

Chapter 10 – Stacked Widget

Tabwidgetstacked application

Step 1 Open Qt Designer

- Open the QtDesigner by selecting the Qt Designer program icon



Step 2 Open the Tab Widget application

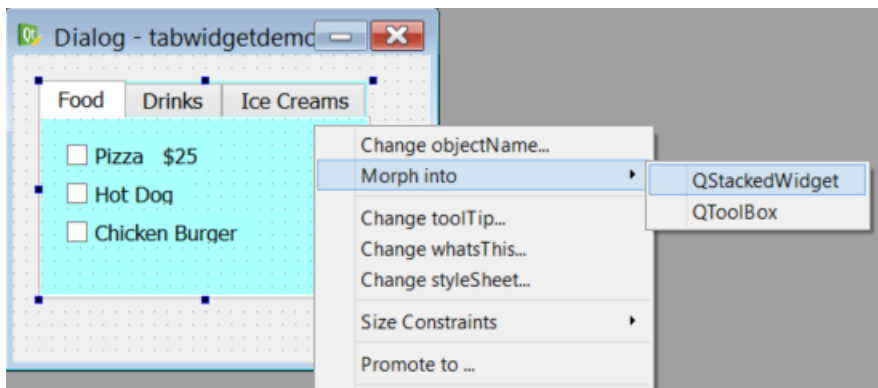
- Click the Open Button
- Locate the tabwidgetdemo application previously created

Step 3 Save the application as

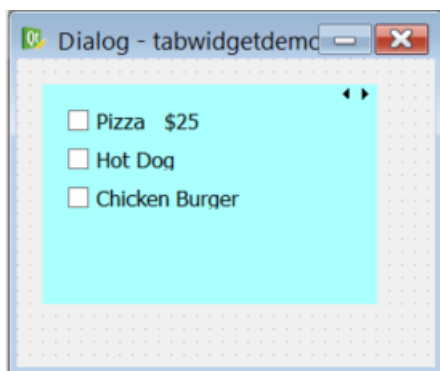
- Click File > Save As
- Save the file as `tabwidgetstacked.ui`

Step 4 Convert the tab widget to a Stacked Widget

- A tacked widget provides a stack of widgets where only one is visible at a time. By default it does not have a way of switching tabs and you will have to create one.
- Right click on the tab widget select Morph into



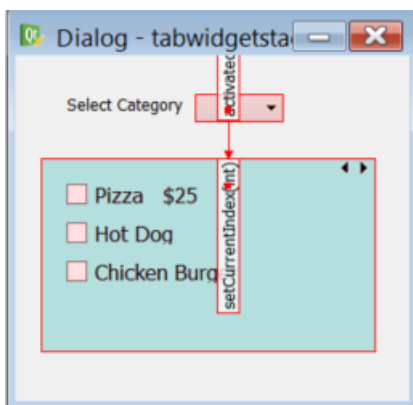
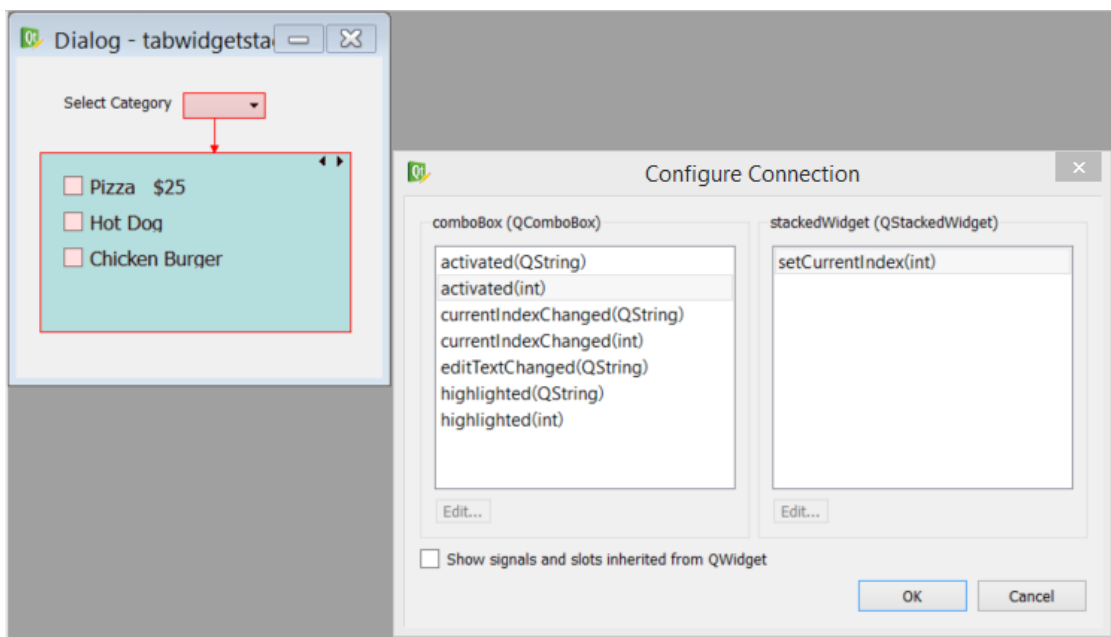
- The tab widget will change into a stacked widget



- Remember there is by default no switching available and we will create a way by adding a combobox. Each widget in the stack has an index number which we can use to access it.
- Add the following widgets to the form:

Widget	Property	Value
QLabel (Display widgets section)	objectName text	label Select Category
QComboBox (Input widgets section)	objectName	comboBox

- Link the activated(int) signal of the combobox to the setCurrentIndex(int) slot of the stacked widget. Open the signal and slots view and drag the comboBox onto the stacked widget. Select activate(int) in the comboBox list and then setCurrentIndex(int) in the stackedWidget list



Step 5 Save the form

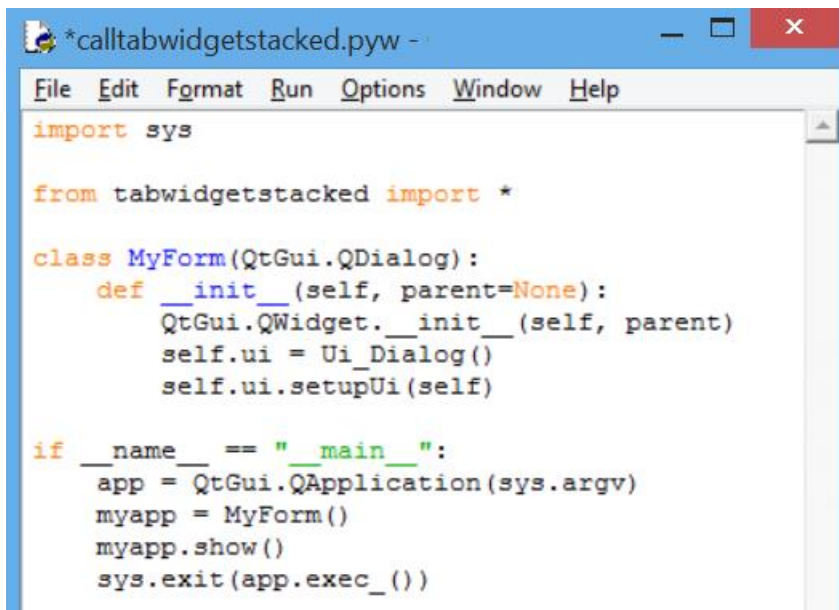
- Save the file as `tabwidgetstacked.ui` **(note the case!! Python is case sensitive)**

Step 6 Convert the .ui file to a .py file

- Convert the `tabwidgetstacked.ui` file to `tabwidgetstacked.py` using `pyuic4`. **(note the case!! Python is case sensitive, so even on file names the case must be the same throughout)**

Step 7 Create a source file (.pyw) that imports the .py file

- Create a source file that will import the .py file created in step above and from which we will invoke the user interface
- We import the reservation form we created in Qt Designer
- Use the following code **(note the indentation and case!!)**



```
*calltabwidgetstacked.pyw -
File Edit Format Run Options Window Help
import sys

from tabwidgetstacked import *

class MyForm(QtGui.QDialog):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_Dialog()
        self.ui.setupUi(self)

if __name__ == "__main__":
    app = QtGui.QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```

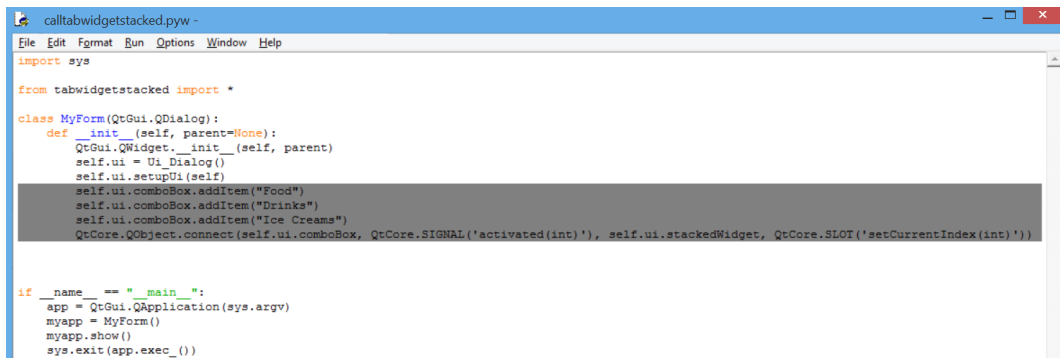
- Save the file as `calltabwidgetstacked.pyw`
- Run and test the application upto this point.

Step 7 Add code to create the combobox items and signal/slots

- Add the following code: (Note the case and indentation)

```
self.ui.comboBox.addItem("Food")
self.ui.comboBox.addItem("Drinks")
self.ui.comboBox.addItem("Ice Creams")
```

```
QtCore.QObject.connect(self.ui.comboBox,  
QtCore.SIGNAL('activated(int)'), self.ui.stackedWidget,  
QtCore.SLOT('setCurrentIndex(int)'))
```



```
calltabwidgetstacked.pyw -  
File Edit Format Run Options Window Help  
import sys  
from tabwidgetstacked import *  
class MyForm(QtGui.QDialog):  
    def __init__(self, parent=None):  
        QtGui.QWidget.__init__(self, parent)  
        self.ui = Ui_Dialog()  
        self.ui.setupUi(self)  
        self.ui.comboBox.addItem("Food")  
        self.ui.comboBox.addItem("Drinks")  
        self.ui.comboBox.addItem("Ice Creams")  
        QtCore.QObject.connect(self.ui.comboBox, QtCore.SIGNAL('activated(int)'), self.ui.stackedWidget, QtCore.SLOT('setCurrentIndex(int)'))  
if __name__ == "__main__":  
    app = QtGui.QApplication(sys.argv)  
    myapp = MyForm()  
    myapp.show()  
    sys.exit(app.exec_())
```

- Save the file as calltabwidgetstacked.pyw
- Run and test the application.

