## **Raspberry PI PICO**



Nous allons utiliser la Raspberry PI PICO (appeler PICO par la suite) pour faire notre projet, avec ce document nous allons voir les bases pour l'utiliser.

 Nous utiliserons un IDE (Integrated Developpement Environment ou environnement de développement intégrer (l'endroit où on mettra notre code)), <u>Thonny</u> téléchargeable sur <u>https://thonny.org/</u>

Choisir votre distribution, (Linux dispo dans une autre version) ex : Windows



- a. Première chose à faire l'installer sur votre PC il y a une version Mac, Linux et Windows
- b. Deuxième lancer Thonny et le paramétrer pour faire fonctionner la PICO

Français ou à votre convenance	Language:	Français	~
Et bien mettre standard	Initial settings:	Standard	~
			Let's go!

c. Se familiariser avec les mots clés (voir annexe)

Cette fenêtre devrait apparaitre :

Thonny - <untitled> @ 1:1</untitled>	-		×
Fichier Édition Affichage Exécuter Outils Aide			
🕂 🖞 🖞 🖸 🔲 🔲 🔲 🔘 🔘			
<untitled>  X Variables  X</untitled>			
1 Nom		Valeur	
Console X			
Python 3.7.9 (bundled)			- 11
>>> 			~
		Pythor	n 3.7.9

On va ensuite sélectionner l'interpréteur :

	-		$\times$
	1		
F5 Ctrl+F5 Shift+F5 Ctrl+Shift+B	×	Valeur	Â
F6 F7			
F8 Ctrl+F8 Ctrl+B			
Ctrl+T	1		- 1
Ctrl+F2 Ctrl+C Ctrl+D			
	1		
	Ţ	×	Ç. Pytho

🔣 Opti	ons de Tho	nny					×
Géné	Interprét	Édite	Thème et po	Exécuter et débi	Termiı	Const	Assista
Que	l interpréteu roPvthon (f	ur ou ap Raspber	pareil Thonny rv Pi Pico)	doit-il utiliser pour	r exécute	r votre c	ode ?
Déta	iils						
Connectez votre appareil à l'ordinateur et sélectionnez le port correspond: (recherche du nom de votre appareil, « USB Serial » ou « UART »). Si vous ne le trouvez pas, vous devriez d'abord installer un pilote USB corre							
Port							
	coody cr uc	uttette	. ie pore datom	induction .			
				<u>Installer ou mett</u>	re à jour	le firmw	are
					ОК	4	Annuler

On met l'interpretteur MicroPython (Raspbery PI Pico), on peut laisser le port en automatique, on valide avec OK.

Si ultérieurement, il y a des problèmes et que le port n'est pas détecté automatiquement, il faudra revenir ici pour le sélectionner.

Un message dans la console indique que le composant n'est pas trouvé (device), c'est normal, il faut relancer *Thonny*. Ensuite, on peut brancher et débrancher la PI PICO quand on le souhaite, pour faire les branchements des composants par exemple.

Les différents codes qui seront utilisés par la suite sont disponibles en ligne. À vous de les télécharger si vous le souhaitez, sinon il est possible de les réécrire, à vous de choisir (préconisation : réécrire les choses importantes C-a-d faire du copier/coller de son propre code).

2. Voici les différentes PIN de notre PICO :



3. Faire clignoter la LED (GP25) présente sur la PICO.

Pour cela on doit écrire le code suivant :

1 2	<pre>import machine import utime</pre>
3 4	<pre>led = machine.Pin(25, machine.Pin.OUT)</pre>
5 6 7	while True :
8	<pre>led.value(1) utime.sleep(2)</pre>
10	led.value(0)
11	utime.sleep(2)
12	

Et on la renomme led.PY : On n'oublie pas de valider avec OK.

Raspberry Pi Pic	o		= 🗍
Nom		Taille (o	ctets)
			- 1
			- 1
			- 1
			- 1
			- 1

Ensuite on l'enregistre sur la PICO :

chier Édition Affichage Exe	écuter Outils	The Where to save to?
Nouveau	Ctrl+N	
Ouvrir	Ctrl+O	
Fichiers récents		
Fermer	Ctrl+W	Cet ordinateur
Tout fermer	Ctrl+Shift+\	Cet ordinatedi
Enregistrer	Ctrl+S	
Sauvegarder Tous les fichiers	Ctrl+Alt+S	
Enregistrer sous	Ctrl+Shift+S	
Enregistrer une copie		
Move / rename		Raspberry Pi Pico
Imprimer	Ctrl+P	
Quitter	Alt+F4	

Il n'y a plus qu'à appuyer sur la flèche verte pour faire fonctionner le programme. Pour le stopper on utilise le carré rouge



Il est possible qu'une fenêtre d'erreur apparaisse, si c'est le cas il faut simplement appuyer sur installer et attendre que la barre de chargement se termine pour finir appuyer sur fermer.

The Install MicroPython firmware for Raspberry Pi Pico $ imes$	$\mathbf{\tilde{k}}$ Install MicroPython firmware for Raspberry Pi Pico $\qquad \times$
Here you can install or update MicroPython firmware on Raspberry Pi Pico. 1. Plug in your Pico while holding the BOOTSEL button. 2. Wait until device information appears. 3. Click 'Install'. When the process finishes, your Pico will be running the latest version of MicroPython. Close the dialog and start programming!	Here you can install or update MicroPython firmware on Raspberry Pi Pico. 1. Plug in your Pico while holding the BOOTSEL button. 2. Wait until device information appears. 3. Click 'Install'. When the process finishes, your Pico will be running the latest version of MicroPython. Close the dialog and start programming!
Version à installer: v1.18 (2022-01-17) Emplacement du : E:\ Modèle de l'appareil cible: Raspberry Pi RP2 Installer Annuler	Version à installer:       v1.18 (2022-01-17)         Emplacement du :       E:\         Modèle de l'appareil cible:       Raspberry Pi RP2         Done!       Installer

Cette étape ne devrait arriver qu'une seule fois, c'est pour « initialiser » la carte et faire en sorte qu'elle puisse communiquer avec THONNY

Quelques explications : import machine import utime	# On ajoute les bibliothèques machine et utime
<pre>led = machine.Pin(25, machine.Pin.OUT)</pre>	# On initialise la variable led pour dire que c'est une # sortie sur la PIN 25 (sur le schéma précédent le # numéro des pins correspond au nombre après le GP
while True :	# La boucle infinie, le corps du programme
led.value(1) utime.sleep(2) led.value(0) utime.sleep(2)	# On allume la led # On attend 2sec # On éteint la led # on attend 2 sec

Derrière les # on peut mettre du commentaire qui ne sera pas regardé par le programme c'est pour que nous humain nous comprenions et donnions des informations aux autres humains qui lisent le code

4. Le micro-rupteur

Branchements :





1	<pre>import machine</pre>	# On ajoute les bibliothèques
2		# machine et utime
3	import utime	
4 5	<pre>microRupteur = machine.Pin(16, machine.Pin.IN)</pre>	# On initialise la variable led
6		<pre># pour dire que c'est une</pre>
7		# entrée sur la PIN 16
8	while Taur	
9	while frue:	
10	utime.sleep_ms(200)	# On attend 200ms
11	<pre>print(microRupteur.value())</pre>	# On affiche la valeur lue
12		<pre># sur microRupteur</pre>
13		

Objectif voir dans la console ce qui se passe quand on appuis sur le « bouton » 0 ou 1 apparait.

Petit exercice :

Allumer la led quand vous appuyez sur le micro-rupteur. (Réponse à télécharger)

5. Le Servo moteur

Branchements :



Programme :

1	import machine	
2	import utime	
5 4 5 6	stop = 1500000 back = 1000000 go = 2000000	# valeur permettant de stopper le servo moteur # valeur permettant de faire tourner le servo moteur dans un sens # valeur permettant de faire tourner le servo moteur dans l'autre sens
7 <b>8</b> 9	<pre>servo = machine.PWM(machine.Pin(28))</pre>	# on déclare le servo moteur comme PWM brancher sur la PIN 1
10	servo.freq(50)	# on donne la fréquence (50 par défaut) # on appête le convergetour
12	servo.ducy_ns(scop)	# ON ANTELE IE SELVO MOLEUN
13	while True :	
14	<pre>servo.duty_ns(stop)</pre>	# on arrête le servo moteur
15	utime.sleep(1)	# on attend 1 sec (servo arrêté 1 sec)
16	<pre>servo.duty_ns(back)</pre>	# on arrête le servo moteur
17	utime.sleep(2)	# on attend 2 sec (servo tourne dans un sens 2 sec)
18	<pre>servo.duty_ns(stop)</pre>	# on arrête le servo moteur
19	utime.sleep(1)	# on attend 1 sec (servo arrêté 1 sec)
20	servo.duty_ns(go)	# on arrête le servo moteur
21	utime.sleep(2)	# on attend 2 sec (servo tourne dans l'autre sens 2 sec)

Actuellement il fait des allers-retours avec une pause, il y a la possibilité de changer les valeurs.

Petit exercice :

Avec 2 boutons, si j'appuie sur un bouton je tourne dans un sens, si j'appuie sur l'autre, dans l'autre sens sinon si j'appuie sur les deux OU sur aucun je m'arrête. Possibilité de s'amuser avec « and » et « or ».

6. Ultrason

Petite subtilité il faut aller télécharger la bibliothèque <u>HCSR04</u> sur le lien : (aussi dispo dans les documents fournis)

https://github.com/rsc1975/micropython-hcsr04

Il faut ensuite l'ouvrir avec *Thonny* Faire « enregistrer sous » et l'enregistrer sur la PICO attention à bien respecter le nom : hcsr04.py

Ensuite comme on commence à avoir l'habitude

Branchement :



Programme :

```
1
   from hcsr04 import HCSR04  # https://github.com/rsc1975/micropython-hcsr04
 2
   import utime
 4
   #definition ultrason attention brancher sur 5V le VCC
5 sensor = HCSR04(trigger_pin=27, echo_pin=26) # On déclare le capteur sur la PIN 27 Pour le TRIGGER
                                                # et sur la PIN 26 pour le ECHO
6
8 #boucle de fonctionnement
9
10 while True:
11
12
       utime.sleep_ms(200)
                                                 # on attend 20ms
                                                 # on récupère la valeur dans la variable distance
13
       distance = sensor.distance_cm()
14
       print(distance)
                                                 # on affiche la distance dans la console
15
```

Petit exercice :

Allumer 1,2,3 LEDs en fonction de la distance de détection 100cm, 50cm, 25 cm par exemple

Infrarouge

2 étapes :

- Allumer la LED IR voir 3
- Détecter la présence d'infrarouge (même principe qu'un bouton)

Attention, la diode IR fonctionne avec une tension de 1.5V, 20mA et notre PICO fourni 3.3V il faut donc rajouter une résistance 3.3-1.5 = 1.8V R=U/I donc R= 1.8/0.02 = 90 ohms on utilisera une résistance de 1000hms.

En réalité pour la LED IR on n'est pas obligé de la commander, elle peut rester allumer continuellement.

On vous laisse faire, avec ce qu'on a vu précédemment vous devriez y arriver.

## Annexes

Nous avons réellement besoin des conditions (if),

des boucles (while) puisqu'on l'utilise dans tous nos programme éventuellement « for » pour certain d'entre vous.

Et des fonctions (même si non obligatoire mais bien plus pratique pour les feignants, il faut être un feignant intelligent)

Les conditions :

```
Le si ... sinon si ... sinon ...
Le principe : si j'ai faim
je mange
sinon si j'ai envie d'un bonbon
je mange un bonbon,
sinon
je ne fais rien
```

Il est possible de ne mettre que le si, ex : if a==b :

print(a égale b) donc tant que ce n'est pas vrai il n'y a rien d'affiché

```
on peut mettre si et sinon ex :

if a==b :

print(« a égale b »)

else :

print(« a différend de b »)

ou la totale
```

etc. par contre il faut forcément que la première comparaison soit un « si » ou if .

Les opérateurs de comparaison possible sont :

x == y x est égal à y
x != y x est différent de y
x > y x est plus grand que y
x < y x est plus petit que y</li>
x >= y x est plus grand ou égal à y
x <= y x est plus petit ou égal à y</li>

Le cours :

https://courspython.com/tests.html

Pour la suite autant utiliser les cours disponibles sur les liens suivants

Les boucles :

Permettant de répéter une opération sans réécrire X fois le code

https://courspython.com/boucles.html

Les fonctions :

Permettant de faire une action spécifique à plusieurs endroit dans le code et en ne l'écrivant qu'une fois (c'est du bonus)

Dispo ici :

https://courspython.com/fonctions.html

Toutes les autres bases disponibles <u>ici</u>: (pas forcément utile pour ce qu'on a à faire, ça peut toujours être utile.

https://courspython.com/bases-python.html