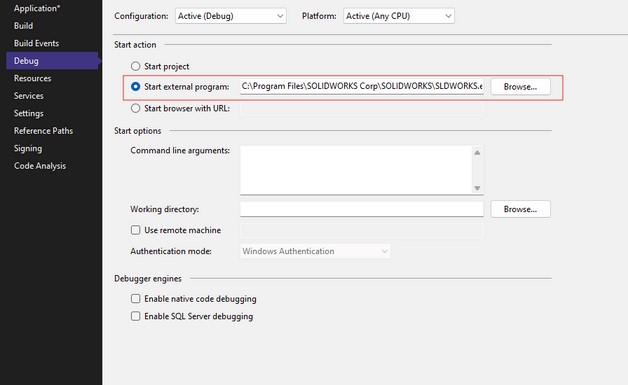
Select the project name and press right click, and choose “Properties”, follow the next screens, and make sure to tick “Make assembly COM-Visible”



1

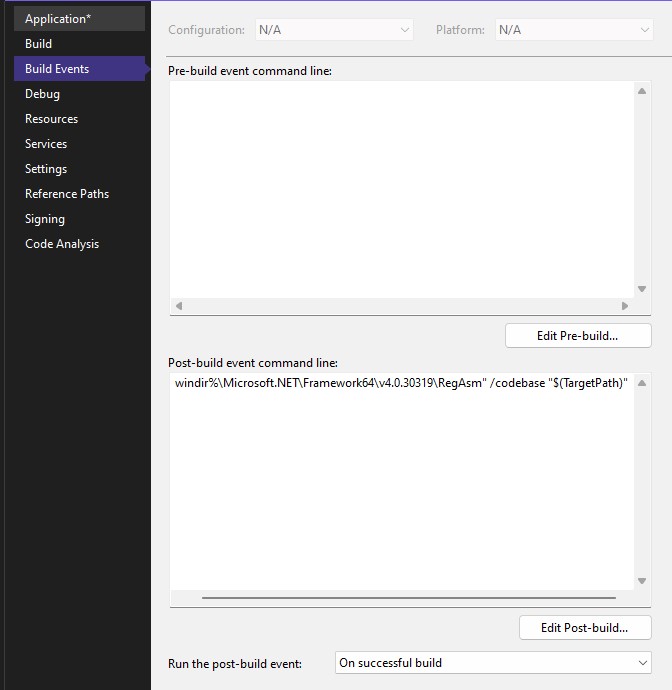
2

While still in Option, choose debug, and click BROWSE, then head to “SLDWORKS.exe” in the main SOLIDWORKS folder in Program Files



# #03 PREPARE PROJECT (2)

Finally, select “Build Events”



In post-build event command line, type the following code:

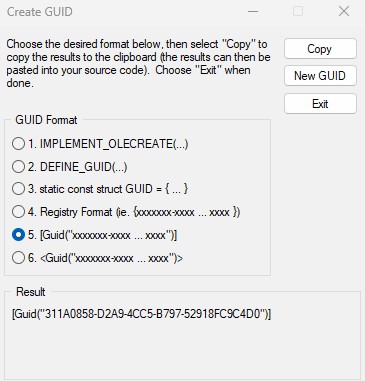
"%windir%\Microsoft.NET\Framework64\v4.0.30319\RegAsm" /codebase "$(TargetPath)"

Now you are ready for writing your code

# #04 PREPARE FOR CODING

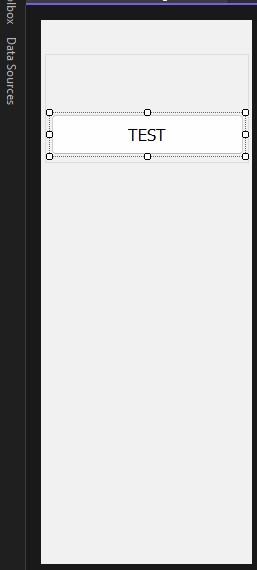
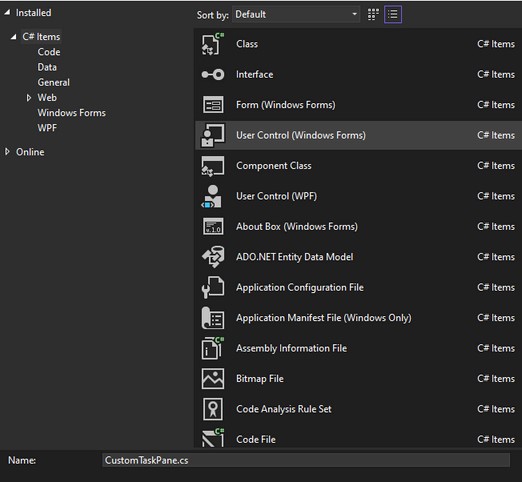
It is better to rename “class 1” to something more readable, let’s rename it

“TaskPaneIntegration”, and make sure to press yes to the message that tells you that all the relative objects will be renamed as well.

 You will need a unique GUID for the registry. To get one from visual studio press on Tools > Create GUID, and from the pop-up windows copy your GUID.

To make your add-in has its own picture in the task pane, in the main folder of your project, put the photo you want.

# #05 DESIGN THE TASK PANE

 Right click on the project name on the right panel and choose Add > User control. from the appeared window choose “User Control”

, and name it “CustomTaskPane”, Then press Add

After that, in the right side, right click on the

“CustomTaskPane” and choose “View Designer”.

From there the design of the task pane will appear, you can add buttons, text, labels, and all the tools you normally assign in your visual studio projects. you can continue to create your custom add-in function, then come back to complete adding it to task pane, or vice versa.

# #06 START CODING

Right click on “TaskPaneIntegration.cs” file in the right section and choose “View Code”.

In the code area, paste the following code, make sure that all the code will be inside the “namespace” section.

|  |
| --- |
| **[ComVisible(true)]**  **[Guid("A418D0B5-FB95-4E5A-B842-8137D0B4CF45")] public class TaskPaneIntegration : ISwAddin**  **{**  **private int InnwCookie;**  **private TaskpaneView CustomTaskpaneView; private CustomTaskPane TaskPaneHost; public static SldWorks SolidworksApplication;**  **public const string swProgID = "Enable Addin-For-Solidworks";**  **public static string workPath =**  **Path.GetDirectoryName(typeof(TaskPaneIntegration).Assembly.CodeBase).Replace(@"file:\", string.Empty);**  **public bool ConnectToSW(object ThisSW, int Cookie)**  **{ }**  **public bool DisconnectFromSW()**  **{ }**  **private void LoadUI()**  **{ }**  **private void UnLoadUI()**  **{**  **}**  **[ComRegisterFunction()]**  **private static void ComRegister(Type t)**  **{**  **}**  **[ComUnregisterFunction()]**  **private static void ComUnRegister(Type t)**  **{**  **}**  **}** |

Inside this class we will write all the variables deceleration, and our 6 needed methods.

Make sure to change the GUID in this code with the one you already have.

Now, fill the 6 methods as follows.

|  |
| --- |
| **public bool ConnectToSW(object ThisSW, int Cookie)**  **{**  **SolidworksApplication = (SldWorks)ThisSW; InnwCookie = Cookie;**  **var ok = SolidworksApplication.SetAddinCallbackInfo2(0, this, InnwCookie); LoadUI(); return true;**  **}** |
| **public bool DisconnectFromSW()**  **{**  **UnLoadUI(); return true;**  **}** |
| **private void LoadUI()**  **{**  **// Find location to our taskpane image**  **var imagePath =**  **Path.Combine(Path.GetDirectoryName(typeof(TaskPaneIntegration).Assembly.CodeBase).Replace(@"file:\", string.Empty), "Logo.png");**  **// Craete our Taskpane**  **CustomTaskpaneView = SolidworksApplication.CreateTaskpaneView2(imagePath, "custom Add-in");**  **// Load our UI into our Taskpane**    **TaskPaneHost = (CustomTaskPane)CustomTaskpaneView.AddControl(TaskPaneIntegration.swProgID,**  **string.Empty);**    **}** |

In the previous method, make sure to write the name of the picture of your add-in as it exists in the main folder.

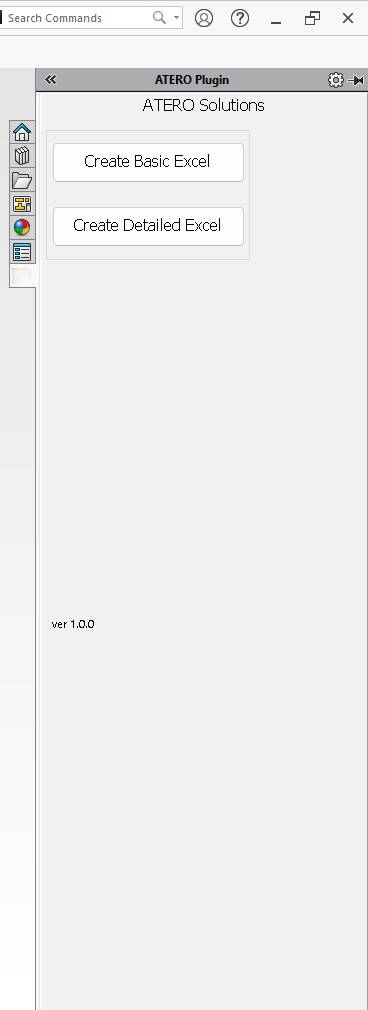
Continue to fill the 6 methods as follows.

|  |
| --- |
| **private void UnLoadUI()**  **{**  **TaskPaneHost = null;**  **// Remove Taskpane view**  **CustomTaskpaneView.DeleteView();**  **// Release COM reference and cleanup memory**  **Marshal.ReleaseComObject(CustomTaskpaneView);**  **//**  **CustomTaskpaneView = null; }** |
| **[ComRegisterFunction()]**  **private static void ComRegister(Type t)**  **{ var keyPath = string.Format(@"SOFTWARE\Solidworks\AddIns\{0:b}", t.GUID);**  **using (var rk = Microsoft.Win32.Registry.LocalMachine.CreateSubKey(keyPath)) {**  **// Load Addin when solidworks start rk.SetValue(null, 1);**  **rk.SetValue("Title", "custom Add-in");**  **rk.SetValue("Description", "For any help info@aterosolutions.com");**  **} var keyPath2 = string.Format(@"Software\SolidWorks\AddInsStartup\{0:b}", t.GUID);**  **using (var rk2 = Microsoft.Win32.Registry.CurrentUser.CreateSubKey(keyPath2))**  **{**  **rk2.SetValue(null, 1);**  **}**  **}** |

Continue to fill the 6 methods as follows.

|  |
| --- |
| **[ComUnregisterFunction()]**  **private static void ComUnRegister(Type t)**  **{ var keyPath = string.Format(@"SOFTWARE\Solidworks\AddIns\{0:b}", t.GUID); Microsoft.Win32.Registry.LocalMachine.DeleteSubKeyTree(keyPath); var keyPath2 = string.Format(@"Software\SolidWorks\AddInsStartup\{0:b}", t.GUID);**  **Microsoft.Win32.Registry.CurrentUser.DeleteSubKeyTree(keyPath2); }** |

# #08 RUN YOUR PROJECT



Congratulations. You have created your custom add-in and attached it to the task pane.

To run your code, make sure that you open

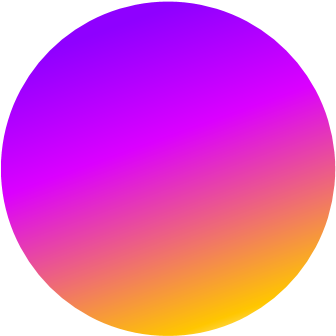
Visual Studio as administrator, and there is no SOLIDWORKS instances are opened.

Press “Start” in the standard tool bar, you will see that a new instance of SolidWorks is now opening, and after that you will see a new tab in the task pane, with the custom photo you have put. press on that new tab, and you will see the task pane design you have created, and you can use it. This procedure must be done one time only, every time you open SOLIDWORKS again you will see your custom add-in exist.

ATERO

Solutions

Mastering SOLIDWORKS API is a huge thing, but it seems very easy if you are following us.



We have a numerous number of experts in SOLIDWORKS, in addition to other software.

Always be prepared for every guide and everything new from us

www.aterosolutions.com