





Lire des données dans le port série avec Python

Prérequis :  Ecrire dans le port série

 **Application : récupérer via python un flux de données, les visualiser en cours d'acquisition et les sauvegarder**

 Instructions


millis() Renvoie le nombre de millisecondes écoulées depuis que le sketch a été téléversé et exécuté.

  Ecrire un programme (dans l'IDE Arduino) permettant de **simuler** un flux de données dans le port série en respectant le cahier des charges ci-dessous. La boucle simulant les mesures est interrompue lorsqu'un temps maximum (défini dans le sketch) est atteint.

Le programme doit :

- définir une variable `tMax = 5000` représentant la durée totale du processus de « mesure » ;
- initialiser une variable `t0` en capturant le temps au début de la boucle `loop()` (cf. instruction `millis()` ci-dessus) ;
- initialiser un compteur de mesures nommé `cpt` au début de la boucle `loop()` ;
- initialiser une variable `ti = t0` représentant l'instant d'acquisition de la mesure n° *i* ;
- exécuter en boucle les instructions suivantes tant que la durée du processus n'excède pas le temps maximal :
 - attendre 100 ms ;
 - incrémenter le compteur de mesures ;
 - capturer la nouvelle valeur de `ti` ;
 - calculer le temps écoulé `t` depuis `t0` en s ;
 - calculer une expression quelconque à partir de ce temps (relation affine par exemple) afin de simuler la mesure d'une grandeur au cours du temps (utiliser une variable nommée `mesure_capteur`) ;
 - écrire les valeurs de `t`, `mesure_capteur` et `cpt` dans le port série (séparés par des points-virgules).
- écrire "Stop" dans le port série une fois la boucle précédente achevée.

Programme Python (tutoriels détaillés : documents « Python – Fichiers texte » et « Python – Acquisition de données via le port série »)


```
 1 import serial
2
3 # Communication
4 port_serie = 'COM3'           # Cf. Arduino IDE (port sélectionné)
5 bauds = 115200                # Cf. sketch arduino : Serial.begin(bauds)
6 # Enregistrement des mesures
7 dossier = ""                  # Dossier courant (dossier de ce fichier python)
8 nom_fichier = "exemple2.txt"  # A personnaliser - Préférer un chemin ABSOLU
9 chemin = dossier + nom_fichier
10
11 ps = serial.Serial(port_serie, bauds) # Ouverture du port série
12 fichier = open(chemin, "w+")          # Ouverture du fichier en écriture
13 while True:                          # Boucle "infinie"
14     ligne = ps.readline()             # Lecture d'une ligne sur le port série
15     ligne = ligne.decode("utf-8")     # readline -> binaire, conversion
16     ligne = ligne.strip('\n')         # Traitement (suppression \n fin de ligne)
17     if 'Stop' in ligne:                # Message d'arrêt défini dans sketch Arduino
18         break                         # Sortie de la boucle while
19     print(ligne)                      # Vérification visuelle dans le shell
20     fichier.write(ligne)              # Ecriture de la ligne dans le fichier
21 fichier.close()                     # Fermeture du fichier
22 ps.close()                          # Fermeture du port série
```

 **Protocole**

1. Téléverser le sketch sur la carte Arduino.
2. **Fermer le moniteur série Arduino** (ne pas déconnecter la carte du port USB).
3. **Vérifier le paramétrage lignes 4 (port) et 5 (vitesse de transmission)** dans le programme python.
4. Exécuter le programme python : les données lues doivent s'afficher dans le shell.
5. Vérifier le contenu du fichier en l'ouvrant (possible depuis pyzo).

 **L'onglet "Serial Monitor" doit rester fermé pour que python puisse ouvrir le port série.**

Erreur python : `serial.serialutil.SerialException: could not open port 'COM3': PermissionError(13, 'Accès refusé.', None, 5)`

 Il existe de nombreuses façons d'interrompre la lecture dans le programme python (durée d'acquisition, nombre de mesures...) qui ne nécessitent pas d'inclure dans le sketch Arduino l'envoi d'une instruction particulière.