**Application:** ExternalCommand  
**Revit Platform:** All  
**Revit Version:** 2011  
**First Released For:** 2011  
**Programming Language:** C#  
**Skill Level:** Beginning  
**Category:** Basics  
**Type:** ExternalCommand  
  
**Subject:** External Command Registration in Revit  
**Summary:**   
This sample is used to demonstrate RevitAPI user how to use new external command registration more effectively, we provide two samples with functionalities below:

* ***RevitAddInUtility***:

Show user how to use RevitAddInUtility to create\edit .addin manifest file, or retrieve information from manifest file and get installed Revit product information.

* ***New features of External Command registration***

1. Visibility Mode: show user how to control the visibility of each external command base on the different product type and document type.
2. IAvailabilityClass: show user how to enable/disable each external command base on user’s selection or application information.
3. Icon and tooltip: show user how to add icon and tooltip
4. Localization: show user how to localize strings in the addin manifest file

**Classes:**

Autodesk.Revit.UI.IExternalCommand

Autodesk.Revit.UI.IExternalCommandAvailability

Autodesk.RevitAddIns.RevitAddInManifest

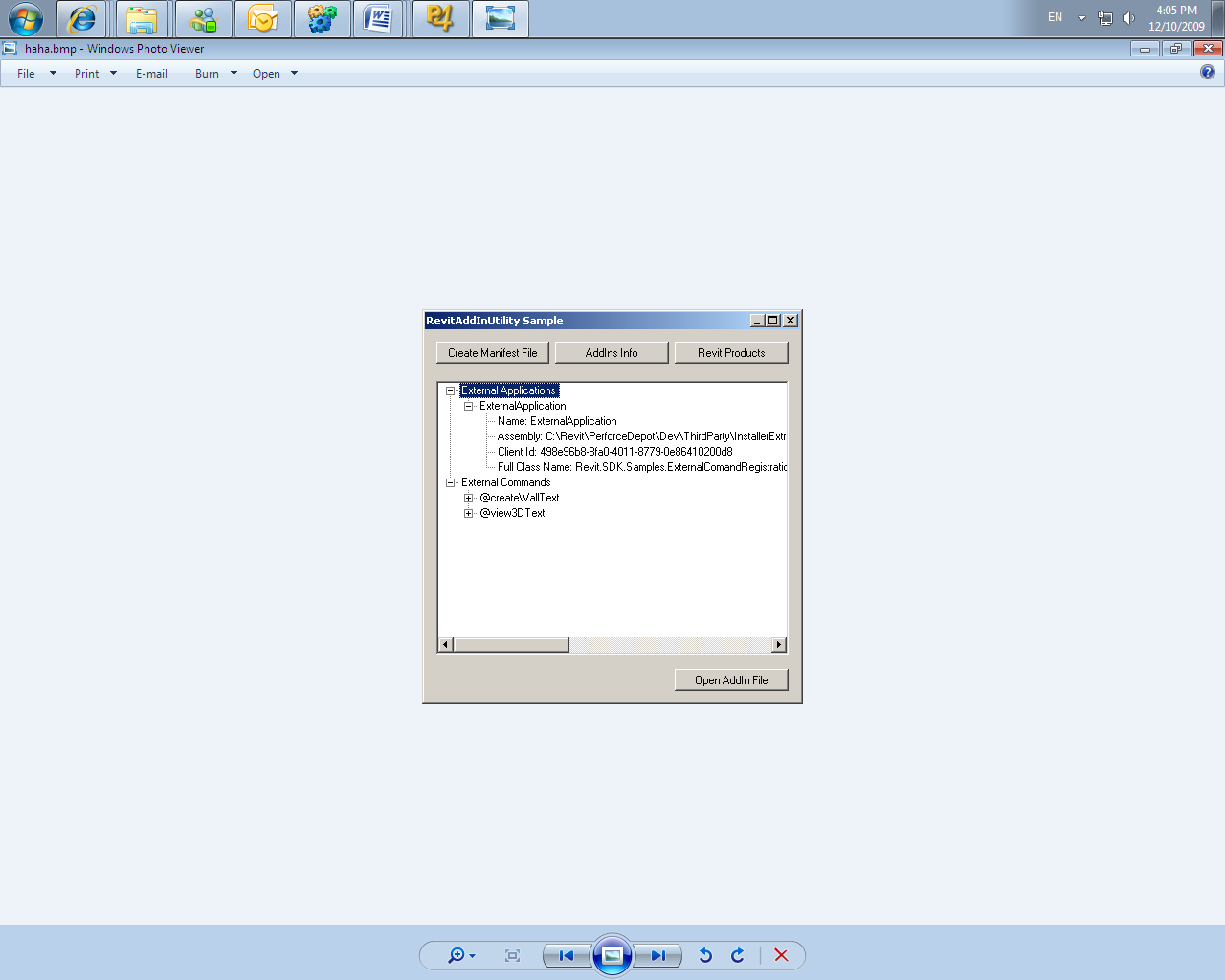
Autodesk.RevitAddIns.RevitAddInApplication

Autodesk.RevitAddIns.RevitAddInCommand

**Description:**

1. ***RevitAddInUtility***

Demonstrate user how to use RevitAddInUtility to write a new .addin manifest file, or how to retrieve information from specific manifest file or folder with manifest files in it. Or so, show user how to get information of installed Revit products in local machine.



Functionalities provided in this sample:

1. Creates a windows form application with three buttons and a tree view to demonstrate functions of RevitAddInUtility.
2. The first button is used to create a new addin manifest file with an external application and an external command in it. Each external command or application should contain all the properties provided by RevitAddInUtility.
3. The second button is used to get external command and application from the new created addin manifest and display all the properties of each command in a tree view.
4. The third button is used to get the information of installed Revit products and corresponding addin manifest folder, this information will be displayed in the tree view too.
5. ***New features of External Command registration***

Demonstrate user how to register external commands with new features provided in Revit.

Functionalities provided in this sample:

1. Provides a console application in which we create an .addin manifest which contains two external commands by RevitAddInUtility.
2. Set “VisibilityMode” node of these two external commands with different values to demonstrate how to visualize external command base on the document type or project type change. Set visibility mode of first command with “[NotVisibleInStructure](http://www.aisto.com/roeder/dotnet/Default.aspx?Target=code://RevitAddInUtility:2011.0.0.0/Autodesk.RevitAddIns.VisibilityMode/NotVisibleInStructure) | [NotVisibleWhenNoActiveDocument](http://www.aisto.com/roeder/dotnet/Default.aspx?Target=code://RevitAddInUtility:2011.0.0.0/Autodesk.RevitAddIns.VisibilityMode/NotVisibleWhenNoActiveDocument)”, Set visibility mode of second command with “[NotVisibleInMEP](http://www.aisto.com/roeder/dotnet/Default.aspx?Target=code://RevitAddInUtility:2011.0.0.0/Autodesk.RevitAddIns.VisibilityMode/NotVisibleInMEP) | [NotVisibleInFamily](http://www.aisto.com/roeder/dotnet/Default.aspx?Target=code://RevitAddInUtility:2011.0.0.0/Autodesk.RevitAddIns.VisibilityMode/NotVisibleInFamily)”.
3. Provides user an assembly with two classes which inherited from IAvailabilityClass interface:
4. First Class: return false from IsCommandAvailabilable(UIApplication^ app, CategorySet^ categories) when user selected a wall. Use CategorySet property to judge user’s selection.
5. Second Class: return false from IsCommandAvailabilable(UIApplication^ app, CategorySet^ categories) when current document is in 3D view.
6. Set “AvailabilityClass” node of these two external commands with class name provided in step 3.
7. Provides resource files for both English and Chinese language. Including both picture and text string to demonstrate the localization, icon and tooltip.
8. Set icon and tooltip image of these two external commands with images above, and set <LanguageType> node as “English\_USA”. And tell user how to change language to “Chinese\_Simplified” to display external command by Chinese.
9. Provides an easy add-on application to bind with each external command.

**Instructions:**

1. Open the ExternalCommand.sln and rebuild solution. (make sure Revit has been installed)
2. Run the new created RevitAddInUtiliySample.exe. Click “Create Manifest File” button in the popup dialogue to create a new addin manifest which contains one external application and two external commands.
3. All the new created external commands and applications will be displayed in the tree view below.
4. Copy the new created addin manifest file (ExternalCommand.Sample.addin) to corresponding addin folder of Revit. (you can check this information using RevitAddInUtiliySample.exe by “Revit Products” button)
5. Run you revit.
6. There will be two external commands under your add-in tab after you opened a project document. The first external command “Create Wall” will be disabled when you selected a wall. And the second external command “3D View” will only be enabled when you in a 3D view.
7. Switch between different product (Architecture, Structure, and MEP) and document (Project, Family, and No Active Document) to check if the visibility mode works.
8. Change the LanguageType node in addin manifest file to “Chinese\_Simplified” to check if localization works well.