

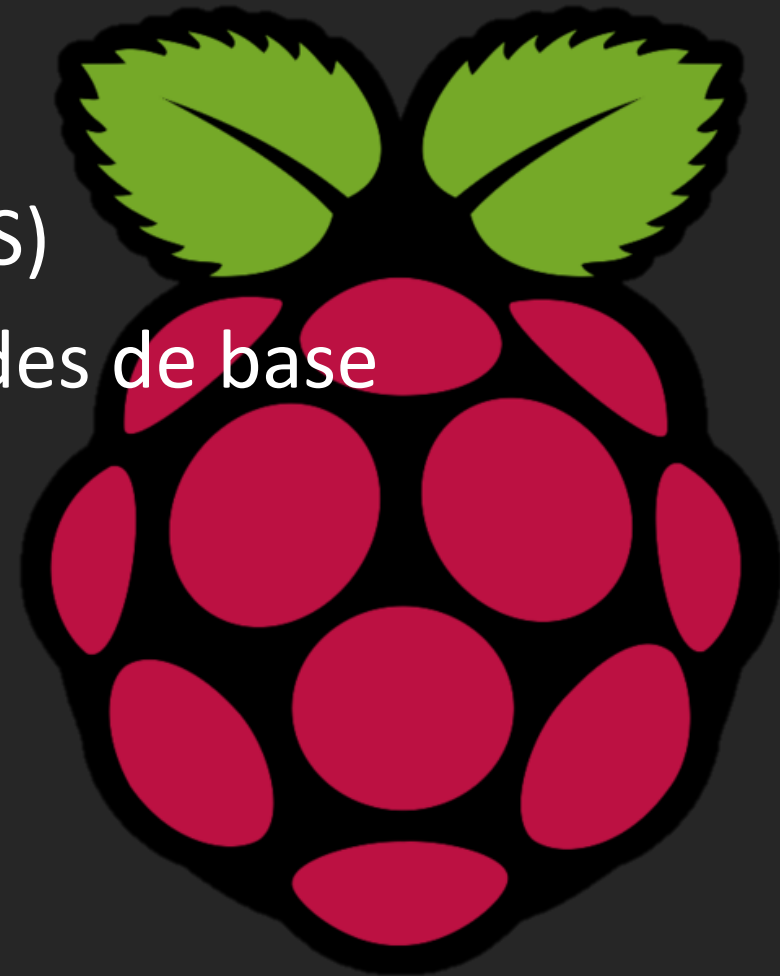
créa*ct*ifs!

Raspberry Pi

Introduction

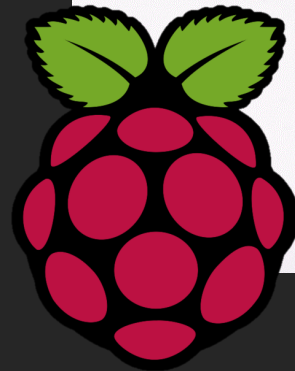
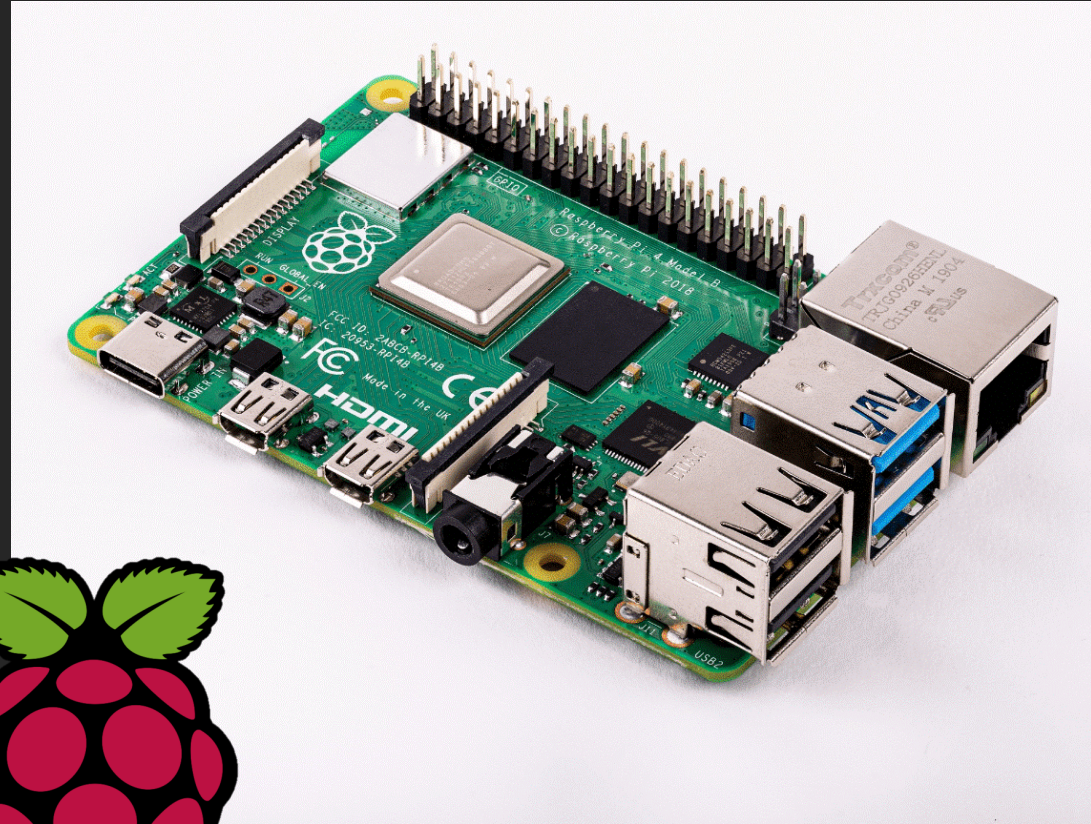
Plan de la séance 8

- Présentation de la carte
- Installation de l'OS (Raspberry Pi OS)
- Utilisation de SSH/SCP et commandes de base
 - Installation d'une clef public
 - Mise à jour de la carte
 - Activation de VNC
- Manipulation des broches !
- Blink



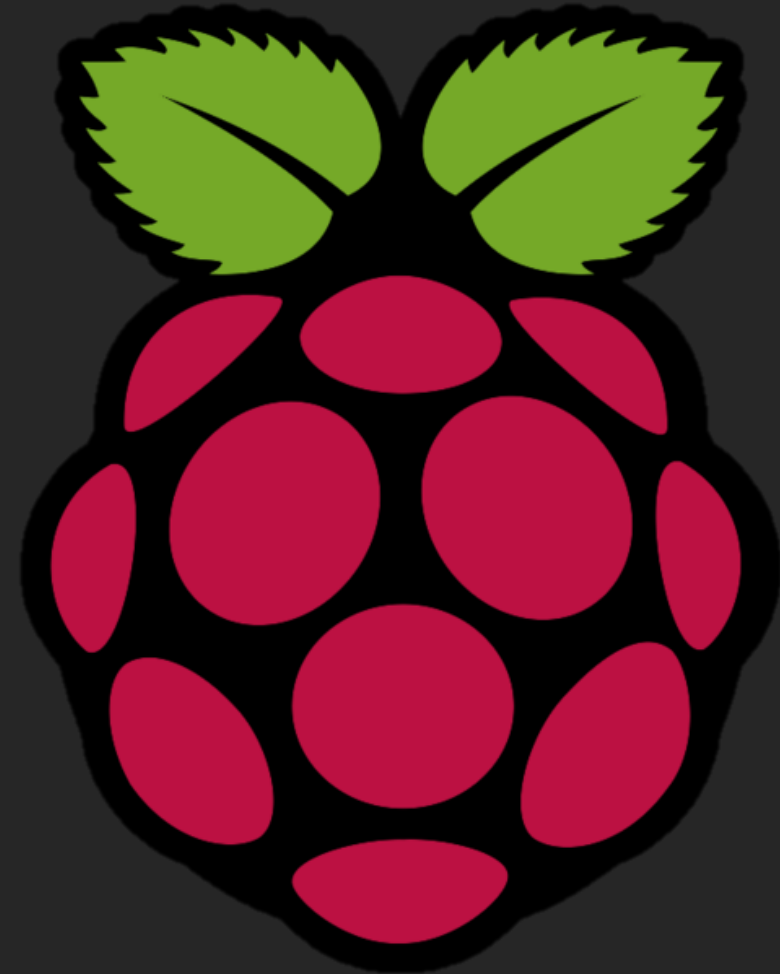
Kesako Raspberry Pi ?

- Ordinateur embarqué !
- Carte de développement idéale pour :
 - projet de serveur web basse consommation (serveur domotiques, petit site web, etc.)
 - projet de vision par ordinateur
 - interaction nécessitant un écran, des périphériques USB, caméra, etc.
- Ne pas utiliser pour :
 - projet ayant de forte contrainte énergétique
 - projet nécessitant du « temps » réel
- Utilise Linux
- Attention niveau de tension logique de 3,3 V



Installation de Raspberry Pi OS

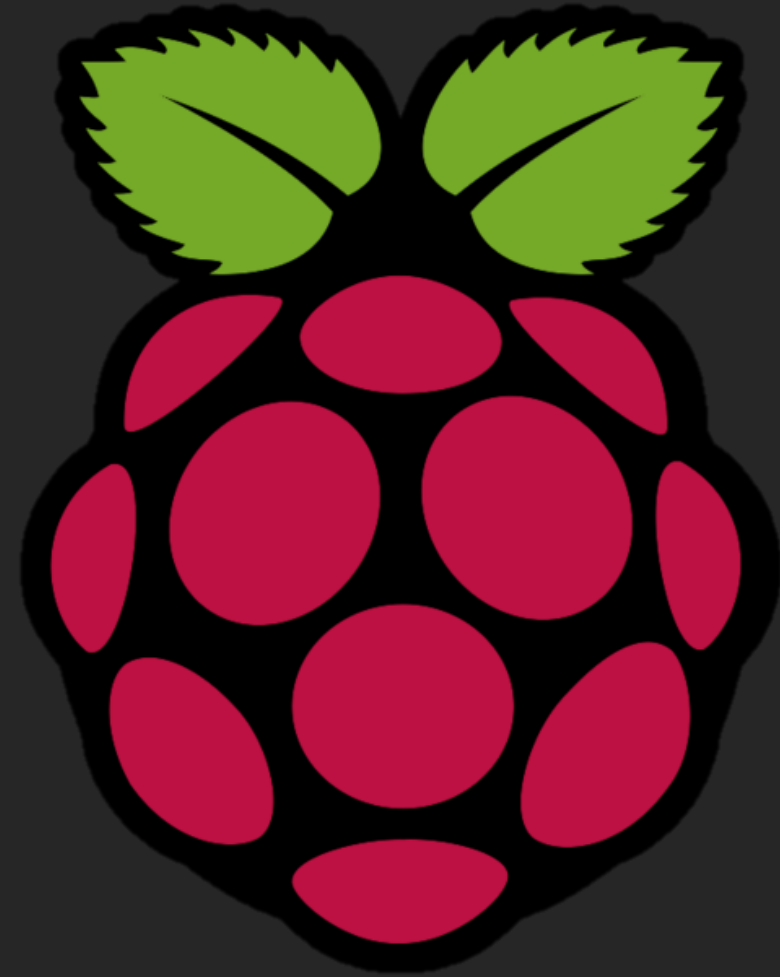
- Télécharger Pi Imager
 - Choisir l'OS
 - Choisir la carte (attention !)
 - Configurer l'image :
 - hostname
 - SSH
 - AP
 - etc.



SSH

Dans un terminal :

```
ssh creactifs@creactifs-1
```



SSH sans mot de passe !!!

Dans un terminal :

```
ssh-keygen -t rsa
```

```
cat .ssh/id_rsa.pub | ssh creactifs@creactifs-0 'cat >> .ssh/authorized_keys'
```

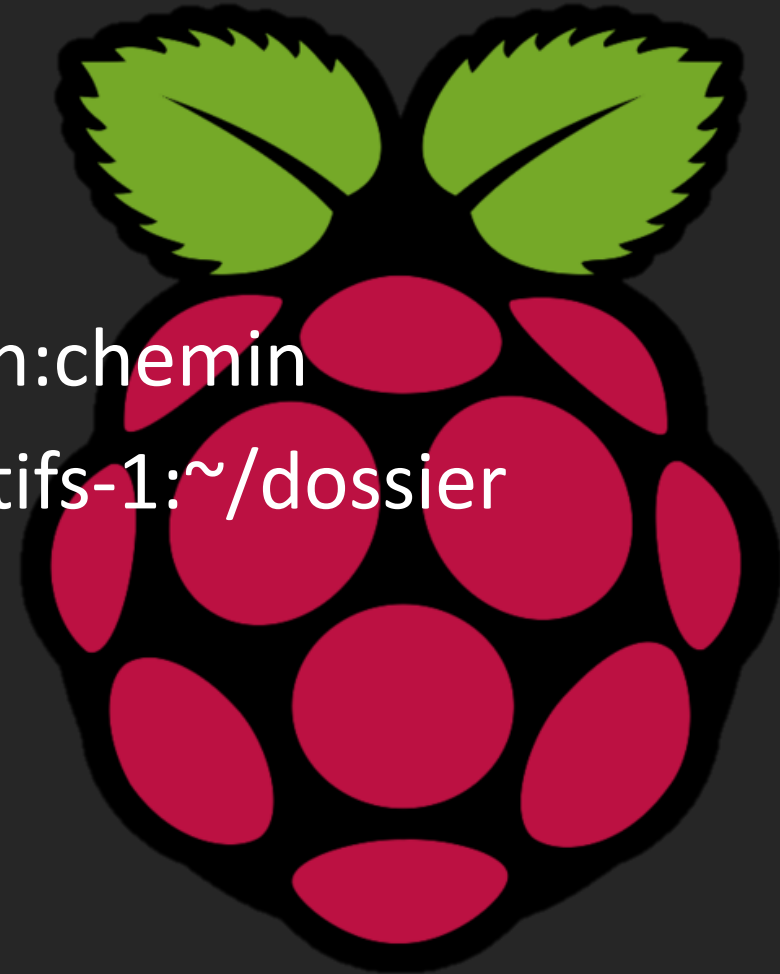
Source : http://linuxproblem.org/art_9.html



SCP

Dans un terminal :

```
scp -r source:chemin destination:chemin  
scp -r ./dossier creactifs@creactifs-1:~/dossier
```

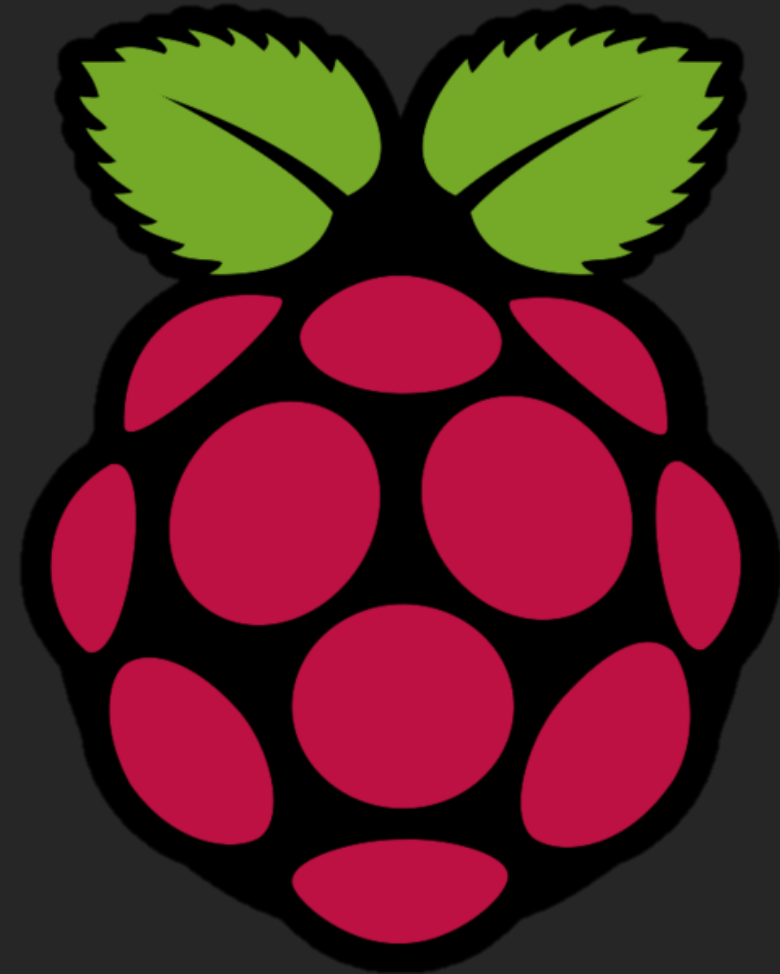


Mise à jour

Dans un terminal (SSH) :

```
sudo apt update
```

```
sudo apt upgrade -y
```

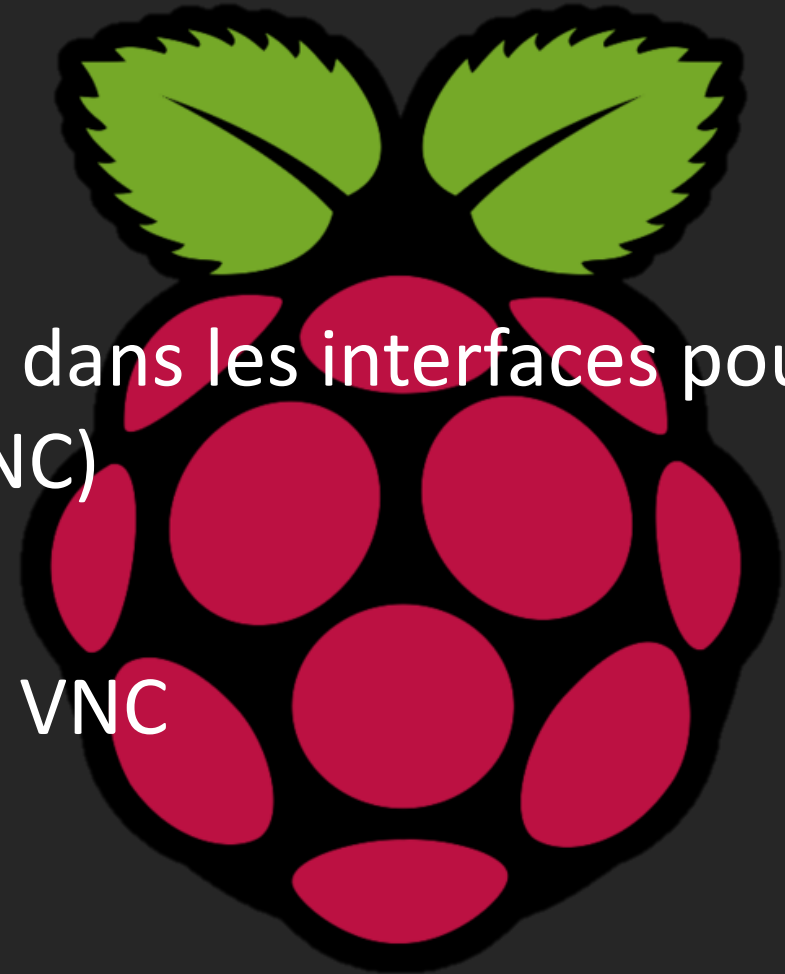


Activer VNC

Dans un terminal (SSH) :

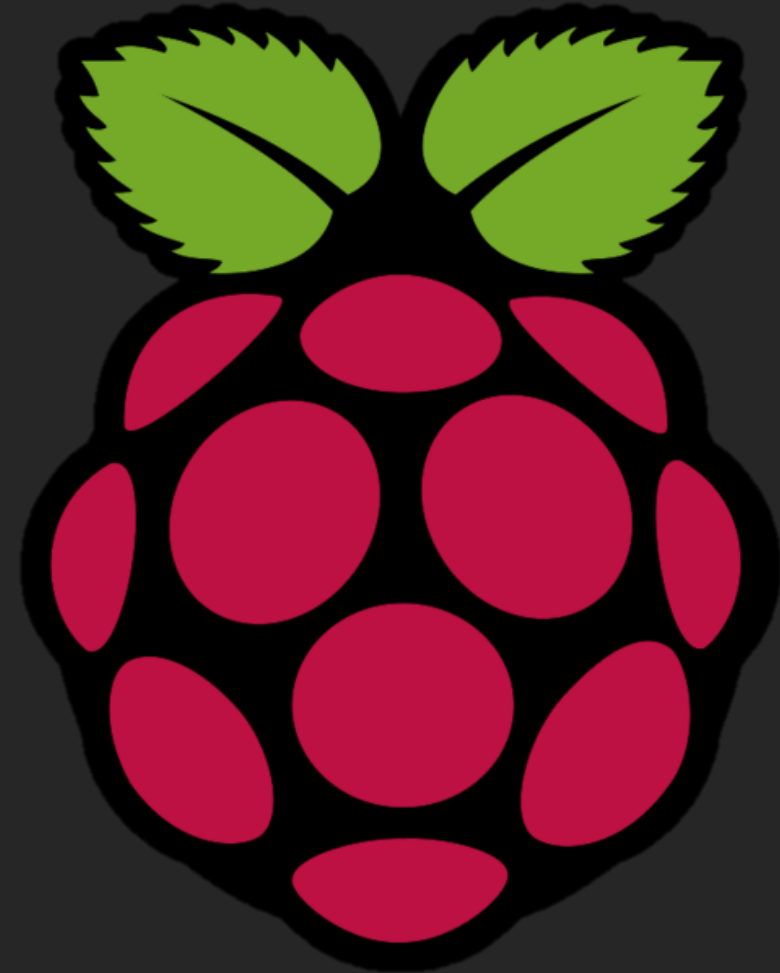
`sudo raspi-config` (puis naviguer dans les interfaces pour activer VNC)

Sur machine locale, installer un client VNC



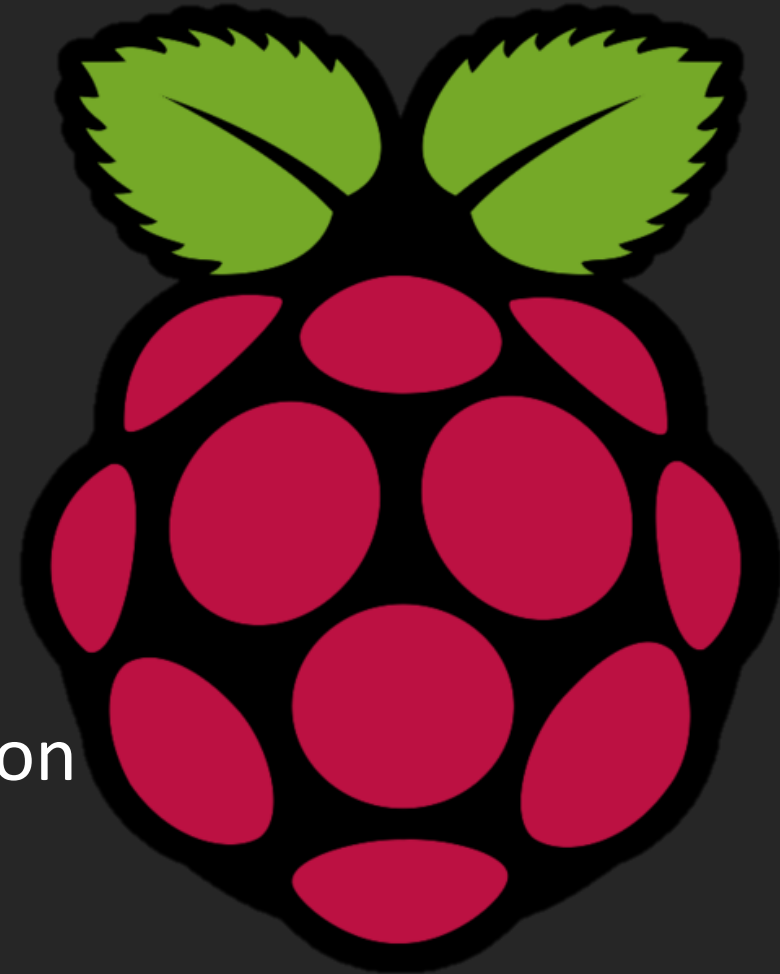
Commandes Linux de base

- Se déplacer
 - `cd ./un/chemin/relatif`
 - `cd /un/chemin/`
- Créer des dossiers
 - `mkdir nouveau_dossier`
 - `mkdir -p ./nouveau/dossiers`
- Créer, éditer un fichier
 - `touch nom_du_fichier`
 - `nano nom_du_fichier`



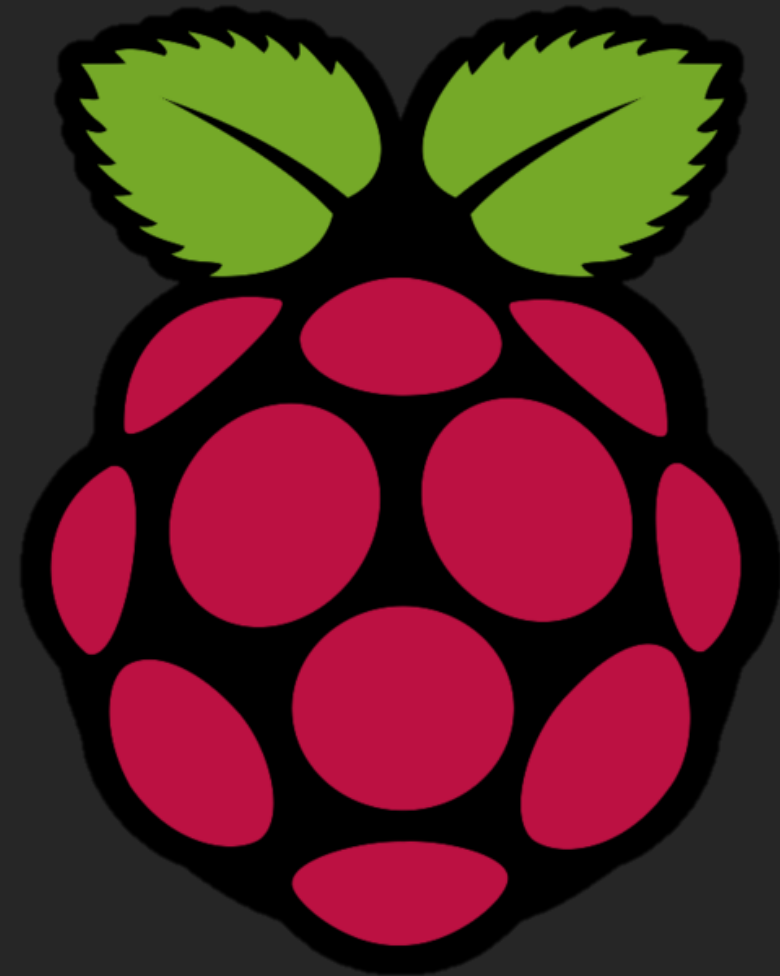
Commandes Linux de base

- Supprimer un fichier
`rm ./fichier`
- Supprimer un dossier
`rm -r ./dossier`
- Lister les fichiers et dossier
`ls`
`ls -a`
- Voir les processus en cours d'exécution
`top`
`htop`



Commandes Linux de base

- Afficher le contenu d'un fichier
`cat nom_du_fichier`
- Écrire dans un fichier
`echo contenu > fichier`

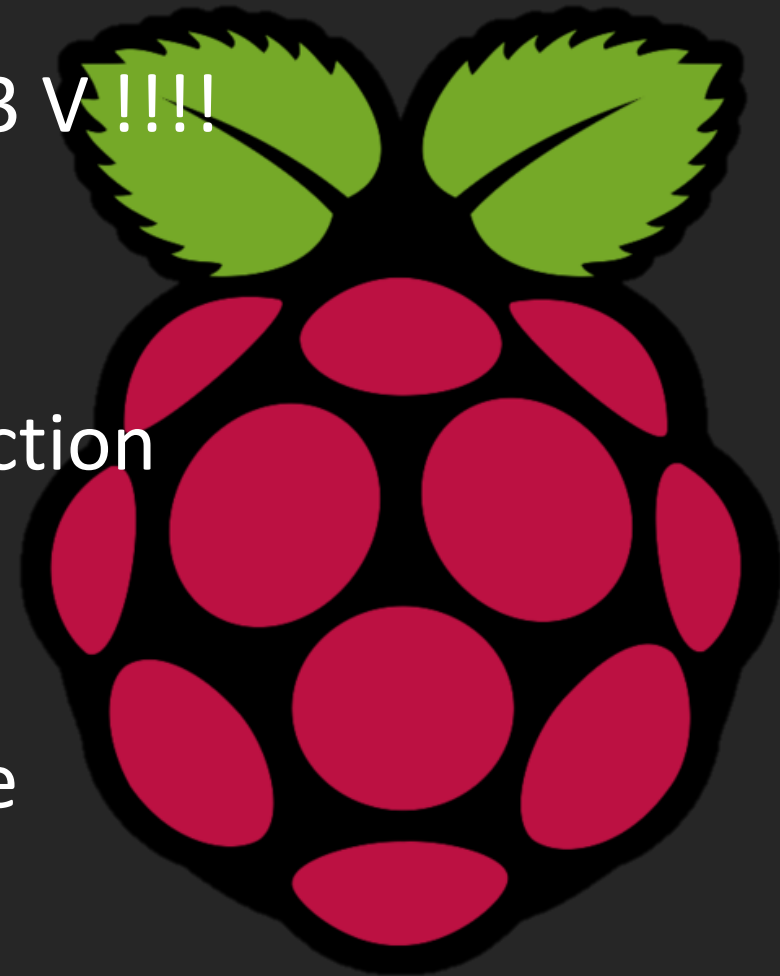


Tout est fichier !

ATTENTION NIVEAUX LOGIQUE DE 3,3 V!!!!

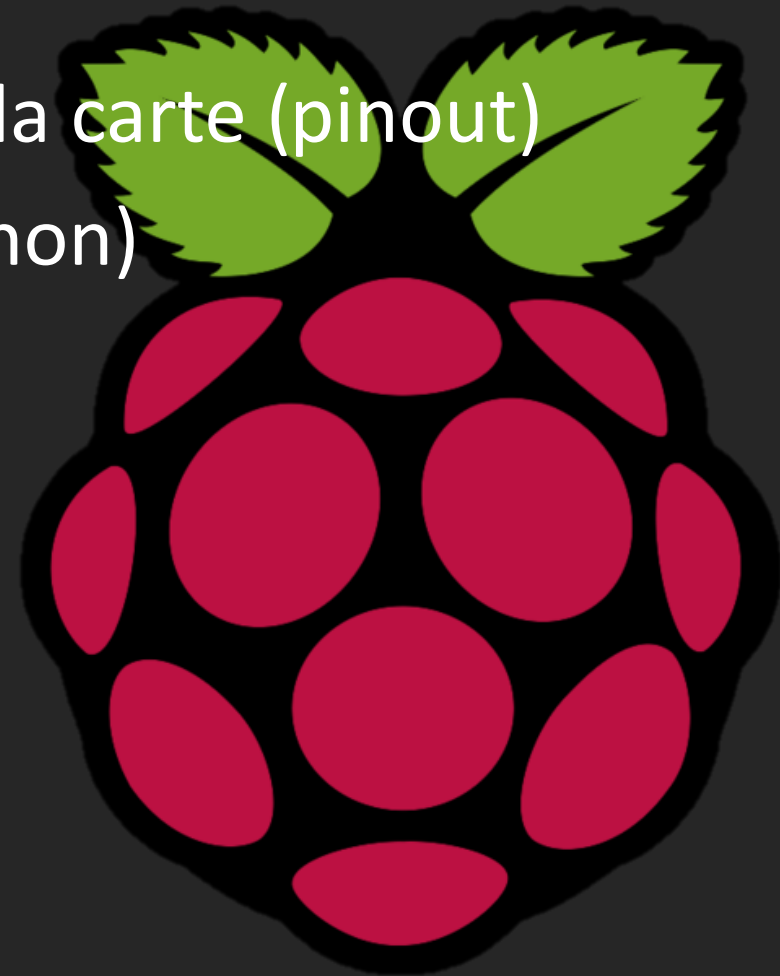
```
echo 18 > /sys/class/gpio/export  
echo out /sys/class/gpio/gpio18/direction
```

```
cat /sys/class/gpio/gpio18/value  
echo 1 > /sys/class/gpio/gpio18/value  
cat /sys/class/gpio/gpio18/value
```



Plan de la séance 9

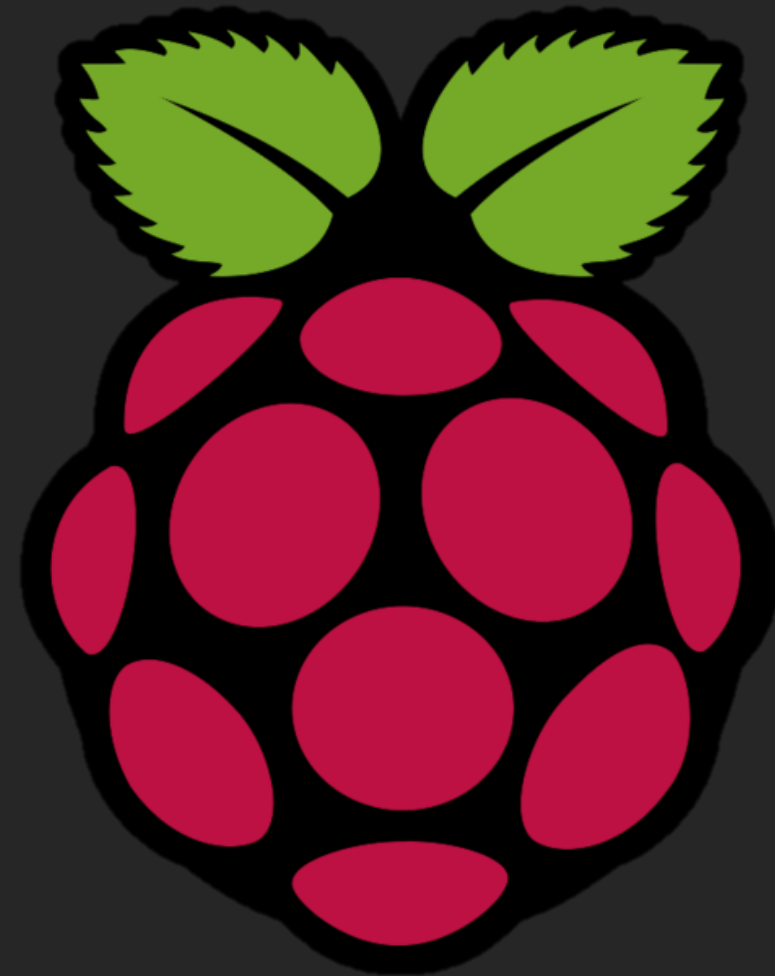
- Présentation des entrées sortie de la carte (pinout)
- Interaction avec les GPIO (C++, Python)
 - C++
 - Base de Python
 - if/elif/else, for, while, etc.
- Exercice Blink avec Python
- Exercice bouton + LED en Python



« Pinout »

3v3 Power	1	2	5v Power
GPIO 2 (I2C1 SDA)	3	4	5v Power
GPIO 3 (I2C1 SCL)	5	6	Ground
GPIO 4 (GPCLK0)	7	8	GPIO 14 (UART TX)
Ground	9	10	GPIO 15 (UART RX)
GPIO 17	11	12	GPIO 18 (PCM CLK)
GPIO 27	13	14	Ground
GPIO 22	15	16	GPIO 23
3v3 Power	17	18	GPIO 24
GPIO 10 (SPI0 MOSI)	19	20	Ground
GPIO 9 (SPI0 MISO)	21	22	GPIO 25
GPIO 11 (SPI0 SCLK)	23	24	GPIO 8 (SPI0 CE0)
Ground	25	26	GPIO 7 (SPI0 CE1)
GPIO 0 (EEPROM SDA)	27	28	GPIO 1 (EEPROM SCL)
GPIO 5	29	30	Ground
GPIO 6	31	32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33	34	Ground
GPIO 19 (PCM FS)	35	36	GPIO 16
GPIO 26	37	38	GPIO 20 (PCM DIN)
Ground	39	40	GPIO 21 (PCM DOUT)

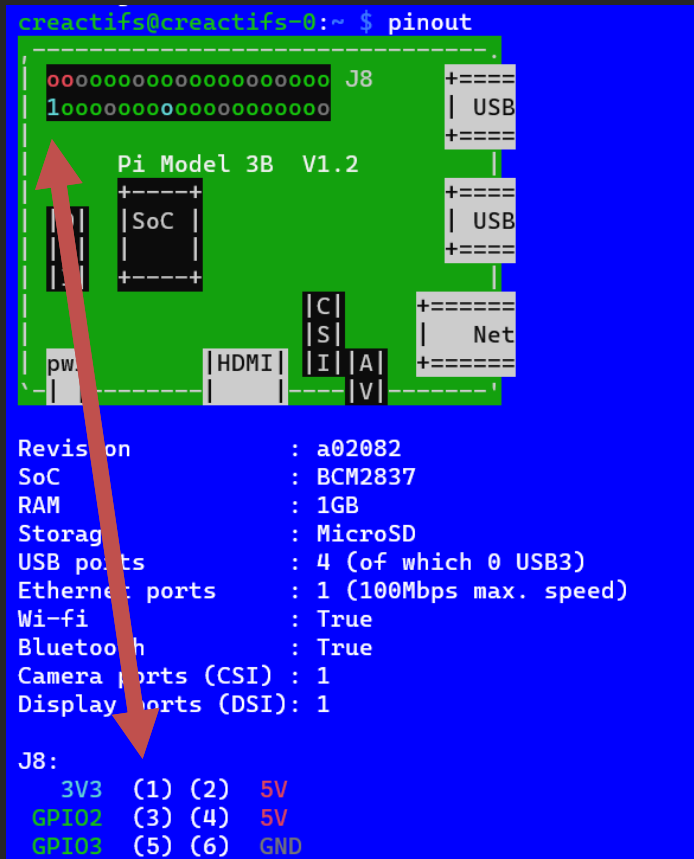
<https://pinout.xyz/>



« Pinout », un doute ?

- Saisissez la commande « pinout » dans un terminal de votre RPi

```
creactifs@creactifs-0:~$ pinout
```

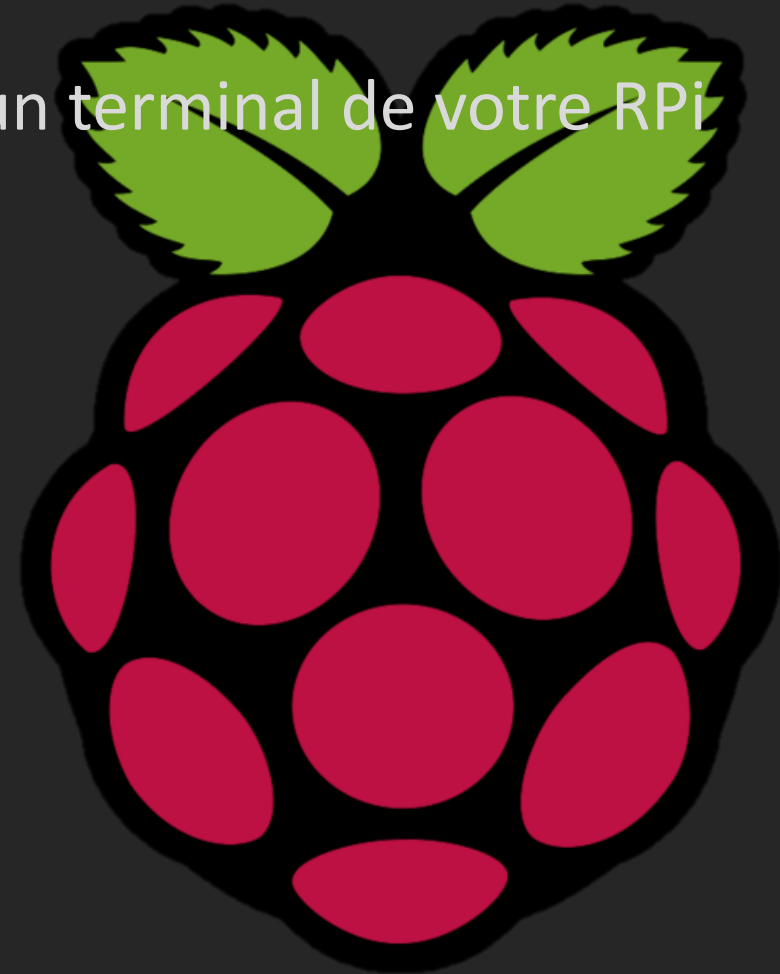


```

+-----+
| 00000000000000000000 J8 | +-----+
| 10000000000000000000 | | USB |
+-----+ +-----+
|                               | +-----+
| Pi Model 3B V1.2             | | USB |
|                               | +-----+
| SoC                           | +-----+
|                               | | Net |
+-----+ +-----+
| C | | S | | A | | I | | V | +-----+
|   | |   | |   | |   | |   |
+-----+ +-----+
| pw | | HDMI | | I | | A | | V | +-----+
|   | |   | |   | |   | |   |
+-----+ +-----+

Revision      : a02082
SoC           : BCM2837
RAM           : 1GB
Storage       : MicroSD
USB ports     : 4 (of which 0 USB3)
Ethernet ports : 1 (100Mbps max. speed)
Wi-fi         : True
Bluetooth     : True
Camera ports (CSI) : 1
Display ports (DSI): 1

J8:
 3V3  (1) (2) 5V
GPIO2 (3) (4) 5V
GPIO3 (5) (6) GND
```



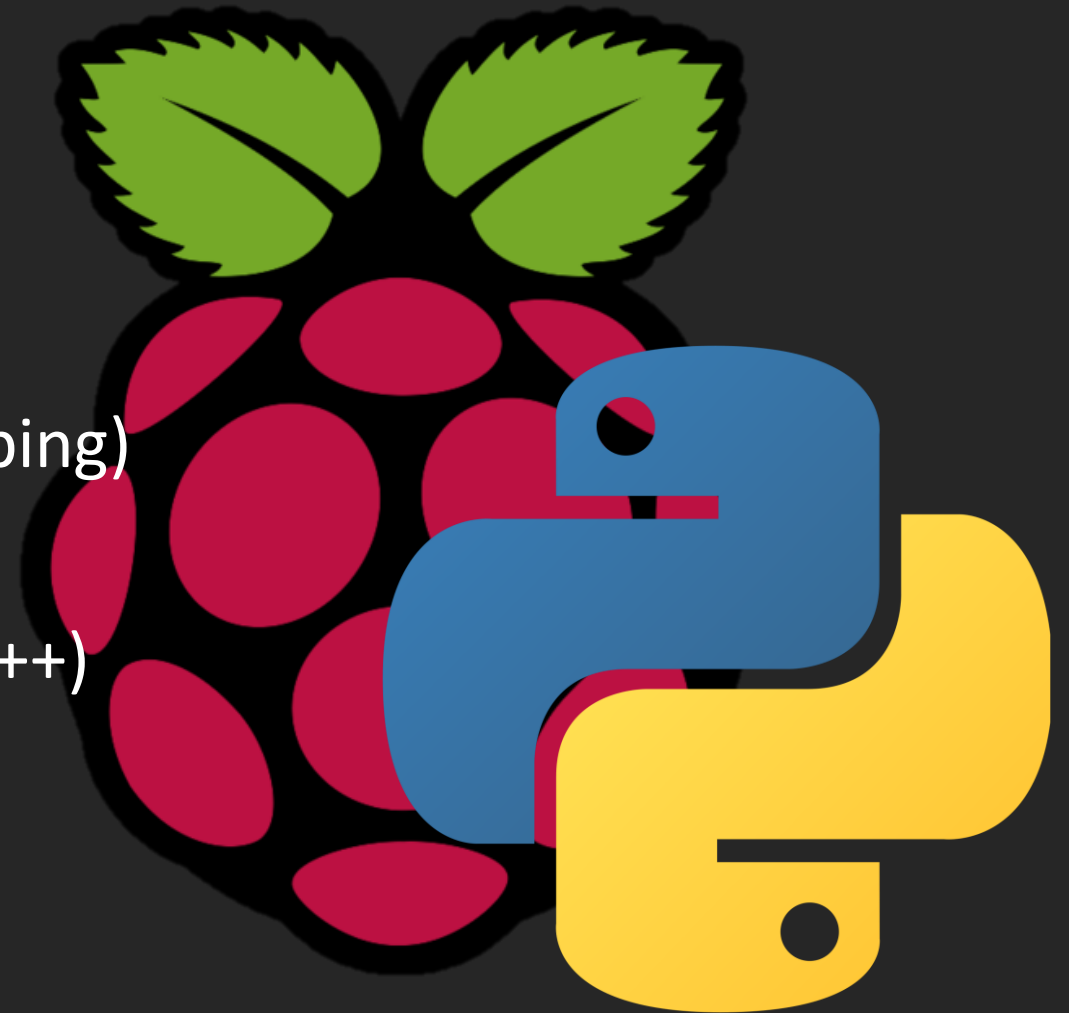
C++

- Installer WiringPi (<https://github.com/WiringPi/WiringPi>, <http://wiringpi.com/download-and-install/>)
- Créer un fichier « hello.cpp » dans ~/test_cpp
- Récupérer l'exemple de (<https://solarianprogrammer.com/2018/12/23/raspberry-pi-cpp-control-led/>)
- Compiler l'exemple : `g++ -o creactifs -I/usr/local/include -L/usr/local/lib creactifs.cpp -lwiringPi`
- Lancer le programme : `./creactifs`



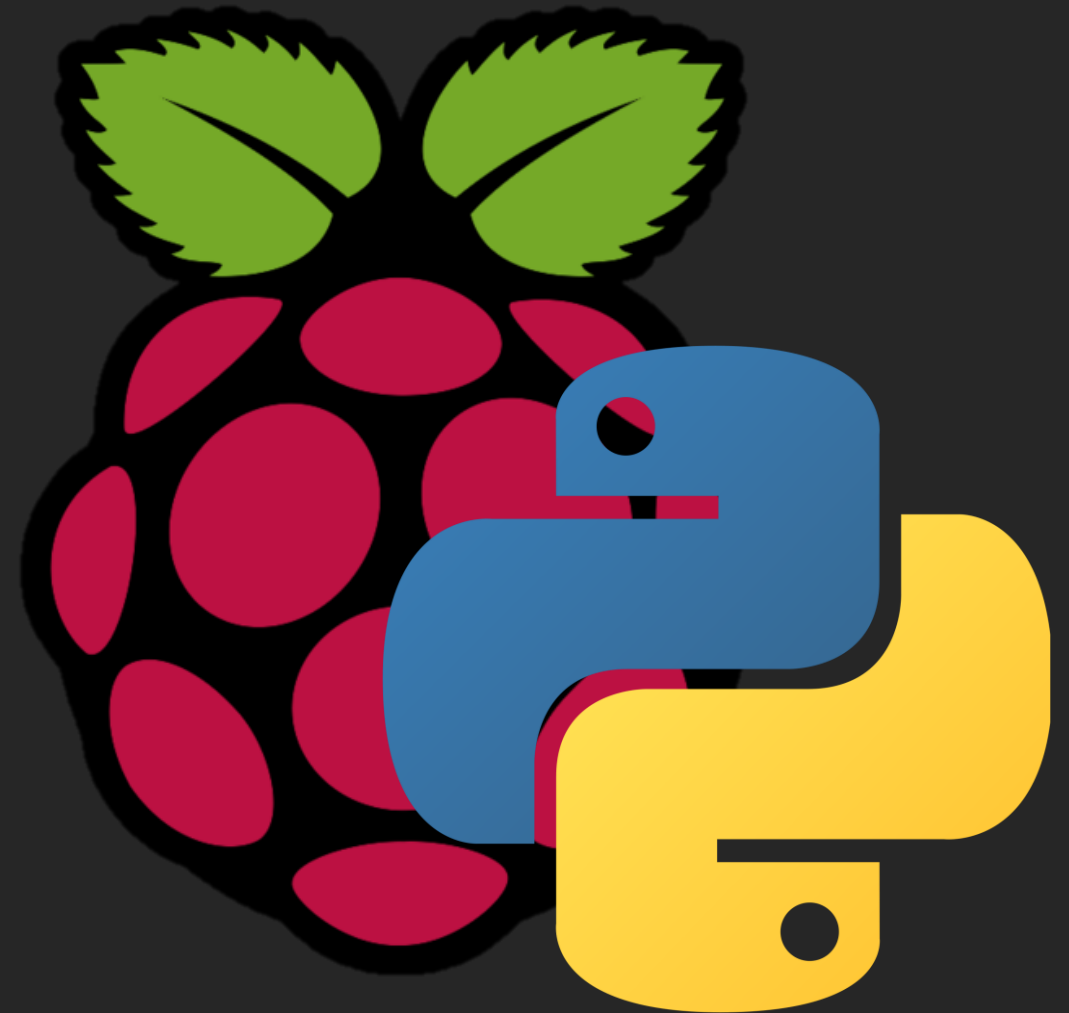
Python !

- Différences avec C++
 - Pas de fonction `setup()`, `loop()`
 - Pas de « ; » en fin de ligne
 - Pas de type strict (concept de ducktyping)
 - Pas d'accolades
 - Langage interprété (plus lent que C/C++)



Python ! – if, elif et else

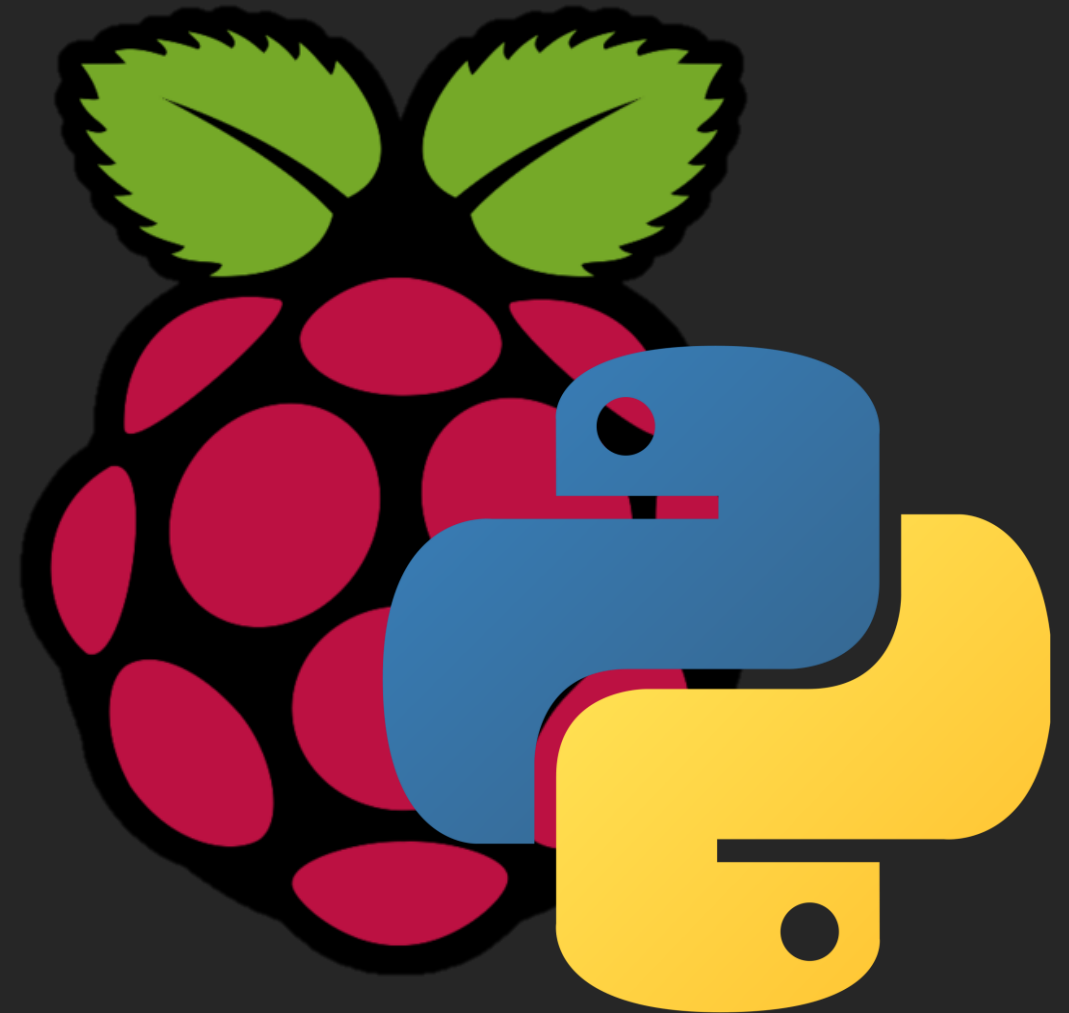
```
if condition_0 or condition_1:  
    faire_ceci()  
elif condition_2 and condition_3:  
    faire_cela()  
else:  
    la_manifestation()
```



Python ! – while

```
une_variable = True
```

```
while une_variable:  
    faire_en_boucle()  
    # Pour sortir de la boucle  
    une_variable = False  
    # Autre solution, utiliser break  
    break
```



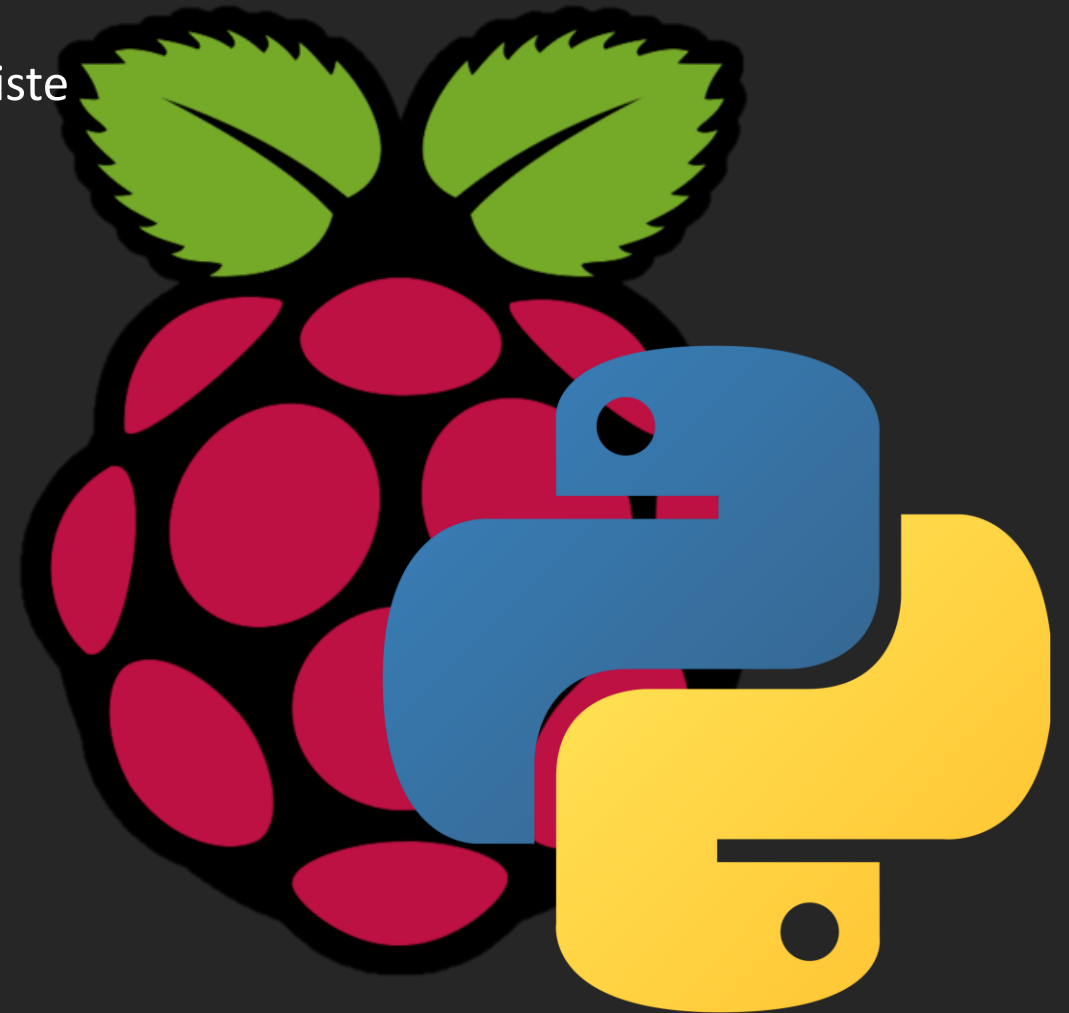
Python ! – for

```
mon_tuple = (3, 4, 7, 9) # peut aussi fonctionner avec une liste  
ma_liste = ['25', 52, '33']
```

```
# Extraction des valeurs du tuple  
for valeur in mon_tuple:  
    print(valeur)
```

```
# Extraction des valeurs du tuple et de l'index de la valeur  
for index, valeur in enumerate(mon_tuple):  
    print(valeur)
```

```
# Extraction de l'index de la liste puis de la valeur  
for index in range(ma_liste):  
    print(ma_liste[index])
```

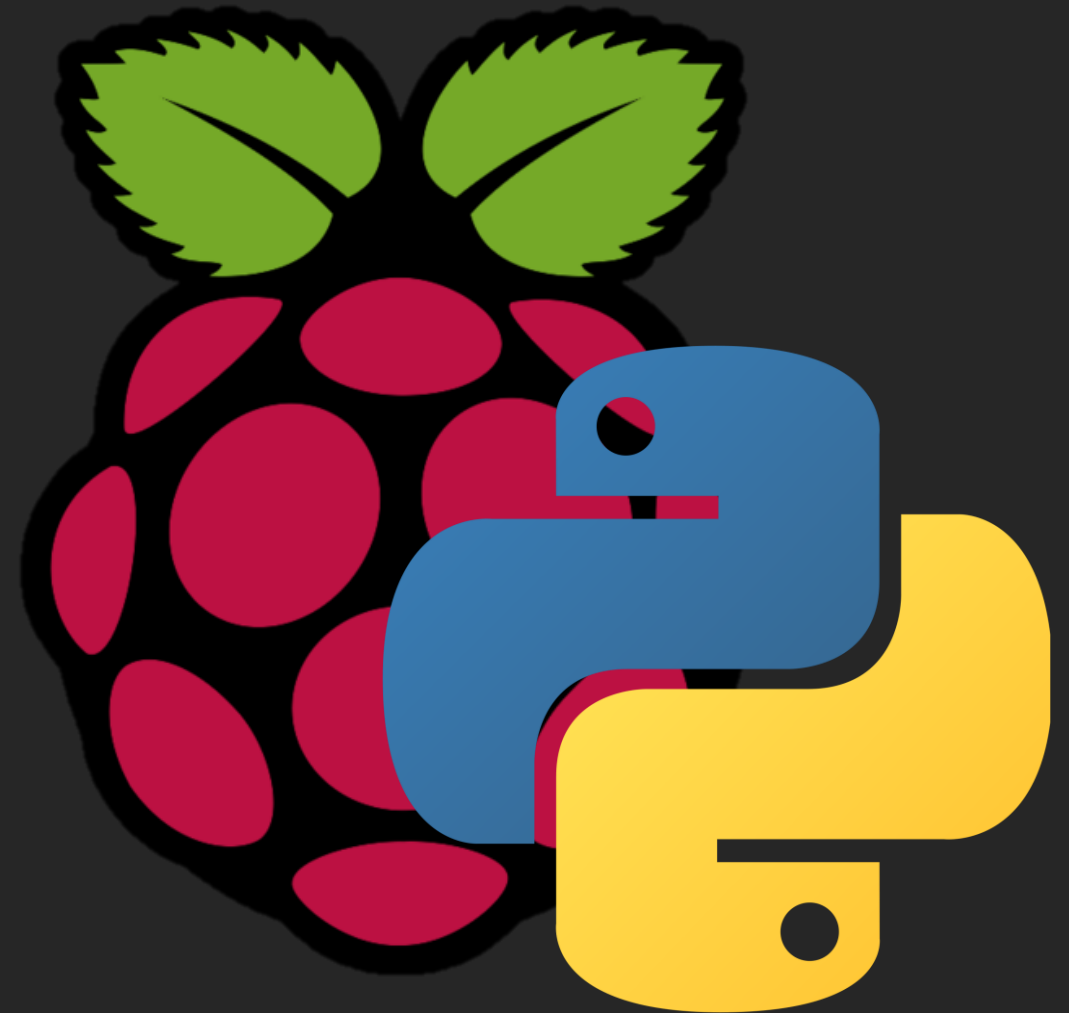


Python ! – les fonctions !

```
def addition(parametre_0, parametre_1):  
    return parametre_0 + parametre_1
```

```
def fonction_complexe(tableau):  
    resultat = 0
```

```
    def une_autre(valeur):  
        return valeur ** 2 # valeur au carré  
    for element in tableau:  
        resultat += une_autre_fonction()  
    return resultat
```

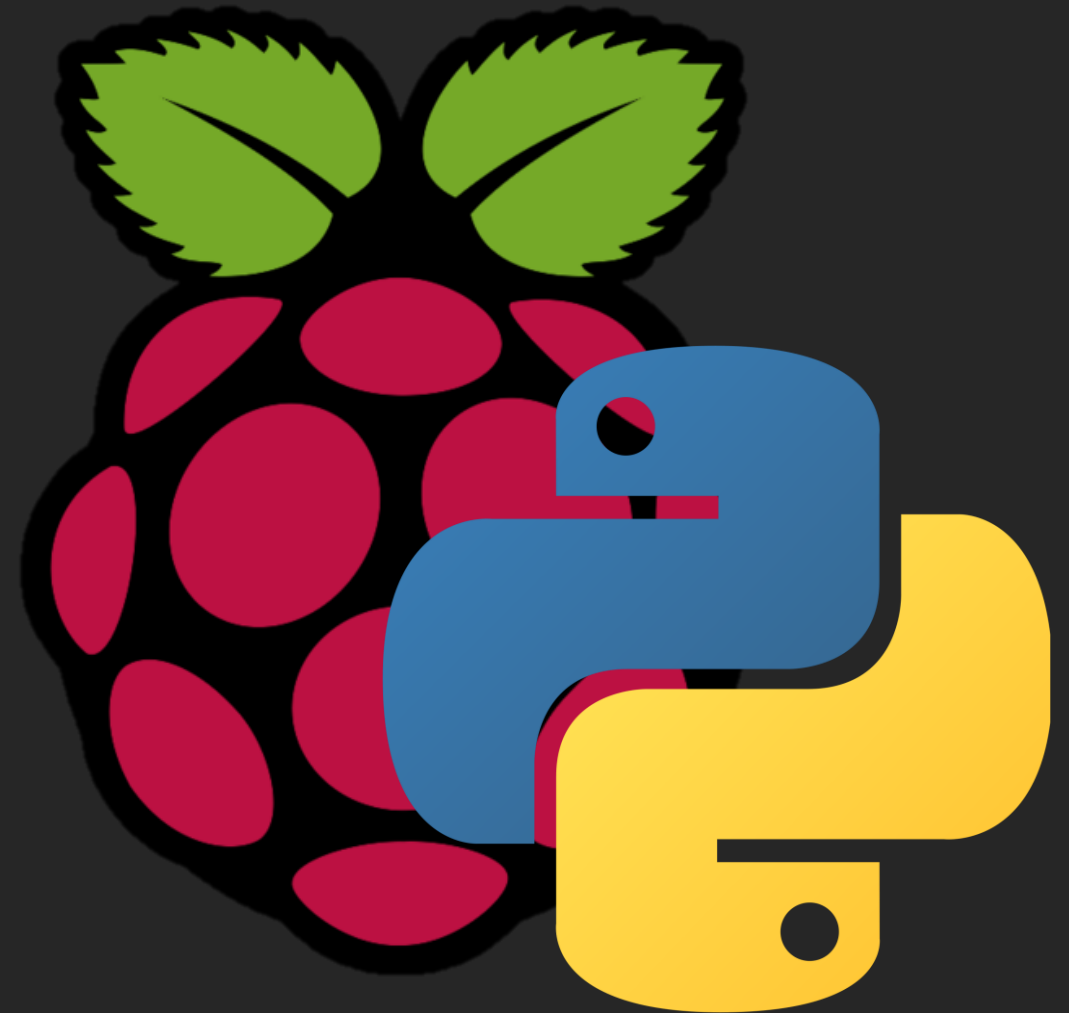


Python ! – Blink

```
from gpiozero import LED  
from time import sleep
```

```
led = LED(17)
```

```
while True:  
    led.on()  
    sleep(1)  
    led.off()  
    sleep(1)
```



Plan de la séance 10

- Présentation des entrées sortie de la carte (pinout)
- Interaction avec les GPIO (c++, Python)
- Base de Python
 - if/elif/else, for, while, etc.
- Exercice Blink avec Python
- Exercice bouton + LED en Python

