

# GUI Programming in Python

A gateway to create Python-MySQL projects

# Introduction

- ▶ Like many other popular programming languages Python too supports GUI development.
- ▶ Python provides several different options for writing GUI applications
  - ▶ TkInter - Python's standard GUI library
  - ▶ PyQt5 - A specific installable library written in C++ for rapid GUI development
  - ▶ wxPython - A cross platform GUI toolkit for python
  - ▶ <https://wiki.python.org/moin/GuiProgramming> contains a detailed list

# GUI Form Builders

- ▶ Creating a good looking GUI by manual coding can be tedious. A visual GUI designer tool is always handy.
- ▶ Many GUI development IDEs targeted at wxPython are available. Following are some of them:
  - ▶ **wxFormBuilder** - an open source, cross-platform WYSIWYG GUI builder that can translate the wxWidget GUI design into C++, Python, PHP or XML format
  - ▶ **wxGlade** - a GUI designer written in Python with the popular GUI toolkit wxPython, that helps you create wxWidgets/wxPython user interfaces
  - ▶ **BoaConstructor** - GPL A RAD GUI Building IDE for wxPython.

# Working with wxFormBuilder - Installation

- ▶ Installer for wxFormBuilder can be downloaded and installed from <http://sourceforge.net/projects/wxformbuilder/>
- ▶ Install the wxPython package - pip install -U wxPython. For additional notes visit <https://wxpython.org/pages/downloads/>

# Basic GUI application

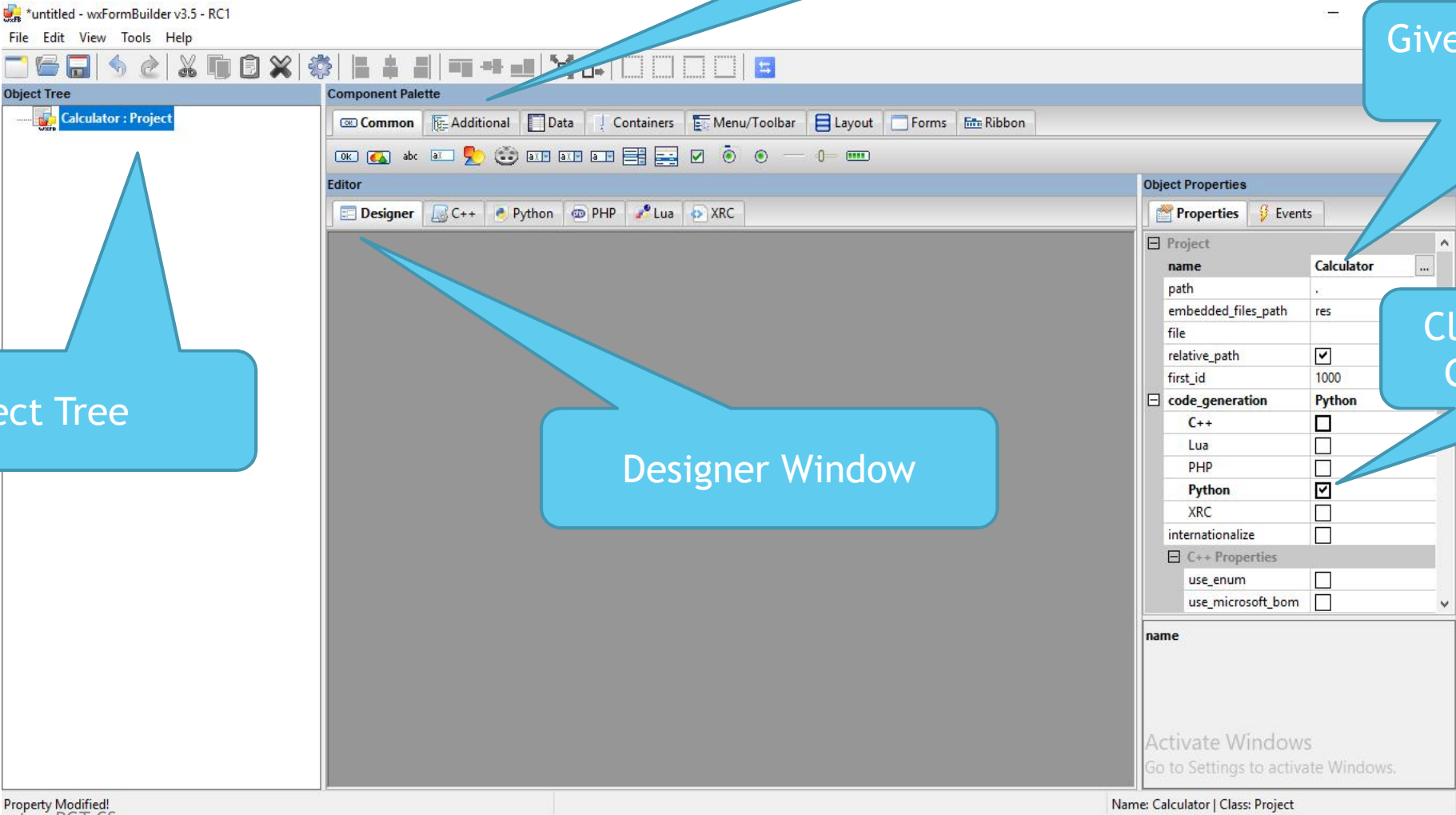
Component Palette

Give a sensible name to your project

Object Tree

Designer Window

Click Python in Code Generation option



# Adding a layout on the frame

Add a Box Layout or resizer to the frame

The screenshot shows the wxFormBuilder interface. The Object Tree on the left shows a project named 'Calculator : Project' containing a frame 'CalcFrame : Frame' with a 'bSizer1 : wxBoxSizer' child. The Component Palette in the center has the 'Layout' tab selected, showing various layout managers. The Editor window displays a 'Calculator' window with a red border. The Object Properties panel on the right shows the properties for the selected 'wxBoxSizer' object, including 'name' (bSizer1), 'orient' (wxVERTICAL), 'minimum\_size' (-1; -1), and 'permission' (none). A status bar at the bottom indicates 'Object 'bSizer1' of class 'wxBoxSizer' created.' and 'Name: bSizer1 | Class: wxBoxSizer'.

Property	Value
name	bSizer1
orient	wxVERTICAL
minimum_size	-1; -1
permission	none

Layout visible only in object tree

We can select Vertical or Horizontal

In a box resizer we can dynamically add number of control elements horizontally or vertically in rows or columns depending upon the boundaries of the container Frame.

# Creating the Basic Form

1. Click Forms Palette

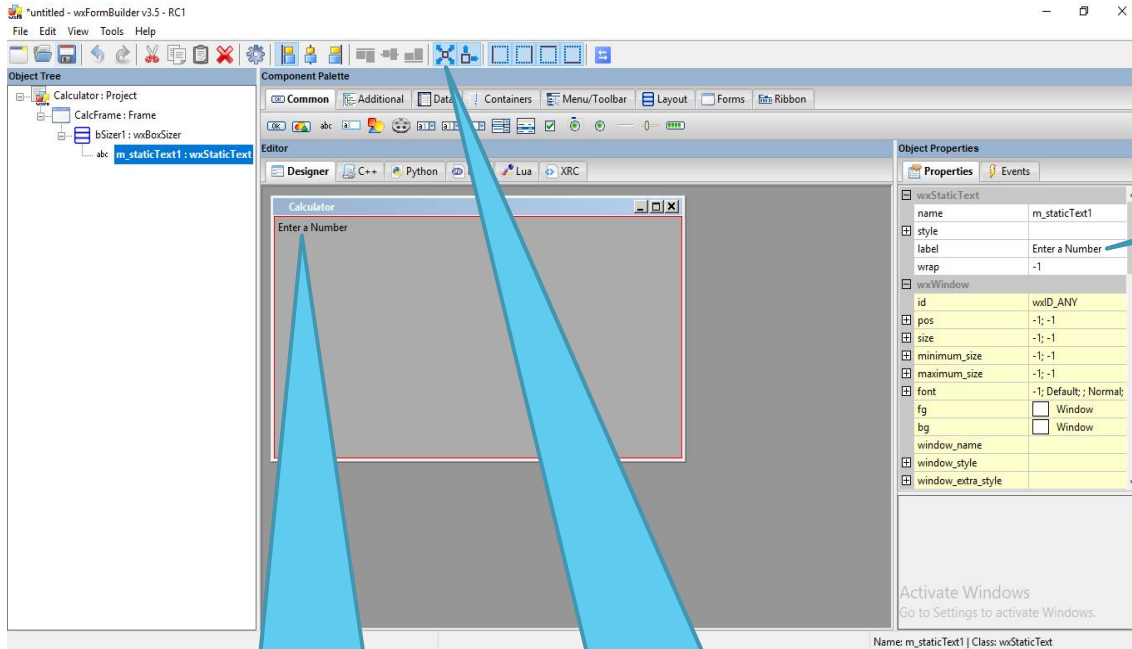
2. Click Frame control to place the Frame

3. Give sensible names to Frame and a Title too.

The screenshot shows the wxFormBuilder v3.5 - RC1 interface. The 'Component Palette' is open to the 'Forms' tab, and a 'Frame' control is placed in the 'Editor' area. The 'Object Tree' shows a project named 'Calculator : Project' with a sub-item 'CalcFrame : Frame'. The 'Object Properties' panel is open, showing the properties for the selected 'Frame' control. The 'name' property is set to 'CalcFrame' and the 'title' property is set to 'Calculator'. The 'wxWindow' section shows the 'id' property set to 'wxID\_ANY', 'pos' set to '-1; -1', 'size' set to '500; 300', 'minimum\_size' set to '-1; -1', and 'maximum\_size' set to '-1; -1'. The 'font' property is set to '-1; Default; Normal'. The status bar at the bottom right shows 'Name: CalcFrame | Class: Frame'.

Property	Value
name	CalcFrame
title	Calculator
style	wxDEFAULT_FRAME_...
extra_style	
center	wxBOTH
xrc_skip_sizer	<input checked="" type="checkbox"/>
event_handler	impl_virtual
au_i_managed	<input type="checkbox"/>
au_i_manager_style	wxAUI_MGR_DEFAULT...
wxWindow	
id	wxID_ANY
pos	-1; -1
size	500; 300
minimum_size	-1; -1
maximum_size	-1; -1
font	-1; Default; Normal

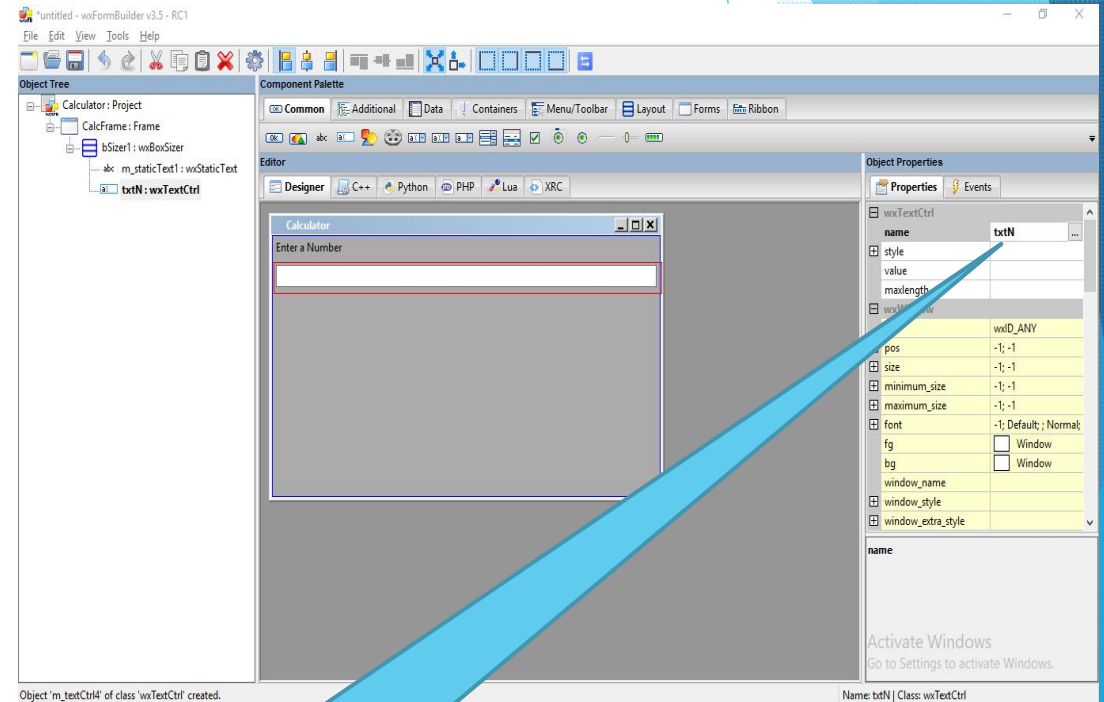
# Adding and configuring Controls



1. Click Label control from common palette to place on the Frame

3. Click Expand to fit the label on frame

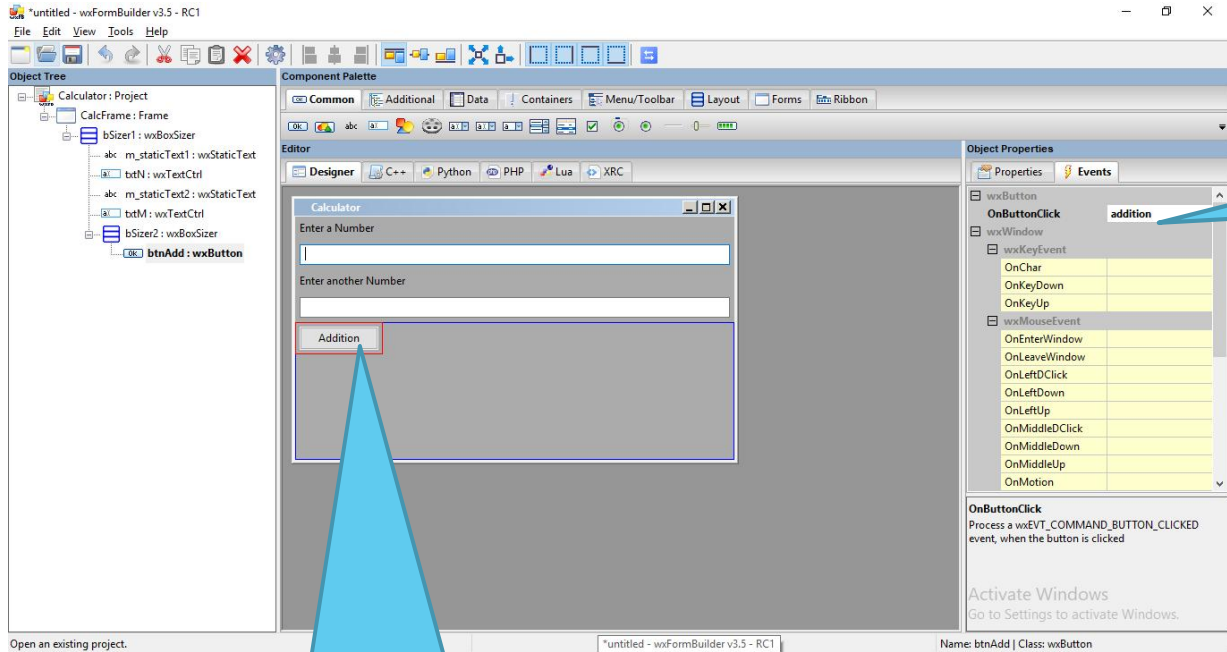
2. Give a name on Label Property



4. Add a Text Control and give it a name



# Adding buttons and their events

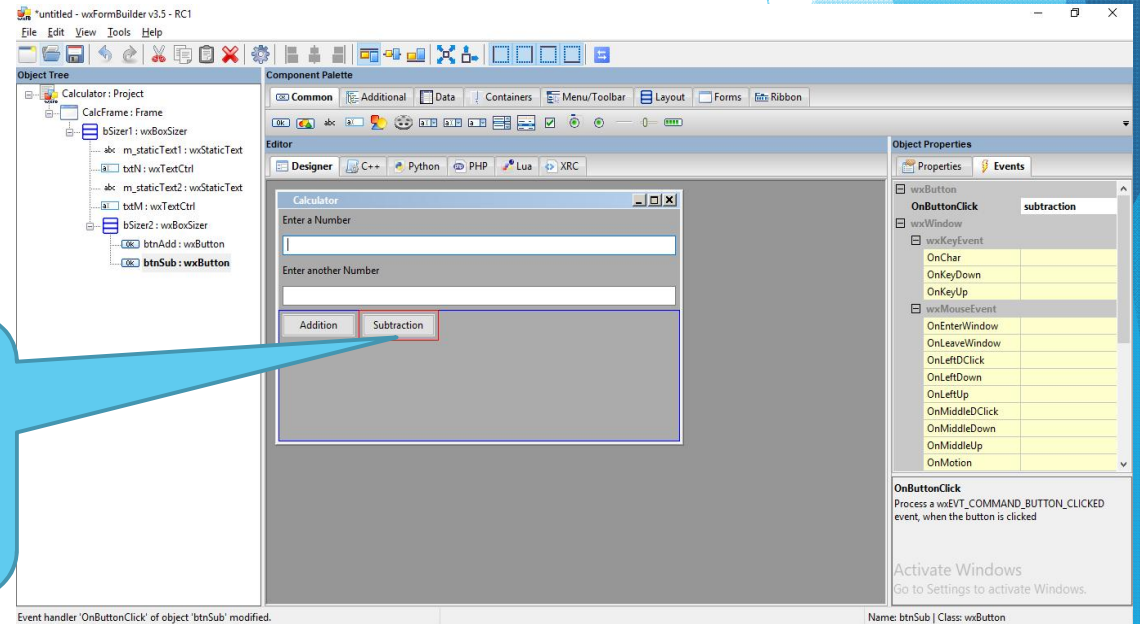


1. Add a Button under another box resizer with wxHORIZONTAL orientation and give it a name and label (caption)

Ashok Sengupta - PGT CS

2. Click the Events Tab and give an appropriate name for the OnButtonClick Event

3. Similarly add another button with name caption and event as subtraction



Event handler 'OnButtonClick' of object 'btnSub' modified.

Name: btnSub | Class: wxButton

# Giving Final touch

The screenshot shows the wxFormBuilder v3.5 - RC1 interface. The Object Tree on the left shows a project named 'Calculator' with a frame 'CalcFrame' containing several components: 'bSizer1', 'm\_staticText1', 'txtN', 'm\_staticText2', 'txtM', 'bSizer2', 'btnAdd', 'btnSub', 'bSizer3', 'm\_staticText3', and 'txtRes'. The Editor in the center displays a window titled 'Calculator' with two input fields, two buttons labeled 'Addition' and 'Subtraction', and a 'Result' label above a text field. The Object Properties panel on the right shows the properties for the selected 'm\_staticText3' component, including 'name', 'style', 'label', 'wrap', 'id', 'pos', 'size', 'minimum\_size', 'maximum\_size', 'font', 'fg', 'bg', 'window\_name', 'window\_style', and 'window\_extra\_style'. A blue callout box points to the 'Result' label and text field in the Editor, containing the text: '1. Added another label and a Text Field (txtRes) to display the result of the operations'. The status bar at the bottom right shows 'Name: m\_staticText3 | Class: wxStaticText'.

# Generating Python Guard Code for GUI

- Now press F8 to generate the Guard python code that will be saved in the same location where the wxFormBuilder file `.fbp` is saved. The default name for the file is `noname.py`.
- Rename the `noname.py` to `calculatorwx.py` and please do not edit that file
- Create a python black file and name it as `calc.py` in the same folder. We shall code it further to apply our business logic.

# The final Guard Code

```
calculatorwx.py - D:\pyproj\calculatorwx.py (3.6.5)
File Edit Format Run Options Window Help
# -*- coding: utf-8 -*-

#####
## Python code generated with wxFormBuilder (version Jun 17 2015)
## http://www.wxformbuilder.org/
##
## PLEASE DO "NOT" EDIT THIS FILE!
#####

import wx
import wx.xrc

#####
## Class CalcFrame
#####

class CalcFrame ( wx.Frame ) :

    def __init__ ( self, parent ):
        wx.Frame.__init__ ( self, parent, id = wx.ID_ANY, title = u"Calc" )
```

```
calculatorwx.py - D:\pyproj\calculatorwx.py (3.6.5)
File Edit Format Run Options Window Help

        self.SetSizeHintsSz( wx.DefaultSize, wx.DefaultSize )

        bSizer1 = wx.BoxSizer( wx.VERTICAL )

        self.m_staticText1 = wx.StaticText( self, wx.ID_ANY, u"Enter a N
        self.m_staticText1.Wrap( -1 )
        bSizer1.Add( self.m_staticText1, 0, wx.ALL|wx.EXPAND, 5 )

        self.txtN = wx.TextCtrl( self, wx.ID_ANY, wx.EmptyString, wx.Def
        bSizer1.Add( self.txtN, 0, wx.ALL|wx.EXPAND, 5 )

        self.m_staticText2 = wx.StaticText( self, wx.ID_ANY, u"Enter ano
        self.m_staticText2.Wrap( -1 )
        bSizer1.Add( self.m_staticText2, 0, wx.ALL|wx.EXPAND, 5 )

        self.txtM = wx.TextCtrl( self, wx.ID_ANY, wx.EmptyString, wx.Def
        bSizer1.Add( self.txtM, 0, wx.ALL|wx.EXPAND, 5 )

        bSizer2 = wx.BoxSizer( wx.HORIZONTAL )
```

Please do not Edit this Guard Code. We are going to inherit this python class CalcFrame in Calculator class to be defined in another file calc.py . Also we are going the to override the event methods addition and subtraction in the Calculator class.

# Writing code for the calc.py

```
*calc.py - D:\pyproj\calc.py (3.6.5)*
File Edit Format Run Options Window Help

import wx
import calculatorwx # importing the Guard code module calculatorwx

class Calculator(calculatorwx.CalcFrame): # Calculator class inheriting calculatorwx
    def __init__(self,parent): # the constructor for the class
        calculatorwx.CalcFrame.__init__(self,parent) # explicit call to base class const.

    def addition(self,event): # Defining addition event that is being overridden
        a = int(self.txtN.GetValue()) # Fetching data from first text field
        b = int(self.txtM.GetValue()) # Fetching data from second text field
        self.txtRes.SetValue(str(a+b)) # Displaying result

    def subtraction(self,event):
        a = int(self.txtN.GetValue())
        b = int(self.txtM.GetValue())
        self.txtRes.SetValue(str(a-b))

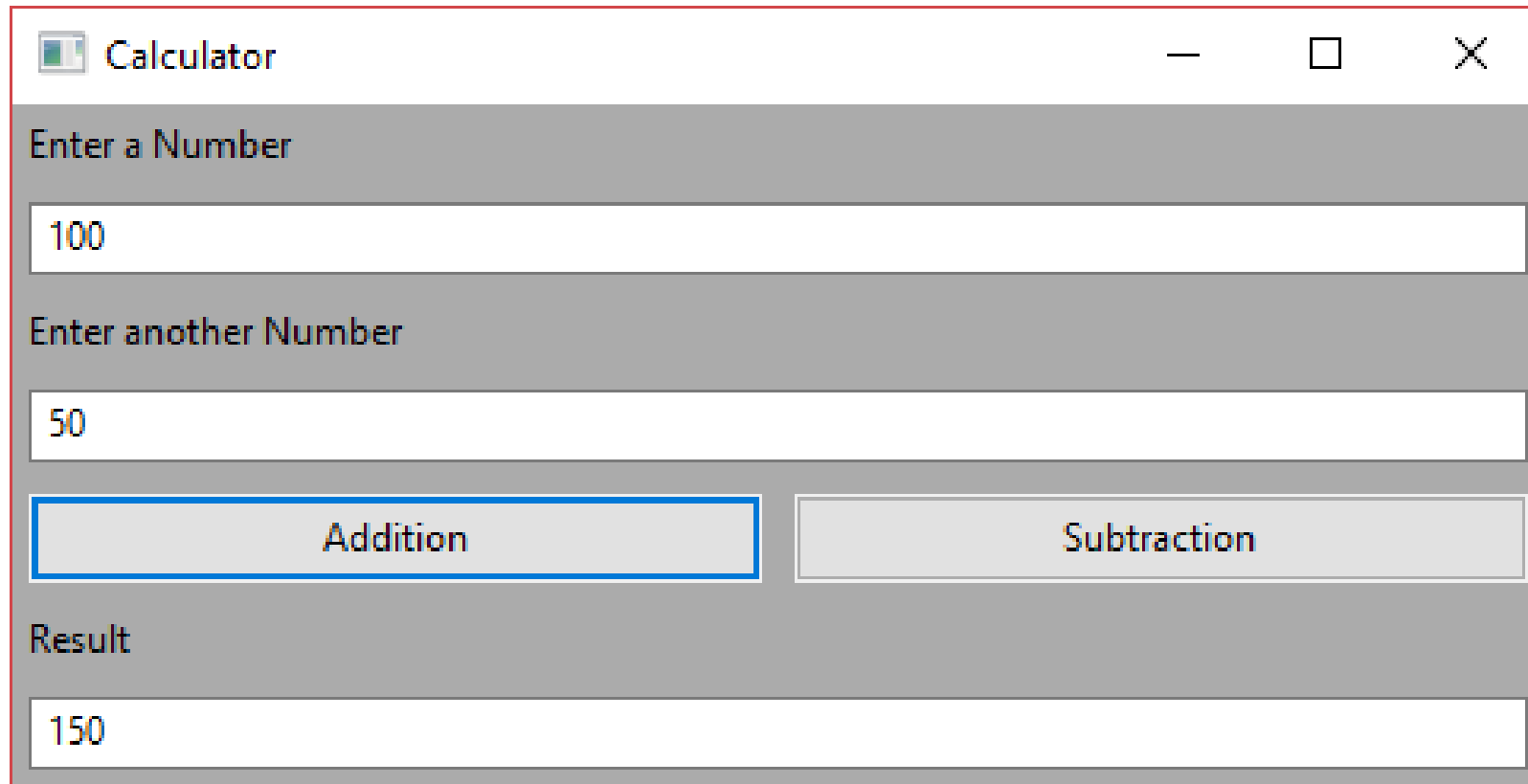
app = wx.App(False) # instantiating a wx application
frame = Calculator(None) # instantiating a Calculator object
frame.Show(True) # Enabling the object for display
#start the applications
app.MainLoop() # running the application|

Abhishek Sengupta - PUPCS
```

Activate Windows  
Go to Settings to activate Windows.

Ln: 23 Col: 41

# Running the Application - Save and execute the calc.py file to get the desired application running



The image shows a screenshot of a Python application window titled "Calculator". The window has a standard Windows-style title bar with minimize, maximize, and close buttons. The application interface is as follows:

- Enter a Number:** A text input field containing the value "100".
- Enter another Number:** A text input field containing the value "50".
- Operations:** Two buttons are present: "Addition" (highlighted with a blue border) and "Subtraction".
- Result:** A text output field displaying the value "150".

# References

- ▶ <https://wxpython.org/>
- ▶ <https://www.tutorialspoint.com/wxpython/>