MS Office PowerPoint Plugin

Pre-Request

* Visual Studio (I am working on VS-2013).
* MS Power Point (I am working on MS-Power Point 2013).

Assumptions

* I assume you have good knowledge about c#.

Index (What I am going to cover)

* How to set up power point plugin project in visual studio.
* How to show your plugin in power point plugin tab.
* How to change image of your icon.
* How to use resource folder.
* How to invoke your methods.
* How to display result on power point.
* How to read slide data.
* How to write in power point file.

How to set up power point plugin project in visual studio?

File => New Project => Templates => Visual C# => Office/Share Point => PowerPoint 2013 Add-In

Check screenshot => Project Setup

How to show your plugin in power point plugin tab?

For this you to create a ribbon.

Right Click on your project => select add new item => Ribbon (XML)

Check screenshot => ribbon

After adding this, visual studio open a <ribbon name>.cs file and when you read the comment you will get to know what to do next.

Example

// TODO: Follow these steps to enable the Ribbon (XML) item:

// 1: Copy the following code block into the ThisAddin, ThisWorkbook, or ThisDocument class.

// protected override Microsoft.Office.Core.IRibbonExtensibility CreateRibbonExtensibilityObject()

// {

// return new Ribbon1();

// }

// 2. Create callback methods in the "Ribbon Callbacks" region of this class to handle user

// actions, such as clicking a button. Note: if you have exported this Ribbon from the Ribbon designer,

// move your code from the event handlers to the callback methods and modify the code to work with the

// Ribbon extensibility (RibbonX) programming model.

// 3. Assign attributes to the control tags in the Ribbon XML file to identify the appropriate callback methods in your code.

// For more information, see the Ribbon XML documentation in the Visual Studio Tools for Office Help.

1. So first you have to copy the code from first comment and paste it into ThisAddin class
2. Now we are adding a button in ribbon to trigger an event from PowerPoint

Paste this code into your xml or you can write your own code: D

<?xml version="1.0" encoding="UTF-8"?>

<customUI xmlns="http://schemas.microsoft.com/office/2009/07/customui" onLoad="Ribbon\_Load">

<ribbon>

<tabs>

<tab idMso="TabAddIns">

<group id="Mygroup" label="TestAdd-in" >

<button id="AddAnimationButton"

label="Check"

size="large"

getImage="GetImage"

supertip="Check"

enabled="true"

onAction="AddAnimationButtonClick"/>

</group>

</tab>

</tabs>

</ribbon>

</customUI>

Now run the program you can see one ADD-IN tab and inside this tab you can see one button named check.

Here you can see getImage tag calling a function "GetImage" to get the icon image, so how it’s working …

We have inbuilt resource file. This file can save any object in xml format.

So first we have to insert icon file in resource file.

Type resource in your solution explorer search box

Now you can see Resources.resx file, open this file.

Click on Add Resources => add existing file add one image file.

Now you have to implement "GetImage" method

This is the function of ribbon so you have to implement this into Ribbon.cs file.

This function takes one argument which is Microsoft.Office.Core.IRibbonControl

And in return you have to return bitmap file.

Example:

public Bitmap GetImage(Office.IRibbonControl control)

{

return new Bitmap(PluginPowerPoint.Properties.Resources.icon);

}

Now run the program and see the magic.

That’s great

But it’s not working why????? Hay hay you didn’t implement any action on this button so let’s implement.

You can see “ onAction="AddAnimationButtonClick"/> “ line on ribbon.xml file so this is the trigger point and you have to implement this method in your ribbon.cs file.

Ok I want to display a message box saying “Hello world”.

Copy and paste this method in your ribbion.cs file. All the function takes IRibbonControl interface object as argument to control things.

public void AddAnimationButtonClick(Office.IRibbonControl control)

{

MessageBox.Show("Hello World");

}

Now run the code ..

..

..

..

Hayyy what happened ….

Hurry success

Deep Dive Into your code

I want to tell you something guys till now are only setting up our project. Now we have to involve in real mess.

What we need

1. Slide handle of every slide
2. Handle of every object in the every slide

For first thing you need

Microsoft.Office.Interop.PowerPoint.Slides object

How can you get this?

It’s one line code only:

Globals.slideQAddIn.Application.ActivePresentation.Slides (in power point plugin only)

This syntax will give you array of slide object.

Let’s play with it.

Previously we show “Hello World ” on button click. Can we display number of slide instead of this?

Yaa why not

If above syntax gives us an array then we can show the array count so replace your code with this code:

public void AddAnimationButtonClick(Office.IRibbonControl control)

{

MessageBox.Show(Globals.ThisAddIn.Application.ActivePresentation.Slides.Count.ToString());

}

Now run the application

Olla,,,,,,….

What’s your count?? ;)

Come to second point

Now you have slide object, and now you need object of every shape.

For this you have to extract shapes property from slide object, and iterate throw a loop like:

foreach (PPT.Shape shape in slide.Shapes)

{

}

Where PPT is “using PPT = Microsoft.Office.Interop.PowerPoint;”

So now you have shape objects but is not solved here what if you have group shape (group of many shapes) in this case from above code will gives you only group object and one more worse case what if you have placeholder in your slide.

Q. Why placeholder is the worst case?

ANS. If you have group object you can easily get the inherit objects only with the help of single loop.

But if you have placeholder you can’t first you have to judge type of object inside the placeholder then you can fetch the object.

So what should we do?

Here is the plan

1. We have to use recursion for the extraction of every single shape in the slide.
2. We should know how to compare type of object.

I’m not going to teach you recursion let’s start with point number 2.

To get the type of shape you need to visit https://msdn.microsoft.com/en-us/library/office/ff860759.aspx page. And you have to search about enum MsoTriState.

Let’s see how its work..

# First we will see how to extract shape from group object

There is two way to do this

1. Ungroup all the shapes
2. With simple iteration of shape object property

First we have to judge whether its group item or not

You can do this with MsoShapeType as I already mentioned in the link

Example:

if (shape.Type == MsoShapeType.msoGroup)

{

}

Type is the property of shape object and in right hand side MsoShapeType enum is there.

After this condition you have to iterate all the shapes contained by shape object with the help of “GroupItems” property.

Example

foreach (PPT.Shape myShape in shape.GroupItems)

{

}

Hola done.

# Now come to the second shape (worse case) “Placeholder”

In this first you have to judge what type of object/shape placeholder having.

You have to use same approach as above I mentioned earlier.

I am giving you a function which cover most of the shapes.

private void extractInfoFromShape(PPT.Shape shape)

{

if (shape. .PlaceholderFormat.ContainedType == MsoShapeType.msoGroup)

{

}

else if (shape. .PlaceholderFormat.ContainedType == MsoShapeType.msoSmartArt )

{

}

else if (shape. .PlaceholderFormat.ContainedType == MsoShapeType.msoTable)

{

}

else if (shape. .PlaceholderFormat.ContainedType == MsoShapeType.msoPlaceholder)

{

}

}

Ok now you can check the type of shape now what right now I don’t know how to extract shape directly so you have to hit and trial for every shape.

Now if you want to check what the shape object contained you can use “Has” property.

Like if you want to check TextFrame in a shape you can check like this:

if (shape.HasTextFrame == MsoTriState.msoTrue)

{

extractInfo(shape);

}

Let’s see a live example

In this example we will show the total number of smart Art shape nodes in a PowerPoint file.

What we need for this

1. Every slide object
2. Every shape object

First try yourself ..

..

..

..

..

..

Ok then hope you did it.

Append your project

Note: I am not going to maintain code quality here.

So add a class here named counter (as per your wish)

Create a method which takes Slides interface object as an argument.

Now follow the stapes

1. Iterate every slide
2. Use recursive method to extract every single shape object
3. Check for three shape SmartArt, group, Placeholder
4. If it’s SmartArt count the nodes
5. If its group or placeholder extract all the shapes
6. Again start with step 3

Here is the code for this

class counter

{

public int detectSmartArt(Slides slides)

{

int count = 0;

GetCount(slides, ref count);

return count;

}

void GetCount(Slides slides,ref int count)

{

foreach (Slide slide in slides)

{

extractSlideInfo( slide,ref count);

}

}

private void extractSlideInfo(Slide slide,ref int count)

{

foreach (PPT.Shape shape in slide.Shapes)

extractInfoFromShape(shape,ref count);

}

private void extractInfoFromShape(PPT.Shape shape,ref int count)

{

if (shape.Type == MsoShapeType.msoGroup)

{

GetDataFromGroupItem(shape,ref count);

}

else if (shape.Type == MsoShapeType.msoSmartArt)

{

GetDataFromSmartArt(shape,ref count);

}

else if (shape.Type == MsoShapeType.msoPlaceholder && shape.PlaceholderFormat.ContainedType == MsoShapeType.msoSmartArt)

{

GetDataFromPlaceHolder(shape,ref count);

}

}

private void GetDataFromPlaceHolder(PPT.Shape shape,ref int count)

{

try

{

GetDataFromSmartArt(shape, ref count);

}

catch

{

try

{

GetDataFromGroupItem(shape, ref count);

}

catch

{

}

}

}

private void GetDataFromSmartArt(PPT.Shape shape,ref int count)

{

SmartArtNodes nodes = shape.SmartArt.AllNodes;

foreach (SmartArtNode node in nodes)

{

count++;

}

}

private void GetDataFromGroupItem(PPT.Shape shape,ref int count)

{

foreach (PPT.Shape myShape in shape.GroupItems)

extractInfoFromShape(myShape,ref count);

}

}

And ribbon.cs class replace this method

public void AddAnimationButtonClick(Office.IRibbonControl control)

{

counter contobj = new counter();

MessageBox.Show( contobj.detectSmartArt( Globals.ThisAddIn.Application.ActivePresentation.Slides).ToString());

}

So what’s your count ;)

What if you want to show your output in batter way. What will we do?

We can use pane for this.

Pane use to show some information and for user interaction.

How we can use the pane let’s see..

1. Add a wpf user control in your project
   1. Add a label in this control
2. In your xaml.cs file create a public label object
   1. Like : public static Label counter ;
3. Initialize this object with your ui label object
4. Bind your ui label with this object.

Code :

XAML CODE

<UserControl x:Class="PluginPowerPoint.Display"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="300">

<Grid Background="#FFFDFDFD">

<Label x:Name="count" HorizontalAlignment="Left" Margin="55,74,0,0" VerticalAlignment="Top" Height="142" Width="171" HorizontalContentAlignment="Center" VerticalContentAlignment="Center" Content="{Binding counter}"/>

</Grid>

</UserControl>

C# code XAML.CS file

public static Label counter ;

public Display()

{

InitializeComponent();

counter = count;

}

1. Now add WinForm User control in your project
2. Build your project.
3. You can see your above wpf control in your toolbox dock this into your winform controller.
4. Now add two private properties in your thisaddin class
   1. private BackPane panebase;
   2. private static Microsoft.Office.Tools.CustomTaskPane TaskPaneObj;
5. For controlling task pane from outside you have to make a public property of taskpane object.

public Microsoft.Office.Tools.CustomTaskPane TaskPane

{

get

{

return TaskPaneObj;

}

}

1. Now you have to initialize this pane at the time of plugin startup so you can see a method named ThisAddIn\_Startup is there in your thisaddin class.
2. Initialize all the object in ThisAddIn\_Startup method
   1. panebase = new BackPane();
   2. TaskPaneObj = this.CustomTaskPanes.Add(panebase, "SmartArt Count");
   3. TaskPaneObj.Visible = false;
3. As you can see at time of taskpane object initialization you have to pass one winform control object and name of the task pane.
4. Now come to ribbon.cs file.
5. Go to your cbutton\_click\_event method (“AddAnimationButtonClick”)
6. Use your wpf public label and set the value of label.
7. Update the layout.
8. And set the visibility to true of taskpane with the help of global object.

CODE:

counter contobj = new counter();

int count= contobj.detectSmartArt( Globals.ThisAddIn.Application.ActivePresentation.Slides);

Display.counter.Content= count;

Display.counter.UpdateLayout();

Globals.ThisAddIn.TaskPane.Visible = true;

Now run your program ..

Hay what happened.

So what’s your count ;)

# Now we have to write in the slide