# Tkinter

**Intro Tkinter**

**references:**

<http://www.pythonlake.com/python/tkinter>

<https://www.geeksforgeeks.org/python-gui-tkinter/>

**stap 1 : library importeren**

**from tkinter import \***

**stap 2 : main window definieren**

window = Tk()

window.geometry("800x400") # not \*

window.title("CVOFOCUS THERMOSTAAT")

**stap 3 : widgets definieren**

mijnLabel1= **Label**( window, text = "Level=40")

mijnKnop1 = **Button**(window , text = "LEVEL 35" , width = 15 , height = 1, command=fct\_knop1)

mijnSlider1 = **Scale**(window, from\_=20, to=50, tickinterval=10, orient=HORIZONTAL, command = fct\_scale1 )

mijnTekstvak1 = **Entry**(window, width = 40 )

**stap 4: window.mainloop() starten**

Alternatief => window.update()

**stap 5 : Events opvangen**

Events van widgets worden bij voorkeur via hun callback functies ( command= ) afgehandeld

**stap6 : Andere code uitvoeren, bijvoorbeeld checken van GPIO’s**

Via window.after(100, callback\_functienaam) in de main loop en zichzelf elke X msec blijven aanroepen in de functie die repetitief uitgevoerd moet worden.

**Default tekst en run time tekst aanpassen :**

**button**

* mijnKnop1 = **Button**(window , text = "default tekst" , …….)
* mijnknop1["text"]="tekst changed by button2"

**label**

* mijnLabel1= **Label**( window, text = " default tekst ")
* mijnLabel1["text"]="run time tekst"

**entry**

* mijnTekstvak1.insert(0,"default tekst")
* mijnTekstvak1..delete(0, "end")
* mijnTekstvak1..insert(0,"tekst changed by button3")

from tkinter import \*

window = Tk()

print("\n\*\*\*\*\*\*\*\*\*\*\*\*window.keys()\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print(window.keys())

print("\n\*\*\*\*\*\*\*\*\*\*\*\*\*my\_lable.keys()\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

my\_lable=Label()

print(my\_lable.keys())

print("\n\*\*\*\*\*\*\*\*\*my\_button.keys()\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

my\_button=Button()

print(my\_button.keys())

print("\n\*\*\*\*\*\*\*\*\*\*\*\*\*my\_entry.keys()\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

my\_entry=Entry()

print(my\_entry.keys())

print("\n\*\*\*\*\*\*\*\*\*\*\*frame1.keys()\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

frame1=Frame()

print(frame1.keys())

print("\n\*\*\*\*\*\*\*\*\*\*\*y\_canvas.keys()\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

my\_canvas=Canvas()

print(my\_canvas.keys())

**OEFENINGEN**  
  
Oef 1

Stuur met een GUI een led verbonden met GPIO 25.

Plaats een knop ON en een knop OFF op je GUI

Oef2

Stuur met een GUI een led verbonden met GPIO 25.

Plaats een knop TOGGLE op je GUI

Oef3

Herhaal oef 1 en 2 en toon in de GUI de status van een LED ( 0 of 1) in een label

Oef4

Gebruik een slider om de knippersnelheid van je LED in te stellen tussen 1 en 10 keer per seconde

Oef5

Gebruik een slider om de tijdsduur van je LED in te stellen tussen 1 en 100 seconden.

Met een knop met opschrift “LICHT” start je de led met de ingestelde tijd.

Uitbreiding : Geef feedback over de status met kleuren of met afbeeldingen ( zelf opzoeken hoe dat moet)

Oef6 schrijf de toestand aan een RP-pin 1 of 0 in een label!

Of7 Verander de kleur van een label van groen naar rood indien input van gesloten naar open verandert!

Oef8 toon een open of gesloten schakelaar in je GUI die de toestand van je schakelaar aan een pin volgt.