Introduction to python : Interface

Alexandre LETOIS

8 décembre 2020

1 Basics

Interface

Python have multiple libraries to create interfaces. In this lesson, we will learn the basics to create an interface with Tkinter, a standard library of Python that is usable on most OS.

Basic window

First of all, you need to create a window that will hold your interface. This is a **Tk()** object that you need to create first. Every other details of your interface will then be added one by one on this object. You can decide for a size of your interface with the function **geometry("heightxwidth")**.

```
from tkinter import *

interface = Tk()

interface.title("My first interface !")
interface.geometry("400x300")

interface.mainloop()
```

Button

To place different **widgets** on your interface, Tkinter creates an invisible grid. You can then select the column and row where you want to place your different widgets. To place the widgets, you need to use the function **grid()**. We will now create and place our first widget : a **button**.

```
button = Button(interface, text="New Button")
button.grid(column=0, row=0)
```

Label

We will place our second widget : a **label**. Labels are used to show text. You can personalize the label with different sizes, font, colors, etc. You can check the documentation for the list of all personnalizations.

```
label = Label(window, text="Our button", \
fg="red",bg="blue", font=("Helvetica",20))
label.grid(column=0, row=1)
```

Entry

The third widget is the **entry**. An entry in Tkinter is a small box where you can write a text. An entry have multiple methods, but the most useful is probably **get()** that returns the value in the entry. Entry have specific keywords like **ANCHOR**, that represents the start of the selection, **END** that represents the position after the last character of the entry and **INSERT** that represent the position where the cursor is actually. There is also a **delete(first,last=None)** function that delete the character at the "first" position or at a given range between "first" and "last". For example delete(0,END) removes everything in the entry.

```
entry = Entry(interface,width=20)
entry.grid(column=0, row=2)
```

2 More functionality

Click on a button

Now that we have seen the basic widgets, we can use some functions linked to them. First, we will see how to assignate a function with our button. For that, we use the parameter **command** when we create our button. We assign to the command, the name of the function that will be executed during the click.

```
count = 0
def click_fct():
    global count
    count += 1
    label.configure(text="Button has been \
    clicked %s times" %count)
button = Button(interface, \
text="New Button", command=click_fct)
```

Configure

You can also use the method **configure()** to configure again a widget. For example in the previous slide, the labeled is configured to be updated with a new text. Other methods exist for widgets or for specific widgets and you can check for them on the official documentation of Tkinter.

3 More widgets

Checkbutton

The **checkbutton** is a widget that can be used to create small check boxes.

```
checkbox = Checkbutton(interface, \
text='Example of a checkbutton', \
var=checkbox_state)
```

You need to create a variable associated with this checkbutton, in this example it's "checkbox_state". You can define a default value for this variable with the method **set(True)** or **set(False)**.

Radiobutton

The **radiobutton** is a widget similar to the checkbutton. One of the difference is that you can only have one option checked at a time.

```
radio_var = IntVar()
radio_var.set(1)

radio1 = Radiobutton(interface,text='First',\
variable=radio_var, value=1)
```

Spinbox

The **spinbox** is a widget typically used to give a variable int value in a set range.

```
spinbox = Spinbox(interface, from_=0, to=10)
```

Menu

You can create a **Menu** with the following commands. You can create a simple clickable menu or add a menu with a cascade and different categories within.

```
from tkinter import Menu

menu = tk.Menu(interface)
file_c = tk.Menu(menu)

menu.add_command(label='New')
menu.add_cascade(label='File', menu=file_c)
file_c.add_command(label='New File')
file_c.add_command(label='Load File')
file_c.add_command(label='Edit File')
file_c.add_command(label='Delete File')
```

4 Still more widgets

Menu

We have seen how to create a menu. We can also assign commands, just like with buttons by also using the argument **command**.

```
from tkinter import Menu

def quit_app():
    global interface
    interface.destroy()

menu = tk.Menu(interface)
file_c = Menu(menu)

menu.add_cascade(label='Quit', menu=file_c)
file_c.add_command(label='Quit', \
command=quit_app)
```

Scrolled Text

Another widget you can use is the **ScrolledText**. Like an entry, you can enter text in a scrolled text. The good point of the scrolled text is that you can decide of a fixed box size that can contain as much text as you want.

```
from tkinter import scrolledtext

s_text = scrolledtext.ScrolledText( \
interface ,width=20,height=10)
```

Message Box

You can create a **Message Box** for specific events. Like clicking on a button for example.

```
from tkinter import messagebox

def click_mess():
    messagebox.showinfo( \
    'Title of the message', 'Message itself')

button = Button(interface, text= \
"Message Button", command=click_mess)
```

Different kind of message box exists. You can use those for showing messages, errors, asking for an answer. You an see all those different message boxes in the official documentation.

5 More control on the interface

Frame

You can create a **frame** that will contain your widgets. You can then after that place your frames and all the widget within will be placed accordingly.

```
frame1 = Frame(interface)

button = Button(frame1, text="New Button", \
command=click_fct)
button.grid(column=0, row=0)
```

Padx and Pady

You can chose to use **padx** and **pady** to decide a value for the height and the width of a widget.

```
from tkinter import *

interface = Tk()

button = Button(interface, text="New Button", p
```

Sticky

You can use the parameter sticky to stick the widget to a specific cardinal. The keywords are N, NE, E, SE, S, SW,W,NW.

```
from tkinter import *

interface = Tk()

button = Button(interface, text="New Button")
button.grid(column=0, row=0, sticky=E)
```

6 Photos and drawings

Photo

You can import an image with **PhotoImage**

```
from tkinter import *

cat = PhotoImage(file="cat.png")
img_label = Label(interface, image=cat)
img_label.grid(column=0,row=0)
```

Canvas

You can make drawings in Tkinter with a **Canvas** widget. You can use different functions to draw different kind of things with **create_line**, **create_rectangle**, **create_arc**, **create_oval**, **create_polygon**.

```
canvas = Canvas(interface, width=200, \
height=100)
canvas.grid(column=0,row=0)

canvas.create_line(0, 0, 200, 100)
canvas.create_line(0, 100, 200, 0, \
fill="red", dash=(1, 10))
canvas.create_rectangle(50, 25, 150, 75, \
fill="blue")
```