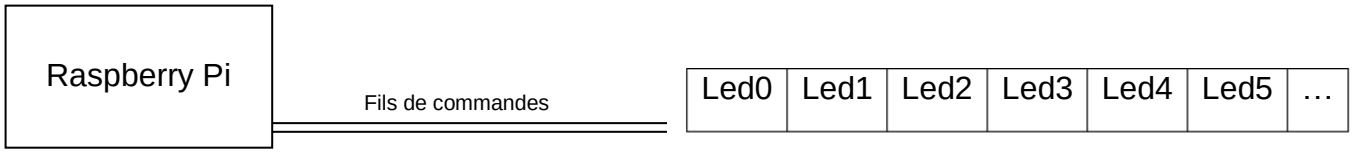
Exercice 5

Bandeau à LED

Thème abordé : l'objectif de cet exercice est de commander un bandeau de diodes électroluminescentes (LED) à l'aide d'un nano-ordinateur Raspberry Pi en langage Python.

Chacune des LEDs du bandeau pourra être commandée individuellement pour l'allumer avec une couleur choisie. Une LED est considérée comme éteinte lorsque la couleur est noire (pas de lumière), allumée pour toutes les autres couleurs.

Pour cela, nous allons utiliser un bandeau comportant au maximum 32 LEDs branché sur un nano-ordinateur Raspberry Pi sur lequel est installé le module (librairie) Adafruit_WS2801, à importer dans le script Python.



Après avoir lu la documentation donnée en annexe 2, répondre aux questions suivantes.

1. Compréhension des méthodes avec un bandeau de 8 LEDs

Un script a été exécuté au préalable et le bandeau de LEDs est dans l'état suivant : toutes les LEDs sont éteintes sauf trois

La LED « LED0 » est rouge

La LED « LED1 » est bleue

La LED « LED5 » est verte

ROUGE	BLEU				VERT		

L'objet bandeau est créé au début de notre script dans la variable `Obj_bandeau`

1.a. **Que va retourner** l'instruction : `Obj_bandeau.get_pixel_rgb(1)` ?

1.b. **Que va retourner** l'instruction `Adafruit_WS2801.RGB_to_color(0,0,255)` ?

1.c. On exécute les instructions suivantes :

```
coul = Obj_bandeau.get_pixel_rgb(0)
print(Adafruit_WS2801.RGB_to_color(coul[0],coul[1],coul[2]))
```

Expliquer brièvement le rôle de chacune des deux lignes.

2. On dispose désormais d'un bandeau de 15 LEDs

Pour chacun des 2 scripts suivants, écrits en langage Python, **recopier et compléter** les tableaux correspondants au bandeau dans son état final après exécution des différents scripts.

Exemple : Pour un bandeau avec les trois premières LEDs rouges, les deux suivantes blanches et les quatre dernières bleues, on obtient le tableau donné ci-dessous :

ROUGE	ROUGE	ROUGE	BLANC	BLANC								BLEU	BLEU	BLEU	BLEU

2.a. En vous inspirant de l'exemple ci-dessus, **dessiner** le tableau correspondant au bandeau dans son état final après exécution du script suivant:

```
import RPi.GPIO as GPIO
import Adafruit_WS2801
import Adafruit_GPIO.SPI as SPI

Obj_bandeau=Adafruit_WS2801.WS2801Pixels(15,spi=SPI.SpiDev(0,0),gpio=GPI
O)
Obj_bandeau.clear()

for i in range(5):
    Obj_bandeau.set_pixel(i,16711680)
for i in range(5,10):
    Obj_bandeau.set_pixel(i,16777215)
for i in range(10,15):
    Obj_bandeau.set_pixel(i,255)
Obj_bandeau.show()
```

2.b. **Dessiner** le tableau correspondant au bandeau dans son état final après exécution du script ci-dessous:

```
import RPi.GPIO as GPIO
import Adafruit_WS2801
import Adafruit_GPIO.SPI as SPI

Obj_bandeau=Adafruit_WS2801.WS2801Pixels(15,
spi=SPI.SpiDev(0,0),gpio=GPIO)
Obj_bandeau.clear()
for i in range(15):
    if i%3 == 0:
        Obj_bandeau.set_pixel(i,32768)
    else:
        Obj_bandeau.set_pixel(i,65535)
Obj_bandeau.show()
```

3. Utilisation de classe.

On souhaite ajouter quelques fonctions supplémentaires, pour cela on crée une classe `Bandeau`. Le script est donnée ci-dessous :

```
1  import RPi.GPIO as GPIO
2  import Adafruit_WS2801
3  import Adafruit_GPIO.SPI as SPI
4
5  class Bandeau:
6          """ Classe définissant un bandeau  """
7
8     def __init__(self,Pixel_COUNT):
9        """ Commentaire n°1 """
10       self.__nbpixels=Pixel_COUNT
11       self.__tous_pixels=Adafruit_WS2801.WS2801Pixels(Pixel_COUNT,
                                                    spi=SPI.SpiDev(0, 0), gpio=GPIO)
12
13    def estallume(self):
14       """ Renvoie True si au moins un pixel est allumé de l objet bandeau
15           Renvoie False sinon """
16       for i in range(self.__nbpixels):
17          r, g, b =self.__tous_pixels.get_pixel_rgb(i)
18          if (r+g+b>0):
19             return True
20       return False
21
22    def setpixel(self,i,c):
23       """ Affecte au pixel i la couleur c et affiche directement la modification sur
le
24           bandeau de leds        i est un entier compris entre 0 et Pixel_COUNT
25                                  c est un entier représentant une couleur
num_color"""
26       self.__tous_pixels.set_pixel(i,c)
27       self.__tous_pixels.show()
28
29    def effacetout(self):
30       """ Eteint tous les pixels du bandeau et l'affiche directement sur le bandeau
"""
31       self.__tous_pixels.clear()
32       self.__tous_pixels.show()
33
34 # Création de l'objet mon_bandeau avec 8 leds
35 mon_bandeau=Bandeau(8)
36 # Eteint tous les pixels du bandeau et l'affiche directement sur le bandeau de leds
37 mon_bandeau.effacetout()
38 # Commentaire n°2
39 mon_bandeau.setpixel(6,16711680)
40 mon_bandeau.setpixel(7,16711680)
41 # Affiche un message si au moins un pixel de mon_bandeau est allumé
42 if mon_bandeau.estallume():
43         print("Il y au moins une led d'allumée")
```

3.a. On rappelle que la documentation (ou docstring) d'une fonction ou d'une méthode consiste à expliquer ce qu'elle fait, ce qu'elle prend en paramètre et ce qu'elle renvoie. **Proposer** une documentation à écrire à la ligne 9.

3.b. **Proposer** également un commentaire à écrire à la ligne 38.

Annexe 2 (exercice 5)
(à ne pas rendre avec la copie)

Documentation de la librairie Adafruit_ws2801

1) Méthodes de la librairie Adafruit_WS2801 :

`WS2801Pixels(nb_px, spi, gpio)` : création d'un objet représentant le bandeau de LEDs.

- `nb_px` : nombre de LEDs du bandeau
- `spi` (Serial Peripheral Interface) : interface du micro-processeur du Raspberry.
- `gpio` (General Purpose Input/Output) : gestion des ports entrée sortie du Raspberry.

`RGB_to_color(r , g , b)` : renvoie un entier (`num_color`) correspondant à la couleur RGB. (Voir tableau des correspondances des couleurs ci-dessous)

`Adafruit_WS2801.WS2801Pixels(32,spi=SPI.SpiDev(0,0), gpio=GPIO)` : crée et renvoie un objet représentant le bandeau de 32 LEDs dans un script en python.

2) Methodes sur un objet `Obj_bandeau` **de type** `Adafruit_WS2801.WS2801Pixels`

`count()` : Retourne le nombre de LEDs de `Obj_bandeau`

`show()` : Affiche l'ensemble des modifications effectuées sur `Obj_bandeau` au bandeau.

A noter que `Obj_bandeau` représente **indirectement** les LEDs du bandeau de LEDs, c'est-à-dire que l'on modifie d'abord `Obj_bandeau` (couleur des pixels, effacement…) puis on applique les modifications sur le bandeau de LEDs avec la méthode `show()`.

`clear()` : Eteint toutes les LEDs de `Obj_bandeau`.

Remarque : `clear()` n'éteint pas directement les LEDs du bandeau de LEDs.

`set_pixel(i, num_color)` : Attribue à la LED n°`i` de l'`Obj_bandeau` la couleur `num_color`.

`i` : entier compris entre `0` et `count()-1`.

`num_color`: entier correspondant à la couleur obtenue avec la méthode `Adafruit_WS2801.RGB_to_color(r,g,b)`

Remarque : les modifications de `set_pixel` ne seront affichées sur le bandeau de LEDs qu'à la suite d'une instruction `show()`.

`get_pixel_rgb(i)` : Retourne un tuple de trois entiers correspondant à la couleur RGB de la LED n°`i` de `Obj_bandeau`.

Remarque : pour une LED éteinte, `get_pixel_rgb` retourne (0,0,0).

Tableau de correspondance des couleurs :

Couleur	Rouge	Bleu	Vert	Jaune	Orange	Noir	Blanc
RGB	255,0,0	0,0,255	0,128,0	255,255,0	255,65,0	0,0,0	255,255,255
`num_color`	255	16711680	32768	65535	16895	0	16777215