A. tkinter_introduction

March 21, 2024

1 Introduction to Tkinter

By Karthik Nair

Tkinter is a Python binding to the Tk GUI toolkit. Tk is a FOSS cross-platform widget toolkit that provides a library of basic elements of GUI widgets for building a graphical user interface in many programming languages.

Tcl (Tool Command Language) internally handles the application's main window as a widget. Tkinter calls are translated into Tcl commands which are fed to this embedded interpreter, thus making it possible to mix Python and Tcl in a single application.

Pros

- Built into the Python standard library
- lightweight and easy to build something quickly
- Cross-platform (Windows, macOS, Linux, UNIX)
- Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they're run.

Cons

- Outdated look and feel
- Not as many widgets as other toolkits like Qt
- Since visual elements are rendered using native operating system elements, applications built with Tkinter look like they belong on the platform where they're run. This means that the appearance of your application can vary from platform to platform, which can be a problem if you want your application to have a consistent look and feel across platforms.

1.0.1 Installation

1. Windows

tkinter is included in the standard Windows Python distribution.

2. Mac OS

`pip3 install tk` might work, if it doesn't for you, try using `brew` or any other alternative methods on the internet

3. Linux

Debian based (Debian/Ubuntu/Mint/Pop!_OS/..) `sudo apt-get install python3-tk` Red Hat based (Red Hat/Fedora/CentOS/..) `sudo yum install python3-tkinter` Arch based (Arch/Manjaro/..) `sudo pacman -S tk`

[1]:
```
from tkinter import *
from tkinter import ttk
root = Tk()
frm = ttk.Frame(root, padding=50)
frm.grid() # same as frm.grid(column=0, row=0)
ttk.Label(frm, text="Hello World!").grid(column=3, row=1)
ttk.Button(frm, text="Quit", command=root.destroy).grid(column=1, row=0)
root.mainloop()
```

- ttk provides widgets
- root is the main window
- `root = tk.Tk()` creates a top level window, known as the root window
- root window is the main window of the application
- `frm = ttk.Frame(root)` creates a frame widget inside the root window
  - padding is the space between the frame and the widgets inside it
- `frm.grid()` makes the frame visible
- `ttk.Label(frm, text="Hello World!").grid(column=0, row=0)`
  - Label is a widget that displays text or images
  - grid with column and row arguments places the widget in the specified location
- `ttk.Button(frm, text = "Quit", command = root.destroy).grid(column=1, row=0)`
  - Button is a widget that performs an action when clicked
  - command argument specifies the function to be called when the button is clicked
  - root.destroy is a function that closes the window
- `root.mainloop()` starts the event loop
  - event loop is an infinite loop that waits for events to happen and then processes the event accordingly
  - event is an action that the user performs, such as clicking a button or resizing a window
  - event loop is responsible for dispatching events to the widgets, such as button clicks and key presses
  - event loop is terminated when the window is closed

1.0.2 Important Widget Classes

- Label
  - Displays text or images
- Button
  - Performs an action when clicked
- Entry

  - Accepts a single line of text input from the user
- Checkbutton
  - Allows the user to select one or more options from a set
- Radiobutton
  - Allows the user to select one option from a set
- Scale
  - Allows the user to select from a range of values by moving a slider
- Listbox
  - Displays a list of options from which the user can select one or more
- Text
  - Displays multiple lines of text that the user can edit
- Canvas
  - Displays graphics and other widgets such as buttons and entry boxes
- Menu
  - Displays a menu of options that the user can select from
- Menubutton
  - Displays a menu of options that the user can select from
- Dialog
  - Displays a dialog box for the user to make a choice or enter information

[ ]:
```
# label showing text "Hello World!"

from tkinter import *
from tkinter import ttk
root = Tk()
frm = ttk.Frame(root, padding=10)
frm.grid()
ttk.Label(frm, text="Hello World!").grid(column=0, row=0)
ttk.Button(frm, text="Quit", command=root.destroy).grid(column=1, row=0)
root.mainloop()
```

[1]:
```
# label showing image img.png

from tkinter import *
from tkinter import ttk

root = Tk()
frm = ttk.Frame(root, padding=10)
frm.grid()
img = PhotoImage(file="images/img.png")
ttk.Label(frm, image=img).grid(column=0, row=0)
ttk.Button(frm, text="Quit", command=root.destroy).grid(column=1, row=0)
root.mainloop()
```

[3]:
```
# check box

from tkinter import *
```

```
from tkinter import ttk

root = Tk()
frm = ttk.Frame(root, padding=10)
frm.grid()
tea = StringVar()
coffee = StringVar()
ttk.Checkbutton(frm, text="tea", variable=tea).grid(column=0, row=0)
ttk.Checkbutton(frm, text="coffee", variable=coffee).grid(column=0, row=1)
ttk.Button(frm, text="Quit", command=root.destroy).grid(column=2, row=0)
root.mainloop()
```

[5]:
```
# radio buttons

from tkinter import *
from tkinter import ttk

root = Tk()
frm = ttk.Frame(root, padding=10)

tea_var = StringVar()
coffee_var = StringVar(value="coffee")
sugar_var = StringVar()

frm.grid(column=0, row=0)
# tea_rb = ttk.Radiobutton(frm, text="Tea", variable=tea_var).grid(column=0,␣
 ↪row=0, sticky=W)
tea_rb = ttk.Radiobutton(frm, text="Tea", variable=tea_var).grid(column=0,␣
 ↪row=0)
coffee_rb = ttk.Radiobutton(frm, text="Coffee", variable=coffee_var,␣
 ↪value="coffee").grid(column=0, row=1)
sugar_rb = ttk.Radiobutton(frm, text="Sugar", variable=sugar_var).
 ↪grid(column=0, row=2)

# frm.grid(column=0, row=0)
#tea_rb.grid(column=0, row=0, sticky=W)
#coffee_rb.grid(column=0, row=1, sticky=W)
#sugar_rb.grid(column=0, row=2, sticky=W)

ttk.Button(frm, text="Quit", command=root.destroy).grid(column=1, row=0)
root.mainloop()
```

[5]:
```
# tkinter dialog box

from tkinter import *
from tkinter import ttk
from tkinter import messagebox
```

```
root = Tk()
root.withdraw()
messagebox.showinfo(title="A Friendly Message", message="Hello, Tkinter!")
answer = messagebox.askquestion(title="Hungry?", message="Do you want Coffee?")
if answer == "yes":
    print("Coffee")
    root.destroy()
else:
    print("Tea?")
    root.destroy()
root.mainloop()
```

```
Tea?
```

[4]:
```
# input dialog box
# input name and display name in a label

from tkinter import *
from tkinter import ttk
from tkinter import simpledialog
root = Tk()

root.withdraw()
name = simpledialog.askstring(title="Name", prompt="What is your name?")
root.deiconify() # deiconify function : to show the window again
ttk.Label(root, text="Hello, " + name + "!").grid()
ttk.Button(root, text="Quit", command=root.destroy).grid()


root.mainloop()
```

Entry widget and working of functions

[15]:
```
import tkinter as tk
from tkinter import ttk

def display_greeting(name):
    greeting_label.config(text=f"Hello, {name}!")

def get_name():
    name = name_entry.get()
    display_greeting(name)

root = tk.Tk()
root.title("Greeting")

frame = ttk.Frame(root, padding=20)
frame.grid(row=0, column=0, padx=10, pady=10)
```

```
# widgets
name_label = ttk.Label(frame, text="Enter your name:")
name_entry = ttk.Entry(frame)
greeting_button = ttk.Button(frame, text="Greet", command=get_name)
greeting_label = ttk.Label(frame, text="")

name_label.grid(row=0, column=0, sticky="E")
name_entry.grid(row=0, column=1, padx=5, pady=5)
greeting_button.grid(row=1, column=0, columnspan=2, pady=10)
greeting_label.grid(row=2, column=0, columnspan=2)

root.mainloop()
```

- ttk.Entry(frame) creates an entry widget inside the frame
  - entry widget is used to accept a single line of text input from the user
  - name_entry.get() gets the text from the entry widget
- ttk.Button(frame, text=“Greet”, command=get_name) creates a button widget inside the frame
  - button widget is used to perform an action when clicked
  - command argument specifies the function to be called when the button is clicked
- greeting_label.config(text=f”Hello, {name}!“) changes the text of the label widget
  - label widget is used to display text or images
  - config method is used to change the configuration of the widget

1.0.3 Simple login program

[23]:
```
from tkinter import *
from tkinter import ttk
from tkinter import messagebox

# Declare root as a global variable
root = None

def home():
    global root
    if root:
        root.destroy()  # Destroy the existing root window if it exists
    root = Tk()
    root.title("HOME")
    root.resizable(False, False)
    frm = ttk.Frame(root, padding=10)
    frm.grid()

    global pass_var
    global user_var
    user_var = StringVar()
```

```
    pass_var = StringVar()

    ttk.Label(frm, text="Username:").grid(column=0, row=0, sticky=W)
    user_entry = ttk.Entry(frm, textvariable=user_var)
    user_entry.grid(column=1, row=0, sticky=(W, E))
    user_entry.focus()

    ttk.Label(frm, text="Password:").grid(column=0, row=1, sticky=W)
    pass_entry = ttk.Entry(frm, textvariable=pass_var, show="*")
    pass_entry.grid(column=1, row=1, sticky=(W, E))

    ttk.Button(frm, text="Login", command=login).grid(column=1, row=2, sticky=E)
    ttk.Button(frm, text="Quit", command=quit).grid(column=2, row=2, sticky=E)

    root.mainloop()


def login():
    global root
    if user_var.get() == "admin" and pass_var.get() == "1234":
        root.destroy()
        root = Tk()
        root.title("Login Successful")
        root.resizable(False, False)
        frm = ttk.Frame(root, padding=10)
        frm.grid()
        ttk.Label(frm, text="Login Successful").grid(column=0, row=0, sticky=W)
        ttk.Button(frm, text="Home", command=home).grid(column=1, row=1,␣
 ↪sticky=E)
        ttk.Button(frm, text="Quit", command=quit).grid(column=2, row=1,␣
 ↪sticky=E)
    else:
        root.destroy()
        root = Tk()
        root.title("Login Unsuccessful")
        root.resizable(False, False)
        frm = ttk.Frame(root, padding=10)
        frm.grid()
        ttk.Label(frm, text="Login Unsuccessful").grid(column=0, row=0,␣
 ↪sticky=W)
        ttk.Button(frm, text="Home", command=home).grid(column=1, row=1,␣
 ↪sticky=E)
        ttk.Button(frm, text="Quit", command=quit).grid(column=2, row=1,␣
 ↪sticky=E)

def quit():
    global root
```

```
    root.destroy()


home()
```

- workflow
  - home function creates the root window and the widgets inside it and destroys the existing root window if it exists
  - login function checks the username and password and creates a new window based on the result
  - quit function closes the window
- `root.resizable(False, False)` makes the window non-resizable
  - first argument is for the width
  - second argument is for the height
- `user_entry.focus()` sets the focus to the entry widget
  - focus is the widget that receives keyboard input
- `show="*"` hides the text in the entry widget
  - show is the character to be displayed instead of the actual text
  - useful for password fields, for example
- `user_var.get()` gets the text from the entry widget
  - get method is used to get the text from the entry widget

1.0.4 fill_zeros program

[27]:
```
import pandas as pd
from tkinter import *
from tkinter import filedialog
from tkinter import messagebox
from tkinter import ttk
import os


def fill_zeros(filename, new_filename, fill_value):
    try:
        df = pd.read_csv(filename, header=None)
        df.replace(0, fill_value, inplace=True)
        df.to_csv(new_filename, index=False, header=False)
        messagebox.showinfo("Success", "Operation completed successfully!")
    except Exception as e:
        messagebox.showerror("Failed", f"An error occurred: {str(e)}")


def choose_file(entry_widget):
    filename = filedialog.askopenfilename()
    entry_widget.delete(0, END)
    entry_widget.insert(0, filename)
```

```
def choose_folder(entry_widget):
    foldername = filedialog.askdirectory()
    entry_widget.delete(0, END)
    entry_widget.insert(0, foldername)


def main():
    root = Tk()
    root.title("Fill Zeros")
    root.resizable(False, False)

    frm = ttk.Frame(root, padding=10)
    frm.grid(row=0, column=0)

    Label(frm, text="Choose the input file:").grid(row=0, column=0, sticky=W)
    filename_entry = Entry(frm)
    filename_entry.grid(row=0, column=1)
    Button(frm, text="Browse", command=lambda: choose_file(filename_entry)).
 ↪grid(row=0, column=2)

    Label(frm, text="Choose the output folder:").grid(row=1, column=0, sticky=W)
    folder_entry = Entry(frm)
    folder_entry.grid(row=1, column=1)
    Button(frm, text="Browse", command=lambda: choose_folder(folder_entry)).
 ↪grid(row=1, column=2)

    Label(frm, text="Enter the new file name:").grid(row=2, column=0, sticky=W)
    new_filename_entry = Entry(frm)
    new_filename_entry.grid(row=2, column=1)

    Label(frm, text="Enter the fill value:").grid(row=3, column=0, sticky=W)
    fill_value_entry = Entry(frm)
    fill_value_entry.grid(row=3, column=1)

    Button(frm, text="Fill Zeros", command=lambda: fill_zeros(filename_entry.
 ↪get(), os.path.join(folder_entry.get(), new_filename_entry.get()),
                                                              fill_value_entry.
 ↪get())).grid(row=4, column=1)

    Button(frm, text="Quit", command=root.destroy).grid(row=5, column=1)

    root.mainloop()


if __name__ == "__main__":
    main()
```

Functions

- `fill_zeros(filename, new_filename, fill_value)`
  - Parameters:
    - `filename`: str, path of the input CSV file.
    - `new_filename`: str, path of the output CSV file.
    - `fill_value`: int, value to fill zeros with.
  - shows a success message box if the operation is successful.
  - shows an error message box if the operation fails.
- `choose_file(entry_widget)`: Opens a file dialog to choose a file and inserts its path into the specified entry widget. - `filedialog.askopenfilename()`: Opens a file dialog to choose a file and returns the file path. - `entry_widget`: Entry, widget where the selected file path will be inserted. - entry_widget.delete(0, END) deletes the current text in the entry widget. - entry_widget.insert(0, filename) inserts the selected file path into the entry widget.
- `choose_folder(entry_widget)`: Opens a folder dialog to choose a folder and inserts its path into the specified entry widget. - `filedialog.askdirectory()`: Opens a folder dialog to choose a folder and returns the folder path. - `entry_widget`: Entry, widget where the selected folder path will be inserted. - entry_widget.delete(0, END) deletes the current text in the entry widget. - entry_widget.insert(0, foldername) inserts the selected folder path into the entry widget.

Main Function

- `main()`: Main function to create the GUI for filling zeros in a CSV file.
  - Creates a Tkinter window.
  - Creates labels, entry widgets, and buttons for file selection and parameter input.
  - Calls `fill_zeros` function with user-input parameters on button click.
  - Allows quitting the application.

GUI Elements

- `root.resizable(False, False)`: Makes the window non-resizable.
  - First argument is for the width.
  - Second argument is for the height.
- `filename_entry`: Entry widget to input the path of the input file.
- `folder_entry`: Entry widget to input the path of the output folder.
- `new_filename_entry`: Entry widget to input the new file name.
- `fill_value_entry`: Entry widget to input the fill value.
- `Button(frm, text="Fill Zeros", command=lambda: fill_zeros(filename_entry.get(), folder_entry.get() + "/" + new_filename_entry.get(), fill_value_entry.get()))`: Button to trigger the `fill_zeros` function with user-input parameters.
- `Button(frm, text="Quit", command=root.destroy)`: Button to close the window.