UI Programming with tkinter

olsen software

Contents

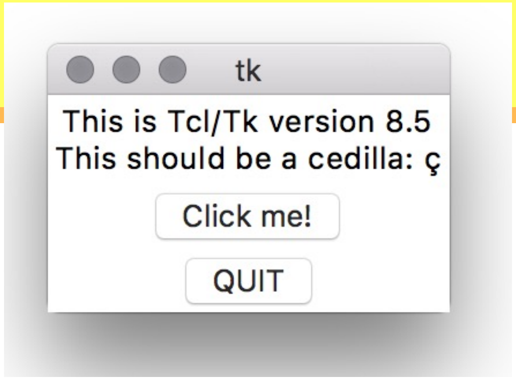
- Getting started with tkinter
- Widgets

Introducing tkinter

- tkinter is Python's standard UI library
  - Also Gtk, Qt, wxWidgets

- Update to latest version
  - Download from ActiveState

```
import tkinter      # Python 3 name

tkinter._test()     # Should say 8.5 or higher
```



This is Tcl/Tk version 8.5
This should be a cedilla: ç

- Need to know three things
  - Widgets
  - Geometry Managers
  - Events

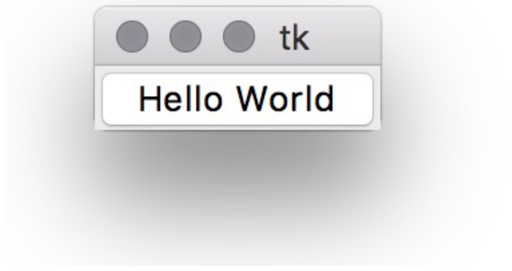
Hello World!

- We have to start with Hello World...

```
from tkinter import *
from tkinter import ttk

# Create a root
root = Tk()

# Add a button
ttk.Button(root, text="Hello World").grid()

# Listen for events
root.mainloop()
```

hello.py

- Important concepts
  - The root
  - tkinter supports "themed widgets"
  - Widgets have parents
  - Tk is event-driven



Widgets

- Widgets are the objects that make up your UI

- They form a hierarchy
  - Only the root doesn't have a parent

- Widgets are usually hosted in a frame

```
root = Tk()
content = ttk.Frame(root)     # root is the parent of the Frame
button = ttk.Button(content)  # the Frame is the parent of the Button
```

- You don't have to save a reference to a widget

Configuration

- Widgets typically have lots of options
  - Held in a dictionary
  - They try to be consistent with naming

- Use configure to change
  - Or use the dictionary directly

```
txt = button['text']                # check the text...

button['text'] = 'goodbye'          # change the text
button.configure(text='goodbye')    # another way to do the same thing


button.configure('text')            # get info about the 'text' option

button.configure()                  # get info about all options
```

The ttk Widget Set

- Button
- CheckButton
- RadioButton
- Entry
- Frame
- Label
- LabelFrame
- MenuButton
- TreeView
- Scale
- Scrollbar
- ComboBox
- Notebook
- ProgressBar
- Separator
- PanedWindow
- SizeGrip

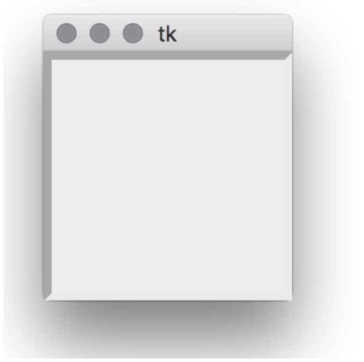
Basic Widgets - Frame

- A Frame is a container
  - Used to hold other widgets
  - Displays as a blank rectangle

```
root = Tk()
content = ttk.Frame(root)      # root is the parent of the Frame
button = ttk.Button(content)  # the Frame is the parent of the Button
```

- Set the width and height if you want an empty Frame!
  - Or the geometry manager will size it to nothing

```
root = Tk()
content = ttk.Frame(root, width=150, height=150,
      borderwidth=5, relief="sunken")
content.grid(row=0, column=0)
```



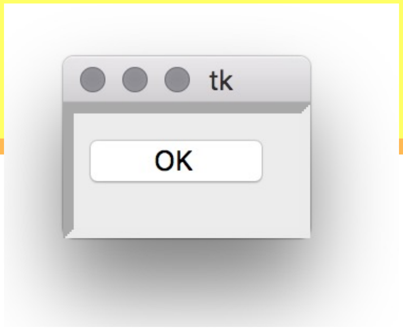
Basic Widgets - Frame

- Use padding to control internal margins
  - ttk Frames are different to plain tkinter Frames

- Add a border with a style and thickness

- You can specify dimensions
  - Defaults are pixels

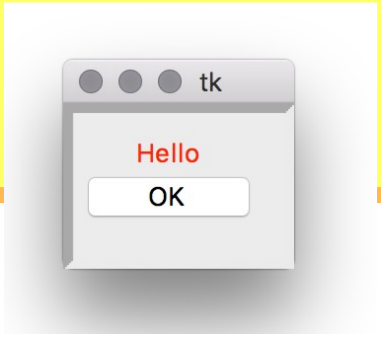
```
root = Tk()
content = ttk.Frame(root, padding='5 10 15 20 ', borderwidth=5, relief="sunken")
content.grid(row=0, column=0)

btn = Button(content, text='OK')
btn.grid(row=0, column=0)
```



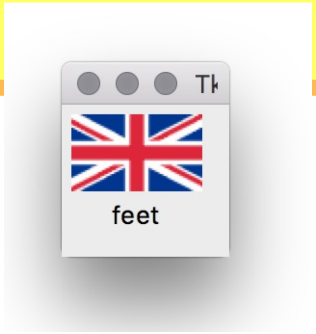
Basic Widgets - Label

- A Label can display text and/or an image
  - ttk Labels have different options to plain Labels

```
root = Tk()
content = ttk.Frame(root, padding='5 10 15 20 ', borderwidth=5, relief="sunken")
content.grid(row=0, column=0)

lbl = ttk.Label(content, text="Hello", foreground="red")
lbl.grid(row=0, column=0)

btn = ttk.Button(content, text='OK')
btn.grid(row=1, column=0)
```



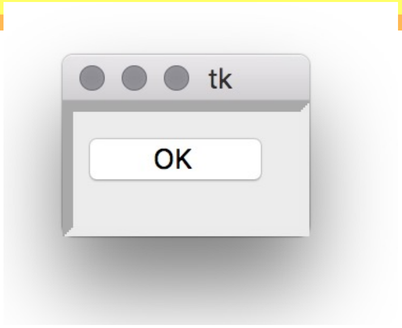
```
img = PhotoImage(file='UK-flag.gif')
lbl = ttk.Label(mainframe, text="feet", image=img, compound=TOP)
lbl.grid(column=1, row=1, sticky=S)
```



Basic Widgets - Button

- Button represents a simple button
  - Takes the same text and image options as Label
- Buttons typically specify a callback
  - The command to execute when pressed

```
root = Tk()
content = ttk.Frame(root, padding='5 10 15 20 ', borderwidth=5, relief="sunken")
content.grid(row=0, column=0)

button = ttk.Button(parent, text='Okay', command=submitForm)
btn.grid(row=0, column=0)
```



- Styled buttons don't support all options

Basic Widgets - Button

- A callback is simply a function
  - It doesn't have a special signature
  - Don't put the name in quotes!

```
# Callback function for button
def calculate():
    try:
        value = float(feet.get())
        meters.set((0.3048 * value * 10000.0 + 0.5) / 10000.0)
    except ValueError:
        pass

calcBtn = ttk.Button(mainframe, text="Calculate", command=calculate)
calcBtn.grid(column=3, row=3, sticky=W)
```

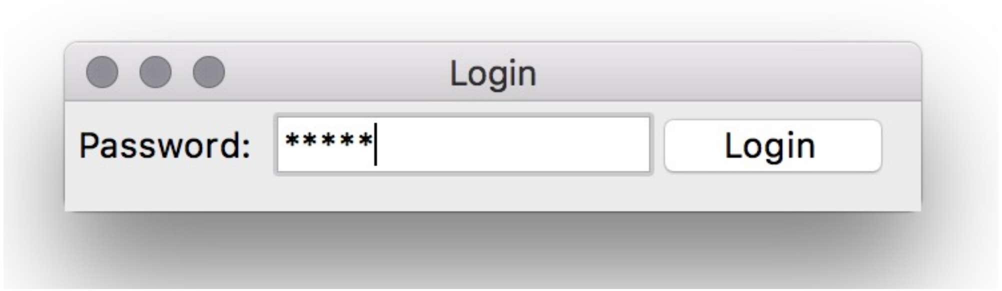
- Use a lambda to pass an argument

```
# Callback function for button
def calculate(number):
    # code using number

calcBtn = ttk.Button(mainframe, text="Calculate",
    command= lambda: calculate(1))
```

Basic Widgets - Entry

- An Entry is a single-line textbox
  - Can specify width
  - Can bind to a variable (see next slide)
  - Use 'show' for password fields
  - They support validation

```
pwd_entry = ttk.Entry(mainframe, width=15, textvariable=pwd, show='*')
pwd_entry.grid(column=1, row=0, sticky=(W, E))
ttk.Label(mainframe, text="Password: ").grid(column=0, row=0, sticky=W)

loginBtn = ttk.Button(mainframe, text="Login", command=check_login)
loginBtn.grid(column=2, row=0, sticky=W)
```



Data Binding

- You can bind a variable to a control
  - Changes in one will affect the other

- There are four types
  - StringVar, IntVar, DoubleVar, BoolVar
  - Use get() and set()

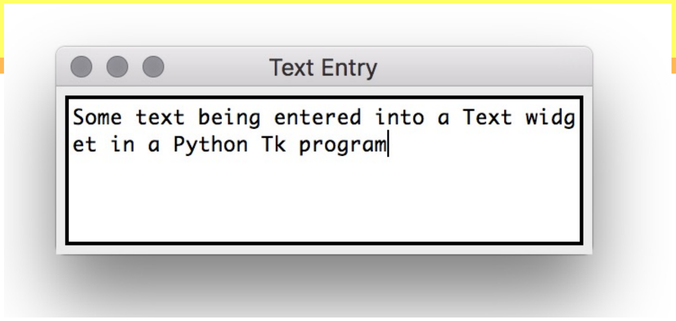
```
def check_login():
    pw = pwd.get()    # Get value of variable
    do_stuff(pw)

pwd = StringVar()     # Create a bindable variable

pwd_entry = ttk.Entry(mainframe, width=15, textvariable=pwd, show='*')
pwd_entry.grid(column=1, row=0, sticky=(W, E))
ttk.Label(mainframe, text="Password: ").grid(column=0, row=0, sticky=W)

loginBtn = ttk.Button(mainframe, text="Login", command=check_login)
loginBtn.grid(column=2, row=0, sticky=W)
```

Basic Widgets - Text

- Text is a multi-line text control
  - Not part of ttk
  - Can be used as a word processor

- Highly configurable
  - Fonts and colours
  - Line spacing
  - Text wrapping

```
txt = Text(c, width=40, height=5)
txt.grid(column=0, row=0)
```

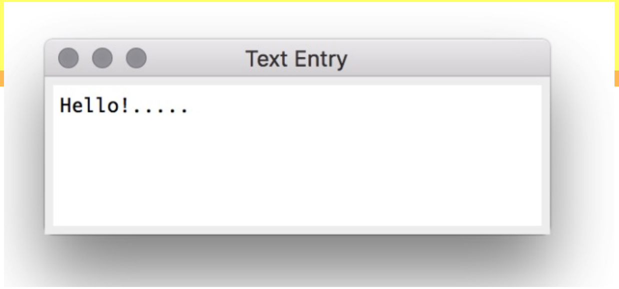


```
Some text being entered into a Text widg
et in a Python Tk program
```

Basic Widgets - Text

- Insert, retrieve and delete text from code
    - Positions use "line.char" format
    - Special END and INSERT positions

```
txt = Text(c, width=40, height=5)
txt.grid(column=0, row=0)

txt.insert(INSERT, 'Hello!')
txt.insert(END, '.....')
```



```
Hello!.....
```

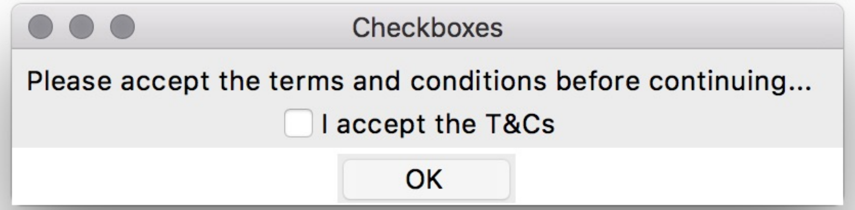
- Marks and Tags
    - Marks are bookmarks
    - Tags let you name ranges of characters

Basic Widgets - Checkbutton

- A Checkbutton is a button
  - It can have a command
  - It has a checkbox

- They usually have a StringVar
  - 'onvalue' and 'offvalue' set the value

```
TCs = StringVar()
check = ttk.Checkbutton(c, text='I accept the T&Cs',
                        command=acceptTCs, variable=TCs,
                        onvalue='accept', offvalue='reject')
```



Please accept the terms and conditions before continuing...

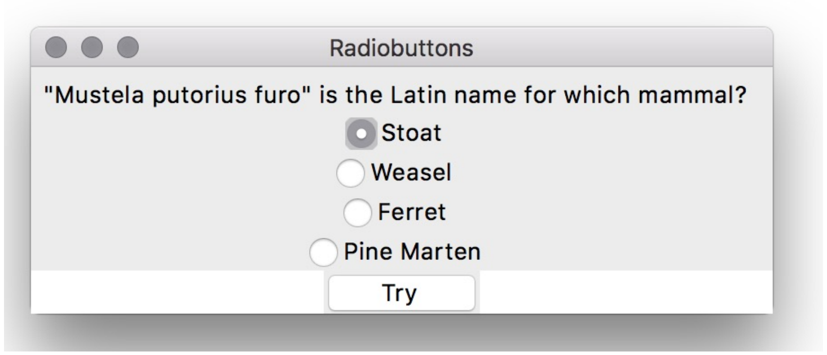
☐ I accept the T&Cs

Basic Widgets - Radiobutton

- Radiobuttons work in groups
  - A Radiobutton has a variable
  - All buttons with the same variable work together

- They have an associated StringVar
  - It has the value of the currently selected button

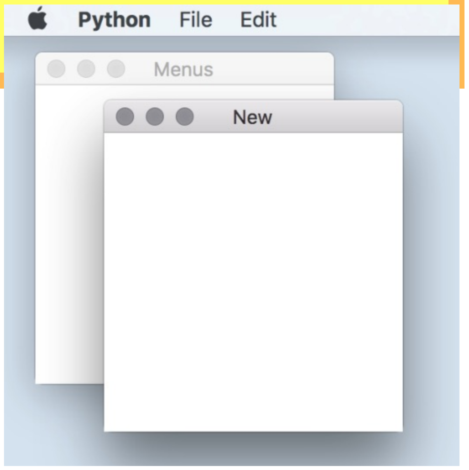
```
phone = StringVar()
home = ttk.Radiobutton(parent, text='Home', variable=phone, value='home')
office = ttk.Radiobutton(parent, text='Office', variable=phone, value='office')
cell = ttk.Radiobutton(parent, text='Mobile', variable=phone, value='cell')
```



Windows and Frames

- The Tk() command creates a top-level window
  - You can create more using Toplevel()
  - Frames live within windows

```
root = Tk()                    # Create a top-level window
content = ttk.Frame(root)      # A Frame lives in a window
win = Toplevel(root)           # Create a new toplevel window
```



- You can set the window geometry
  - Size and position

```
root = Tk()                    # Create a top-level window
root.title('Test')
root.geometry('300x300')
```

Any Questions?