**Python - attività 4: interfaccia grafica**

*Lo scopo di questa attività è gestire tramite interfaccia grafica il sensore e l’attuatore sviluppati nella attività precedente.*

*I programmi su Arduino non cambiano, mentre le applicazioni Python saranno da modificare per gestire la grafica.*

*La GUI per l’attuatore è più semplice da realizzare, nella sua versione base, di quella del sensore.*

**Parte 1: la libreria Tkinter**

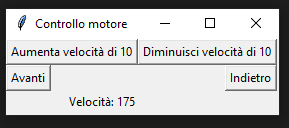
**Fase 1.1:** studiare il funzionamento dell’esempio a [questo link](http://www.youtube.com/watch?v=2vWCzB9HBgk), almeno fino al minuto 15:38.

**Parte 2: GUI per l’attuatore**

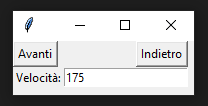
*La GUI invia all’attuatore un pacchetto contenente la velocità e la direzione, secondo il formato atteso dall’attuatore stesso (vedi esercitazione precedente). Per il posizionamento dei widgets mediante grid, fare riferimento a* [*questo link*](https://www.delftstack.com/it/tutorial/tkinter-tutorial/tkinter-geometry-managers/)*.*

*Vengono richieste più soluzioni: almeno la prima deve essere realizzata da tutti.*

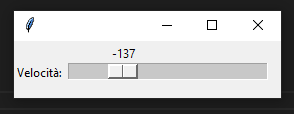
**Soluzione 1:** realizzare una GUI che permetta di modificare la velocità mediante una coppia di bottoni, uno dei quali aumenta la velocità e l’altro la diminuisce. Una semplice versione potrebbe risultare così:



**Soluzione 2:** realizzare una GUI con un campo di tipo Entry (vedi [questo link](https://www.youtube.com/watch?v=ITaDE9LLEDY)), che permetta di specificare la velocità voluta (0-255) e due bottoni per la scelta della direzione: quando si preme un bottone o quando si modifica la velocità, si invia all’attuatore la stringa con velocità e direzione. Una semplice versione potrebbe risultare così:



**Soluzione 3:** realizzare una GUI con un campo di tipo Scale (vedi [questo link](https://tkdocs.com/tutorial/morewidgets.html#scale)), che permetta di specificare la velocità voluta (da -255 a +255): quando si modifica la velocità, si invia all’attuatore la stringa con velocità e direzione. La direzione dipende dal segno della velocità. Una semplice versione potrebbe risultare così:



**Parte 3: GUI per il sensore**

*La GUI deve controllare la presenza di dati provenienti dal sensore con una frequenza superiore a quelle dell’invio da parte del sensore.*

*Per eseguire la funzione di controllo all’interno del mainloop di tkinter, con frequenza prefissata, utilizzare il metodo* **after***, come nell’esempio che segue:*

import tkinter

# funzione che viene chiamata ogni secondo

def scatta(finestra):

    s=int(secondi["text"])+1

    secondi.config(text=s)

    finestra.after(1000, scatta, finestra)    #richiamo della funzione dopo un secondo

finestra = tkinter.Tk()

finestra.title("Prova")

secondi=tkinter.Label(finestra,text="0")

secondi.grid(row=0,column=0)

finestra.after(1000, scatta, finestra) #richiamo della funzione dopo un secondo

finestra.mainloop()

**Soluzione 1:** realizzare una GUI che visualizzi nella finestra grafica il dato ricevuto dal sensore

**Soluzione 2:** realizzare una GUI che visualizzi nella finestra grafica un rettangolo la cui dimensione cambi al variare del dato ricevuto. (utilizzare il widget [*Canvas*](https://www.tutorialspoint.com/python/tk_canvas.htm)per disegnare il rettangolo)

**Soluzione 3:** realizzare una GUI che un grafico con i dati ricevuti in un certo lasso di tempo, ad esempio l’ultimo minuto. (vedi il seguente [esempio](https://www.andreaminini.com/python/matplotlib/come-creare-un-grafico-in-python) oppure [quest'altro](https://it.softpython.org/visualization/visualization1-sol.html))