Craft Your Own GUIs with Python and Tkinter

# Penn State MacAdmins 2016
Talking Tkinter

Grab the source code for all examples here:

https://github.com/brysontyrrell

“MacAdmins-2016-Craft-GUIs-with-Python-and-Tkinter”
Talking Tkinter

Run the examples using:

/usr/bin/python example.py
Talking Tkinter

We are gathered here to talk about...

- Python and Tkinter
- Adding another tool to your toolbox

We aren’t here to talk about...

- Convincing you that Tkinter is the best option
- Not using other solutions
What are some pros of using this toolkit?

- Python and Tkinter are standard on macOS
- All of your code can be in one file/script/program
- Your GUIs don’t need to close to process actions
What comes with Tkinter?
What comes with Tkinter?
What comes with Tkinter?
What comes with Tkinter?

- The examples in ‘Tkinter_Widget_Examples.py’ explore different widgets and different means of interacting with those widgets.
- Several make use of Tkinter variables for storing and retrieving values from the inputs.
- Comboboxes, Listboxes and OptionMenus can all be dynamically generated and updated using Python lists and dictionaries for mappings.
The Things You Can Do
Getting Started

‘Tkinter’ v. ‘ttk’

• ‘Tkinter’ is the base library that creates the GUI app and contains all the widgets

• ‘ttk’ is an add-on that provided themed versions of several Tkinter widgets and several special widgets not available in Tkinter
Getting Started

Creating a “boilerplate”

- Yes, there’s a bit of code to go with Tkinter
- The “boilerplate” is a basic template we will use as the base for any future GUI script

We’re going to do a walkthrough of this code
Basic and Ugly

The boilerplate will try to match the basic look of an OS X modal dialog.

The dialog on the top is from Mavericks.

The dialog on the bottom is El Capitain.
Basic and Ugly

import Tkinter as tk

class App(tk.Frame):
    def __init__(self, master):
        tk.Frame.__init__(self, master)
        self.pack()
        self.master.title("Hello World")

        tk.Label(self, text="This is your first GUI. (highfive)" ).pack()

if __name__ == '__main__':
    root = tk.Tk()
    app = App(root)
    app.mainloop()
import subprocess

class App(tk.Frame):
    def __init__(self, master):
        self.master.resizable(False, False)
        self.master.tk_setPalette(background='#ececec')

        x = (self.master.winfo_screenwidth() -
             self.master.winfo_reqwidth()) / 2
        y = (self.master.winfo_screenheight() -
             self.master.winfo_reqheight()) / 3
        self.master.geometry("+{}+{}").format(x, y)

        self.master.config(menu=tk.Menu(self.master))

if __name__ == '__main__':
    subprocess.call(['/usr/bin/osascript', '-e', 'tell app "Finder" to set frontmost of process "Python" to true'])
class App(tk.Frame):
    def __init__(self, master):
        ...
        tk.Button(self, text='OK', default='active',
                  command=self.click_ok).pack(side='right')
        ...
        tk.Button(self, text='Cancel',
                  command=self.click_cancel).pack(side='right')
        ...
    
def click_ok(self):
        print("The user clicked 'OK'")
    
def click_cancel(self):
        print("The user clicked 'Cancel'")
        self.master.destroy()
class App(tk.Frame):
    def __init__(self, master):
        ...
    dialog_frame = tk.Frame(self)
dialog_frame.pack(padx=20, pady=15)

    tk.Label(dialog_frame, text="This is your first GUI. (highfive)").pack()

    button_frame = tk.Frame(self)
button_frame.pack(padx=15, pady=(0, 15), anchor='e')

    tk.Button(button_frame, text='OK', default='active',
              command=self.click_ok).pack(side='right')

    tk.Button(button_frame, text='Cancel',
              command=self.click_cancel).pack(side='right')
    ...

Organize With Frames
import AppKit
...

class App(tk.Frame):
    def __init__(self, master):
        ...
        self.master.protocol('WM_DELETE_WINDOW', self.click_cancel)
        self.master.bind('<Return>', self.click_ok)
        self.master.bind('<Escape>', self.click_cancel)
        ...

if __name__ == '__main__':
    info = AppKit.NSBundle.mainBundle().infoDictionary()
    info['LSUIElement'] = True
    ...
Example Time

We’re going to check out two example Tkinter GUI prompts that use more advanced options built on top of our boilerplate code.
Password Prompt

Please authenticate with your username and password before continuing.

Username: 
Password: 

Cancel  OK

Please authenticate with your username and password before continuing.

Username: 
Password: 

Cancel  OK
class App(tk.Frame):
    def __init__(self, master):
        ...

        tk.Message(self, text="Please authenticate with your username and password before continuing.", font='System 14 bold', justify='left', aspect=800).pack(pady=(15, 0))
        ...

```python
Password Prompt

dialog_frame = tk.Frame(self)
dialog_frame.pack(padx=20, pady=15, anchor='w')

tk.Label(dialog_frame, text='Username:').grid(row=0, column=0, sticky='w')
self.user_input = tk.Entry(dialog_frame, background='white', width=24)
self.user_input.grid(row=0, column=1, sticky='w')
self.user_input.focus_set()

tk.Label(dialog_frame, text='Password:').grid(row=1, column=0, sticky='w')
self.pass_input = tk.Entry(dialog_frame, background='white', width=24, show='*')
self.pass_input.grid(row=1, column=1, sticky='w')
...

def click_ok(self, event=None):
    print('The user clicked 'OK':
    Username: {}
    Password: {}".format(self.user_input.get(), self.pass_input.get()))
```
File Uploader
self.file_count = tk.StringVar(value='')

self.file_label = tk.Label(file_frame, textvariable=self.file_count, anchor='e')
self.file_label.pack(side='right')

tk.Label(self, text="Add a comment:").pack(padx=15, pady=(15, 0), anchor='w')

self.text = tk.Text(text_frame, width=30, height=4, highlightbackground='ffffff',
                    highlightcolor="#7baedc",
                    bg='ffffff', wrap=tk.WORD, font=('System', 14))
self.text.focus_set()
self.text.pack()
import tkFileDialog

def file_picker(self):
    self.selected_files = tkFileDialog.askopenfilenames(parent=self)
    self.file_count.set('{} file(s)'.format(len(self.selected_files)))
def _toggle_state(self, state):
    state = state if state in ('normal', 'disabled') else 'normal'
    widgets = (self.file_button, self.file_label, self.text, self.submit_button, self.cancel_button)
    for widget in widgets:
        widget.configure(state=state)

...
class LoadingFrame(tk.Frame):
    def __init__(self, master, count):
        tk.Frame.__init__(self, master, borderwidth=5, relief='groove')
        self.grid(row=0, column=0)

        tk.Label(self, text="Your files are being uploaded").pack(padx=15, pady=10)

        self.progress = ttk.Progressbar(self, orient='horizontal', length=250, mode='determinate')
        self.progress.pack(padx=15, pady=10)
        self.progress['value'] = 0
        self.progress['maximum'] = count
        ...

File Uploader
def click_submit(self, event=None):
    print("The user clicked 'OK'")
    comment = self.text.get('1.0', 'end')

    if comment.rstrip():
        print('The user entered a comment:"
        print(comment.rstrip())

    if self.selected_files:
        loading = LoadingFrame(self.master, len(self.selected_files))
        self._toggle_state('disabled')
        print('The user has selected files:')
        for path in self.selected_files:
            loading.progress['value'] += 1
            self.update()
            print('File {}/{}'.format(loading.progress['value'], loading.progress['maximum']])
            time.sleep(2)
            with open(path) as f:
                print('Opened file: {}\'.format(path, f))

        print('Loading screen finished')
        loading.destroy()
        self._toggle_state('normal')
        ...

File Uploader
Interested in More?

Official python.org Docs
https://docs.python.org/2/library/tkinter.html

Stack Overflow
http://stackoverflow.com/questions/tagged/tkinter

Effbot (old Tkinter docs)
http://effbot.org/tkinterbook/

TkDocs (new Tkinter docs)
http://www.tkdocs.com/
Thank you!

Q&A?