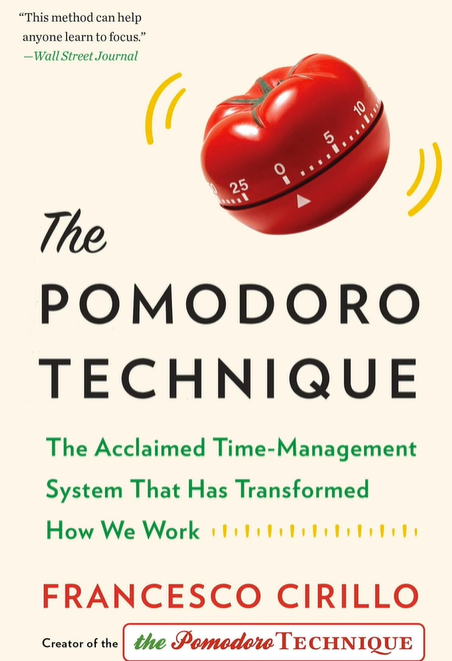
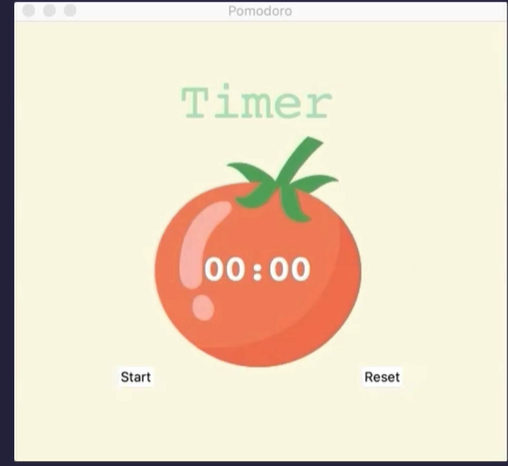
Day-28-Tkinter dynamic typing and Pomodoro App

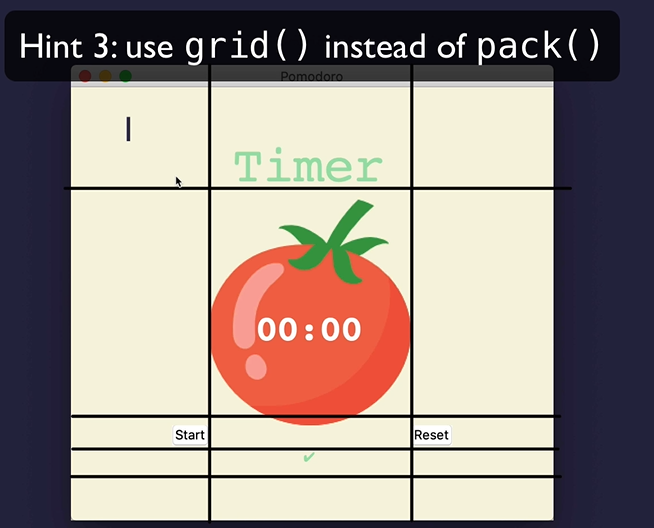








<https://colorhunt.co/>

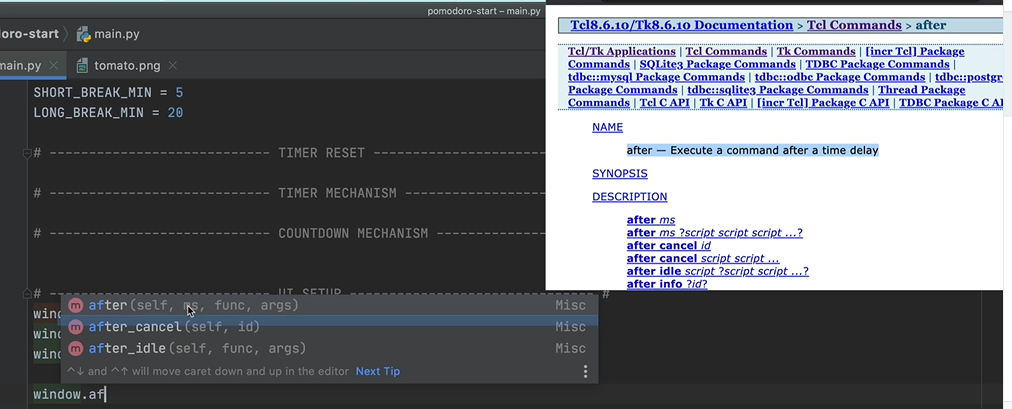


<http://tcl.tk/man/tcl8.6/TclCmd/after.htm>

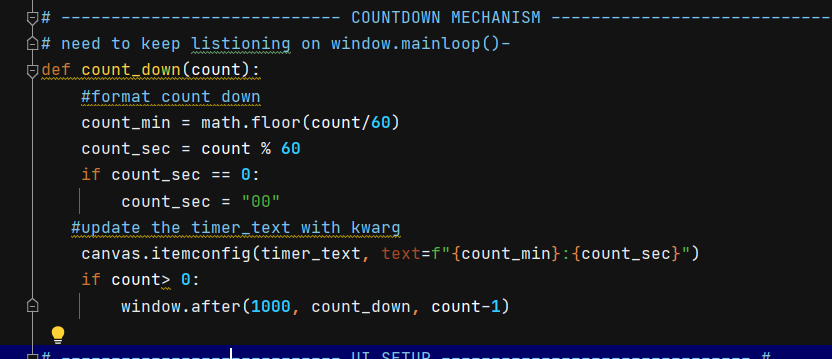
To find a check mark go to Wikipedia

<https://en.wikipedia.org/wiki/Check_mark>

Listen on the windows.mainloop() for the countdown



<https://stackoverflow.com/questions/11328920/is-python-strongly-typed>



Python is strongly, dynamically typed.

* **Strong** typing means that the type of a value doesn't change in unexpected ways. A string containing only digits doesn't magically become a number, as may happen in Perl. Every change of type requires an explicit conversion.
* **Dynamic** typing means that runtime objects (values) have a type, as opposed to static typing where variables have a type.

GUI is event driven

from tkinter import \*

import math

# ---------------------------- CONSTANTS ------------------------------- #

PINK = "#e2979c"

RED = "#e7305b"

GREEN = "#9bdeac"

YELLOW = "#f7f5dd"

FONT\_NAME = "Courier"

WORK\_MIN = 25

SHORT\_BREAK\_MIN = 5

LONG\_BREAK\_MIN = 20

reps = 0

timer = None

# ---------------------------- TIMER RESET ------------------------------- #

*def* reset\_timer():

    window.after\_cancel(timer)

    canvas.itemconfig(timer\_text, *text*="00:00")

    title\_label.config(*text*="Timer")

    check\_marks.config(*text*="")

# ---------------------------- TIMER MECHANISM ------------------------------- #

*def* start\_timer():

    global reps

    reps += 1

    work\_sec = WORK\_MIN \* 60

    short\_break\_sec = SHORT\_BREAK\_MIN \* 60

    long\_break\_sec = LONG\_BREAK\_MIN \* 60

    if reps % 8 == 0:

        count\_down(long\_break\_sec)

        title\_label.config(*text*="Break", *fg*=RED)

    elif reps % 2 == 0:

        count\_down(short\_break\_sec)

        title\_label.config(*text*="Break", *fg*=PINK)

    else:

        count\_down(work\_sec)

        title\_label.config(*text*="Work", *fg*=GREEN)

# ---------------------------- COUNTDOWN MECHANISM ------------------------------- #

*def* count\_down(*count*):

    count\_min = math.floor(count / 60)

    count\_sec = count % 60

    if count\_sec < 10:

        count\_sec = *f*"0{count\_sec}"

    canvas.itemconfig(timer\_text, *text*=*f*"{count\_min}:{count\_sec}")

    if count > 0:

        global timer

        timer = window.after(1000, count\_down, count - 1)

    else:

        start\_timer()

        marks = ""

        work\_sessions = math.floor(reps/2)

        for \_ in range(work\_sessions):

            marks += "✔"

        check\_marks.config(*text*=marks)

# ---------------------------- UI SETUP ------------------------------- #

window = Tk()

window.title("Pomodoro")

window.config(*padx*=100, *pady*=50, *bg*=YELLOW)

title\_label = Label(*text*="Timer", *fg*=GREEN, *bg*=YELLOW, *font*=(FONT\_NAME, 50))

title\_label.grid(*column*=1, *row*=0)

canvas = Canvas(*width*=200, *height*=224, *bg*=YELLOW, *highlightthickness*=0)

tomato\_img = PhotoImage(*file*="tomato.png")

canvas.create\_image(100, 112, *image*=tomato\_img)

timer\_text = canvas.create\_text(100, 130, *text*="00:00", *fill*="white", *font*=(FONT\_NAME, 35, "bold"))

canvas.grid(*column*=1, *row*=1)

start\_button = Button(*text*="Start", *highlightthickness*=0, *command*=start\_timer)

start\_button.grid(*column*=0, *row*=2)

reset\_button = Button(*text*="Reset", *highlightthickness*=0, *command*=reset\_timer)

reset\_button.grid(*column*=2, *row*=2)

check\_marks = Label(*fg*=GREEN, *bg*=YELLOW)

check\_marks.grid(*column*=1, *row*=3)

window.mainloop()