



Solid Edge ST7 AddIn Architecture Overview



Jason Newell

10 Nov 2014 CPOL

Solid Edge ST7 AddIn Architecture Overview

This article was marked as deleted at 10 Mar 2016.

This article appears in the Third Party Products and Tools section. Articles in this section are for the members only and must not be used to promote or advertise products in any way, shape or form. Please report any spam or advertising.

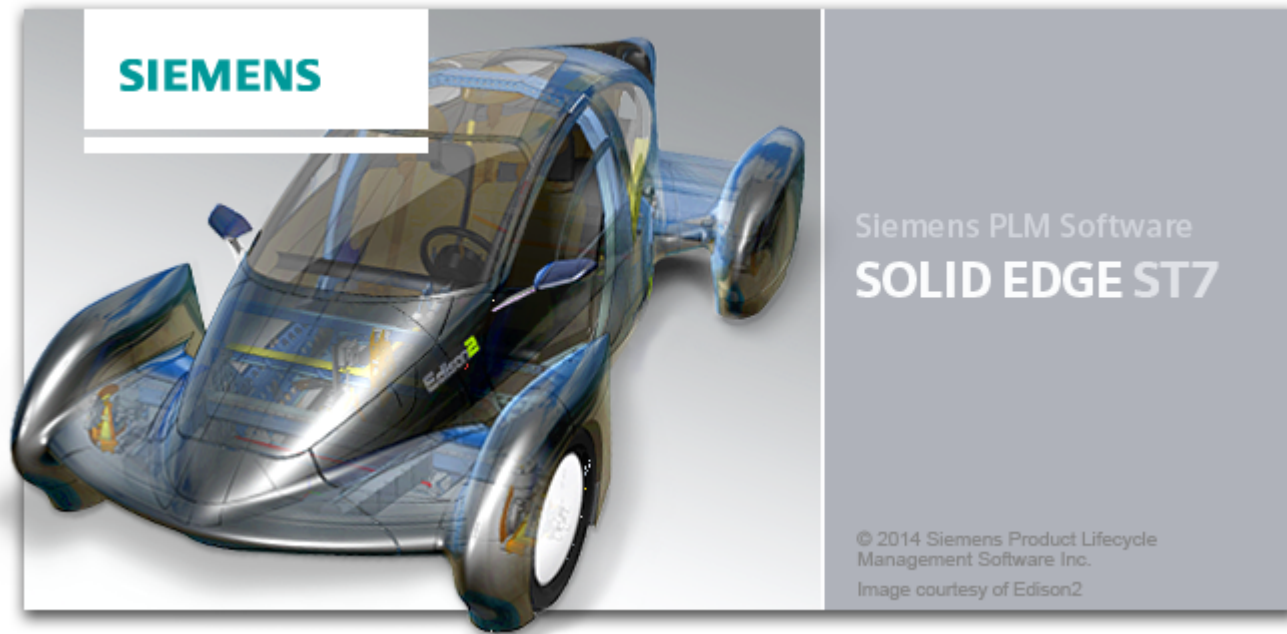


Table of content

- [Introduction](#)
- [Overview](#)
- [Solid Edge SDK](#)
- [Category Identifiers](#)
- [Type Libraries](#)
- [Interfaces](#)
 - [ISolidEdgeAddIn](#)
 - [ISEAddInEx](#)
 - [ISEAddInEx2](#)
 - [ISEAddInEvents](#)
- [Registration](#)
- [Resources](#)

Introduction

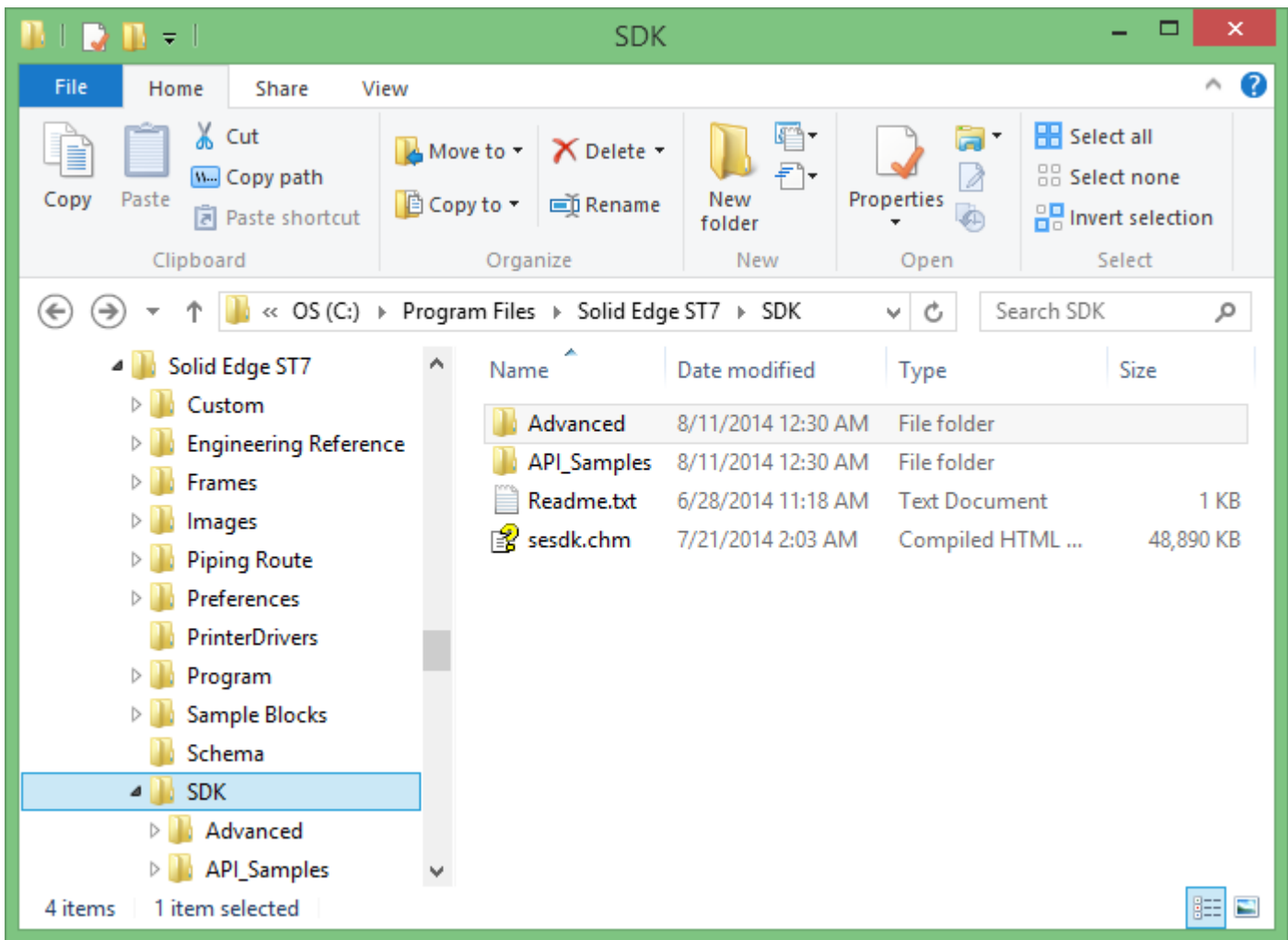
Solid Edge is a 3D CAD application. It is not uncommon for customers or partners to have a need to develop custom code that executes directly inside the Solid Edge process. This article aims to explain the Solid Edge addin architecture from a high level. It will also serve as the foundation for future programming language specific articles.

Overview

Solid Edge has a [COM](#) based API for automation and integration. Since most modern programming languages support COM, integrating with Solid Edge is fairly straightforward. Depending on your programming language of choice, the details of how to write an addin will vary but the requirements are the same.

Solid Edge SDK

The Solid Edge SDK is not installed by default when you install Solid Edge ST7 but there is a SDK folder on the DVD. You can simply copy the SDK folder from the DVD to your local machine. This is useful as there are several files in the SDK folder that you'll likely need to reference.



There are several files of interest so I'll list the most important ones below.

- **sesdk.chm** - The SDK compiled help file. This same content is also [available online](#).
- **\Advanced\include\secatids.h** - C++ include file that defines CATIDs (GUIDs) used by Solid Edge. Regardless of programming language, these CATIDs are important for addin development.
- **\Advanced\include\igl.h** - C++ include that defines an IGL (OpenGL) interface. This interface is not defined in any type libraries and must be referenced if you are planning on working with OpenGL in Solid Edge.
- **\Advanced\samples\Addins\VC** - Sample Solid Edge addin written in C++ (MFC).
- **\Advanced\samples\Addins\Doc\addin.doc** - A must read document for Solid Edge addin developers. Much of the material in this article was gleaned from this document.

Category Identifiers (secatids.h)

The following CATIDs (GUIDs) are defined only in secatids.h and are important to understand because we will need to use them during addin development. I have bolded the most common CATIDs.

CATID	GUID	Remarks
CATID_SolidEdgeAddIn	{26B1D2D1-2B03-11d2-B589-080036E8B802}	Used when registering your addin.
CATID_SEApplication	{26618394-09D6-11d1-BA07-080036230602}	Application Evenvironment
CATID_SEAssembly	{26618395-09D6-11d1-BA07-080036230602}	Assembly Environment
CATID_SEMotion	{67ED3F40-A351-11d3-A40B-0004AC969602}	Simply Motion Environment
CATID_SEPart	{26618396-09D6-11d1-BA07-080036230602}	Ordered Part Environment
CATID_SEProfile	{26618397-09D6-11d1-BA07-080036230602}	Profile Environment
CATID_SEFeatureRecognition	{E6F9C8DC-B256-11d3-A41E-0004AC969602}	Feature Recognition Environment
CATID_SESheetMetal	{26618398-09D6-11d1-BA07-080036230602}	Ordered Sheet Metal Environment
CATID_SEDraft	{08244193-B78D-11d2-9216-00C04F79BE98}	Draft Environment
CATID_SEWeldment	{7313526A-276F-11d4-B64E-00C04F79B2BF}	Weldment Environment
CATID_SEXpresRoute	{1661432A-489C-4714-B1B2-61E85CFD0B71}	XpresRoute Environment
CATID_SEExplode	{23BE4212-5810-478b-94FF-B4D682C1B538}	Explode Environment
CATID_SESimplify	{CE3DCEBF-E36E-4851-930A-ED892FE0772A}	Simplify Environment
CATID_SEStudio	{D35550BF-0810-4f67-97D5-789EDBC23F4D}	Explode Render Animate Environment
CATID_SELayout	{27B34941-FFCD-4768-9102-0B6698656CEA}	Sketch Environment
CATID_SESketch	{0DDABC90-125E-4cfe-9CB7-DC97FB74CCF4}	Profile/Sketch Environment
CATID_SEProfileHole	{0D5CC5F7-5BA3-4d2f-B6A9-31D9B401FE30}	Profile Environment
CATID_SEProfilePattern	{7BD57D4B-BA47-4a79-A4E2-DFFD43B97ADF}	Profile Environment
CATID_SEProfileRevolved	{FB73C683-1536-4073-B792-E28B8D31146E}	Profile Environment
CATID_SEDrawingViewEdit	{8DBC3B5F-02D6-4241-BE96-B12EAF83FAE6}	Draft View Edit Environment

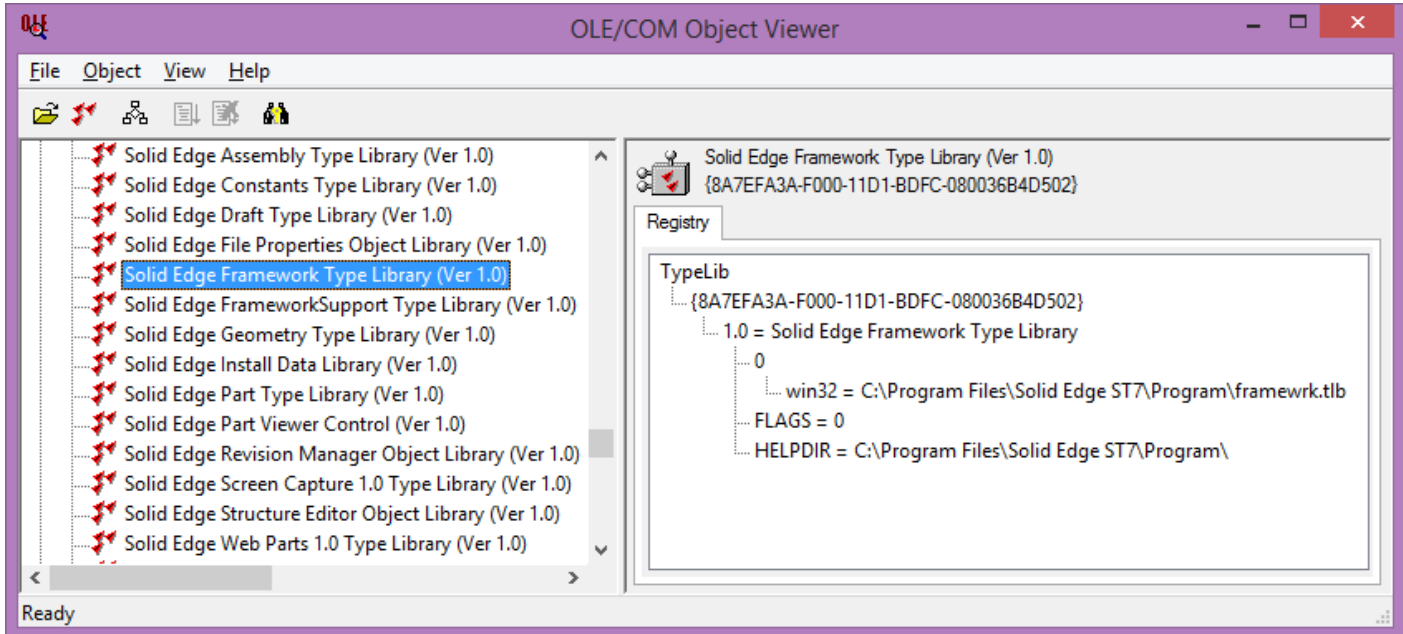
CATID	GUID	Remarks
CATID_SERefAxis	{B21CCFF8-1FDD-4f44-9417-F1EAE06888FA}	Profile Environment
CATID_SECuttingPlaneLine	{7C6F65F1-A02D-4c3c-8063-8F54B59B34E3}	Cutting Plane Line Environment
CATID_SEBrokenOutSectionProfile	{534CAB66-8089-4e18-8FC4-6FA5A957E445}	Broken Out Section Profile Environment
CATID_SEFrame	{D84119E8-F844-4823-B3A0-D4F31793028A}	Frame Environment
CATID_2dModel	{F6031120-7D99-48a7-95FC-EEE8038D7996}	2D Model Environment
CATID_EditBlockView	{892A1CDA-12AE-4619-BB69-C5156C929832}	Open Block Environment
CATID_EditBlockInPlace	{308A1927-CDCE-4b92-B654-241362608CDE}	Edit Block Environment
CATID_SEComponentSketchInPart	{FAB8DC23-00F4-4872-8662-18DD013F2095}	Sketch Environment
CATID_SEComponentSketchInAsm	{86D925FB-66ED-40d2-AA3D-D04E74838141}	Sketch Environment
CATID_SEHarness	{5337A0AB-23ED-4261-A238-00E2070406FC}	Harness Design Environment
CATID_SEAll	{C484ED57-DBB6-4a83-BEDB-C08600AF07BF}	All Environments
CATID_SEAllDocumentEnvironments	{BAD41B8D-18FF-42c9-9611-8A00E6921AE8}	All Document Environments
CATID_SEDMPart	{D9B0BB85-3A6C-4086-A0BB-88A1AAD57A58}	Synchronous Part Environment
CATID_SEDMSheetMetal	{9CBF2809-FF80-4dbc-98F2-B82DABF3530F}	Synchronous Sheet Metal Environment
CATID_SEDMAsssembly	{2C3C2A72-3A4A-471d-98B5-E3A8CFA4A2BF}	*Not used
CATID_SESimplifiedAssemblyPart	{E7350DC3-6E7A-4D53-A53F-5B1C7A0709B3}	Simplified Assembly Part Environment
CATID_Sketch3d	{07F05BA4-18CD-4B87-8E2F-49864E71B41F}	3D Sketch Environment

Type Libraries

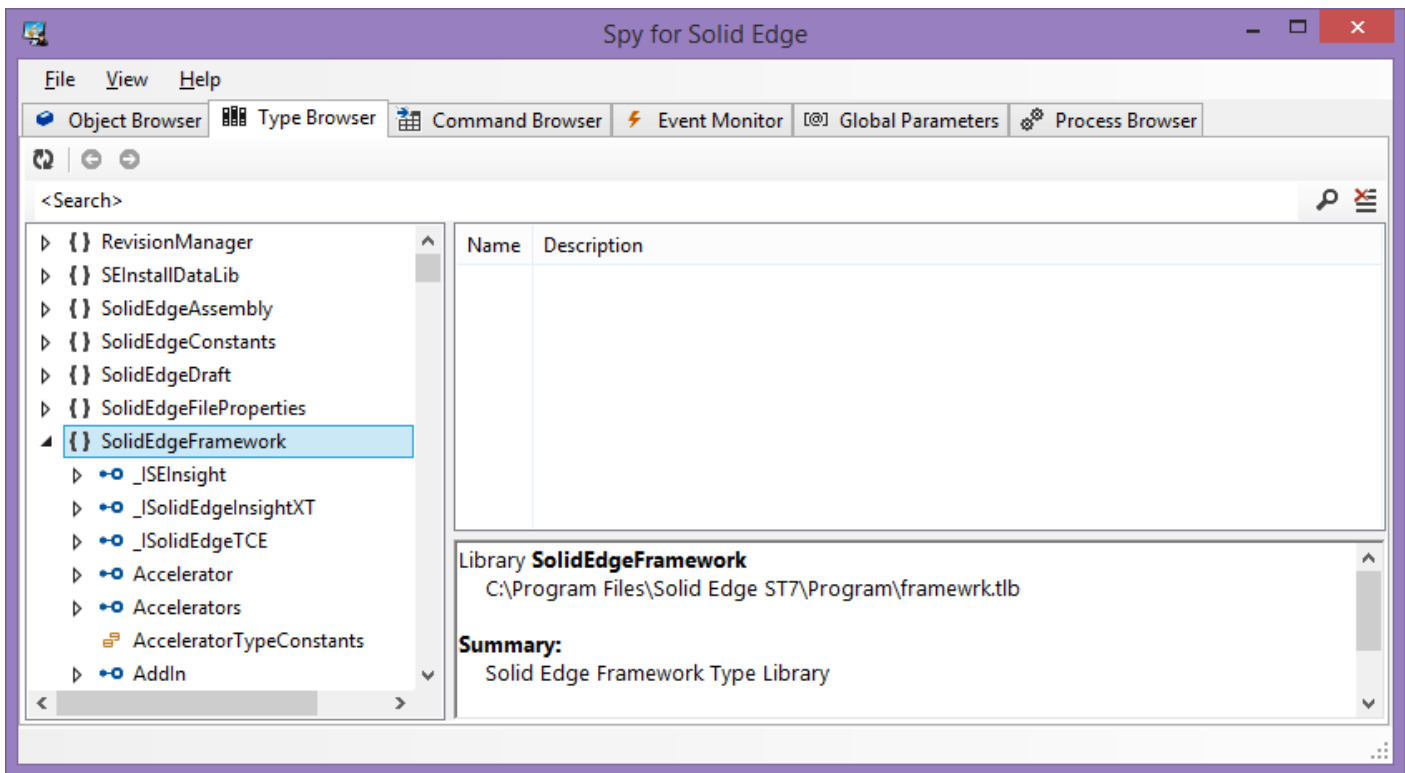
All Solid Edge COM type libraries can be found in the %PROGRAMFILES%\Solid Edge ST7\Program directory. These type libraries define the Solid Edge COM APIs. The SolidEdgeFramework (framework.tlb) type library is the base library. At a minimum, you must reference this type library to write a Solid Edge addin.

Library	Namespace	Description	LIBID
assembly.tlb	SolidEdgeAssembly	Solid Edge Assembly Type Library	{3E2B3BD4-F0B9-11D1-BDFD-080036B4D502}
constant.tlb	SolidEdgeConstants	Solid Edge Constants Type Library	{C467A6F5-27ED-11D2-BE30-080036B4D502}
draft.tlb	SolidEdgeDraft	Solid Edge Draft Type Library	{3E2B3BDC-F0B9-11D1-BDFD-080036B4D502}
PropAuto.dll	SolidEdgeFileProperties	Solid Edge File Properties Object Library	{AED8FE52-3129-11D1-BC83-0800360E1E02}
framework.tlb	SolidEdgeFramework	Solid Edge Framework Type Library	{8A7EFA3A-F000-11D1-BDFC-080036B4D502}
fwksupp.tlb	SolidEdgeFrameworkSupport	Solid Edge FrameworkSupport Type Library	{943AC5C6-F4DB-11D1-BE00-080036B4D502}
geometry.tlb	SolidEdgeGeometry	Solid Edge Geometry Type Library	{3E2B3BE1-F0B9-11D1-BDFD-080036B4D502}
SEInstallData.dll	SEInstallDataLib	Solid Edge Install Data Library	{42E04299-18A0-11D5-BBB2-00C04F79BEA5}
Part.tlb	SolidEdgePart	Solid Edge Part Type Library	{8A7EFA42-F000-11D1-BDFC-080036B4D502}

The OLE/COM Object Viewer (oleview.exe) is an easy way to browse type libraries.



Spy for Solid Edge is another easy way to view the Solid Edge type libraries.



Interfaces

The following interfaces are the most relevant for Solid Edge addins. They are defined in the `framework.tlb` type library and are contained in the `SolidEdgeFramework` namespace.

- `ISolidEdgeAddIn` - Interface for addins to implement for Solid Edge to connect to them.
- `ISEAddInEx` - Interface for configuring your addin.
- `ISEAddInEx2` - Interface for configuring your addin.
- `ISEAddInEvents` - Interface for addin events.

ISolidEdgeAddIn

A Solid Edge addin must implement the `ISolidEdgeAddIn` interface. This interface provides the main entry point between Solid Edge and your addin.

```
[
    odl,
    uuid(D3F30AE5-2582-11D2-BAF9-080036230602),
    helpstring("Provides an interface for add-ins to implement for Solid Edge to connect to them."),
    helpcontext(0x0000c4d2),
    nonextensible,
    oleautomation
]
interface ISolidEdgeAddIn : IUnknown {
    [helpstring("Occurs when Solid Edge connects to an add-in."), helpcontext(0x0000c4d3)]
```

```

HRESULT _stdcall OnConnection(
    [in] IDispatch* Application,
    [in] SeConnectMode ConnectMode,
    [in] AddIn* AddInInstance);
[helpstring("Occurs when a Solid Edge environment and an add-in connect."),
helpcontext(0x0000c4d4)]
HRESULT _stdcall OnConnectToEnvironment(
    [in] BSTR EnvCatID,
    [in] IDispatch* pEnvironmentDispatch,
    [in] VARIANT_BOOL bFirstTime);
[helpstring("Occurs when Solid Edge and an add-in are disconnected."),
helpcontext(0x0000c4d5)]
HRESULT _stdcall OnDisconnection([in] SeDisconnectMode DisconnectMode);
};

```

OnConnection(IDispatch* Application, SeConnectMode ConnectMode, AddIn* AddInInstance)

This method gets called by Solid Edge when your addin is first loaded into the process. When your addin gets loaded depends on which environments you register your addin for.

Application

SolidEdgeFramework::Application pointer to the dispatch interface of the Solid Edge application that is attempting to connect to the addin. The addin uses this pointer to make any necessary calls to the application to connect to Solid Edge event sinks, or to otherwise communicate with Solid Edge to perform whatever tasks the add-in needs when first starting up.

ConnectMode

Connect mode that indicates what caused Solid Edge to connect to the addin. The possible values are:

- **seConnectAtStartup** - Loading the addin at startup.
- **seConnectByUser** - Loading the addin at user's request.
- **seConnectExternally** - Loading the addin due to an external (programmatic) request.

AddIn

SolidEdgeFramework::AddIn pointer to the dispatch interface of a Solid Edge addin object that provides another channel of communication between the addin and Solid Edge.

OnConnectToEnvironment(BSTR EnvCatID, LPDISPATCH pEnvironmentDispatch, VARIANT_BOOL* bFirstTime)

This method gets called by Solid Edge when your addin first connects to an environment. Which environments your addin connects to depends on how your addin is registered.

EnvCatID

The category identifier of the environment as a string. If the addin is registered as supporting multiple environments, the addin can use the string to determine which environment to which it is being asked to connect.

pEnvironment

SolidEdgeFramework::Environment pointer to the dispatch interface of the environment.

bFirstTime

The **bFirstTime** parameter specifies that a Solid Edge environment is connecting to the addin for the first time. When connecting for the first time, the addin, if necessary, should add any needed user interface elements (for example, buttons). On exiting, Solid Edge will save any such buttons so they can be restored during the next session.

*It is important to note that for the Part and SheetMetal environments, you will actually get two **OnConnectToEnvironment()** calls. The reason for this is that Solid Edge supports Ordered (traditional) and Synchronous environments.

- Part
 - **CATID_SEPart** (Ordered)
 - **CATID_SEDMPart** (Synchronous)
- SheetMetal
 - **CATID_SESheetMetal** (Ordered)
 - **CATID_SEDMSheetMetal** (Synchronous)

OnDisconnection(SeDisconnectMode DisconnectMode)

This method gets called by Solid Edge when the process is about to exit.

DisconnectMode

Disconnect mode that indicates what caused Solid Edge to disconnect from the addin. The possible values are:

- **seDisconnectAtShutdown** - Unloading at shutdown.
- **seDisconnectByUser** - Unloading the addin due to a user request.
- **seDisconnectExternally** - Unloading the addin due to an external (programmatic) request.

ISEAddInEx

The **ISEAddInEx** interface provides functionality to configure your addin. One of the main features of this interface is that it allows you to create commands for your addin. Depending on how you create the commands, they get added to the Solid Edge ribbon as buttons that allow users to easily invoke the commands.

When `OnConnection()` is called, Solid Edge passes in the `AddInInstance` argument. You will want to store this object in a local variable for later use. When `OnConnectToEnvironment()` is called, you can cast that local variable as `ISEAddInEx` or `ISEAddInEx2`.

MIDL

```
[
    odl,
    uuid(DC601E2F-5BB3-4BF2-A9C7-03E60975E897),
    helpstring("A control that represents a command added to Solid Edge through the AddIn interface."),
    helpcontext(0x0000c439),
    oleautomation
]
interface ISEAddInEx : ISEAddIn {
    [helpstring("Sets the parameters for the referenced AddIn object."),
    helpcontext(0x0098967f)]
    HRESULT _stdcall SetAddInInfoEx(
        [in] BSTR ResourceFilename,
        [in] BSTR EnvironmentCatID,
        [in] BSTR CategoryName,
        [in] long IDColorBitmapMedium,
        [in] long IDColorBitmapLarge,
        [in] long IDMonochromeBitmapMedium,
        [in] long IDMonochromeBitmapLarge,
        [in] long NumberOfCommands,
        [in] SAFEARRAY(BSTR)* CommandNames,
        [in, out] SAFEARRAY(long)* CommandIDs);
    [propget, helpstring(" "), helpcontext(0x0098967f)]
    HRESULT _stdcall AddInEdgeBarEvents([out, retval] AddInEdgeBarEvents**
    AddInEdgeBarEvents);
};
```

SetAddInInfoEx(...)

This method should be called during a `OnConnectToEnvironment()` call from Solid Edge. It allows you to add commands for your addin to Solid Edge. If your command images are contained in an image strip, you can make a single call to add all of the commands at once. If your command images are separated into individual images, you'll need to make a call for each image (multiple calls).

ResourceFilename

The path to a .dll that contains the WIN32 image resources to be used for the ribbon buttons. This path can be your addin.dll or a separate resource.dll.

EnvironmentCatID

The CATID (GUID) of the environment to install the commands. You should use the `EnvCatID` parameter passed in from the `OnConnectToEnvironment()` call from Solid Edge.

CategoryName

The name the addin associates with the set of commands being added. This name is used in the ribbon tab caption and should be internationalized. e.g. **"My Ribbon Tab"**

IDColorBitmapMedium

The ID of the image resource containing the medium sized images of the commands being added.

IDColorBitmapLarge

The ID of the image resource containing the large sized images of the commands being added.

IDMonochromeBitmapMedium

The ID of the image resource containing the medium sized monochrome images of the commands being added. You can pass -1 if you do not wish to support monochrome displays.

IDMonochromeBitmapLarge

The ID of the image resource containing the large sized monochrome images of the commands being added. You can pass -1 if you do not wish to support monochrome displays.

NumberOfCommands

The number of commands being added to the environment. 1-N commands can be added.

CommandNames

An array of strings containing command information. Each string contains substrings separated by a '\n'. e.g. **"3CE1FCC9_Command1\nCamera\nCamera Supertip\nCamera Screentip"**

String part	Description
Name	This name should be completely unique or it may fail to get added due to naming conflicts with other addins.
Text (caption)	The command text.
Status bar	The command supertip.
Tooltip	The command screentip.
Macro (optional)	e.g. "notepad.exe"
Macro parameter (optional)	e.g. "param1 param2"

CommandIDs

An array of command IDs. Each command id should be unique within the addin for a given environment. These command IDs will be passed back to the addin during [ISEAddInEvents](#).

AddCommandBarButton(...)

If you use [SetAddInInfoEx\(\)](#), you must configure your ribbon buttons manually. If the **bFirstTime** parameter of [OnConnectToEnvironment\(\)](#) is TRUE, call [AddCommandBarButton\(\)](#) for each command.

EnvironmentCatID

The CATID (GUID) of the environment to install the commands. You should use the **EnvCatID** parameter passed in from the [OnConnectToEnvironment\(\)](#) call from Solid Edge.

CommandBarName

The parameter name is a bit misleading due to the API pre-dating the ribbon. In ST7, this will be the ribbon group name. e.g. **"My Group"**

CommandID

The command ID to be assigned to the button.

GuiVersion

It is important to understand that Solid Edge caches the commands that an addin creates. If you make changes to your commands after deployment, you must change the **GuiVersion** of the addin to tell Solid Edge to flush its cache or your commands will likely not get added correctly.

ISEAddInEx2

The **ISEAddInEx2** interface is an extension to the **ISEAddInEx** interface that adds one method, **SetAddInInfoEx2()**. This method allows you to skip the **AddCommandBarButton()** approach necessary when using **SetAddInInfoEx()**.

MIDL

```
[
    odl,
    uuid(8517E9E0-387B-45D1-8DCD-AC9388224E35),
    helpstring("A control that represents a command added to Solid Edge through the AddIn interface."),
    helpcontext(0x0000c439),
    oleautomation
]
interface ISEAddInEx2 : ISEAddInEx {
    [helpstring("Sets the parameters for the referenced AddIn object."),
    helpcontext(0x0098967f)]
    HRESULT _stdcall SetAddInInfoEx2(
        [in] BSTR ResourceFilename,
        [in] BSTR EnvironmentCatID,
        [in] BSTR CategoryName,
        [in] long IDColorBitmapMedium,
        [in] long IDColorBitmapLarge,
        [in] long IDMonochromeBitmapMedium,
        [in] long IDMonochromeBitmapLarge,
        [in] long NumberOfCommands,
        [in] SAFEARRAY(BSTR)* CommandNames,
        [in, out] SAFEARRAY(long)* CommandIDs,
        [in] SAFEARRAY(long)* CommandButtonStyles);
};
```

SetAddInInfoEx2(...)

This method extends the `ISEAddInEx` method `SetAddInInfoEx()` by adding a new parameter, `CommandButtonStyles`. The following list only mentions the differences.

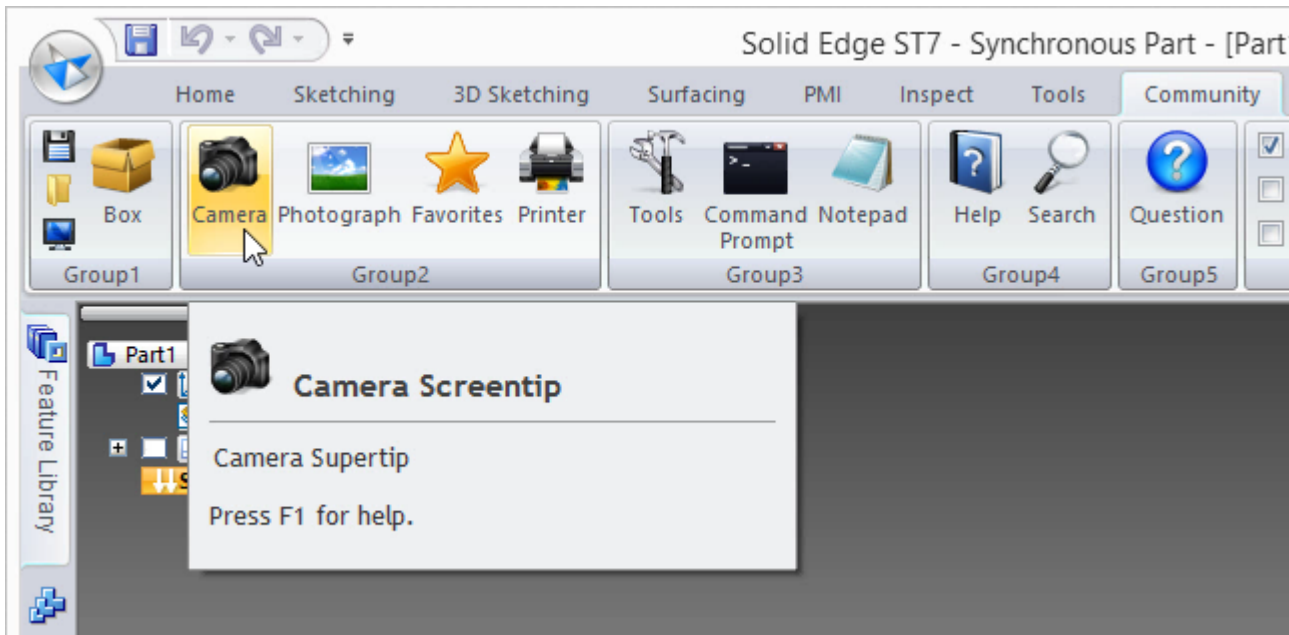
CategoryName

The name the addin associates with the set of commands being added. This name is used in the ribbon tab caption and should be internationalized. e.g. **"My Ribbon Tab\nMy Group"**

CommandButtonStyles

An array of `SeButtonStyle` values for each command being added. This eliminates the need to call `AddCommandBarButton()`.

This is a screenshot of what the added command looks like in the ribbon.



ISEAddInEvents

The `ISEAddInEvents` interface provides command events for your addin. These events allow you to control how your commands behave. A pointer to the events can be obtained from the `ISEAddIn::AddInEvents` property.

MIDL

```
[
    odl,
    uuid(0F539244-4816-11D2-B5AC-080036E8B802),
    helpstring("A collection of AddInEvent objects."),
    helpcontext(0x0000c446),
    oleautomation
]
interface ISEAddInEvents : IUnknown {
    [helpstring("Occurs when the user selects the button for the associated add-in."),
    helpcontext(0x0000c447)]
    HRESULT _stdcall OnCommand([in] long CommandID);
}
```

```

    [helpstring("Occurs when the user requests help for the associated add-in."),
    helpcontext(0x0000c448)]
    HRESULT _stdcall OnCommandHelp(
        [in] long hFrameWnd,
        [in] long HelpCommandID,
        [in] long CommandID);
    [helpstring("Occurs when the user interface needs to be updated."),
    helpcontext(0x0000c449)]
    HRESULT _stdcall OnCommandUpdateUI(
        [in] long CommandID,
        [in, out] long* CommandFlags,
        [out] BSTR* MenuItemText,
        [in, out] long* BitmapID);
};

```

OnCommand(LONG CommandID)

This event is raised by Solid Edge when a specific command is invoked.

CommandID

The command identifier for the selected command.

OnCommandHelp(LONG hFrameWnd, LONG HelpCommandID, LONG CommandID)

This event is raised by Solid Edge when a specific command's help is invoked.

hFrameWnd

The handle to the application window.

HelpCommandID






The type of help to be requested.

CommandID

The command identifier for the selected command.

OnCommandUpdateUI(LONG CommandID, LONG* CommandFlags, BSTR* MenuItemText, LONG* BitmapID)

This event is raised by Solid Edge when a specific command's UI needs to be updated.

-  seCmdActive_Enabled
-  seCmdActive_Checked
-  seCmdActive_ChangeText
-  seCmdActive_UseDotMark
-  seCmdActive_UseBitmap

CommandID

The command identifier for the affected command.

CommandFlags

Bitwise combination of the enum **SECommandActivation**. Setting this value affects how the control is displayed to the user.

MenuItemText

The text that appears on the button.

BitmapID

The image that appears on the button.

Registration

In addition to the normal requirements to register a COM object, Solid Edge addins have additional registration requirements in order for Solid Edge to recognize them as addins. This means that after the addin dll is registered, you must create additional registry entries. This can be done in code or in an installer.

Since all COM objects must have a unique GUID, I will refer to {ADDIN_GUID} in the following examples.

[HKEY_CLASSES_ROOT\CLSID\{ADDIN_GUID}]

In the root key of your addin, there are two values that you need to create.

- **AutoConnect** - A DWORD value indicating whether or not the addin should be enabled by default. A value of 0 is not enabled by default and a value of 1 is enabled by default.
- **[Locale ID]** - The name of the string value must be a hexadecimal value of a [Locale ID assigned by Microsoft](#). The string value itself is the description of the addin that is displayed in the Solid Edge Add-In Manager. In this example, I am using 0x409 which is 1033 in decimal form and represents the "English - United States" locale.

```
[HKEY_CLASSES_ROOT\CLSID\{ADDIN_GUID}]
"AutoConnect"=dword:00000001
"409"="My Solid Edge AddIn"
```

[HKEY_CLASSES_ROOT\CLSID\{ADDIN_GUID}\Environment Categories]

Underneath the root key of your addin, you need to create an **Environment Categories** key with one or more sub-keys that are environment GUIDS from **\Advanced\include\secatids.h**. In the following example, the addin is registered for the **CATID_SEApplication** and **CATID_SEAllDocumentEnvrionments** environments.

```
[HKEY_CLASSES_ROOT\CLSID\{ADDIN_GUID}\Environment Categories\{26618394-09D6-11D1-BA07-080036230602}]
[HKEY_CLASSES_ROOT\CLSID\{ADDIN_GUID}\Environment Categories\{BAD41B8D-18FF-42C9-9611-8A00E6921AE8}]
```

[HKEY_CLASSES_ROOT\CLSID\{ADDIN_GUID}\Implemented Categories]

Underneath the root key of your addin, you need to create an **Implemented Categories** key with a subkey of the **CATID_SolidEdgeAddIn** GUID as defined in **\Advanced\include\secatids.h**. This is how Solid Edge identifies your COM object as a Solid Edge addin.

```
[HKEY_CLASSES_ROOT\CLSID\{ADDIN_GUID}\Implemented Categories\{26B1D2D1-2B03-11D2-B589-080036E8B802}]
```

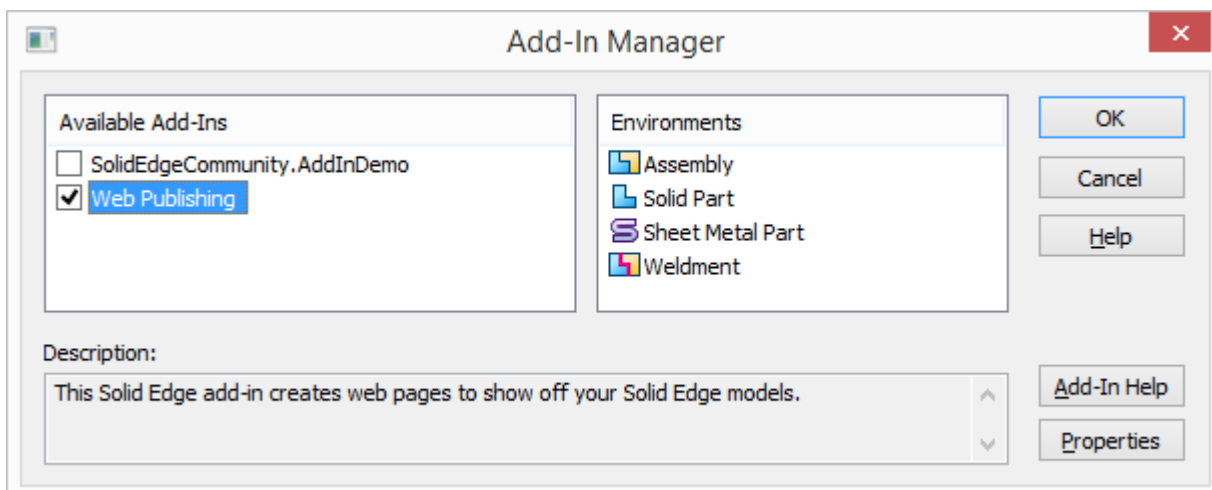
[HKEY_CLASSES_ROOT\CLSID\{ADDIN_GUID}\Summary]

Underneath the root key of your addin, you need to create a **Summary** key with the following value.

- **[Locale ID]** - The name of the string value must be a hexadecimal value of a [Locale ID assigned by Microsoft](#). The string value itself is the summary of the addin that is displayed in the Solid Edge Add-In Manager. In this example, I am using 0x409 which is 1033 in decimal form and represents the "English - United States" locale.

```
[HKEY_CLASSES_ROOT\CLSID\{ADDIN_GUID}\Summary]
"409"="My summary"
```

If you've done everything correctly, your addin should now be available in Solid Edge.



Resources

- [Solid Edge ST7 SDK](#)
- [Solid Edge Developer Community](#)
- [Solid Edge Community on GitHub](#)

License

This article, along with any associated source code and files, is licensed under [The Code Project Open License \(CPOL\)](#)

Written By

Jason Newell


Software Developer (Senior) JasonNewell.NET

 United States

Jason Newell is an Applications Architect for [Ditch Witch](#) and a Siemens PLM Solutions Partner specializing in custom development for [Solid Edge](#). Jason also runs www.jasonnewell.net, a website dedicated to [Solid Edge](#) programming.



Comments and Discussions

 **5 messages** have been posted for this article Visit

<https://www.codeproject.com/Articles/839585/Solid-Edge-ST-AddIn-Architecture-Overview> to post and view comments on this article, or click [here](#) to get a print view with messages.

[Permalink](#)

[Advertise](#)

[Privacy](#)

[Cookies](#)

[Terms of Use](#)

Article Copyright 2014 by Jason Newell
Everything else Copyright © [CodeProject](#),
1999-2023

Web01 2.8:2023-07-24:2