Lecture #14: GUI: Tkinter

Presented by Jamal Bouajjaj

2023–11–05

For University of New Haven's Fall 2023 CSCIxx51 Course

Interface

There are 3 "interface types" that are possible with a modern computer

- CLI: Command-Line Interface. Interface with the application thru a command line sending text to it
- TUI: Terminal User Interface. Interface with the application thru a terminal, but interactivally
- GUI: Graphical User Interface. Interface with the application with graphics

So far, we have only done CLI. We will be going over GUI, but not TUI (if you're intrested, look into *curses*

Widget Toolkit

A *Widget Toolkit* or *GUI Framework* is a library (or multiple) that allows you to make a GUI application. A lot of them are OS-independent.

There are many toolkits for Python, including:

- Tkinter (what we will be using)
- PyQT/PySide
- wxPython
- Kivy

[1]

[1] https://en.wikipedia.org/wiki/Widget_toolkit

tkinter

tkinter

tkinter is the defacto-standard toolkit for making simple GUI in Python. So much in fact that Python ships with it as a standard library.

tkinter is technically a Python wrapper for *tk*, a GUI framework that is ran under the *tcl* programming language.

There is also *tkinter.ttk*, which just has themed widgets, and is also part of the *tk* library.

Widgets

A widget is some GUI element that can be used as an element of interaction. Think like buttons, scroll-bars, labels, etc. Tk has many widgets, all in the official docs. Here are some common/popular ones

- label
- button
- listbox
- checkbutton
- text
- radiobox
- canvas

AND MORE!

Invoking

To get started, we need to make a top-level (a window) that is the root, the top-most level. Then we can run it

```
import tkinter as tk
# The top-level window
root = tk.Tk()
# This runs TK. This will HANG FOREVER until the GUI is
↪  closed
root.mainloop()
```

Widget

Let's add a widget

```
import tkinter as tk
# The top-level window
root = tk.Tk()

label = tk.Label(root, text="Welcome World!")
label.pack()    # We place the label (see next slide)

# This runs TK. This will HANG FOREVER until the GUI is closed
root.mainloop()
```

Placement

How do we place widgets? Tkinter offers 3 "styles.

- pack(): Places widgets on top of one another, or side to side.
- grid(): Places the widget in a grid-like fashion
- place(): Places the widget in direct x-y coordinates in the window

You must stick to one placement style per frame/top-level.

Binding

Each widget can have events binded to it, such as a key press, button press, or other events. Each binding call need to take one argument, which is the event

```
    root = tk.Tk()

    text = tk.Label(root, text="A thing!")
    text.pack()

    text.bind("<Enter>", lambda x: print("Entered"))
    text.bind("<Leave>", lambda x: print("Left"))

    root.bind("q", lambda x: root.destroy())

    # run
    root.mainloop()
```

Messagebox

There are several built-in message box, such as

```
from tkinter import messagebox
# The top-level window
messagebox.showinfo("TITLE", "MESSAGE")
messagebox.showwarning("TITLE", "MESSAGE")
messagebox.showerror("TITLE", "MESSAGE")
messagebox.askquestion("TITLE", "MESSAGE")
messagebox.askokcancel("TITLE", "MESSAGE")
messagebox.askretrycancel("TITLE", "MESSAGE")
messagebox.askyesno("TITLE", "MESSAGE")
messagebox.askyesnocancel("TITLE", "MESSAGE")
```

Dialogs

There are several built-in dialogs into tkinter, such as

```
from tkinter import simpledialog
# The top-level window
simpledialog.askfloat("TITLE", "MESSAGE")
simpledialog.askinteger("TITLE", "MESSAGE")
simpledialog.askstring("TITLE", "MESSAGE")
```

Also file dialogs

```
from tkinter import filedialog
# Actually return a FileIO
filedialog.askopenfile()
filedialog.asksaveasfile()
# Only return file name
filedialog.askopenfilename()
filedialog.asksaveasfilename()
# Ask for directory
filedialog.askdirectory()
```

References

- https://www.tcl.tk/man/tcl8.6/TkCmd/contents.html
- https://docs.python.org/3/library/tk.html
- https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/index.html

End

The end.

End

The end.

Wait, don't we need to know more?

End

The end.

Wait, don't we need to know more?

Well yes, but I will be doing that as an in-class demo/live coding