

# SEATING TODAY

If you spend less than 60 min  
on the homework

If you spend more than  
60 min on the homework

SIT HERE



SIT HERE



# Lecture 7

GUI Programming

Prof. Steven Ludtke  
N410.07, [sludtke@bcm.edu](mailto:sludtke@bcm.edu)

# eval() and exec()

- eval() - evaluates a string expression, returns result
- exec() - executes a string with python code
- Note: These functions are dangerous and can create severe security holes in your applications. Use them with great care !

# Scope

What will this produce ?

```
def f(x):  
    y=x*10  
    return y
```

```
x=5
```

```
y=6
```

```
print y
```

```
print f(x)
```

```
print y
```

# Scope

How about this ?

```
def f():  
    y=x*10  
    return y
```

```
x=5
```

```
y=6
```

```
print y
```

```
print f()
```

```
print y
```

# Scope

- Local scope
  - Variables defined within a function, exist only within the function
- Global scope
  - Variables defined at the “top level” in the program or module.
  - Variables declared using the “global” keyword
- Built-in names
  - Built in functions and variable names

# GUI Programming

Standard



Best  
(IMHO)



- Tkinter, PyQt, PyGTK, wxPython, FXPy
- widget - A graphical object, like a button or a slider
- callback - a function which is called when the user interacts with a widget
- geometry or layout manager - controls where widgets are displayed

# Tkinter

- 'standard' Python GUI toolkit
- Python interface elegant, but built on top of Tcl/Tk
- A bit clunky and slow, but has been used to build some very large applications (eg - Chimera)
- If you have a choice, for larger projects, use PyQt4 (just my suggestion)
- <http://www.pythonware.com/library/tkinter/introduction/index.htm>
- Tkinter extended by PMW and Tix



# Modal Widgets

- Get specific info from the user without writing a full GUI for the program.
- or can be used as part of a full GUI.
  
- `tkFileDialog`
- `tkMessageBox`
- `tkColorChooser`

# tkFileDialog

- `import tkFileDialog`
  - `askdirectory(**options)`
  - `askopenfile(mode='r', **options)`
  - `askopenfilename(**options)`
  - `askopenfilenames(**options)`
  - `askopenfiles(mode='r', **options)`
  - `asksaveasfile(mode='w', **options)`
  - `asksaveasfilename(**options)`

# tkMessageBox

- `import tkMessageBox`
  - `askokcancel(title=None, message=None, **options)`
  - `askquestion(title=None, message=None, **options)`
  - `askretrycancel(title=None, message=None, **options)`
  - `askyesno(title=None, message=None, **options)`
  - `showerror(title=None, message=None, **options)`
  - `showinfo(title=None, message=None, **options)`
  - `showwarning(title=None, message=None, **options)`

# tkColorChooser

- `import tkColorChooser`
- `askcolor(color=None, **options)`

# Tkinter

- Event driven programming
  - Set up all of your widgets
    - Create widget
    - Set callbacks
    - Place widget in window
  - Call the event loop
  - Cleanup

```
from Tkinter import *
root = Tk()          # Initializes Tkinter

###setup widgets

root.mainloop()     # Runs the GUI until the user triggers an exit
root.destroy()      # Cleanup
```

# Simple Tkinter

```
from Tkinter import *
```

```
root = Tk()
```

```
w = Label(root, text="Hello, world!")
```

```
w.pack()
```

```
root.mainloop()
```

# Tkinter Widgets

- BitmapImage
- Button
- Canvas
  - Arc, Bitmap, Image, Line, Oval, Polygon, Rectangle, Text
- Checkbutton
- Entry
- Font
- Frame (window)
- Label
- Listbox
- Menu/Menubutton
- Message
- PhotoImage
- Radiobutton
- Scale
- Scrollbar
- Text
- Toplevel Widget

# Tkinter Misc

- DoubleVar
- IntVar
- StringVar
  
- SimpleDialog
  
- tkFont
  
- For Callbacks, use:
- command, after, bind



# Geometry Managers

- Grid Geometry Manager
  - Arrange widgets like a table
- Pack Geometry Manager
  - Arrange widgets sequentially into the available space
- Place Geometry Manager
  - Explicitly position widgets - tricky

# Button Callback Example

```
from Tkinter import *  
root = Tk()
```

```
def pushed(): print "You pushed me too far!"
```

```
w = Button(root, text="Push Me",command=pushed)  
w.pack()
```

```
root.mainloop()
```

# Entry Example

```
import Tkinter as tk
root = tk.Tk()

def enter(): print "You entered: ",str(v.get())

l=tk.Label(root,text="Enter something:")
l.pack()

v=tk.StringVar()
w = tk.Entry(root, width=40,textvariable=v)
w.pack()
v.set("Start")

w2 = tk.Button(root, text="Push Me",command=enter)
w2.pack()

root.mainloop()
```

# Timer Callback Example

```
from Tkinter import *
root = Tk()

def timeout(): print "It's Time !"

w = Button(root, text="Push Me")
w.pack()
w.after(3000, timeout)

root.mainloop()
```

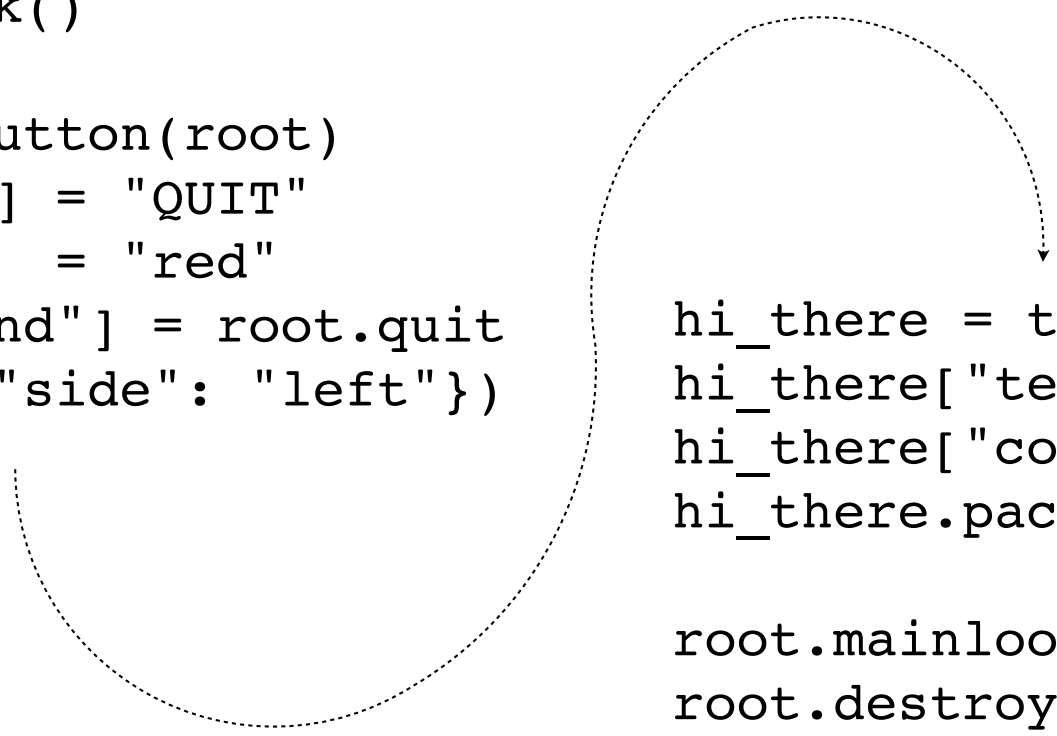
# Full Tkinter Example

```
import Tkinter as tk
import tkMessageBox
```

```
def say_hi():
    tkMessageBox.showinfo("Hi", "Hello There !")
```

```
root = tk.Tk()
```

```
QUIT = tk.Button(root)
QUIT["text"] = "QUIT"
QUIT["fg"] = "red"
QUIT["command"] = root.quit
QUIT.pack({"side": "left"})
```



```
hi_there = tk.Button(root)
hi_there["text"] = "Hello",
hi_there["command"] = say_hi
hi_there.pack({"side": "left"})
```

```
root.mainloop()
root.destroy()
```

# tkinter References

- <http://www.pythonware.com/library/tkinter/introduction/index.htm>
- <http://infohost.nmt.edu/tcc/help/pubs/tkinter.pdf>
- A few kindle books are available as well. “Python and Tkinter Programming” out of print

# in-class lab

- Those in the 'beginner' group:
  - Write a program:
    - User presented with text box to enter a function of  $x$ , i.e. - “ $\cos(x)$ ” and a button
    - When the button is pressed, a window should open showing a plot of this function over the  $X$  range of 0-10 with a step of 0.1.
    - The program can exit or continue after the plot is closed. Your choice.
- Those in the 'advanced' group:
  - Start with the program above, but add some additional features: The user should be able to enter multiple equations to be plotted on the same plot, the user should be able to define the minimum and maximum values for  $X$ , and the user should be able to select a color or line-style for each line.
- You may work alone or in groups of 2-3. You may ask me questions out loud or privately at any time. If you email me a functioning program during class with subject “Lab 8” this will take the place of Friday’s homework.

# Homework 8

- For those who didn't successfully complete Lab 8 or missed class:
- Pick one of the previous homework assignments (one that worked) and add a simple GUI to it. You should not use `sys.argv`, `input` or `raw_input` in the new version of the program, but instead get what you need using a GUI interface. The program should use either Tkinter or PyQt4. I'll give a point of extra credit if you use PyQt4.