**Revit API UI Labs**

**Lab 3 – TaskDialog**

May, 2011 by A. Nagy

Last updated, Date : March 19, 2014

**<VB.NET>**VB.NET Version**</VB.NET>**

**Objective:** In this lab, we will learn how to use the TaskDialog functions. We’ll learn how to:

* Use TaskDialog static Show() function
* Use TaskDialog instance

The following is the breakdown of step by step instructions in this lab:

1. Create a new External Command
2. Pop up messages using TaskDialog.Show()
3. Use TaskDialog instance
4. House creation dialog
5. Summary
6. **Create a new External Command**

We’ll add another external command to the current project.

* 1. Add a new file and define another external command to your project. Let’s name them as follows:
* File name: **3\_TaskDialog.vb (or .cs)**
* Command class name: **UITaskDialog**

(Once again, you may choose to use any names you want here. When you do so, just remember what you are calling your own project, and substitute these names as needed while following the instruction in this document.)

**Required Namespaces:**

Namespaces needed for this lab are:

* Autodesk.Revit.DB
* Autodesk.Revit.UI
* Autodesk.Revit.ApplicationServices
* Autodesk.Revit.Attributes
* Autodesk.Revit.UI.Selection (this is for selection)

Note (VB.NET only): if you are writing in VB.NET and you import namespaces at the project level, (i.e., in the project properties, there is no need to explicitly import in each file.

Let’s declare some variables in the class that will reference the UIApplication and the active UIDocument

**<VB.NET>**  
 Dim \_uiApp As UIApplication

Dim \_uiDoc As UIDocument

Public Function Execute(ByVal commandData As ExternalCommandData, \_

ByRef message As String, \_

ByVal elements As ElementSet) \_

As Result \_

Implements IExternalCommand.Execute

' Get the access to the top most objects. (

‘ we may not use them all in this specific lab.)

\_uiApp = commandData.Application

\_uiDoc = \_uiApp.ActiveUIDocument

' …

**</VB.NET>**

1. **Use TaskDialog static Show() function**

Every now and then we need to inform the user about certain things, and in general we use dialogs for that. In order to better integrate our Add-In into Revit we can use the TaskDialog class to present the user with dialogs that look just like built-in Revit popup dialogs.

TaskDialog offers a static Show() function with many overloads. Since this function is static we do not need to create an instance of TaskDialog prior to calling this function.

All Show() versions require a header string that will be the header of the dialog, and a main instruction string which will be the actual text presented to the user.

Apart from that you can also add buttons to the dialog (Ok/Yes/No/Cancel/Retry/Close) and can also set the default button.

**<VB.NET>**  
 ' (1) simplest of all. title and main instruction. has default [Close]

‘ button at lower right corner.

TaskDialog.Show("Task Dialog Static 1", "Main message")

' (2) this version accepts command buttons in addition to above.

' Here we add [Yes] [No] [Cancel}

Dim res2 As TaskDialogResult

res2 = TaskDialog.Show("Task Dialog Static 2", "Main message", \_

(TaskDialogCommonButtons.Yes Or \_

TaskDialogCommonButtons.No Or TaskDialogCommonButtons.Cancel))

' What did the user pressed?

TaskDialog.Show("Show task dialog", "You pressed: " + res2.ToString)

' (3) this version accepts default button in addition to above.

' Here we set [No] as a default (just for testing purposes).

Dim res3 As TaskDialogResult

Dim defaultButton As TaskDialogResult = TaskDialogResult.Yes

res3 = TaskDialog.Show("Task Dialog Static 3", "Main message", \_

(TaskDialogCommonButtons.Yes Or \_

TaskDialogCommonButtons.No Or TaskDialogCommonButtons.Cancel), \_

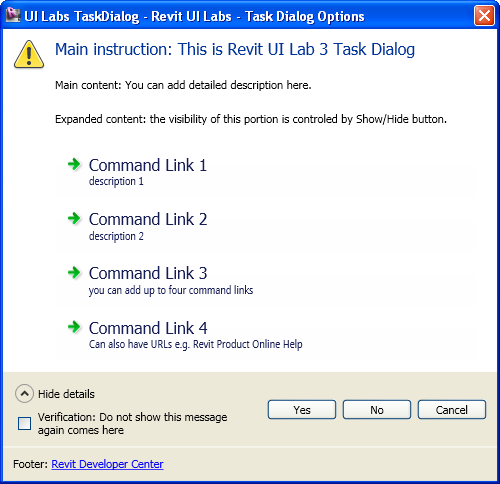
TaskDialogResult.No)

' What did the user pressed?

TaskDialog.Show("Show task dialog", "You pressed: " + res3.ToString)

**</VB.NET>**

1. **Use TaskDialog instance**



If you want to take advantage of the full functionality of TaskDialog then you need to create an instance of it, set its properties and then call the instance’s Show() method.

This way you can set an icon to be shown, add custom options the user can select from and provide hyperlinks to websites that could provide further information for the user.

**</VB.NET>**

' (0) create an instance of task dialog to set more options.

Dim myDialog As New TaskDialog("Revit UI Labs - Task Dialog Options")

' Just declare stepByStep as bool and initialize to True.

If stepByStep Then myDialog.Show()

' (1) set the main area. these appear at the upper portion of the dialog.

'

myDialog.MainIcon = TaskDialogIcon.TaskDialogIconWarning ' or TaskDialogIcon.TaskDialogIconNone.

If stepByStep Then myDialog.Show()

myDialog.MainInstruction = \_

"Main instruction: This is Revit UI Lab 3 Task Dialog"

If stepByStep Then myDialog.Show()

myDialog.MainContent = \_

"Main content: You can add detailed description here."

If stepByStep Then myDialog.Show()

' (2) set the bottom area

myDialog.CommonButtons = TaskDialogCommonButtons.Yes Or TaskDialogCommonButtons.No Or TaskDialogCommonButtons.Cancel

myDialog.DefaultButton = TaskDialogResult.Yes

If stepByStep Then myDialog.Show()

myDialog.ExpandedContent = \_

"Expanded content: the visibility of this portion is controled by Show/Hide button."

If stepByStep Then myDialog.Show()

myDialog.VerificationText = \_

"Verification: Do not show this message again comes here"

If stepByStep Then myDialog.Show()

myDialog.FooterText = \_

"Footer: <a href=""http://www.autodesk.com/developrevit"">Revit Developer Center</a>"

If stepByStep Then myDialog.Show()

' (4) add command links. you can add up to four links

'

myDialog.AddCommandLink(TaskDialogCommandLinkId.CommandLink1, \_

"Command Link 1", "description 1")

If stepByStep Then myDialog.Show()

myDialog.AddCommandLink(TaskDialogCommandLinkId.CommandLink2, \_

"Command Link 2", "description 2")

If stepByStep Then myDialog.Show()

myDialog.AddCommandLink(TaskDialogCommandLinkId.CommandLink3, \_

"Command Link 3", "description 3")

If stepByStep Then myDialog.Show()

myDialog.AddCommandLink(TaskDialogCommandLinkId.CommandLink4, \_

"Command Link 4", "you can add up to four command links")

'If stepByStep Then myDialog.Show()

' Show it.

Dim res As TaskDialogResult = myDialog.Show()

If TaskDialogResult.CommandLink4 = res Then

Dim process As New System.Diagnostics.Process()

process.StartInfo.FileName = \_

"http://wikihelp.autodesk.com/Revit/enu/2013"

process.Start()

End If

TaskDialog.Show("Show task dialog", "The last action was: " & \_

res.ToString())

**</VB.NET>**

1. **Use TaskDialog instance**

Use TaskDialog to prompt the user if he/she wants to create a house interactively or simply create the default house and also let the dialog be cancelled. Depending on the selected option run the interactive house creation or default house creation from the previous lab, or don’t do anything if the dialog got cancelled.

1. **Summary**

In this lab, we learned how to use the TaskDialog functions. We’ve learned how to:

* Use TaskDialog static Show() function
* Use TaskDialog instance

Autodesk Developer Network