# Data Processing and Analysis in Python
# Lecture 9
# Graphical User Interfaces

UNIVERSITY OF MARYLAND

ROBERT H. SMITH SCHOOL OF BUSINESS

DR. ADAM LEE

## Graphical User Interface (GUI)

- GUI displays text as well as small images (called icons) that represent objects such as directories, files, command buttons, and drop-down menus
- In addition to entering text at keyboard, the user of a GUI can select an icon with pointing device, such as mouse, and move that icon around on the display
- GUI displays a window that contains various components, called window **objects** or **widgets**
- User is not constrained to enter inputs in a particular order
- Running different data sets does not require re-entering all of the data

## Event-Driven Programming

- User-generated events (e.g., mouse clicks, keystrokes) trigger operations to respond by pulling in inputs, processing them, and displaying results
- Coding phase:
  - Define a new class to represent the main window
  - Instantiate the classes of window objects needed for this application (e.g., labels, command buttons)
  - Position these components in the window
  - Instantiate and provide any default data in the window objects
  - Register and define controller methods with each window object in which a relevant event might occur
  - Define a main that launches the GUI

## GUI Modules

- There are many libraries and toolkits of GUI components available to the Python programmer
  - **tkinter** includes classes for windows and numerous types of window objects
  - **PyQt** is binding as a plug-in
  - **Kivy** is open source with a natural user interface
  - **PyGUI** is one of the simplest and lightweight libraries
  - **wyPython** is implemented as a Python extension module
  - …

## Window Layout

- Window components are laid out in the window's two-dimensional grid
  - Rows and columns are numbered from the position (0,0) in the upper left corner of the window
  - The programmer can force a horizontal and/or vertical spanning of grid positions

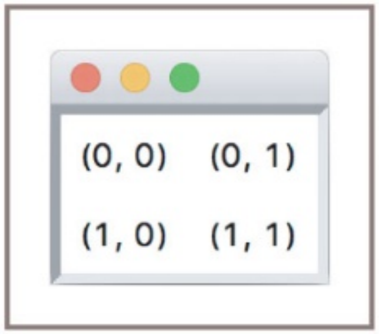


**Figure 8-5** Laying out labels in the window's grid

# Labels and Buttons

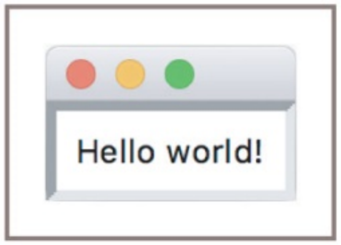


Figure 8-3 Displaying a label with text in a window

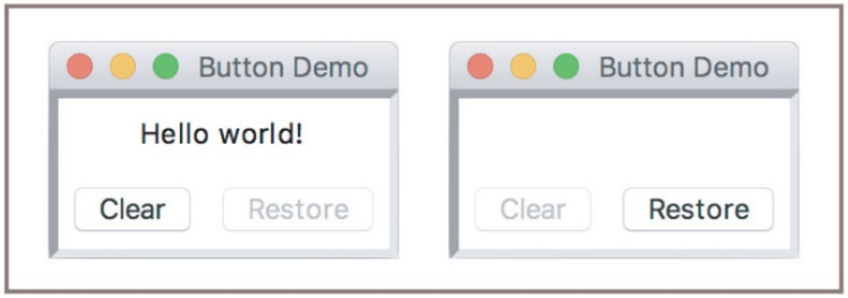
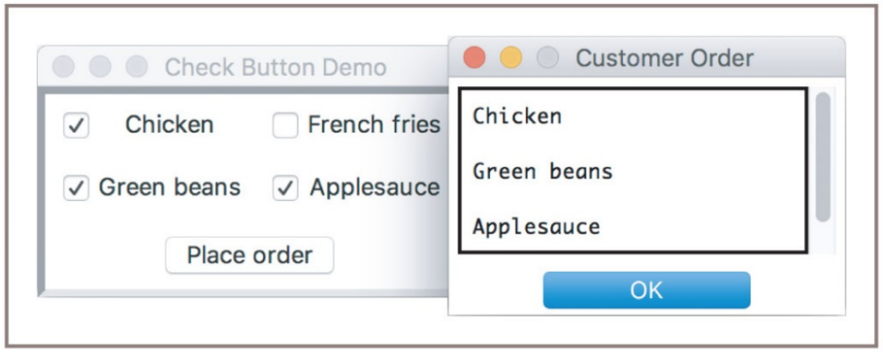
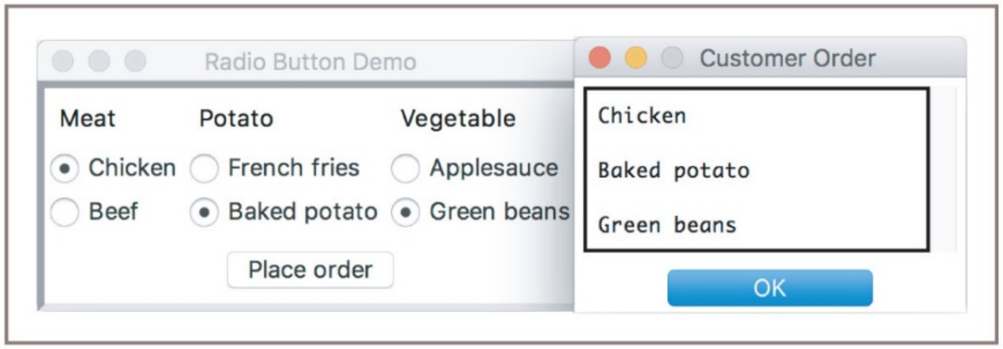


Figure 8-8 Using command buttons



```
Chicken
Green beans
Applesauce
```

Figure 8-20 Using check buttons



```
Chicken
Baked potato
Green beans
```

Figure 8-21 Using radio buttons

## Text Fields and List Boxes

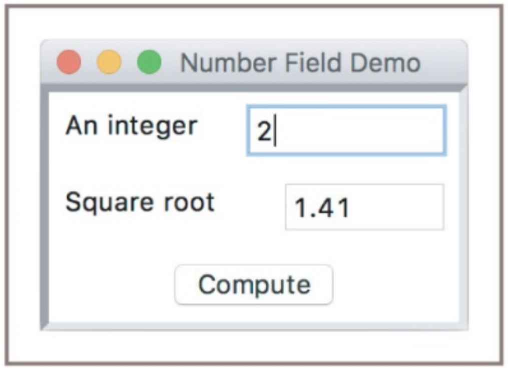
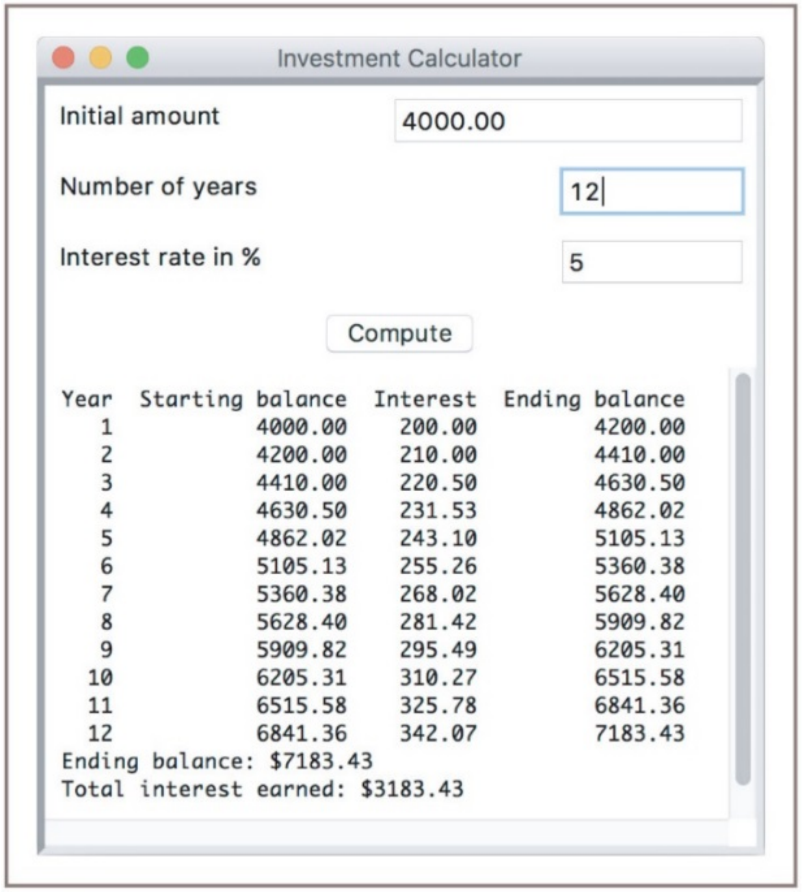


Figure 8-10 Using an integer field and a float field for input and output



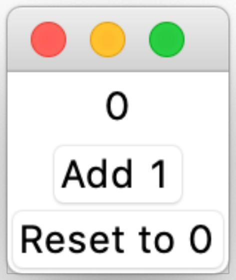
Initial amount 4000.00
Number of years 12
Interest rate in % 5

```
Year  Starting balance  Interest  Ending balance
  1            4000.00    200.00         4200.00
  2            4200.00    210.00         4410.00
  3            4410.00    220.50         4630.50
  4            4630.50    231.53         4862.02
  5            4862.02    243.10         5105.13
  6            5105.13    255.26         5360.38
  7            5360.38    268.02         5628.40
  8            5628.40    281.42         5909.82
  9            5909.82    295.49         6205.31
 10            6205.31    310.27         6515.58
 11            6515.58    325.78         6841.36
 12            6841.36    342.07         7183.43
Ending balance: $7183.43
Total interest earned: $3183.43
```

Figure 8-16 Displaying data in a multi-line text area

## tkinter Example

```
import tkinter as tk
window = tk.Tk() # Create a top-level widget main window
window.title("Terps") # Set window title
txtCount = tk.Label(window, text=counter) # Create a
    text label
txtCount.pack() # Show the text label
btnAdd = tk.Button(window, text="Add 1", command=Add)
    # Create a command button
btnAdd.pack() # Show the command button
btnReset = tk.Button(window, text="Reset to 0",
    command=Reset) # Create another command button
btnReset.pack() # Show the command button
window.mainloop() # Begin waiting for events
```


0
Add 1
Reset to 0