

CIT244 Python II

Program 2: Graphical User Interface

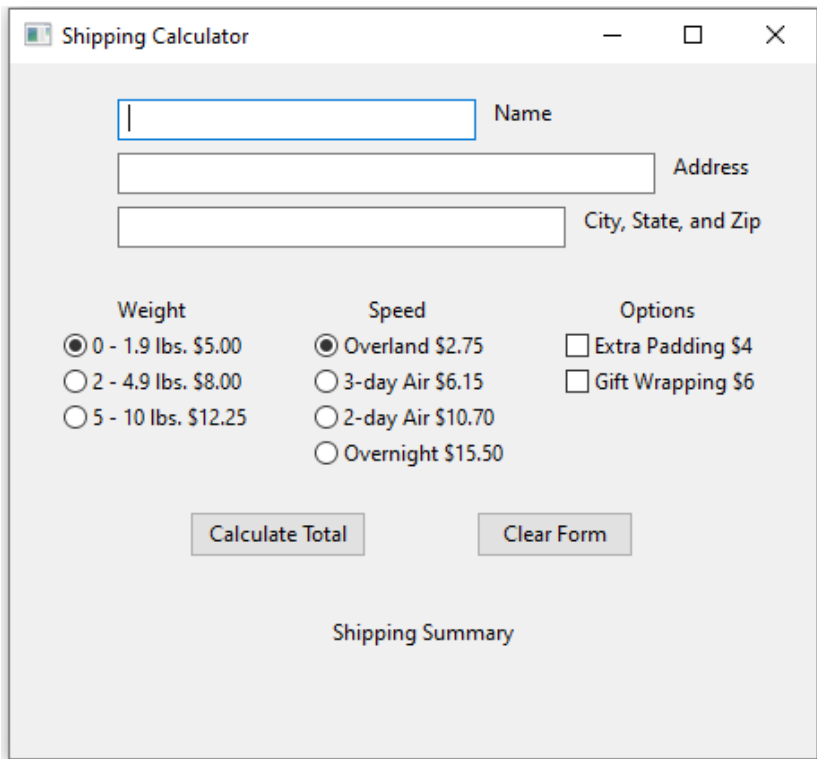
Before looking at this assignment, please give the following video, which gives advice on how and why we learn to program, a good, long, look.

 <i>Writing programs</i>	Program 2
--	------------------

Assignment

This assignment is to use wxPython to make a shipping calculator. The total cost will vary with package weight, shipping speed, and any extras the user might request. The layout should look something like this.

This is the default state of the interface (first radio of each radio button selected, everything else clear). Nothing has been calculated or else the Clear Form button has just been clicked. The prices for various weights and shipping speeds are shown. (0 - 1.9 lbs add \$5, 3-day air shipping add 6.15, etc.).



The screenshot shows a window titled "Shipping Calculator" with a standard Windows title bar (minimize, maximize, close buttons). The interface is organized as follows:

- Input Fields:** Three text boxes at the top for "Name", "Address", and "City, State, and Zip".
- Weight Selection:** Three radio buttons with labels: "0 - 1.9 lbs. \$5.00", "2 - 4.9 lbs. \$8.00", and "5 - 10 lbs. \$12.25". The first option is selected.
- Speed Selection:** Four radio buttons with labels: "Overland \$2.75", "3-day Air \$6.15", "2-day Air \$10.70", and "Overnight \$15.50". The first option is selected.
- Options:** Two checkboxes labeled "Extra Padding \$4" and "Gift Wrapping \$6". Both are currently unchecked.
- Buttons:** Two buttons at the bottom: "Calculate Total" and "Clear Form".
- Summary:** The text "Shipping Summary" is centered at the bottom of the window.

For full credit: your layout does not have to look exactly like this, but the prices must be the same, and it needs the widgets shown, and it should be sensible so someone using it can easily tell what's happening when button is clicked. You'll need these wxPython widgets.

- 3 TextCtrl widgets; one for name, one for address, and one for city-state-zip
- 7 radio buttons total, with a group of 3 for package weight, and a group of 4 for package delivery speed
- 2 check boxes for selecting options
- 2 buttons, one to trigger calculating the total, the other to reset the form in its default position.
- a number of StaticText widgets to make sure everything is labeled clearly
- a label at the bottom of the frame where the results will be printed. Since the form has been cleared in the image above you cannot see the summary label until something is actually calculated. (see the image later on this page)

Please start early in the week that this program is due. I am glad to help troubleshoot programs, but I go to bed reasonably early and am unlikely to stay up until midnight of the due date.

Shown next is an example calculation for a gift-wrapped 3 lb. package shipped 3-day air. Note the 4-line summary printed at the bottom of the form which includes the content of all three TextCtrl widgets plus the calculated total shipping cost. This summary was printed in a single StaticText widget designated for displaying the results when Calculate Total button was clicked.

The screenshot shows a window titled "Shipping Calculator" with the following elements:

- Name:** TextCtrl with "Mr Ralph Cramden"
- Address:** TextCtrl with "12101 Studebaker Drive"
- City, State, and Zip:** TextCtrl with "Grayhound, TX 20002"
- Weight:** Three radio buttons: "0 - 1.9 lbs. \$5.00", "2 - 4.9 lbs. \$8.00" (selected), and "5 - 10 lbs. \$12.25".
- Speed:** Four radio buttons: "Overland \$2.75", "3-day Air \$6.15" (selected), "2-day Air \$10.70", and "Overnight \$15.50".
- Options:** Two checkboxes: "Extra Padding \$4" (unchecked) and "Gift Wrapping \$6" (checked).
- Buttons:** "Calculate Total" (highlighted with a blue border) and "Clear Form".
- Shipping Summary:** A section at the bottom displaying:
Mr Ralph Cramden
12101 Studebaker Drive
Grayhound, TX 20002
\$ 20.15

You may benefit from the following hints.

Recall that to define a group of radio buttons the first radio button of each group must be designated by a `style=wx.RB_GROUP` property. The following 5 buttons form 2 groups. The first 2 radios form a group, and the last 3 radios form a different group. This means you can select one radio from each group.

```
self.a1 = wx.RadioButton(self.panel, -1, 'just 1', pos=(23, 100), style = wx.RB_GROUP)
```

```
self.a2 = wx.RadioButton(self.panel, -1, 'just 2', pos=(23, 100))
```

```
self.r = wx.RadioButton(self.panel, -1, 'red', pos=(23, 100), style = wx.RB_GROUP)
```

```
self.g = wx.RadioButton(self.panel, -1, 'green', pos=(23, 100))
```

```
self.b = wx.RadioButton(self.panel, -1, 'blue', pos=(23, 100))
```

By default the first button in each group will be selected.

And generally, for both radio buttons and checkboxes, you can set the state of the button using the following.

```
self.btn.SetValue(False) # False unchecks a button, True would select it.
```

And, you can make a label print things multi-line if you set a size of both a height and a width, and you concatenate the strings to be printed with `"\n"`, which is the new line character. So this would print "The cat in the hat." on one line.

```
txt = "The cat" + " in the hat."  
self.label.SetValue(txt)
```

But this would print "The cat" on one line, and "in the hat." on the next line.

```
txt = "The cat" + "\n" + "in the hat."  
self.label.SetValue(txt)
```

And finally, when setting the size property of a widget, if you set one of the dimensions as `-1` that dimension will display as the default value. For example, the following would be a `TextCtrl` with a width of 220 pixels, but a height set at the default (which is big enough for any letters)

```
self.name = wx.TextCtrl(self.panel, -1, pos=(90, 30), size=(220, -1))
```

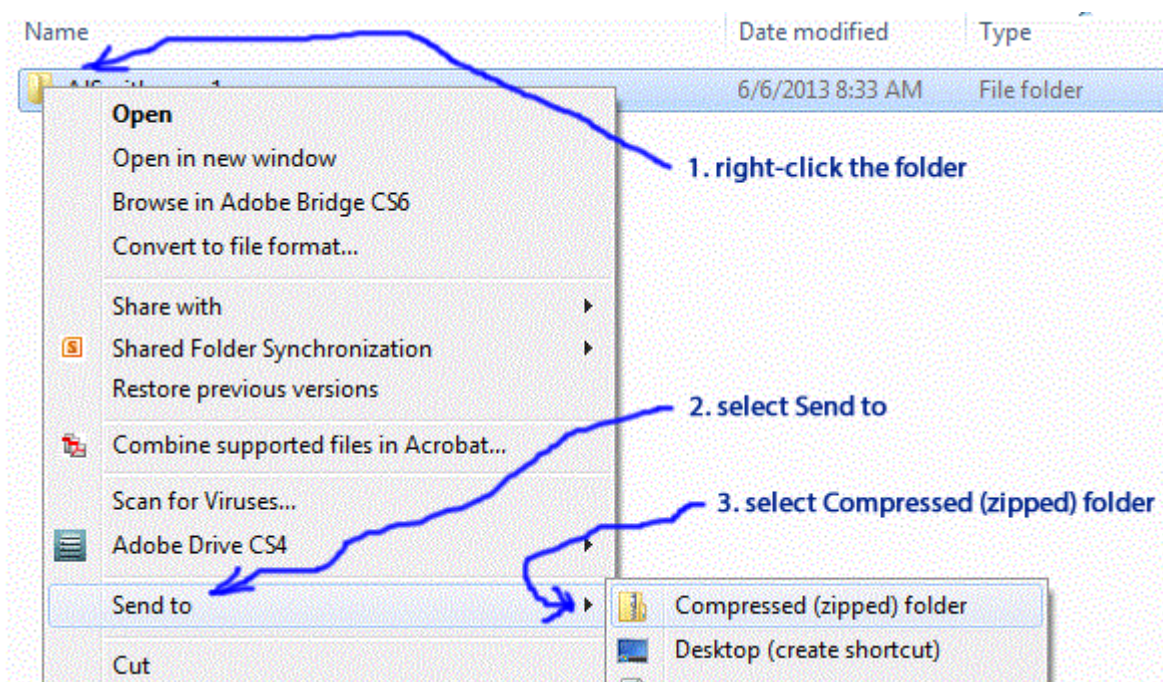
If you get good and stuck attach a compressed version of your code and mail it to me; I'll be glad to help you troubleshoot it.

Grading will follow this approach.

- 100% program runs perfectly with all required items
- 90% program meets all requirement but doesn't always produce the correct output
- 80% program runs without errors, but is missing some requirement.
- 70% program had a good start, most but not all of the code was correct, had one or more fatal errors.
- 40-60% or below. Not very close, many parts missing, would not run, probably a late start.
- 0% no submission or code was not the student's work.

How To Submit Your Program

You need to compress the folder that holds your program file or files before you email it to me. Uncompressed code is likely to be filtered out by a suspicious email server. You probably know this, but to compress an existing folder, *right click it*, choose *Send to*, then choose *Compressed (zipped) folder*, as shown below. The zipped folder has a little zipper on it.



If you email me your finished program, unless it comes in late at night, I will usually grade it that same day. Also, if you get stuck on something you can also send me your code and I'll be happy to help you get it working. The earlier in the week the program is due that you start, the more time we have to sort out any problems there may be. So, please start on these program many days before the due date.

Please clearly name the folder with your code, with *your name, the class, and the program number*. For example, jsmith-pythonprogram3.

I get lots and lots of programs from several different classes, so it helps me a great deal in keeping things straight if I can tell what's in the folder by the folder label.

Once everything is working (or even if it isn't) attach the zipped folder to an email and send it to me..

If you have questions let me know: mark.prather@kctcs.edu.