PY-PROJECT

Première prise en main

1. Mise sous tension

- Brancher le dongle du clavier sur un port USB
- Si vous souhaitez utiliser l'écran du PY-PROJECT basculer le curseur sur le côté droite du coffret vers la position basse, si vous avez un écran externe, glissez le vers le haut pour désactiver l'écran tactile. Vous pouvez utiliser simultanément les deux écrans mais vous serez limité en résolution.
- Brancher l'alimentation secteur 12V-5A fournie
- Attendre le démarrage.

- *Pour éteindre PY-PROJECT il est conseillé d'arrêter proprement la machine (STUTDWON)*

2. Premier programme

Pour les premiers programmes nous pouvons utiliser les éléments qui sont déjà présents sur PY-PROJECT sans connecter des composants externes. Le bouton poussoir, le potentiomètre, le buzzer et les diodes électroluminescentes.

Premier essai : programme bip.py

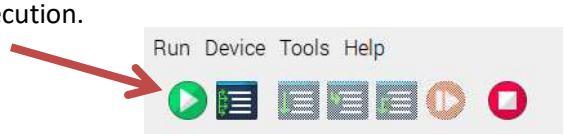
Notre premier programme très simple va lancer une commande en Python pour émettre un bip sur le buzzer.

Ouvrir le logiciel « Thonny Python »

Le programme commence par l'appel d'une petite bibliothèque qui va se charger de la communication entre le Raspberry et l'Arduino, le fichier tools.py doit toujours être dans le même dossier que votre programme. La fonction init() qui suit initialise la communication.

Les programmes devront toujours comporter ces deux lignes.

Le programme en lui-même est composé d'une seule ligne, c'est une commande en sortie qui a pour cible une fonction bip dans l'Arduino. Il suffit de lancer l'exécution.



Le programme émet un seul bip, c'est ce qu'on lui a demandé.

Vous pouvez aussi commander le buzzer en mode ON/OFF avec la cible « buzzer ».

```
from tools import *
import time
init()

sortie('{"cible":"buzzer","etat":"on"}')
time.sleep(2)
sortie('{"cible":"buzzer","etat":"off"}')
```

3. Commande des leds

Trois leds sont intégrées sur la face avant de PY-PROJECT au démarrage la diode verte est allumée. Il y a également deux autres diodes rouge et bleue. Ce programme va allumer les trois leds.

```
from tools import *
import time
init()

sortie('{"cible":"led","couleur":"rouge","etat":"on"}')
sortie('{"cible":"led","couleur":"verte","etat":"on"}')
sortie('{"cible":"led","couleur":"bleue","etat":"on"}')
```

Vous pouvez aussi piloter directement les sorties avec une commande « output » pour les sorties logiques (TOR)

La LED rouge est sur la sortie 25 de l'Arduino
La LED verte est sur la sortie 39 de l'Arduino
La LED bleue est sur la sortie 27 de l'Arduino

```
from tools import *
init()

sortie('{"cible":"output","port":27,"etat":"on"}')
```

Notez ici que la valeur numérique 27 n'est pas encadrée par des guillemets comme le sont les chaines de caractères.

4. Lecture d'une entrée

La face avant de PY-PROJECT dispose d'un bouton poussoir et d'un potentiomètre. Le bouton poussoir bp est sur l'entrée logique 65 et le potentiomètre sur l'entrée analogique A0.

La lecture de l'état de l'entrée du bouton poussoir se fait par une fonction lecture, une boucle sans fin permet d'afficher le résultat, 0 ou 1, en continu. Il est affiché dans la fenêtre du Shell à intervalle de 0.2 seconde.

Un programme similaire permet de lire l'état de l'entrée analogique du potentiomètre.

Pour afficher la tension mesurée.

```
print("tension : " + str(int(100*entree('{"cible":"CAN","port":0}')*5/1023)/100) + " volts")
```

Affichage : `tension : 3.65 volts`

5. Interaction entrée / sortie

- Le bouton poussoir relié au buzzer.

Si on appuie sur le bouton poussoir
le buzzer émet un son
sinon
le buzzer n'émet pas de son.

```
from tools import *
init()

while(True):
    if (entree('{"cible":"bp"}') == 1):
        sortie('{"cible":"buzzer","etat":"on"}')
    else:
        sortie('{"cible":"buzzer","etat":"off"}')
```

- Avec la même simplicité on peut lire les entrées en provenance des capteurs sur les ports externes et piloter les sorties telles que les moteurs.

  Le programme ci-dessous commande la rotation du moteur à courant continu en fonction de la consigne lue sur l'entrée analogique externe nommée CAN4 (sur le port A9 entrée 63 de l'Arduino).

  Un potentiomètre est connecté sur l'entrée CAN4, un moteur à courant continu est sur la sortie de puissance à transistor MOS (0 à 12V)

```
from tools import *
import time
init()

while (True):
    consigne = int(entree('{"cible":"CAN","port":9}')/4)
    sortie('{"cible":"MOS","PWM": '+ str(consigne) + '}')
    time.sleep(0.2)
```

Vidéo de démo