Communication JSON sur le port série (ESP/Arduino)

GrndOs

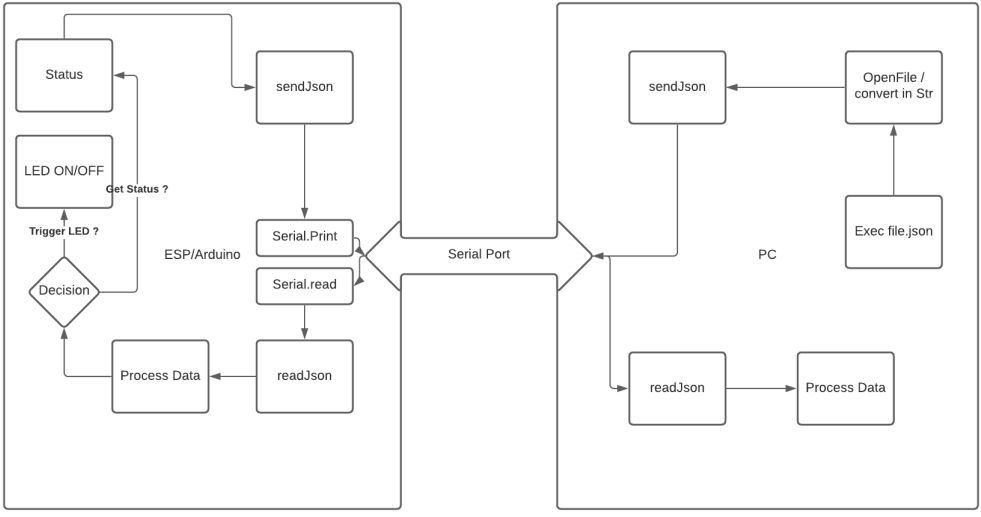
21 avril 2021

Objectif

Objectif

- Pouvoir allumer une led depuis l'ordinateur sur une ESP/Arduino
- Depuis l'ordinateur récupérer des informations envoyée par l'ESP/Arduino

Schéma fonctionnel



ESP/Arduino

Manipulation de JSON

Pour manipuler des JSON avec l'ESP/Arduino il nous faut utiliser une librairie (ici ArduinoJson.h). Ensuite nous allons principalement utiliser deux fonctions :

- deserializeJson (Passage d'un json valide sous format str en objet json)
- serializeJson (Passage d'un objet json sous format str)

```
receivedString = Serial.readString();
Serial.print("RECV: ");
Serial.println(receivedString);
DynamicJsonDocument order(2048);
deserializeJson(order, receivedString);
if (order.containsKey("LED") && order["LED"])
{
  digitalWrite(ledBlue, HIGH);
  ledState = true;
}
```

PC

Communication avec le port série

Pour interagir avec le port série nous allons utiliser la librairie pyserial (Python). Il nous suffit d'une ligne pour ouvrir une communication série.
Exemple : serialCom = serial.Serial("COM6", 9600)

Manipuler des json avec python est très simple. Il nous suffit d'utiliser deux fonctions :

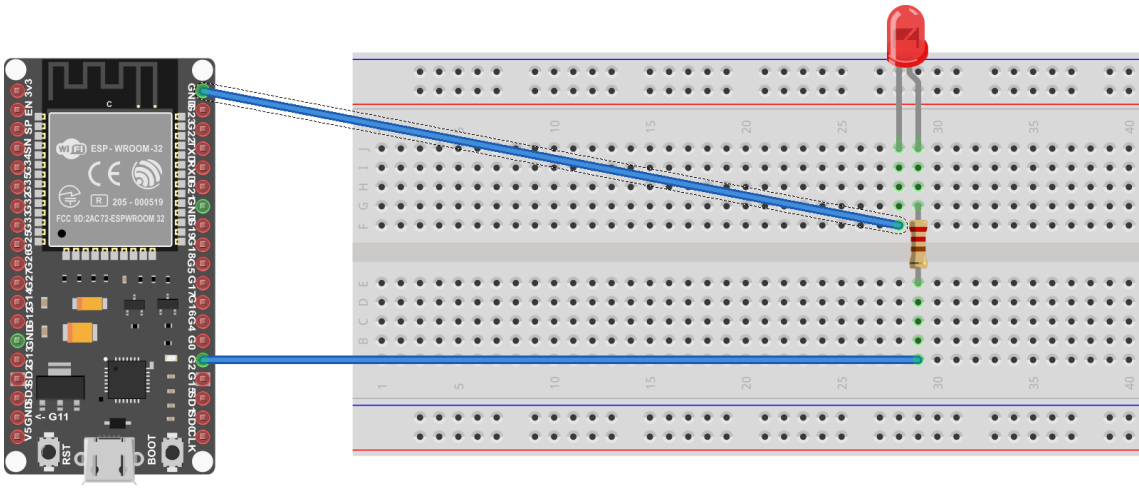
- loads (équivalent de deserialize)
- dumps (équivalent de serialize)

```
def sendJson(serial, filename):
    with open(filename) as jsonFile:
        jsonObject = json.load(jsonFile)
        #Process data before sending on serial
        serial.write(bytes(json.dumps(jsonObject), 'utf-8'))

def read_data(serialCom):
    while serialCom.is_open:
        #Get JSON from serial
        print(str(serialCom.readline(), 'utf-8'))
```

Démonstration

Branchement



Ressources

- PySerial [1]
- Json python [2]
- ArduinoJson.h [3]

1. *https://pythonhosted.org/pyserial/index.html*
2. *https://docs.python.org/fr/3/library/json.html#module-json*
3. *https://arduinojson.org/v6/api/*