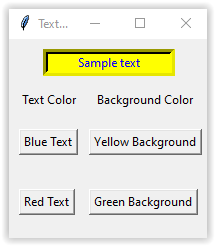
**GEOG 531**

**Tutorial 1: Python Basics**

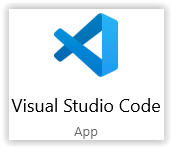
This tutorial will introduce the concepts of creating form’s window and writing code. We will create an application to illustrate some simple interface-building and property setting, both at design-time and run-time.

Your finished application will look something like this:



**Start-up**

1. Start *Visual Studio* (VS) *Code* from the *Start* menu*.*



1. From the start screen for VS Code, go to the menu *File > New File* (or *Ctrl+N*) to open a new *Editor* coding window. From the menu *File > Save* (or *Ctrl+S*), save the new script file within your working folder with name **Tutorial1.py**. Make sure that you have defined the file extension as **py**.
2. You may close *Get Started* window (tab) Graphical user interface, text, application

   Description automatically generated just for conveniences. Also click on the *Explorer*  button on the left to close the *Explorer* window.
3. Now you may see two empty windows in VS Code. The top one is the *Editor* window, and bottom one is the *Terminal* window (the *Terminal* tab should be active).

A screenshot of a computer

Description automatically generated

You will use the codding *Editor* window to enter and execute Python statements, and the *Terminal* window, to see results of code’s executions and to run DOS and Python commands.

1. In the bottom-left *Side Bar*, make sure that your current Python interpreter is set to *Python 3.x.x 64-bit* e.g.,  .

If necessary, see guides and video on how to use VS Code at <https://code.visualstudio.com/>.

**Create the GUI form’s window and controls**

To create a Graphical User Interface (GUI) form in Python, you will need to use a Python package which implements a windows GUI. One of such packages is the **tkinter** package (<https://docs.python.org/3/library/tk.html>). Tkinter is the standard GUI library for Python and not required an installation – it is included in a Python interpreter/environment.

1. Enter the following Python code (you might copy and paste the code) into the *Editor* window:

# Import the GUI package

import tkinter as tk

# Create the GUI application main window

frmTextColours = tk.Tk()

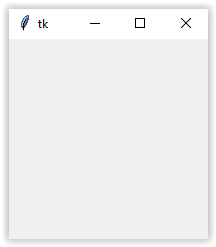
# Code to add widgets will go here...

# Infinite loop of main window

frmTextColours.mainloop()

*Code explanations:* The tkinter module, containing the Tk toolkit, has always to be imported. In our example, we imported tkinter by renaming it into tk variable, which is one of the convenient ways to do it in order to reuse short alias name of module. To initialize tkinter, we must create a Tk main widget, which is a form’s window frmTextColours in our case, with a title bar and other decoration provided by the window manager. The name frmTextColours for variable used is a user choice, we can use e.g., window as this variable name. The root or main window widget must be created before any other widgets and there can only be one window root widget. The window would not appear until we enter the Tkinter event loop mainloop. If we run the script, a main window will remain in the event loop until we close the window.

1. *Run Python File in Terminal* by using button  from the right-top corner.
2. You should be looking at the blank GUI application main frmTextColours window, called *Tk*. You may resize the frmTextColours at the sides and corners of form.



1. You may wish to change the name and size of your frmTextColours. Close the running frmTextColours , and enter the following lines of code after frmTextColours = tk.Tk() line:

# Define the title of the window.

frmTextColours.title("Text color demonstration")

# Define the size of the window

frmTextColours.geometry("200x200")

1. Run the Python script  again. You should see the new title on the top of form resized to dimensions of 200x200 pixels. Close the frmTextColours window.

*Code explanations:* The geometry() method is used to set the dimensions of the Tkinter main window on the user’s desktop.

Tkinter provides various **user’s controls**, such as buttons, labels and text boxes used in a GUI application. These controls are also called **widgets**.

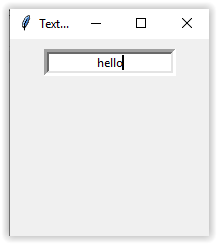
1. First, we will create an Entry textbox control. The Entry widget is used to display a single-line text field for accepting values from a user. Enter the following line of code after the comment # Code to add widgets will go here... and before frmTextColours.mainloop() command, which should be always at the end of a code:

# Create and place the textbox for user entered text on the parent window

txtColouredText = tk.Entry(frmTextColours, bd=5, justify="center")

txtColouredText.pack(side="top", pady=10)

*Code explanations:* The first line of code create a textbox control txtColouredText on the window frmTextColours with border width of 5 pixels and centered justification for text entry (check <https://www.tutorialspoint.com/python3/tk_entry.htm> for more explanations on the widget parameters). The pack()geometry manager method places and organizes the widget txtColouredText in the parent widget frmTextColours. It resizes the txtColouredText on the top of frmTextColours with extra space (padding) of 10 pixels in the y direction (see more at <https://www.tutorialspoint.com/python3/tk_pack.htm>).

1. Run the Python script . You should see the window with the textbox as follows. You can type a text string within the txtColouredText.  Then close the form.
2. Next, we will place two labels on the frmTextColours which will be used as headings, as shown below. Include the following code lines after the line txtColouredText.pack(side="top", pady=10):

# Create and place the labels for Text Color and Background Color

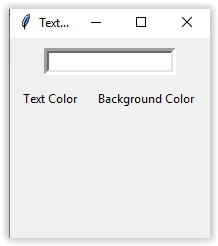
lblTC = tk.Label(frmTextColours, text = "Text Color")

lblTC.place(x=10, y=50)

lblBC = tk.Label(frmTextColours, text = "Background Color")

lblBC.place(x=85, y=50)

*Code explanations:* The Label widget is used to provide a single-line caption text for other widgets (see more about parameters of this widget at <https://www.tutorialspoint.com/python3/tk_label.htm>). It can also contain images.The place() geometry manager method organizes label widgets by placing them in a specific position in the parent widget by using x, y horizontal and vertical offset in pixels from the top-left corner of window (see more at <https://www.tutorialspoint.com/python3/tk_place.htm>). The place() is just different method to position widgets (we used the pack() to place the textbox).

1. Run the Python script . You should see the window with the textbox and two labels as follows.  Then close the form.
2. Now, we will place four buttons, as shown below. The 4 buttons should be the same size and properly aligned. Insert the following code lines after the line lblBC.place(x=85, y=50):

# Create and place the buttons to change text to blue

btnBlueTxt = tk.Button(frmTextColours, text="Blue Text", command=btnBlueTxt\_Click)

btnBlueTxt.place(x=10, y=90)

# Create and place the buttons to change background to yellow

btnYellowBG = tk.Button(frmTextColours, text="Yellow Background", command=btnYellowBg\_Click)

btnYellowBG.place(x=80, y=90)

# Create and place the buttons to change text to red

btnRedTxt = tk.Button(frmTextColours, text="Red Text", command=btnRedTxt\_Click)

btnRedTxt.place(x=10, y=150)

# Create and place the buttons to change background to green

btnGreenBG = tk.Button(frmTextColours, text="Green Background", command=btnGreenBg\_Click)

btnGreenBG.place(x=80, y=150)

*Code explanations:* The Button widget is used to display the buttons in your frmTextColours. The button displays the property text (can be also images) that convey the purpose of the buttons. The command property is used to associate the button with function (e.g., btnBlueTxt\_Click) or method to be called automatically when the button is clicked. Again, the button, as any widget, can be placed on the root window by using place() method.

1. Now if you will try to run the script , you will get the error shown in the *Terminal* window: NameError: name 'btnBlueTxt\_Click' is not defined. In the *Editor* window, you will see that the statement associated with this error is underlined as . To fix this error, we need to create function btnBlueTxt\_Click.
2. Save your work by selecting the menu *File* > *Save* or *Ctrl+S* keyboard combination.

So far, we completed GUI design (root window and user’s controls) and assigned the properties for the form and widgets, e.g., text, bd and justify. In the Tkinter, we can create a root window, controls, assign properties and place controls on the window only via manual codding. That is typical for desktop Python GUI scripting in all Python IDEs. Python GUI design can be very routine if we would like to create a complex windows or mac app.

**Writing Code**

Now, we will create 4 functions for each respective button designed above. We will discuss how to create a Python function in the following classes in detail. In this tutorial, we are just going to use functions provided.

1. In a Python script, a function should be placed before as it is called from respective button. Place the following function before the line btnBlueTxt = tk.Button(frmTextColours, text="Blue Text", command=btnBlueTxt\_Click):

# Function to change textbox foreground property

def btnBlueTxt\_Click():

   txtColouredText.config(fg="Blue")

1. Note that as soon as you will place the function before the button’s code, the underline of command parameter of button will disappear. The error is fixed.

*Code explanations:* A function is a block of code which only runs when it is called. In Python, a function is defined by using the def Python keyword. Statements, which belong to a function’s block, should be **indented** on the same spacing. The config method of widget is used to change the widget parameters, e.g., text, font color fg used to render the text or/and the background color bg displayed behind the text.

1. Similar insert the following 3 functions before their respective buttons:

# Function to change textbox background property

def btnYellowBg\_Click():

   txtColouredText.config(bg="Yellow")

# Function to change textbox foreground property

def btnRedTxt\_Click():

   txtColouredText.config(fg="Red")

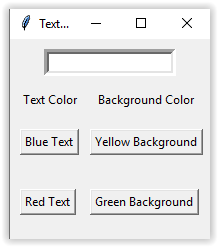
# Function to change textbox background property

def btnGreenBg\_Click():

   txtColouredText.config(bg="Green")

Also, all functions can be placed after the line # Code to add widgets will go here....

**Running the Program**

1. Run  the script. Fix any errors which might occur. Now you must see the following GUI .
2. Type a text into the textbox and test your buttons.
3. Save your script and exit VS Code.