Build Your First Application Using PySimpleGUI

Introduction
In this tutorial we will build a small desktop application using PySimpleGUI. PySimpleGUI provides a super-simple, easy to understand interface to GUIs that can be easily customized.

Importing PySimpleGUI Module

If you are running python 3,

import PySimpleGUI **as** sg

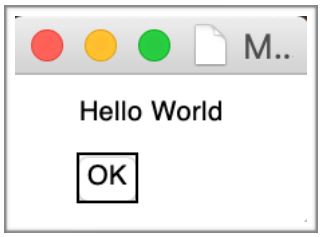
For python 2.7,

import PySimpleGUI27 **as** sg

Our first GUI 'hello world' program using PySimpleGUI

```
import PySimpleGUI as sg

layout=[
        [sg.Text('Hello World')],
        [sg.Button("OK")],
        ]
window = sg.Window('My first GUI', layout)

button, values = window.Read()
```

Output:

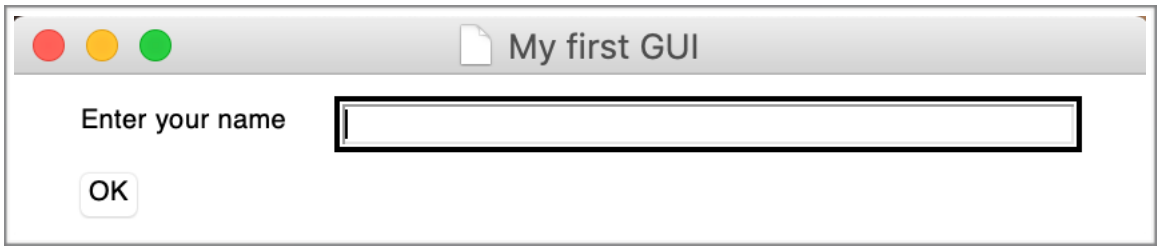


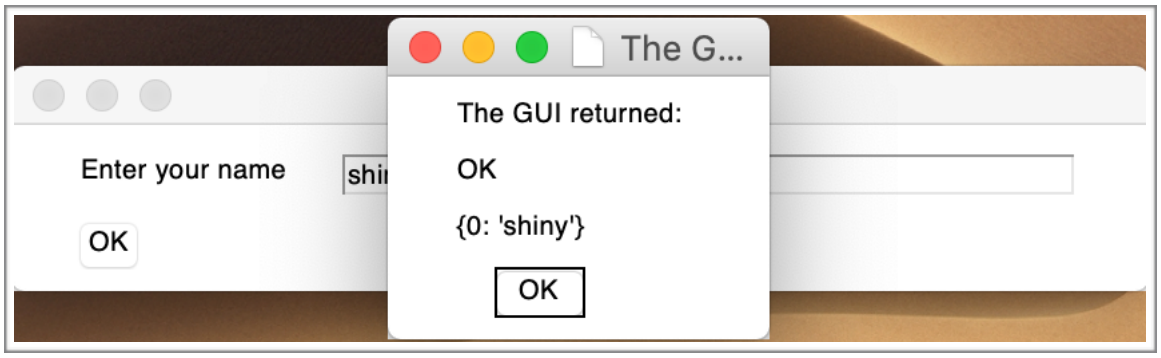
Example 2:

```
import PySimpleGUI as sag

layout=[
        [sg.Text('Enter your name'),sg.InputText("")],
        [sg.Button("OK")],
        ]
```

```
window = sg.Window('My first GUI', layout)

button, values = window.Read()
sg.Popup('The GUI returned:', button, values)
```

Output:





Elements in PySimpleGUI

"Elements" are the building blocks used to create windows. Some GUI APIs use the term "Widget" to describe these graphic elements.

- Text
- Single Line Input
- Buttons including these types:
  - File Browse
  - Folder Browse
  - Calendar picker
  - Date Chooser
  - Read window
  - Close window ("Button" & all shortcut buttons)
  - Realtime
- Checkboxes
- Radio Buttons
- Listbox
- Slider

- Multi-line Text Input/Output
- Multi-line Text Output (Qt only)
- Scroll-able Output
- Vertical Separator
- Progress Bar
- Option Menu
- Menu
- ButtonMenu
- Frame
- Column
- Graph
- Image
- Table
- Tree
- Tab, TabGroup
- StatusBar

Common Element Parameters
Some parameters that you will see on almost all Elements:

Size
Specifies the amount of room reserved for the Element. For elements that are character based, such a Text, it is (# characters, # rows).

Keys
Keys are a way for you to "tag" an Element with a value that will be used to identify that element. After you put a key in an element's definition, the values returned from Read will use that key to tell you the value.
For example, if you have an input field: Input(key='mykey')
And your read looks like this: event, values = Read()
Then to get the input value from the read it would be: values['mykey']
You also use the same key if you want to call Update on an element.

Font
Specifies the font family, size, and style. Font families on Windows include: * Arial * Courier * Comic, * Fixedsys * Times * Verdana * Helvetica

Let's try out few elements

Text Element

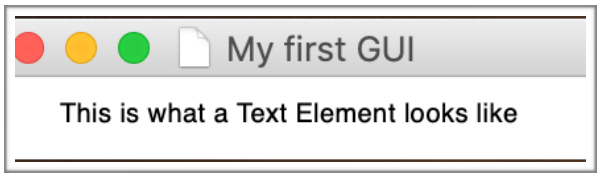
Basic Element. It displays some **text** in the window.

Sample code:

```
import PySimpleGUI as sg

layout = [
  [sg.Text('This is what a Text Element looks like')],
]
```

```
window = sg.Window('My first GUI', layout)

button, values = window.Read()
```

Output:



Text Input Element

Shows a single line of input.

Sample code:

```
import PySimpleGUI as sg

layout = [[sg.InputText('Default text')]]
window = sg.Window('My first GUI', layout)

button, values = window.Read()
```

Output:

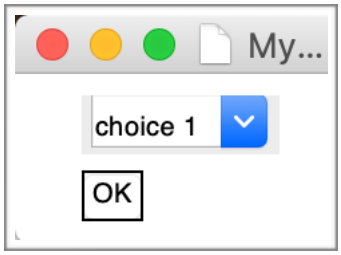
Combo Element

Also known as a drop-down list. Only required parameter is the list of choices.

Sample code:

```
import PySimpleGUI as sg

layout = [[sg.InputCombo(['choice 1', 'choice 2'])],
          [sg.Button("OK")]]
window = sg.Window('My first GUI', layout)

button, values = window.Read()
sg.Popup('The GUI returned:', button, values)
```

Output:



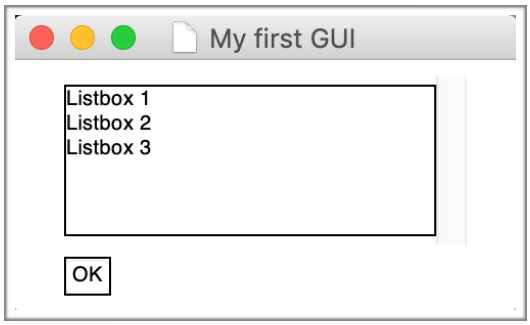
Listbox Element

A List Box. Provide a list of values for the user to choose one or more of. Returns a list of selected rows when a window.Read() is executed.

Sample code:

```
import PySimpleGUI as sg

layout = [[sg.Listbox(values=['Listbox 1', 'Listbox 2', 'Listbox 3'], size=(30, 6))],
          [sg.Button("OK")]]
window = sg.Window('My first GUI', layout)

button, values = window.Read()
sg.Popup('The GUI returned:', button, values)
```

Output:



The details of all the elements supported by PySimpleGUI, it's parameters, and methods can be found here: https://pysimplegui.readthedocs.io/

Also, please visit the Cookbook https://pysimplegui.readthedocs.io/en/latest/cookbook/ for some sample applications

Try these code and see the difference

Example 1 (One-shot-window):

```
import PySimpleGUI as sg

lst=[ ]
layout = [
        [sg.Text('Enter your name'),sg.InputText("", key='name')],
        [sg.Button("Add"),sg.Button('Exit')],
        ]
window = sg.Window('My first GUI', layout)

button, values = window.Read()
if button == 'Add':
    lst.append(values['name'])

sg.Popup('Names are:',lst)
```

Example 2: (Persistent Window)

```
import PySimpleGUI as sg

l=[ ]
layout = [
        [sg.Text('Enter your name'),sg.InputText("", key='name')],
        [sg.Button("Add"),sg.Button('Exit')],
        ]
window = sg.Window('My first GUI', layout)


while True:
    button, values = window.Read()
    if button == 'Add':
        l.append(values['name'])
        window.FindElement('name').Update("")
    elif button == 'Exit':
        sg.Popup('Names are:', l)
        break
```

Exercise:

1. Download the ToDo App and run.
2. Persist the ToDo list in a file
3. Add two attributes; priority and deadline to the tasks
4. Sort the display by priority
5. Include options to show ‘all tasks’ and ‘completed tasks’