

Stack Overflow is a community of 4.7 million programmers, just like you, helping each other.

Join the Stack Overflow community to:

Join them; it only takes a minute:


Join

Ask
programming
questions

Answer and help
your peers


Get recognized for your
expertise

How to add a custom Ribbon tab using VBA?



Use the Industry Leading SolverSDK for Conventional Optimization, Simulation and Stochastic Optimization.

[Download Now!](#)



FrontlineSolvers®
Powered by Solvers for Optimization and Analytics

I am looking for a way to add a custom tab in the Excel ribbon which would carry a few buttons. I chanced on some resources addressing it via Google but all look dodgy and outrageously complicated.

What is a quick and simple way to do that ? I'd like the new tab to get loaded when my VBA gets loaded into Excel..

UPDATE : I tried this example from [here](#) but get an "object required" error on the last instruction :


```
Public Sub AddHighlightRibbon()  
Dim ribbonXml As String  
  
ribbonXml = "<mso:customUI xmlns:mso=""http://schemas.microsoft.com/office/2009/07/  
/customui"">"  
ribbonXml = ribbonXml + " <mso:ribbon>"  
ribbonXml = ribbonXml + " <mso:qat/>"  
ribbonXml = ribbonXml + " <mso:tabs>"  
ribbonXml = ribbonXml + " <mso:tab id=""highlightTab"" label=""Highlight""  
insertBeforeQ=""mso:TabFormat"">"  
ribbonXml = ribbonXml + " <mso:group id=""testGroup"" label=""Test""  
autoScale=""true"">"  
ribbonXml = ribbonXml + " <mso:button id=""highlightManualTasks"" label=""Toggle  
Manual Task Color"" <  
ribbonXml = ribbonXml + " imageMso=""DiagramTargetInsertClassic""  
onAction=""ToggleManualTasksColor""/>"  
ribbonXml = ribbonXml + " </mso:group>"  
ribbonXml = ribbonXml + " </mso:tab>"  
ribbonXml = ribbonXml + " </mso:tabs>"  
ribbonXml = ribbonXml + " </mso:ribbon>"  
ribbonXml = ribbonXml + "</mso:customUI>"  
  
ActiveProject.SetCustomUI (ribbonXml)  
End Sub
```

[excel](#) [excel-vba](#) [excel-2007](#) [ribbonx](#)

edited Sep 11 '15 at 20:59

 [Siddharth Rout](#)
84k 10 96 130

asked Jan 13 '12 at 12:49

 [Jerome](#)
3,072 13 49 100

Please confirm which Excel version you are using? – [Siddharth Rout](#) Jan 13 '12 at 14:41

I am using 2007 – [Jerome](#) Jan 13 '12 at 14:56

4 Answers

AFAIK you cannot use VBA Excel to create custom tab in the Excel ribbon. You can however hide/make visible a ribbon component using VBA. Additionally, the link that you mentioned above is for MS Project and not MS Excel.

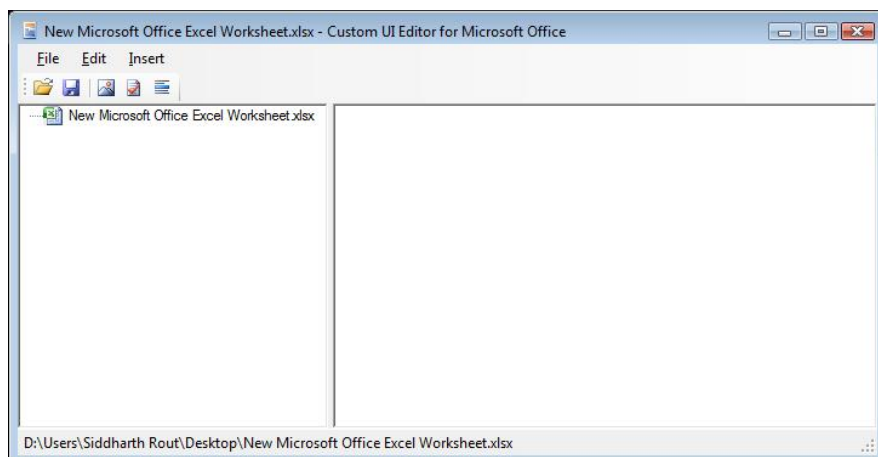
I create tabs for my Excel Applications/Add-Ins using this free utility called [Custom UI Editor](#).

Edit: To accommodate new request by OP

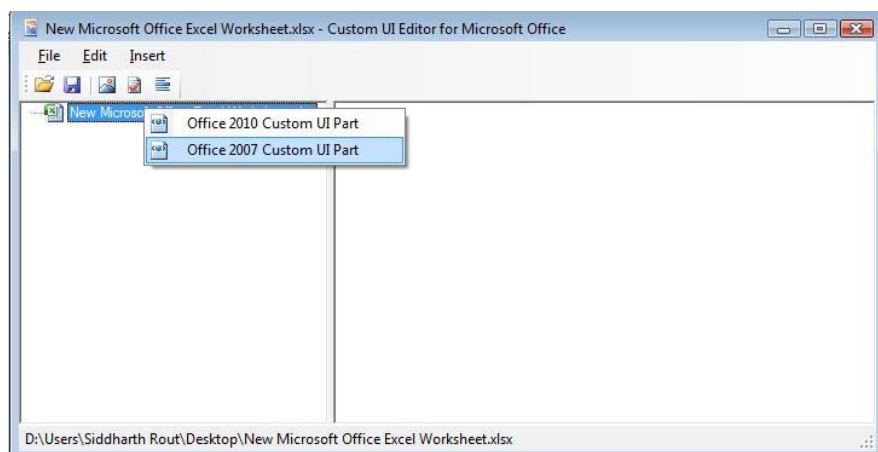
Tutorial

Here is a short tutorial as promised:

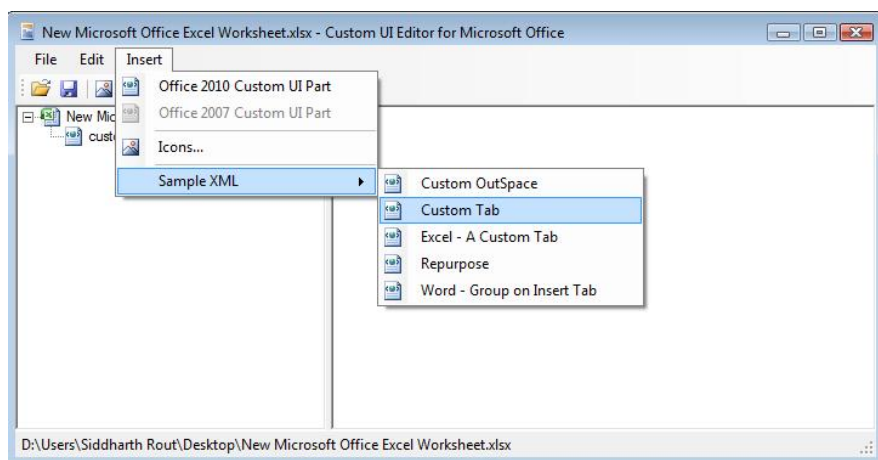
1. After you have installed the Custom UI Editor (CUIE), open it and then click on File | Open and select the relevant Excel File. Please ensure that the Excel File is closed before you open it via CUIE. I am using a brand new worksheet as an example.



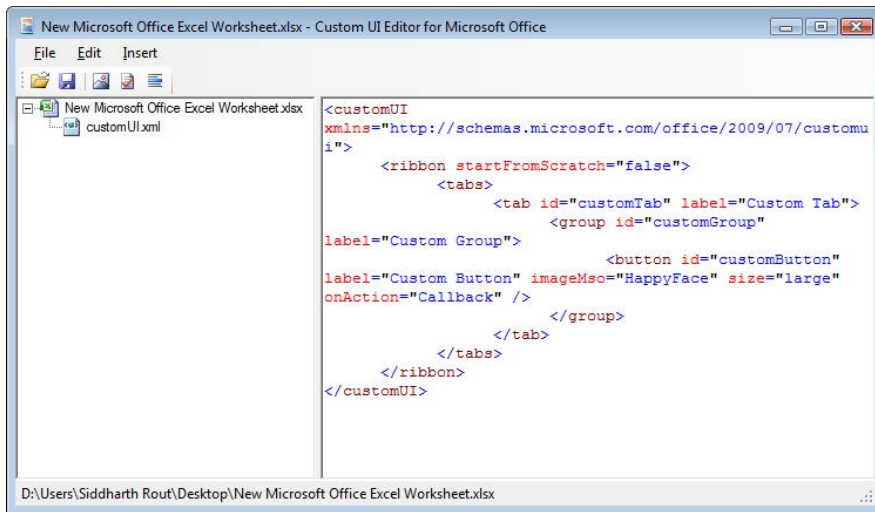
2. Right click as shown in the image below and click on "Office 2007 Custom UI Part". It will insert the "customUI.xml"



3. Next Click on menu Insert | Sample XML | Custom Tab. You will notice that the basic code is automatically generated. Now you are all set to edit it as per your requirements.



4. Let's inspect the code



label="Custom Tab" : Replace "Custom Tab" with the name which you want to give your tab. For the time being let's call it "Jerome".

The below part adds a custom button.

```
<button id="customButton" label="Custom Button" imageMso="HappyFace" size="large"
onAction="Callback" />
```

imageMso : This is the image that will display on the button. "HappyFace" is what you will see at the moment. [You can download more image ID's here.](#)

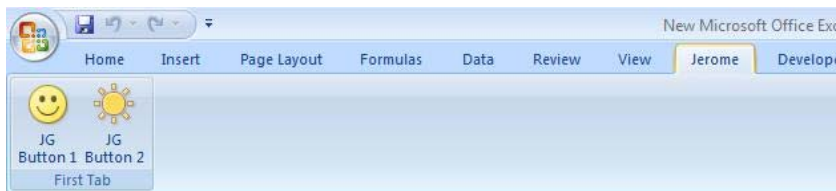
onAction="Callback" : "Callback" is the name of the procedure which runs when you click on the button.

Demo

With that, let's create 2 buttons and call them "JG Button 1" and "JG Button 2". Let's keep happy face as the image of the first one and let's keep the "Sun" for the second. The amended code now looks like this:

```
<customUI xmlns="http://schemas.microsoft.com/office/2006/01/customui">
<ribbon startFromScratch="false">
<tabs>
<tab id="MyCustomTab" label="Jerome" insertAfterMso="TabView">
<group id="customGroup1" label="First Tab">
<button id="customButton1" label="JG Button 1" imageMso="HappyFace" size="large"
onAction="Callback1" />
<button id="customButton2" label="JG Button 2" imageMso="PictureBrightnessGallery"
size="large" onAction="Callback2" />
</group>
</tab>
</tabs>
</ribbon>
</customUI>
```

Delete all the code which was generated in CUIE and then paste the above code in lieu of that. Save and close CUIE. Now when you open the Excel File it will look like this:



Now the code part. Open VBA Editor, insert a module, and paste this code:

```
Public Sub Callback1(control As IRibbonControl)

    MsgBox "You pressed Happy Face"

End Sub

Public Sub Callback2(control As IRibbonControl)

    MsgBox "You pressed the Sun"

End Sub
```

Save the Excel file as a macro enabled file. Now when you click on the Smiley or the Sun you will see the relevant message box:



Hope this helps!

edited Mar 24 '14 at 19:36



eykanal

11k 9 49 78

answered Jan 13 '12 at 15:18



Siddharth Rout

84k 10 96 130

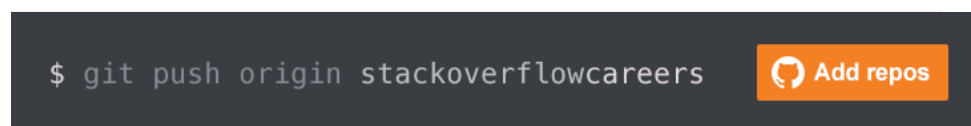
5 Yes :) Ron has plenty of examples in his website. rondebruin.nl/ribbon.htm – Siddharth Rout Jan 13 '12 at 15:29

1 Do you want to create a new Tab? If yes then provide more details and I will give you the XML Code :) – Siddharth Rout Jan 13 '12 at 15:41

2 Done :) Please refresh the page. – Siddharth Rout Jan 13 '12 at 17:17

6 @SiddharthRout +1 - I am finding out I don't need any Excel books, just follow your posts for what I need to learn in a days work (ands this evening, it is the Ribbon XML !):) – Our Man In Bananas Apr 11 '13 at 22:15

2 I'm guessing some idiot clicked down when they meant up and adding my +1 just in case that was so. – pnuts Sep 4 '13 at 9:45



I struggled like mad, but this is actually the right answer. For what it is worth, what I missed was is this:

1. As others say, one can't create the CustomUI ribbon with VBA, **however**, you don't need to!
2. The idea is you create your xml Ribbon code using Excel's File > Options > Customize Ribbon, and then export the Ribbon to a .customUI file (it's just a txt file, with xml in it)
3. **Now comes the trick:** you can **include** the .customUI code in your .xlsm file using the MS tool they refer to here, by copying the code from the .customUI file
4. Once it is included in the .xlsm file, every time you open it, the ribbon you defined is **added** to the user's ribbon - but do use < ribbon startFromScratch="false" > or you lose the rest of the ribbon. On exit-ing the workbook, the ribbon is removed.
5. From here on it is simple, create your ribbon, copy the xml code that is specific to your ribbon from the .customUI file, and place it in a wrapper as shown above (...< tabs> your xml < /tabs...)

By the way the page that explains it on Ron's site is now at <http://www.rondebruin.nl/win/s2/win002.htm>

And here is his example on how you enable /disable buttons on the Ribbon <http://www.rondebruin.nl/win/s2/win013.htm>

For other xml examples of ribbons please also see <http://msdn.microsoft.com/en-us/library/office/aa338202%28v=office.12%29.aspx>

edited Mar 20 '14 at 8:16

answered Mar 20 '14 at 5:40



Jan Wijninckx

191 2 3

I was able to accomplish this with VBA in Excel 2013. No special editors needed. All you need is the Visual Basic code editor which can be accessed on the Developer tab. The Developer tab is not visible by default so it needs to be enabled in File>Options>Customize Ribbon. On the Developer tab, click the Visual Basic button. The code editor will launch. Right click in the Project Explorer pane on the left. Click the insert menu and choose module. Add both subs below to the new module.

```

Sub LoadCustRibbon()

Dim hFile As Long
Dim path As String, fileName As String, ribbonXML As String, user As String

hFile = FreeFile
user = Environ("Username")
path = "C:\Users\" & user & "\AppData\Local\Microsoft\Office\"
fileName = "Excel.officeUI"

ribbonXML = "<mso:customUI          xmlns:mso='http://schemas.microsoft.com/office/2009/07
/customui'" & vbNewLine
ribbonXML = ribbonXML + "    <mso:ribbon>" & vbNewLine
ribbonXML = ribbonXML + "        <mso:qat/>" & vbNewLine
ribbonXML = ribbonXML + "        <mso:tabs>" & vbNewLine
ribbonXML = ribbonXML + "            <mso:tab id='reportTab' label='Reports'
insertBeforeQ='mso:TabFormat'" & vbNewLine
ribbonXML = ribbonXML + "                <mso:group id='reportGroup' label='Reports'
autoScale='true'" & vbNewLine
ribbonXML = ribbonXML + "                    <mso:button id='runReport' label='PTO' " & vbNewLine
ribbonXML = ribbonXML + "                        imageMso='AppointmentColor3'          onAction='GenReport'/" >" &
vbNewLine
ribbonXML = ribbonXML + "                </mso:group>" & vbNewLine
ribbonXML = ribbonXML + "            </mso:tab>" & vbNewLine
ribbonXML = ribbonXML + "        </mso:tabs>" & vbNewLine
ribbonXML = ribbonXML + "    </mso:ribbon>" & vbNewLine
ribbonXML = ribbonXML + "</mso:customUI>"

ribbonXML = Replace(ribbonXML, "''", "")

Open path & fileName For Output Access Write As hFile
Print #hFile, ribbonXML
Close hFile

End Sub

Sub ClearCustRibbon()

Dim hFile As Long
Dim path As String, fileName As String, ribbonXML As String, user As String

hFile = FreeFile
user = Environ("Username")
path = "C:\Users\" & user & "\AppData\Local\Microsoft\Office\"
fileName = "Excel.officeUI"

ribbonXML = "<mso:customUI          xmlns:mso=""http://schemas.microsoft.com/office
/2009/07/customui"">" & _
"<mso:ribbon></mso:ribbon></mso:customUI>"

Open path & fileName For Output Access Write As hFile
Print #hFile, ribbonXML
Close hFile

End Sub

```

Call LoadCustRibbon sub in the Workbook open even and call the ClearCustRibbon sub in the Before_Close Event of the ThisWorkbook code file.

edited Dec 4 '15 at 1:02

answered Jun 17 '15 at 13:41



Roi-Kyi Bryant

110 1 8

This worked for me but with only a couple minor issues that I ran in to. 1) If I kept the Before_Close Event in, the new ribbon did not load. 2) When I managed to get the ribbon to work by removing the Before_Close event, I had to re-load Excel before it would appear. If you could advise why this is happening, that would be great! – Petay87 Aug 18 '15 at 14:16

Is the code exactly the same? Can you post it? The idea is Excel should modify the standard ribbon file before the ribbon is loaded and reset it before the workbook is closed. Resetting the ribbon is necessary so it is not displayed in other workbooks. – Roi-Kyi Bryant Aug 20 '15 at 13:03

The answers on here are specific to using the custom UI Editor. I spent some time creating the interface without that wonderful program, so I am documenting the solution here to help anyone else decide if they need that custom UI editor or not.

I came across the following microsoft help webpage - <https://msdn.microsoft.com/en-us/library/office/ff861787.aspx>. This shows how to set up the interface manually, but I had some trouble when pointing to my custom add-in code.

To get the buttons to work with your custom macros, setup the macro in your .xlam subs to be called as described in this SO answer - [Calling an excel macro from the ribbon](#). Basically, you'll need to add that "control As IRibbonControl" paramter to any module pointed from your ribbon xml. Also, your ribbon xml should have the onAction="myaddin!mymodule.mysub" syntax to

properly call any modules loaded by the add in.

Using those instructions I was able to create an excel add in (.xlam file) that has a custom tab loaded when my VBA gets loaded into Excel along with the add in. The buttons execute code from the add in and the custom tab uninstalls when I remove the add in.

answered Mar 17 '15 at 3:11



Jomtung
98 1 7

This works for those of us who don't have all the development tools. Thanks for the "simple" solution – [Evan](#)
Jun 8 '15 at 17:58

I used to teach my second year programming students this method, but eventually found that using the CustomUI tool is simply less complicated if you can access it. – [Rick Henderson](#) Feb 23 at 20:43

protected by [Community ♦](#) Jan 25 '13 at 22:16

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 [reputation](#) on this site.

Would you like to answer one of these [unanswered questions](#) instead?