TP prise en main de la carte Adafruit Playground Bluefruit:

Qu'est-ce que le circuit Playground Bluefruit :

Il s'agit d'une carte équipée d'un processeur (ici Cortex M4 nRF52840 pour ceux qui aiment les chiffres) avec support bluetooth, d'une mémoire flash de 2 Mo (qui nous servira à stocker nos programmes notamment), et surtout d'une série d'entrées et sorties :

• Outputs:

- o 10 LED NeoPixels pouvant afficher n'importe quelle couleur ;
- o 1 LED rouge D13;
- o 1 LED verte « ON »;
- o 1 mini haut-parleur (7,5 mm) avec amplificateur de classe D :
- 2 14 Pads tout autour permettant différent type d'utilisation (vis, pince crocodile, soudure,...);



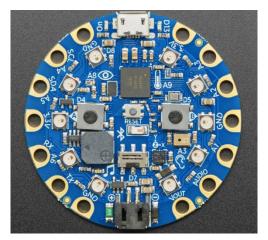
- 1 capteur de luminosité (A8) : Il mesure l'intensité de la lumière ambiante (valeur entre 0 et 1023) ;
- o 1 capteur de température (A9) : Il s'agit s'une thermistance (résistance dont la résistance varie en fonction de la température) ;
- o 1 micro en A3;
- 1 capteur de mouvement accéléromètre : LIS3DH à 3-axes XYZ. Il peut permettre de détecter le mouvement, une chute, une secousse... ;
- o 7 touches capacitives: A1 à A6 et TX sensibles au toucher;
- o 1 entrée pour une batterie avec prise JST;
- o 2 boutons A et B en D4 et D5;
- o 1 bouton reset pour le CPU;
- o 1 switch en D7. Il peut être assigné dans vos programmes.

Les pads:

- Power pads:
 - o GND: Il y en a 3. On les utilise pour la masse;
 - o 3,3 V: Il y a 2 sorties à 3,3V (500 mA max, mais plutôt 250 mA en utilisation);
 - VOut: permet une alimentation directement depuis l'USB ou la batterie sans passer par le régulateur)
- Input/output pads:

Les différents pads peuvent être utilisé en entrées/sorties digitales, pour des LEDs, des boutons. De plus les pads A1 à A6 peuvent être utilisés comme des entrées analogiques.

Nous avons donc de quoi programmer grâce à cet ensemble de capteurs.



L'objectif de ce TP est de faire votre première programmation de la carte Bluefruit.

Nous utiliserons le logiciel **Thonny** pour taper notre code.

Branchez votre carte via le port USB de l'ordinateur.

Pour configurer Thonny:

Dans Outils puis options, choisissez interpréteur. Sélectionnez l'interpréteur MicroPython générique puis sélectionnez le Circuit Playground.

Après avoir tapé votre code sur Thonny, vous allez le sauvegarder en le nommant obligatoirement **code.py** sur la mémoire flash de la carte CIRCUITPY.

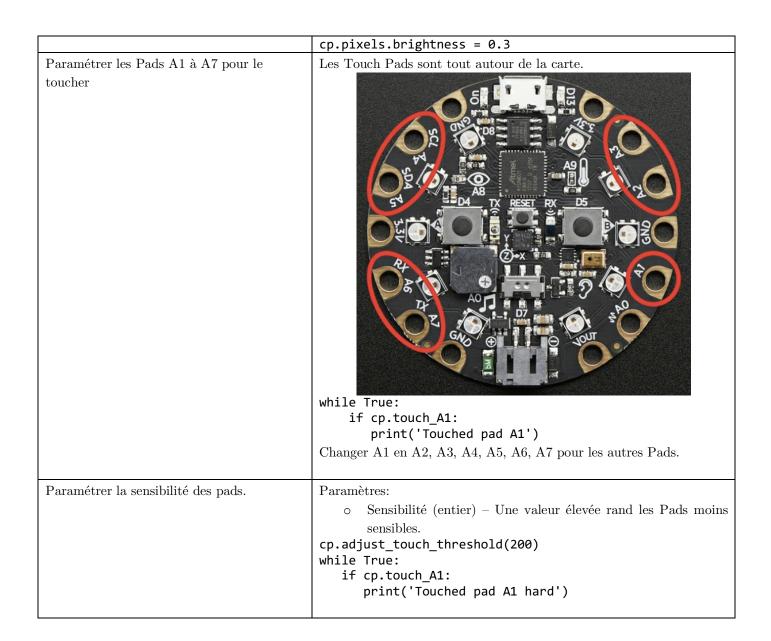
Ne débranchez pas la carte pour tester le code ;), il faut l'alimenter...

Voici quelques exemples de codes pour cette carte que vous allez tester.

Pensez à importer la bibliothèque au début de votre code: from adafruit_circuitplayground import cp

Action	CP library usage
Accéléromètre	<pre>while True: x, y, z = cp.acceleration print((x, y, z))</pre>
Bouton A / B	<pre>while True: if cp.button_a: print("Bouton A pressé!") if cp.button_b: print("Bouton B pressé!")</pre>
Slide Switch	print("Slide switch:", cp.switch) (True à gauche, False à droite)
LED rouge D13	<pre>import time while True: cp.red_led = True time.sleep(1) cp.red_led = False time.sleep(1) (True allume la LED D13, False l'éteint)</pre>
Capteur de température	<pre>temperature_c = cp.temperature temperature_f = temperature_c * 1.8 + 32 print("Temperature celsius:", temperature_c) print("Temperature fahrenheit:", temperature_f) ou plus élaboré: from adafruit_circuitplayground import cp import time while True: temperature_c = cp.temperature temperature_f = temperature_c * 1.8 + 32 print("Temperature celsius:", temperature_c) print("Temperature fahrenheit:", temperature_f) time.sleep(1)</pre>
Capteur de lumière	<pre>import time while True: print("luminosité:", cp.light) time.sleep(1)</pre>
Jouer un fichier WAV (son)	cp.play_file("son.wav")

	Le fichier WAV doit être stocké sur la mémoire flash !
Jouer un son sur le haut-parleur avec une durée fixée Jouer un son sur le hp jusqu'à ce que vous	Paramètres: o fréquence (entier) - Fréquence du son en Hz o durée (décimal) - Durée du son en secondes cp.play_tone(440, 1.0) # 440 hz (La) pour 1 seconde Paramètre:
demandiez de l'arrêter (voir en-dessous)	o fréquence (entier) - Fréquence du son en Hz cp.start_tone(262)
Arrêter un son commencer avec start_tone	<pre>cp.stop_tone()</pre>
Détecter quand la plaque est tapée 1 fois	<pre>cp.detect_taps = 1 if cp.tapped: print("touché 1 fois détecté!")</pre>
Détecter quand la plaque est tapée 2 fois	while True: if cp.tapped: print("Touché 2 fois détecté!")
Détecter quand la plaque est remuée	Paramètre: o Seuil de la secousse (entier) — le seuil doit être franchi pour retourner la valeur True (par défaut: 30). Moins= plus sensible. Garder supérieur à 10. while True: if cp.shake(50): print("Grosse secousse détéctée!")
Paramétrer NeoPixel LEDs	Paramétrer la couleurs des NeoPixels en RGB de 0 à 255. while True: cp.pixels[9] = (30, 0, 0) Sélection de plusieurs LEDs: cp.pixels[0:5] = [(30, 0, 0)]*5 #selectionne les leds 0, 1, 2, 3 et 4. Ne pas oublier le *5.
Paramétrer NeoPixel LEDs (en continue)	Les Leds sont numérotées dans le sens anti-horaire de cpx.pixels[0] à cpx.pixels[9]. Toutes les Leds peuvent être paramétrées à la même couleur avec cp.pixels.fill: while True: cp.pixels.fill((30, 0, 0)) Pour éteindre toutes les Leds les régler à (0, 0, 0):
Paramétrer la luminosité des NeoPixels	cpx.pixels.fill((0, 0, 0)) Régler la luminotité de toutes les Leds (de 0.0 à 1.0):



Un exemple de programme mêlant plusieurs éléments :

```
from adafruit_circuitplayground import cp
cp.pixels.brightness = 0.3
cp.pixels.fill((0, 0, 0))  # Turn off the NeoPixels if they're on!

while True:
    if cp.button_a:
        cp.pixels[0:5] = [(255, 0, 0)] * 5
    else:
        cp.pixels[0:5] = [(0, 0, 0)] * 5

if cp.button_b:
        cp.pixels[5:10] = [(0, 255, 0)] * 5

else:
    cp.pixels[5:10] = [(0, 0, 0)] * 5
```

Essayer de faire de même avec d'autres capteurs...