Revit API UI Labs

Lab 3 – TaskDialog

May, 2011 by A. Nagy

Last Updated, Date : March 03, 2018

<C#>C# Version**</C#>**

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

1. Create a new External Command

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

1.1 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

```
<C#>
  public class UITaskDialog : IExternalCommand
  {
    UIApplication _uiApp;
    UIDocument _uiDoc;

    public Result Execute(
      ExternalCommandData commandData,
      ref string message,
      ElementSet elements )
    {
      _uiApp = commandData.Application;
      _uiDoc = _uiApp.ActiveUIDocument;

      // …
</C#>
```

2. 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.

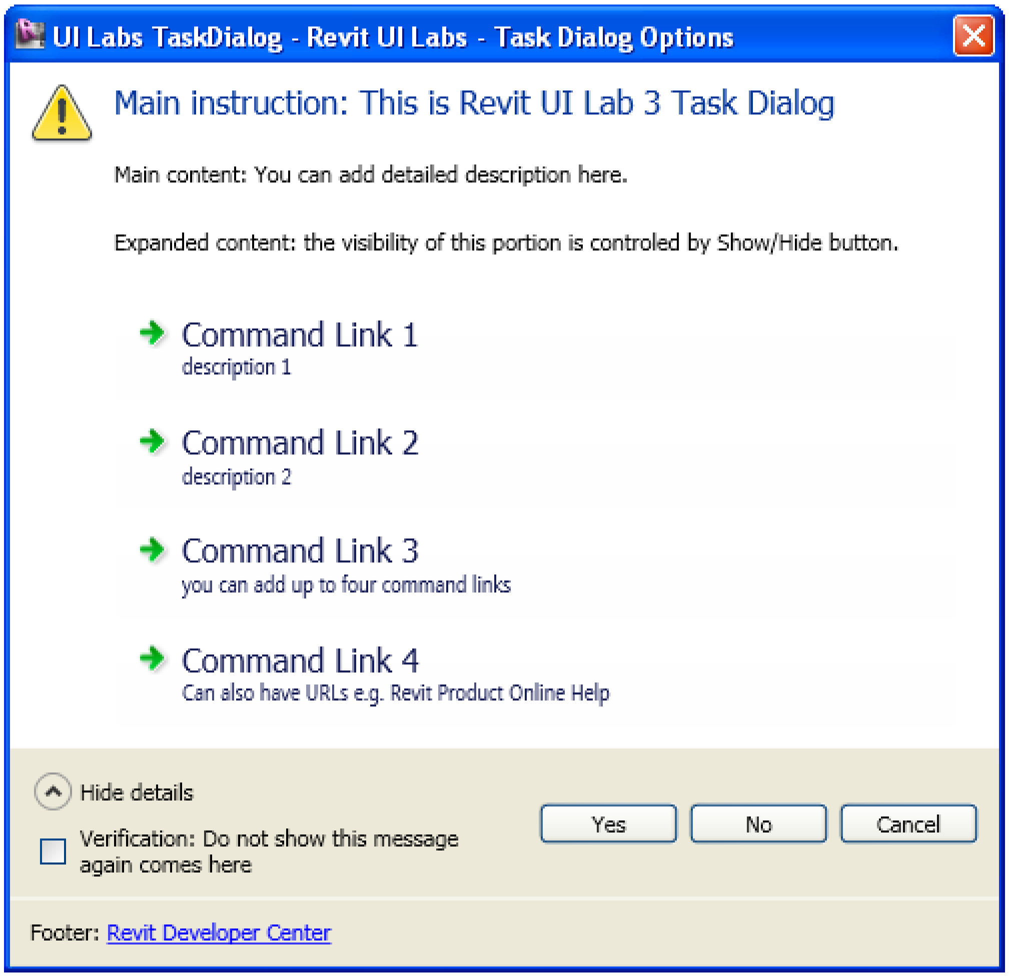
TaskDialog offers a static Show() function with many overloads. Since this function is static we do not need to create an instance of TaskDialog prior to calling this function.

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.

```
<C#>
      // (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]
      TaskDialogResult res2 = default( TaskDialogResult );
      res2 = TaskDialog.Show( "Task Dialog Static 2", "Main message",
( TaskDialogCommonButtons.Yes | TaskDialogCommonButtons.No |
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).

      TaskDialogResult res3 = default( TaskDialogResult );
      TaskDialogResult defaultButton = TaskDialogResult.No;
      res3 = TaskDialog.Show( "Task Dialog Static 3", "Main message",
( TaskDialogCommonButtons.Yes | TaskDialogCommonButtons.No |
TaskDialogCommonButtons.Cancel ), defaultButton );

      // What did the user press?

      TaskDialog.Show("Show task dialog", "You pressed: " + res3.ToString());
```
</C#>

3. Use TaskDialog instance


UI Labs TaskDialog - Revit UI Labs - Task Dialog Options

Main instruction: This is Revit UI Lab 3 Task Dialog

Main content: You can add detailed description here.

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

Command Link 1
description 1

Command Link 2
description 2

Command Link 3
you can add up to four command links

Command Link 4
Can also have URLs e.g. Revit Product Online Help

Hide details

Verification: Do not show this message again comes here

Yes No Cancel

Footer: Revit Developer Center

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.

</C#>
```
      // (0) create an instance of task dialog to set more options.
      TaskDialog myDialog = new TaskDialog( "Revit UI Labs - Task Dialog 
Options" );
      // Just declare stepByStep as bool and initialize to true.
      if( stepByStep ) myDialog.Show(); // Just declare stepByStep

      // (1) set the main area. These appear at the upper portion
      // of the dialog.

      myDialog.MainIcon = TaskDialogIcon.TaskDialogIconWarning;
      // or TaskDialogIcon.TaskDialogIconNone.
      if( stepByStep ) myDialog.Show();

      myDialog.MainInstruction =
        "Main instruction: This is Revit UI Lab 3 Task Dialog";
      if( stepByStep ) myDialog.Show();
      myDialog.MainContent =
        "Main content: You can add detailed description here.";
      if( stepByStep ) myDialog.Show();

      // (2) set the bottom area

      myDialog.CommonButtons = TaskDialogCommonButtons.Yes |
TaskDialogCommonButtons.No | TaskDialogCommonButtons.Cancel;
      myDialog.DefaultButton = TaskDialogResult.Yes;
      if( stepByStep ) myDialog.Show();

      myDialog.ExpandedContent = "Expanded content: the visibility of this
portion is controled by Show/Hide button.";
      if( stepByStep ) myDialog.Show();

      myDialog.VerificationText =
        "Verification: Do not show this message again comes here";
      if( stepByStep ) myDialog.Show();

      myDialog.FooterText = "Footer: <a
href=\"http://www.autodesk.com/developrevit\">Revit Developer Center</a>";
      if( stepByStep ) myDialog.Show();

      // (4) add command links. You can add up to four links

      myDialog.AddCommandLink( TaskDialogCommandLinkId.CommandLink1,
        "Command Link 1", "description 1" );
      if( stepByStep ) myDialog.Show();
      myDialog.AddCommandLink( TaskDialogCommandLinkId.CommandLink2,
        "Command Link 2", "description 2" );
      if( stepByStep ) myDialog.Show();
      myDialog.AddCommandLink( TaskDialogCommandLinkId.CommandLink3,
       "Command Link 3",
"you can add up to four command links" );
      if( stepByStep ) myDialog.Show();
      myDialog.AddCommandLink( TaskDialogCommandLinkId.CommandLink4,
        "Command Link 4",
        "Can also have URLs e.g. Revit Product Online Help" );
      //if (stepByStep) myDialog.Show();

      // Show it.
      TaskDialogResult res = myDialog.Show();
      if( TaskDialogResult.CommandLink4 == res )
      {
        System.Diagnostics.Process process =
          new System.Diagnostics.Process();

        process.StartInfo.FileName =
          "http://wikihelp.autodesk.com/Revit/enu/2022";
        process.Start();
      }

      TaskDialog.Show("Show task dialog", "The last action was: " +
        res.ToString());
</C#>
```

4. 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.

5. 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