

Forum:[Hack Forums](#)

Thread: ["Advanced" Plugin System](#)

imi-tat0r - November 3, 2014, 4:45 am

Hey there.

First of all you should read the tutorial by TheUnknownProgrammer, because this is based on his code or it extends his code a bit.

You can find his tutorial here: <https://hackforums.net/showthread.php?tid=2169571>

In this tutorial I will show you how you can extend his Pluginsystem a bit more.
We will be adding a Tabpage for each Plugin we run, we will be able to load as many plugins as we want and we will be able to communicate from AND to the Plugin.

Remember this is very basic so you can learn from it but it is no way near perfect.

Step 1 - Designing the Main application

At first we will create our Host application which will run all our Plugins. Mine looks like this:

The textbox under the "Run Plug-in" Button will be filled with the Pluginname so we know which one we want to run/close.

The rest should be selfexplaining.

Step 2 - The Connector Library

Our goal is, to create a new Tabpage whenever we load and run a plugin and to delete the Tabpage when we close the Plugin.

Also we want to communicate from the Mainapplication to the Plugin and vice versa

Step 3 - The Connector in the Main Application

After creating the connector library, save and build it and go back to your main application. Add your Plugin connector library reference to the project and create a new class, called Connector. Then add the following code to implement the IConnector interface.

Code Imports PluginConnector

```
Public Class Connector  
    Implements IConnector
```

```
    Dim T As New Dictionary(Of String, TabPage)  
    Dim frm As New Form1
```

```
    Public Sub New(ByVal frm As Form1)
```

```
Me.frm = frm
```

```
End Sub
```

After pressing enter, our subroutines should show up being empty. Now we can fill them with code.

Step 4 - Creating the Plugin

As we are adding the Form we create now to the Tabpage we add in the Main Application, the Form should be borderless and the exact same size than the TabPages are in our Main Application. For me this is: 544; 324

My form looks like this:

Now add a new Class like this:

```
Code Public Class PluginStart  
    Implements IPlugin
```

```
    Public frm As New Form1
```

And the Subs "StartPlugIn" and "DataToPlugin" should appear. In the class of the Form we have to give the buttons something to do.

It is important that, wherever I used "PluginTest", you use your Pluginname. And it always has to be the same name or it might fail.

Our Class "PluginStart" should look like this

DataToPlugin will just reassemble our give Object array to Strings.

Step 5 - Finishing the Main application

At the imports we have to use

```
Code Imports System.Reflection
```

Now on our Load Plugin button we will add this code:

```
Code Dim ofd As New OpenFileDialog
```

```
    If ofd.ShowDialog = Windows.Forms.DialogResult.OK Then
```

```
        'load the assembly
```

```
        Dim pluginassembly As Assembly = Assembly.LoadFile(ofd.FileName)
```

```
        'get the StartOfPlugin class by using GetType("<typename>")
```

```
        Dim pluginclass As IPlugin =
```

```
        CType(Activator.CreateInstance(pluginassembly.GetType(pluginassembly.GetName.Name &  
        ".PluginStart")), IPlugin)
```

```
        Plugs.Add(pluginassembly.GetName.Name,
```

```
        CType(Activator.CreateInstance(pluginassembly.GetType(pluginassembly.GetName.Name &  
        ".PluginStart")), IPlugin))
```

End If

On our Run Plugin Button we add this code:

Code `Plugs(TextBox2.Text).StartPlugin(Connector)TextBox2.Text` is the little Box in our Main application where we have to put in the Pluginname.

On our Close Plugin Button we will do like this:

Code `Connector.Disconnect(TextBox2.Text)`

and on the Send Data to Plugin Button this is our code:

Code `Dim td As Object() = TextBox1.Text.Split(",")`

`Plugs(td(0)).DataToPlugin(td)`

now your code should look like this

Step 6 - Testing

Opening, Running and sending data from a Plugin

Spoiler ([Click to View](#))

Sending data to a Plugin and Closing it

Spoiler ([Click to View](#))

SIM-1847363 - November 3, 2014, 5:10 am

Great tutorial i been looking into plugin systems for awhile now and i can say this tutorial is a great one, nice work!

imi-tat0r - November 3, 2014, 5:12 am

(Nov 3, 2014 06:10 AM)Nettro Wrote: ([CITE](#)) Great tutorial i been looking into plugin systems for awhile now and i can say this tutorial is a great one, nice work!

Thank you for the kind words :)

But as I said it is based on TheUnknownProgrammer's Tutorial

0xfd - November 3, 2014, 6:29 am

great tutorial, thanks for share. keep it up :)

imi-tat0r - November 3, 2014, 11:06 am

(Nov 3, 2014 07:29 AM)sifenks Wrote: ([CITE](#)) great tutorial, thanks for share. keep it up :)

Thx man :)

I just hope that as many people as possible will see this. Cause I never saw a thread like this except for TUP's thread

geekmorpheus403 - November 3, 2014, 11:54 am

So the plugins has to be specific or it can be any?

imi-tat0r - November 3, 2014, 1:53 pm

Read the tutorial please...

The plugins have to be coded for the pluginssystem. They have to use the same connector Class so YES they have to be specific

Washi - November 3, 2014, 4:18 pm

Oh man good ol' times. Not sure what I was thinking at writing this line, since it is quite sloppy and unreliable, o well it's old ;o.

PHP Code

<?php

```
Dim pluginclass As IPlugin = CType(Activator.CreateInstance(pluginassembly.GetType(
pluginassembly.GetName.Name & ".PluginStart")), IPlugin)
```

I have to say your name suites you quite well. Same thread layout and very similar code, with only slight modifications. I don't mind however that much. It is a pretty old system I made like two or three years ago. Moreover, you mentioned my name, giving me enough credit ;).

Some points of interest, I wouldn't go with a form if you are making it borderless and add it to a tab page anyway. Why not a UserControl ? And why are you creating an instance of the IPlugin derived class twice?

imi-tat0r - November 3, 2014, 5:00 pm

Well I used the same Thread layout as it is just like an "addon" to your Pluginsystem so I thought this might be a good way to show this. But as you already said I gave you credits because your post was the first time I actually learned ANYTHING about Plugins.

Usercontrol is a good Idea but the Form was a bit easier to do and to explain for me as English is not my first language, but that should be obvious :D

The double instance :D yeah I didn't notice. I could have just used
Code Plugs.Add(pluginassembly.GetName.Name, pluginclass)

Thanks for pointing out :) wouldn't have noticed that :D

nuk - November 4, 2014, 5:53 am

WOW! This is a GREAT tutorial on this... Props to you for taking the time to write this up and share it with everyone :)

imi-tat0r - November 4, 2014, 12:34 pm

(Nov 4, 2014 06:53 AM)nuk Wrote: [\[CITE\]](#) WOW! This is a GREAT tutorial on this... Props to you for taking the time to write this up and share it with everyone :)

Thanks for the kind words and the Rep :P

Don't forget to thank TUP for his tutorial. This is just an addon