

Introduction à Arduino et l'internet des objets

Trakk Namur - Fevrier-Mars 2019

Disclaimer

qualité de l'air = support pédagogique

capteurs relativement bon marché = résultats potentiellement imprécis par rapport à des solutions industrielles

Présentation du Workshop

- Séance 1 : Introduction et première approche
- Séance 2 : Bases de la programmation
- Séance 3 : Bases de la programmation
- Séance 4 : IoT : communication, collecte et traitement des données
- Séance 5 : IoT : communication, collecte et traitement des données

Qui suis-je ?

- Web developer
- Co-fondateur du studio digital 3kd.be
- Membre du hackerspace de Liège
- Enseignant

Smart Cities

Enjeux

- Data (et open data)
- Bien être
- optimisation des coûts
- motiver des décisions et actions

Smart Cities

Acteurs

- Pouvoirs publics
- Entreprises privées
- Initiatives citoyennes

Smart Cities

Critique

- Tensions entre acteurs publics et privés
- Manque de vision dans la gouvernance
- Trop souvent approche “top to bottom”

Smart Cities

Exemple

- Namur :
 - <https://data.namur.be>
 - <https://www.namur.be/fr/ma-ville/smart-city/namur-ville-intelligente/la-position-de-namur-en-tant-que-smart-city>

Smart Cities

Références

- Rapport Audacities : <https://bit.ly/2HWvRgx>
- HEC-ULiège Smart City Institute :
 - <http://labos.ulg.ac.be/smart-city/>
 - Baromètre Smart Cities Wallonie 2018: <https://bit.ly/2zYxNI1>
 - Guide Pratique : <http://guidesmartcity.be/>

L'internet des objets

Extension d'internet à des choses et des lieux du monde physique

Prise de décisions basée sur les données captées

L'internet des objets

- Corrélation de plusieurs facteurs
 - Coût des capteurs
 - Coût des processeurs
 - Coût de la bande passante
 - Connectivité omniprésente
 - Augmentation du nombre de smartphones

Arduino et les microcontrôleurs

- Microcontrôleur :
 - Circuit intégré reprenant les caractéristiques d'un ordinateur (mémoire et processeur)
 - Souvent programmé pour n'exécuter qu'une seule tâche
 - Performances limitées par rapport à un ordinateur mais son coût l'est également

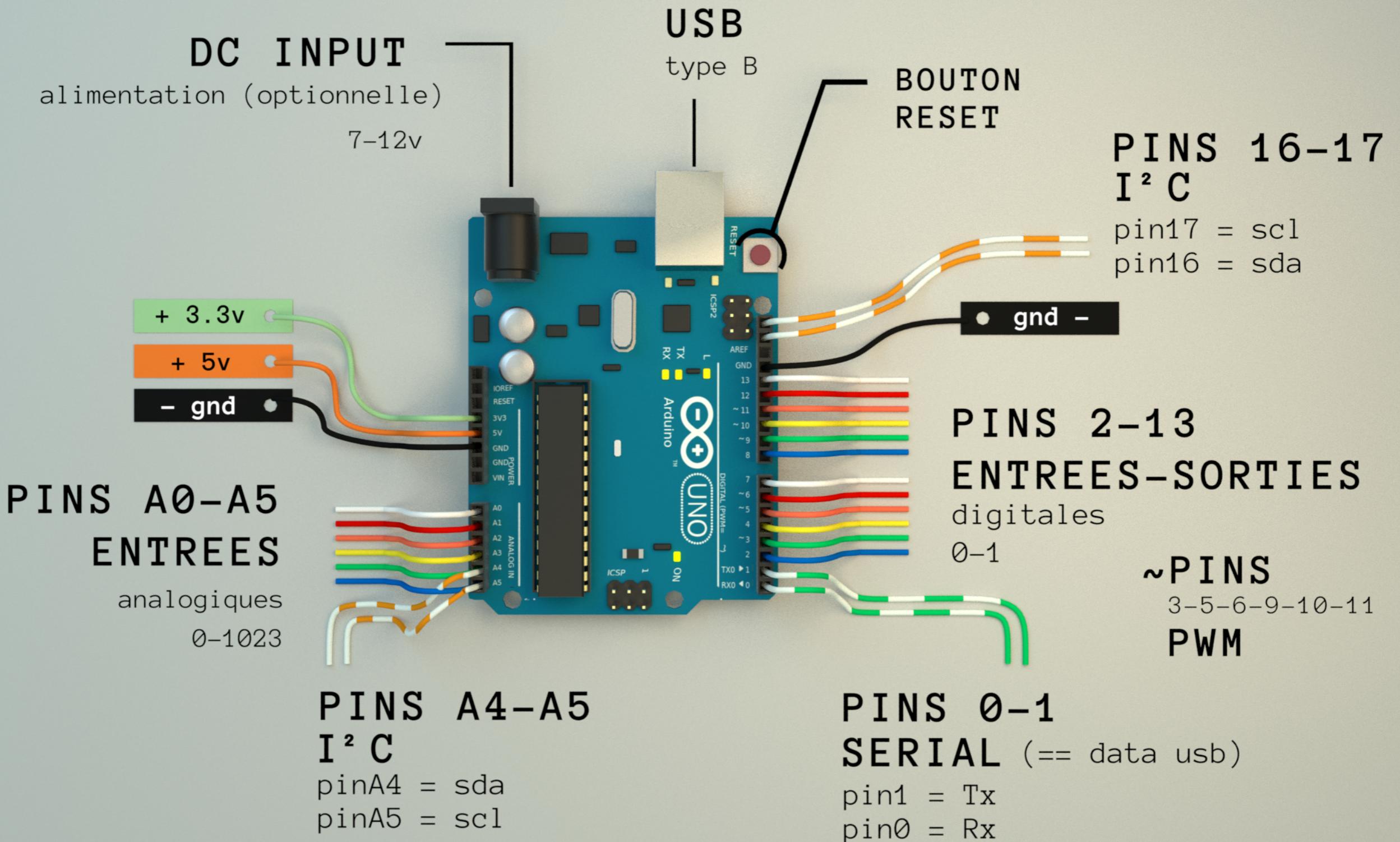
Présentation d'Arduino par l'un de ses créateurs : Massimo Banzi

Ted Talk : How Arduino is open sourcing imagination



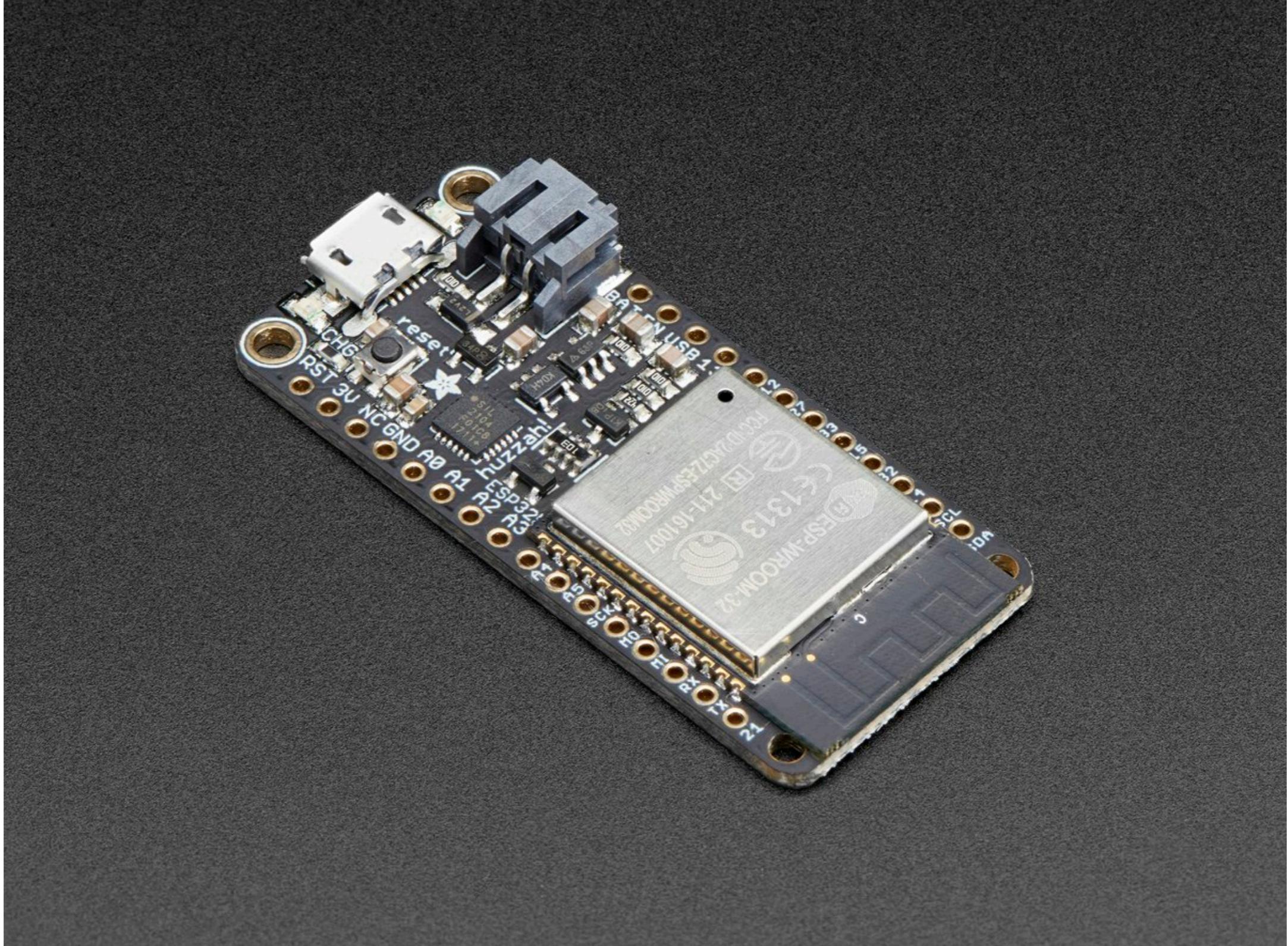
Arduino UNO

- Carte la plus répandue : précurseur
- ATMega 328p (16Mhz - 8 bits)
- 14 Entrées et sorties digitales
- 6 Entrées analogiques (ADC 10 bits)
- Tension de fonctionnement **5v**



Adafruit Feather Huzzah32

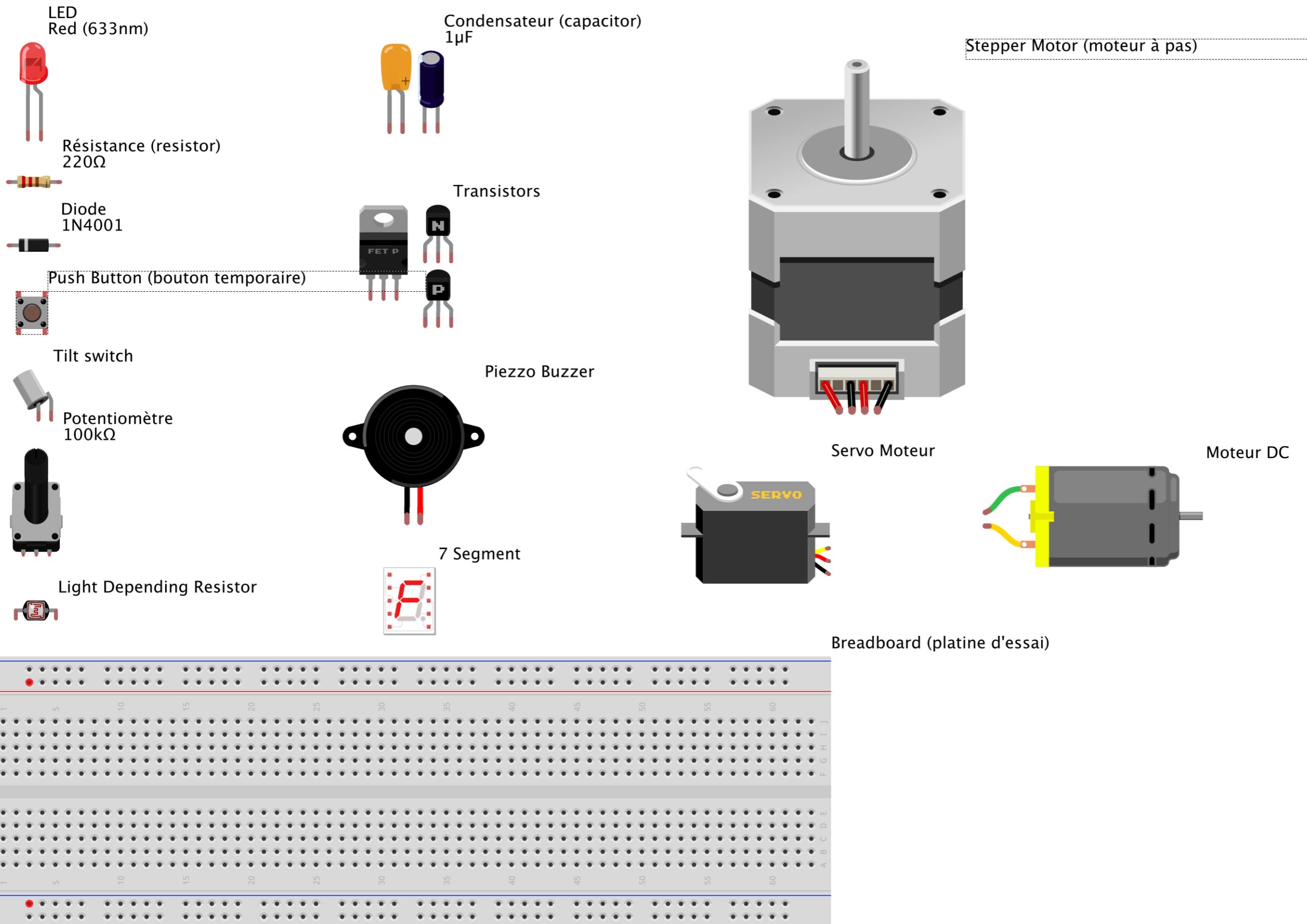
- Espressif ESP32 (Wroom32)
 - Tensilica LX6 dual-core 240Mhz
 - WiFi 2.4Ghz
 - Bluetooth et Bluetooth BLE
 - 4MB Flash
- 10 entrées touch capacitif
- 12 entrées analogiques
- 2 sorties analogique (DAC 10 bits)
- Tension de fonctionnement : **3.3v**



<https://learn.adafruit.com/adafruit-huzzah32-esp32-feather/pinouts>

Matériel

- Feather Huzzah32 :
 - <https://learn.adafruit.com/adafruit-huzzah32-esp32-feather>
- BME280 : air quality monitoring sensor
 - <https://learn.adafruit.com/adafruit-bme680-humidity-temperature-barometric-pressure-voc-gas>
- push buttons
- Leds
- potentiomètre
- phototransistor
- résistances, jumper wire
- breadboard

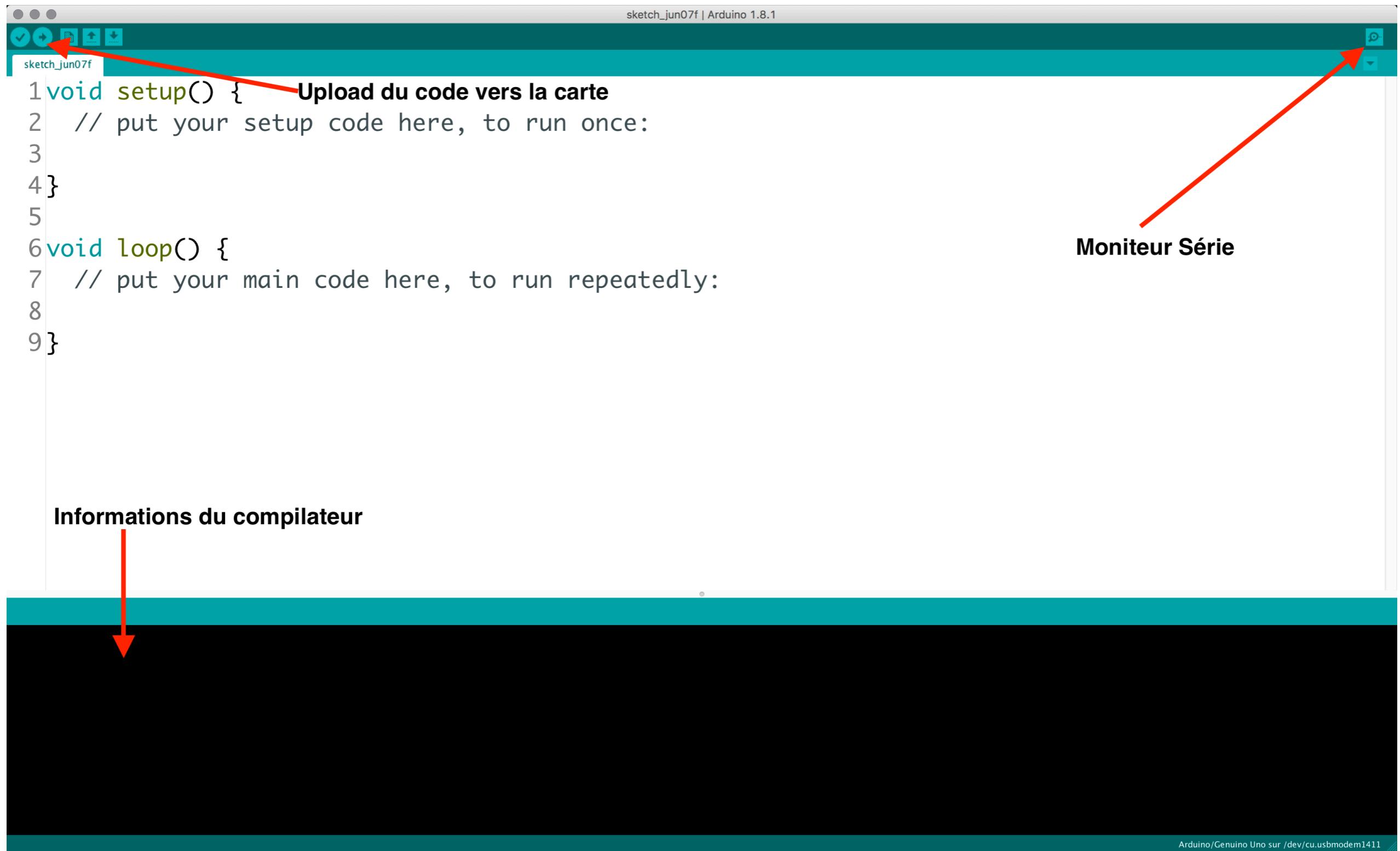


Installation et configuration des logiciels

<https://arduino.cc/downloads>

- **IDE** : Integrated Development Environment, logiciel installé sur la machine contenant tous les outils pour compiler le code et flasher le microcontrôleur

Présentation de l'IDE Arduino



Configuration de l'IDE pour ESP32

Télécharger et installer les
drivers de la carte

<https://frama.link/adafruit-feather-usb-driver>

Configuration de l'IDE pour ESP32

Installer les outils de développement pour ESP32

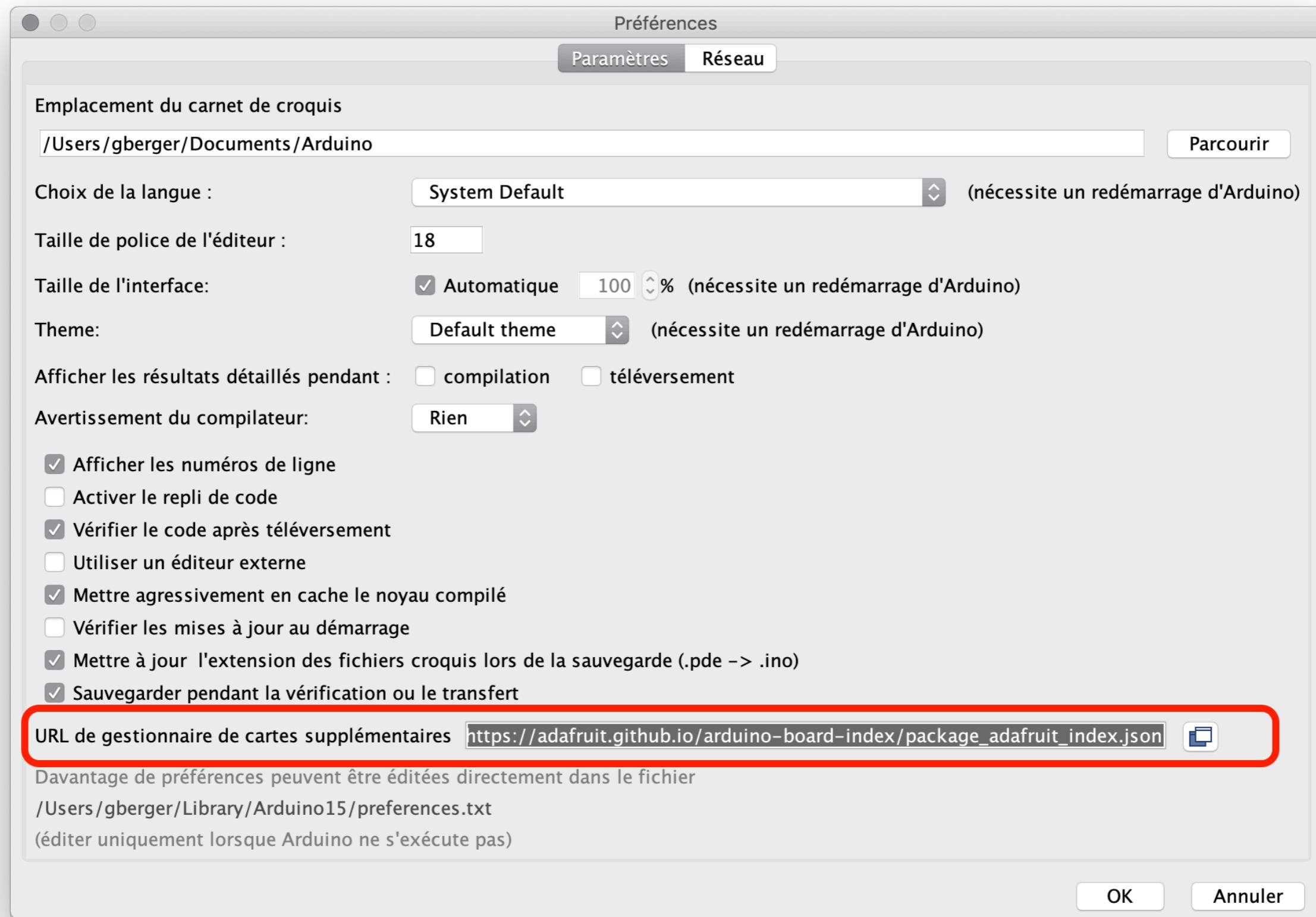
<https://frama.link/esp32-arduino>

Copier le lien vers le fichier de configuration :

https://dl.espressif.com/dl/package_esp32_index.json

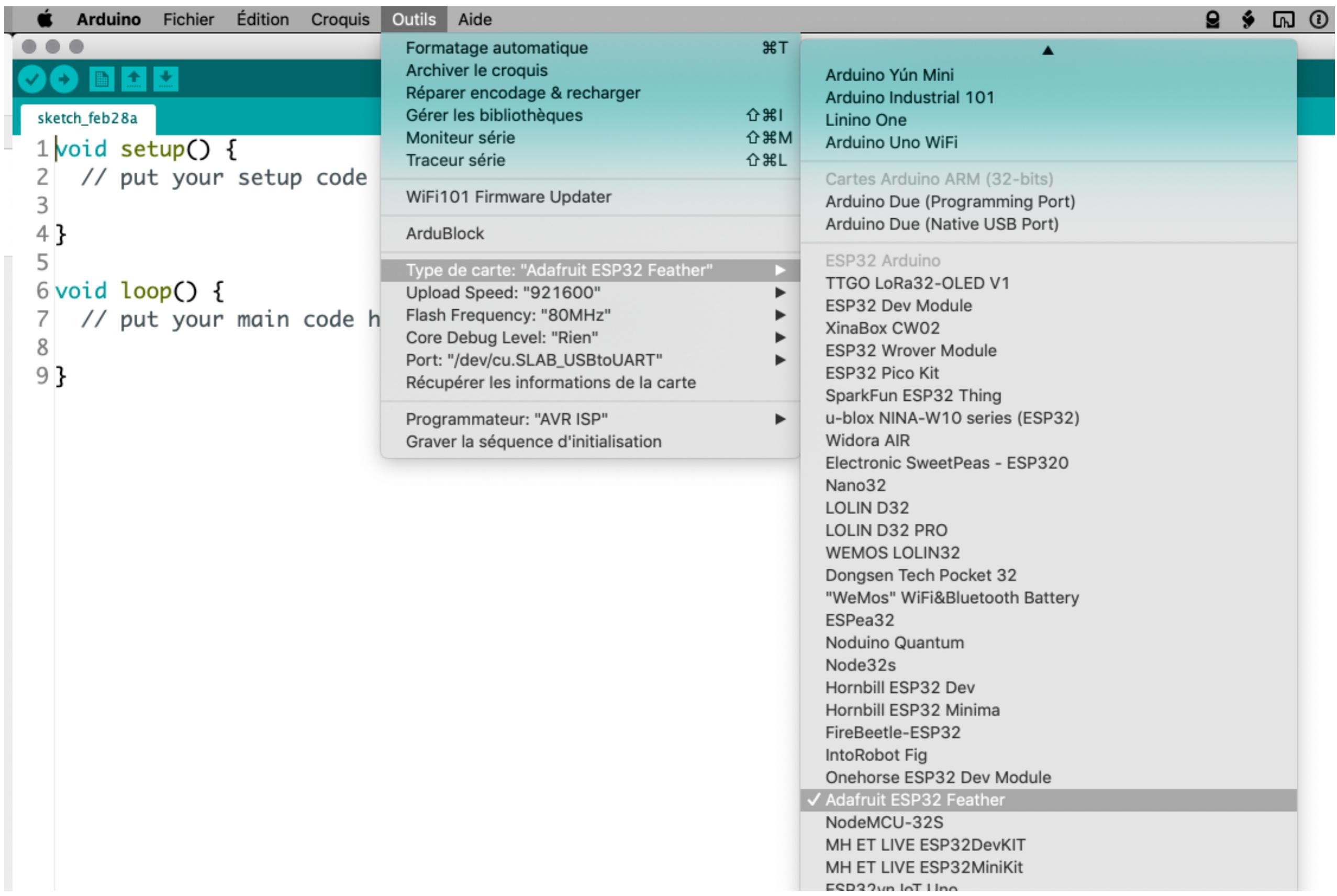
Configuration de l'IDE pour ESP32

Coller le lien dans le champs prévu



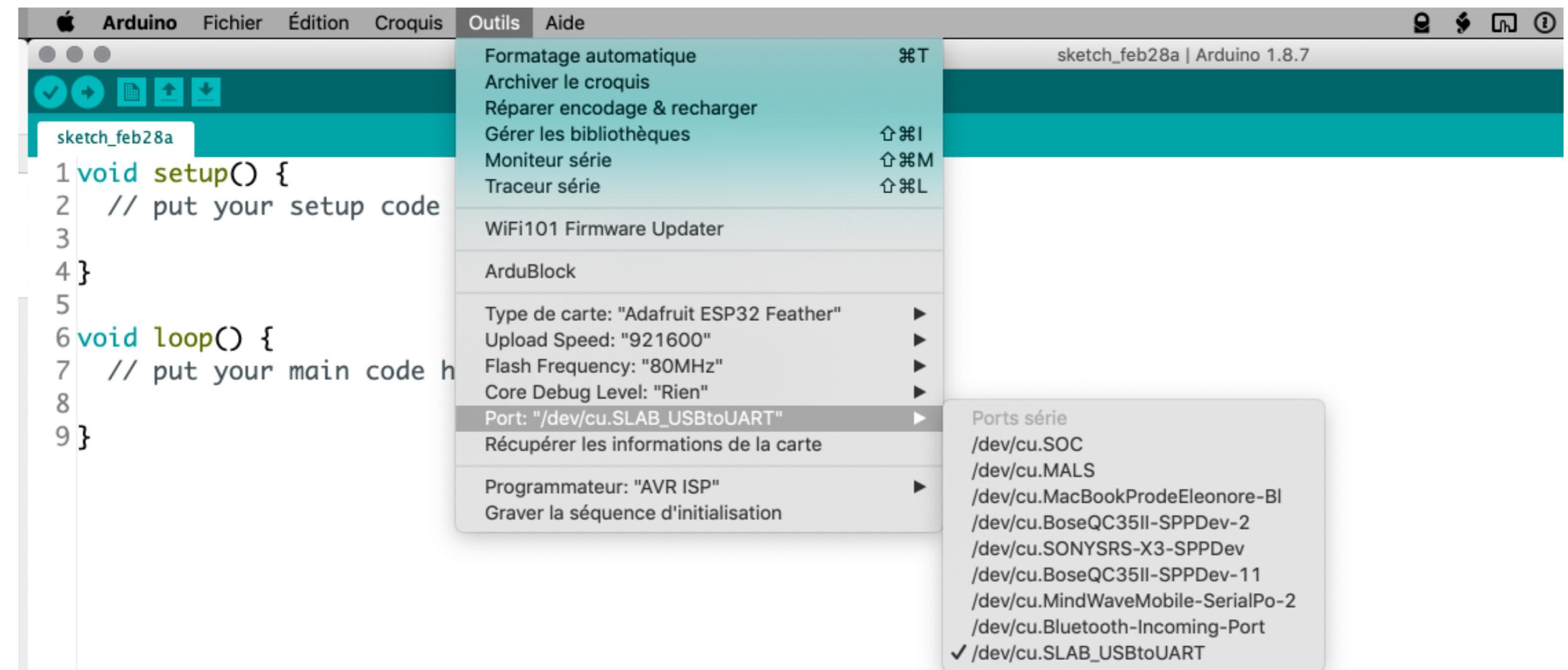
Configuration de l'IDE pour ESP32

Sélectionner la carte dans la liste



Configuration de l'IDE pour ESP32

Sélectionner le port de communication



Introduction à la programmation

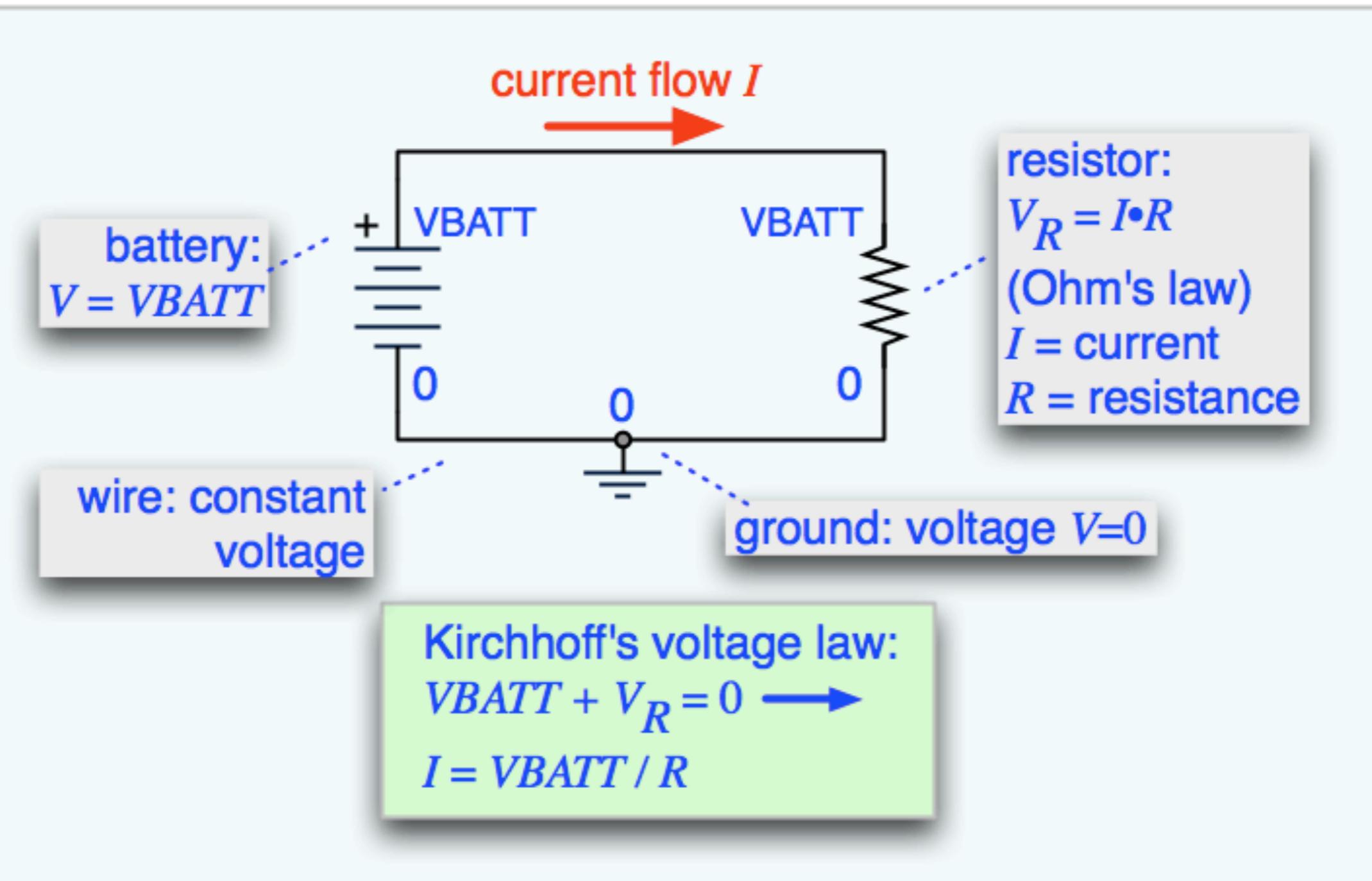
Concepts de base

- Instructions
- Variables
- Conditions
- Boucles
- Fonctions
- Bibliothèques

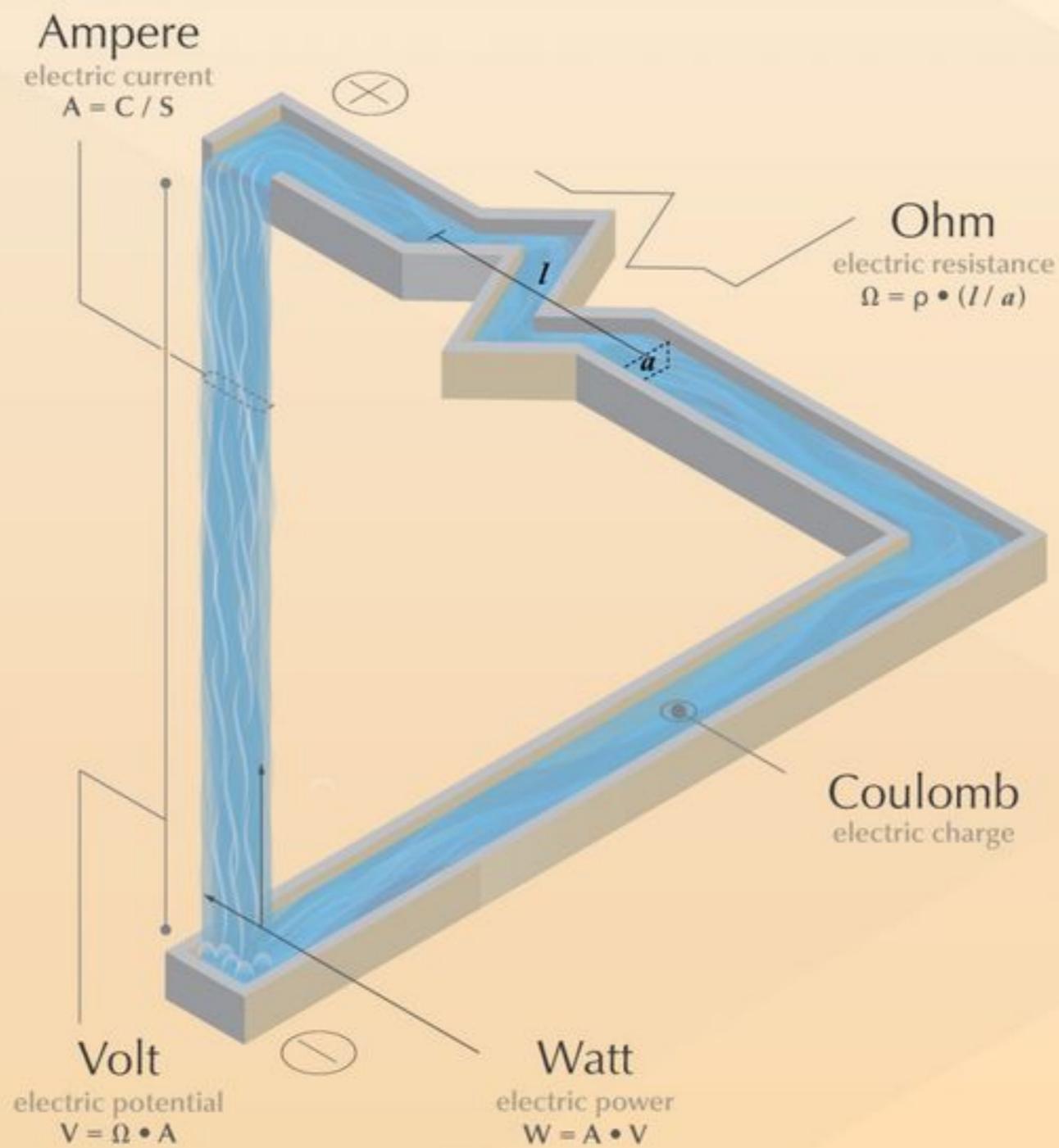
Introduction à l'électronique

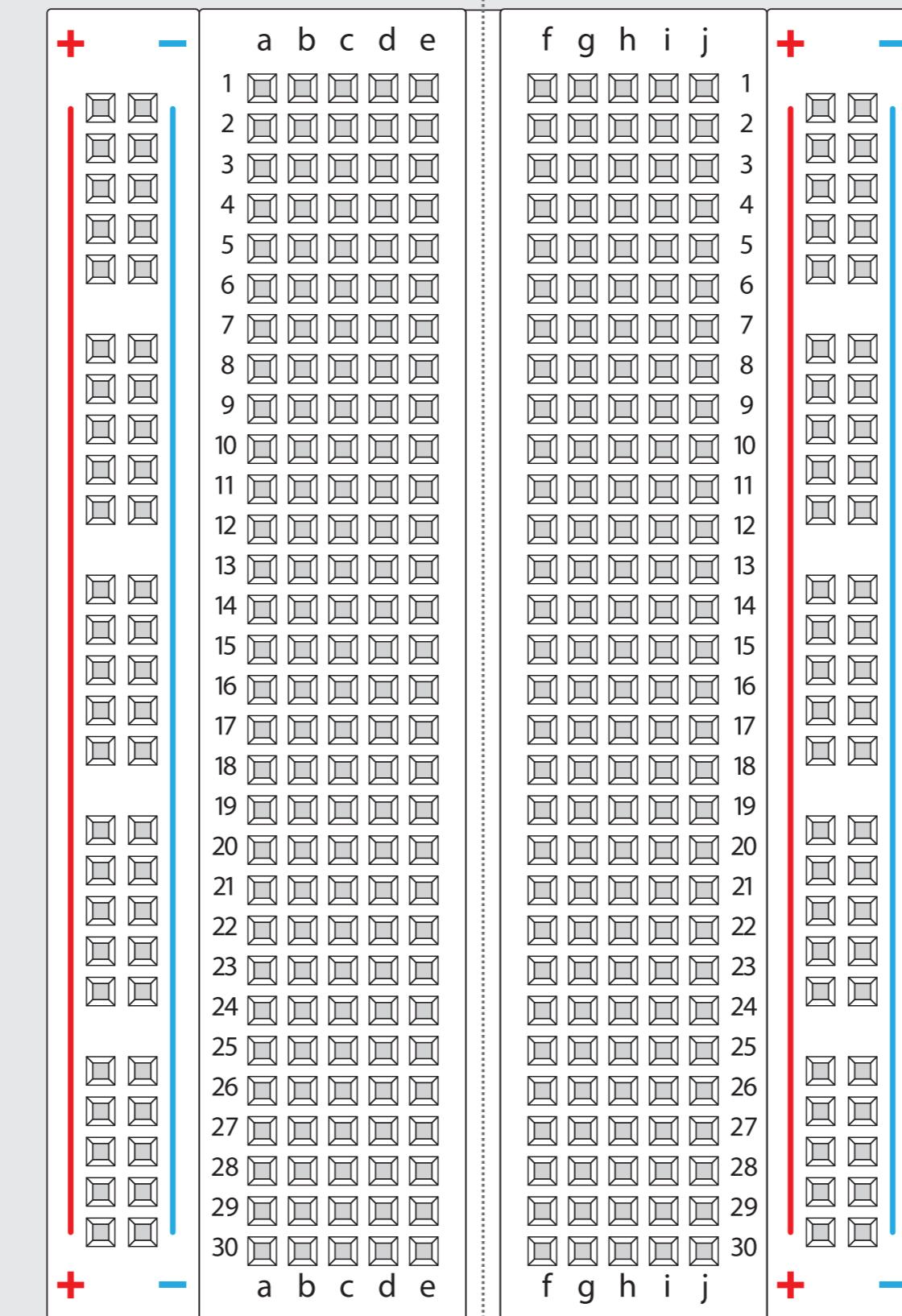
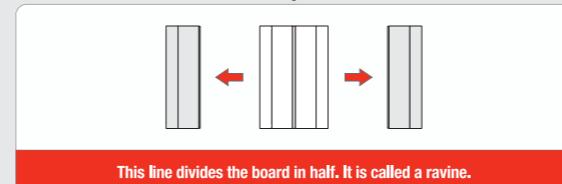
Concepts de base

- Circuit
- La loi d'Ohm
- La breadboard (ou platine d'essai)

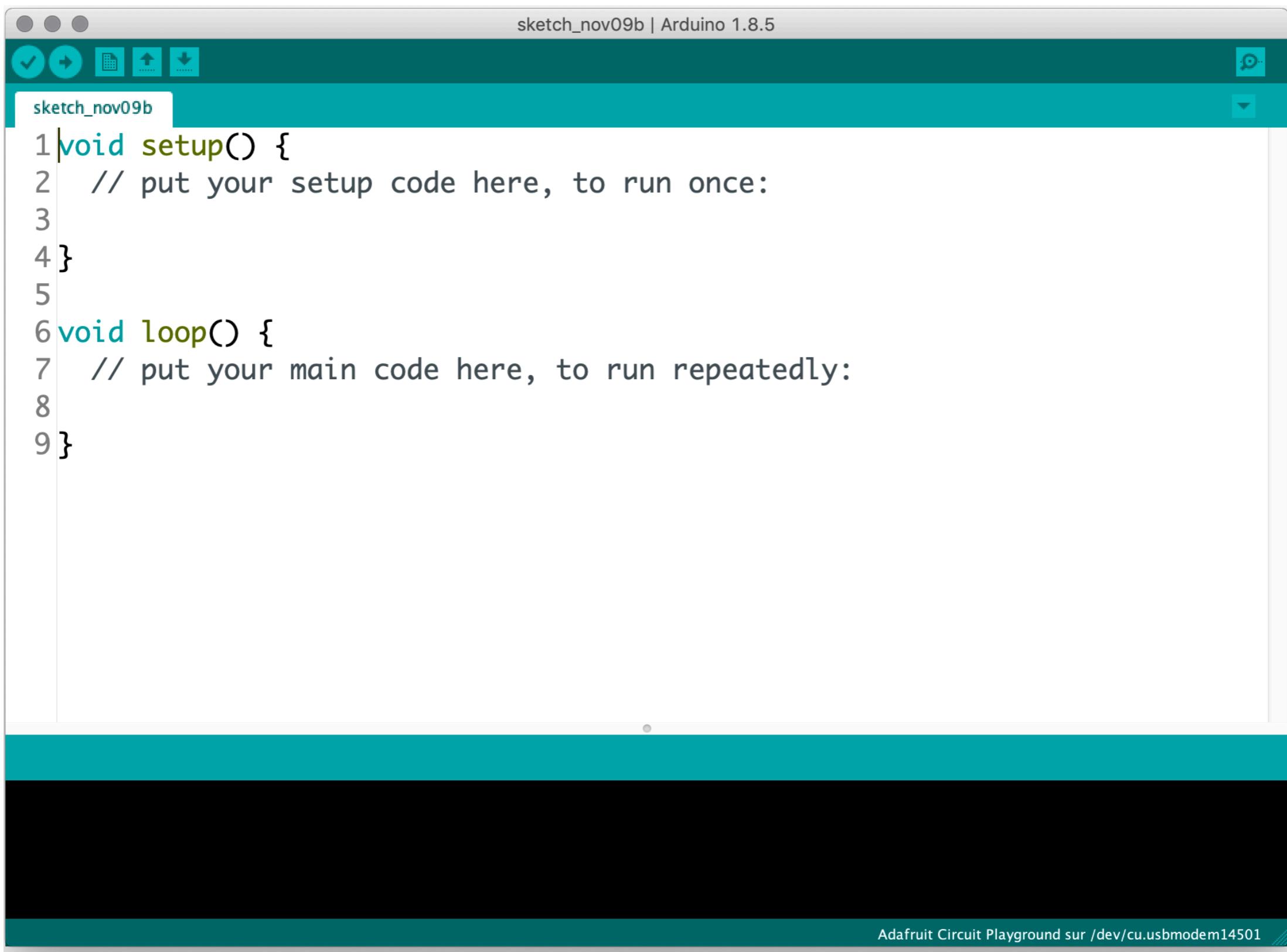


- ELECTRICITY -





Structure de base d'un programme Arduino



The screenshot shows the Arduino IDE interface with the title bar "sketch_nov09b | Arduino 1.8.5". The main window displays the following code:

```
1 void setup() {
2 // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7 // put your main code here, to run repeatedly:
8
9 }
```

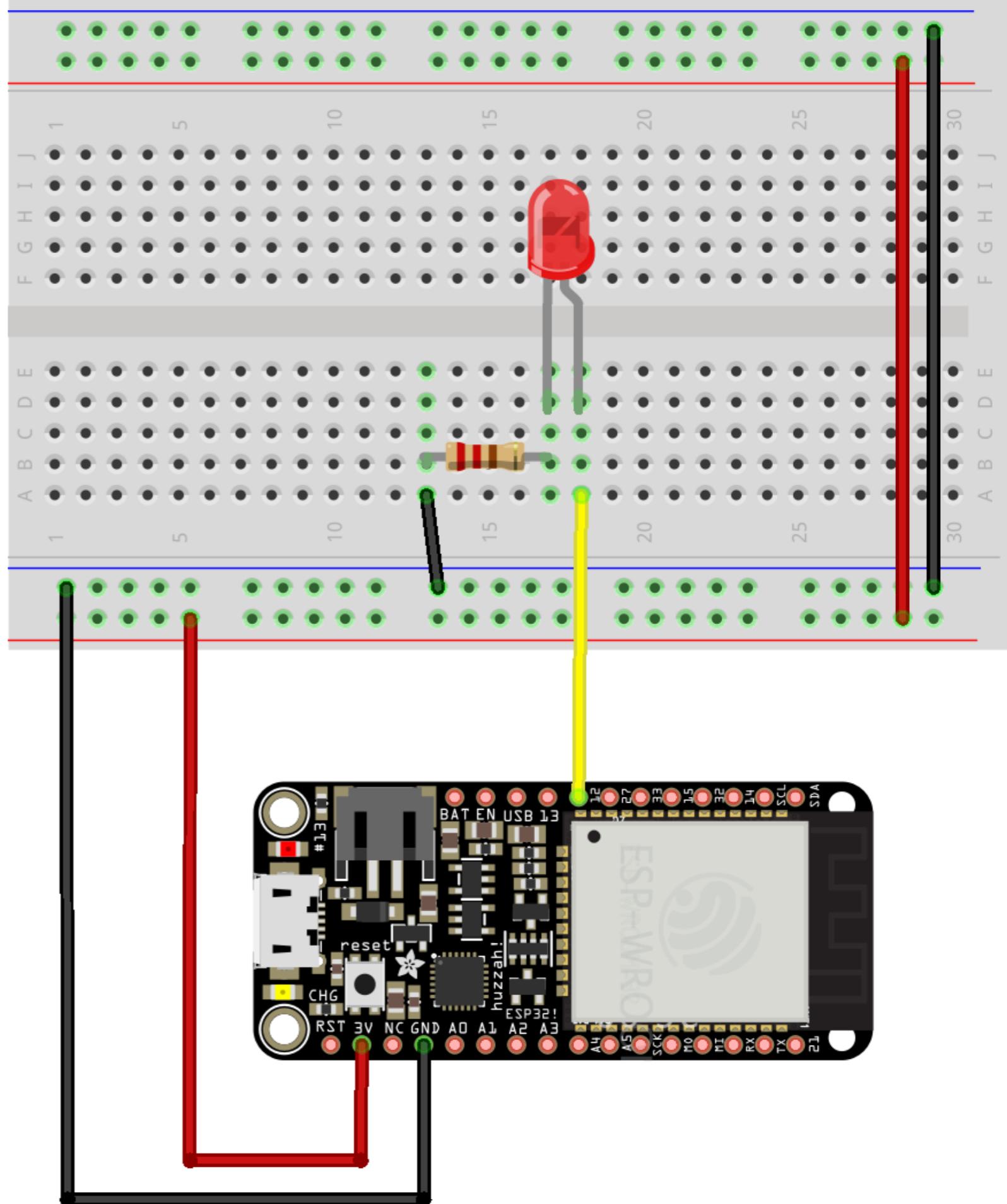
The code consists of two main sections: `setup()` and `loop()`. The `setup()` section is intended for one-time initialization code, and the `loop()` section is for repeated execution.

Premiers montages

1. Sorties Digitales

- `pinMode(pin#, OUTPUT);`
 - configure la direction de la broche digitale **pin#**
 - second paramètre **OUTPUT** (utilisé pour la sortie)
- `digitalWrite(pin#, HIGH/LOW);`
 - modifie l'état de la broche **pin#**
 - le second paramètre peut être soit **HIGH** soit **LOW**

Une LED



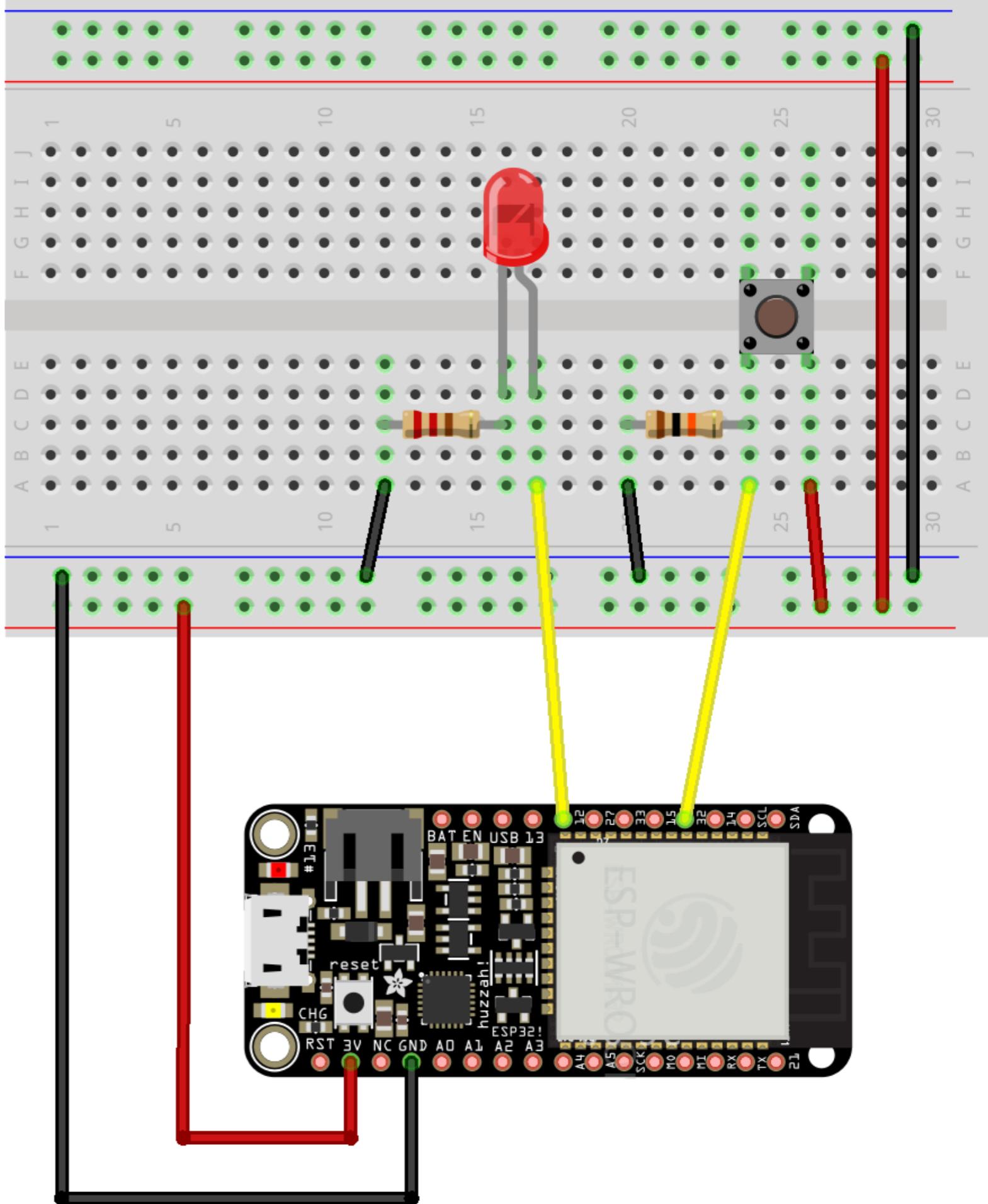
Exercices

- Faire clignoter la LED à des vitesses différentes

2. Entrées Digitales

- `pinMode(pin#, INPUT);`
 - configure la direction de la broche digitale **pin#**
 - le second paramètre peut être **INPUT**,
INPUT_PULLUP
- `digitalRead(pin#);`
 - lit l'état de la broche **pin#**

LED + bouton



fritzing

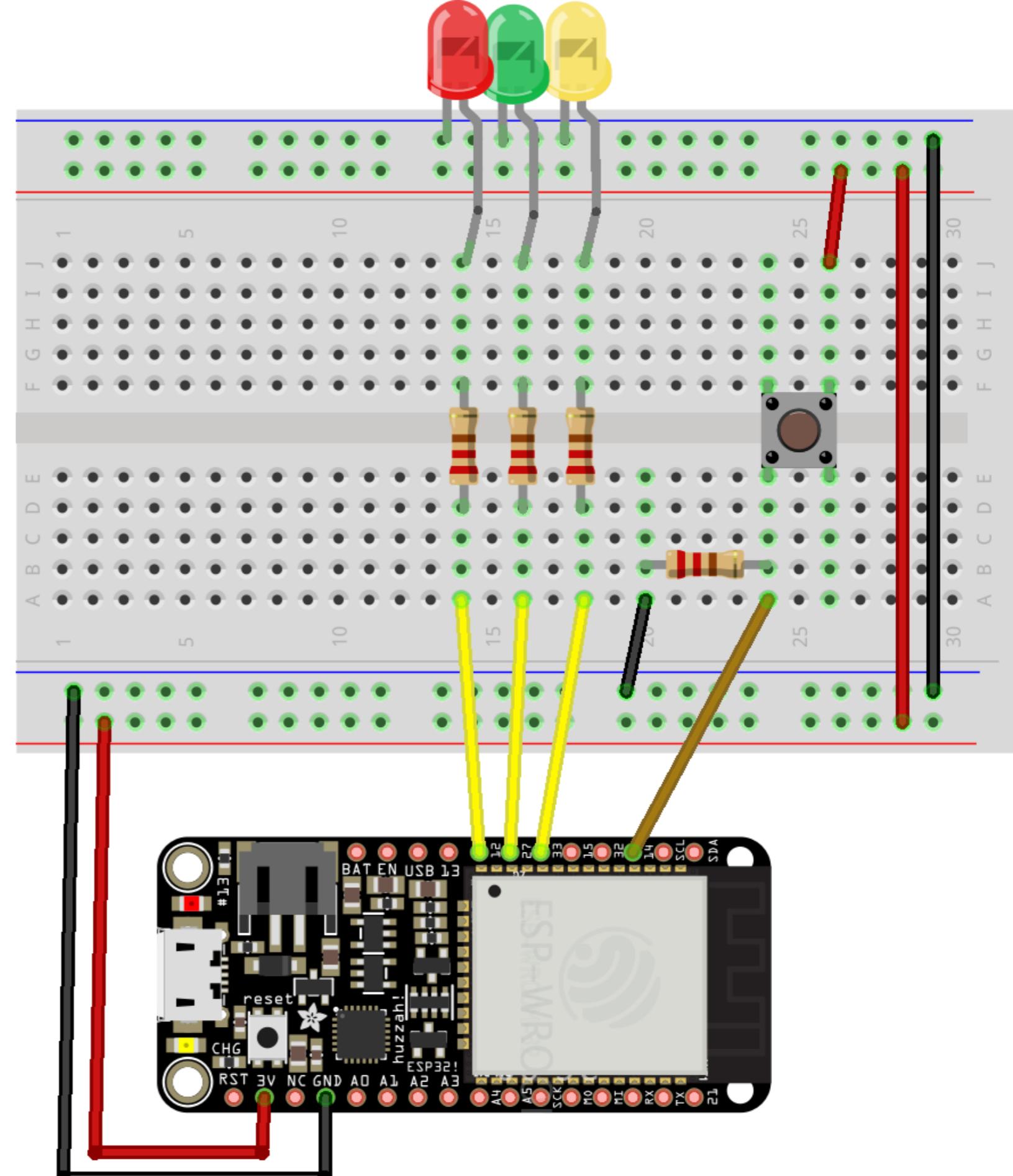
Exemples

- Allumer la LED en fonction de l'état du bouton
- Debounce (anti-rebond)
- Détection de changement d'état

Exercices

- Allumer la LED après 3 appuis sur le bouton
- Changer l'état de la LED au moment de l'appui sur le bouton. Quand le bouton est lâché, la led conserve son état.

Plusieurs LEDs



Exemples

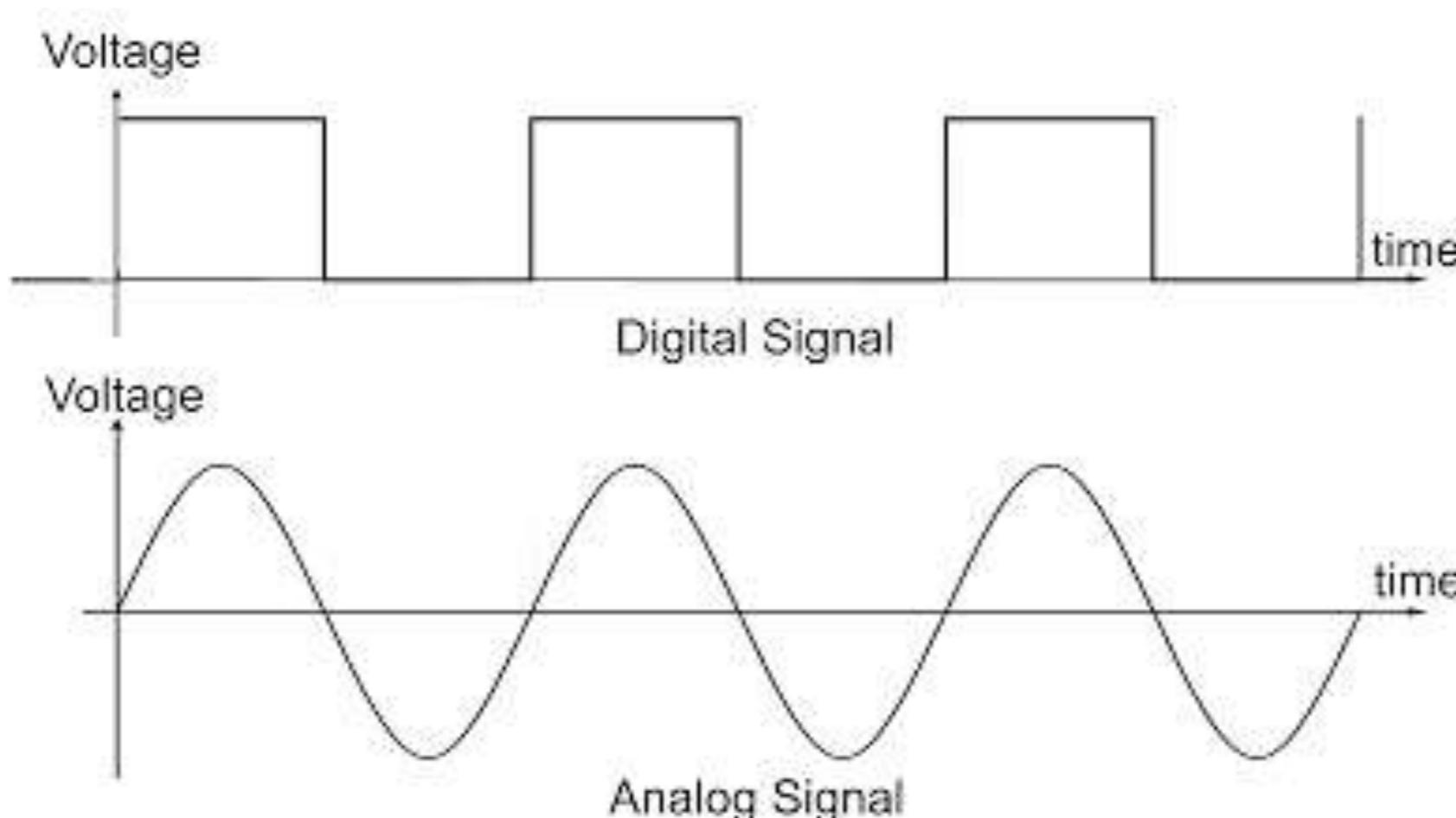
- Utiliser les boucles pour commander l'état de plusieurs LEDS

Exercices

- Utiliser les LEDS pour visualiser le nombre d'appuis sur un bouton

3. Entrées Analogiques

- Signal analogique vs Digital



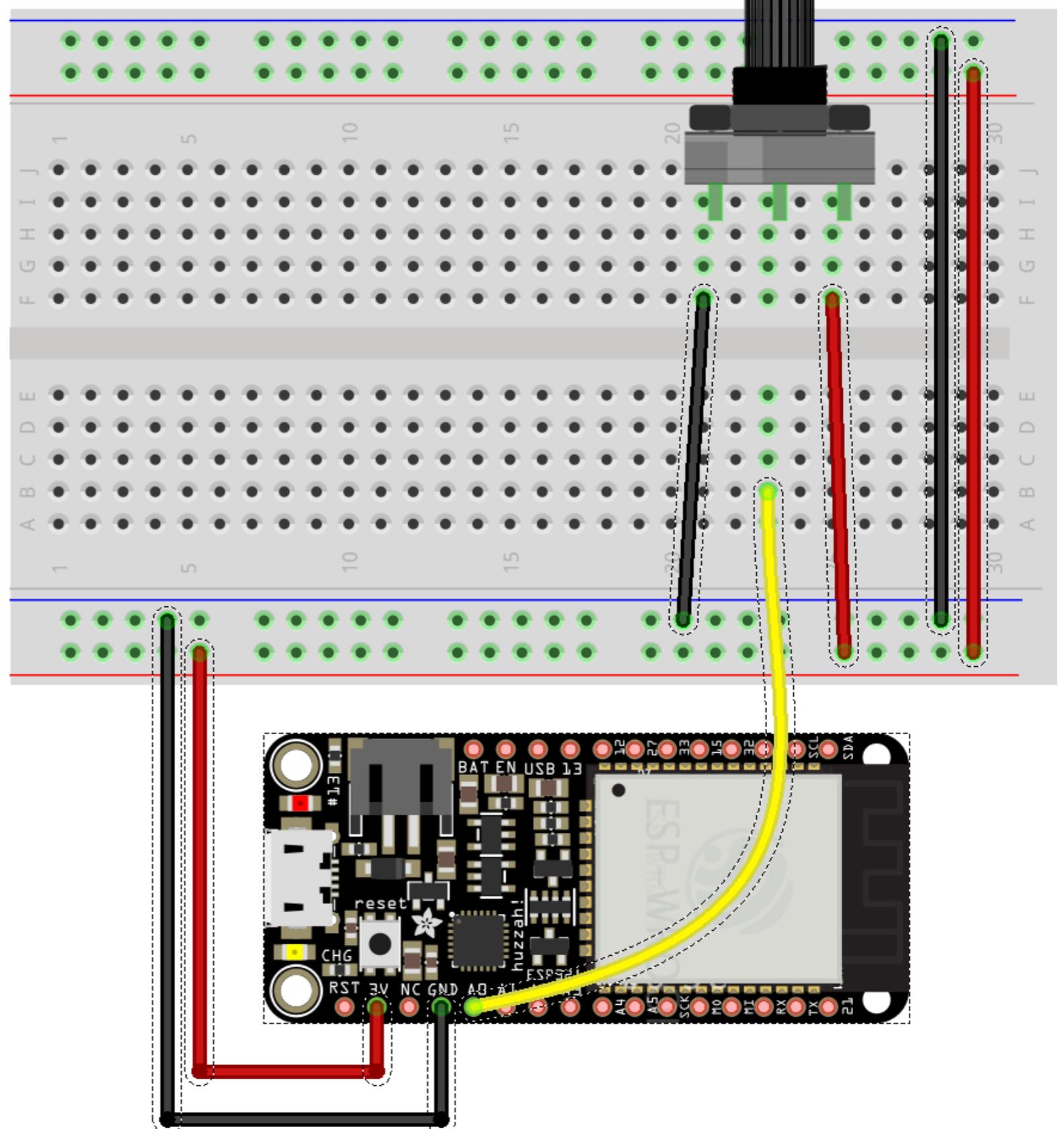
3. Entrées Analogiques

TODO

3. Entrées Analogiques

- **analogRead(pin#);**
 - lit l'état de la broche **pin#**
 - retourne une valeur entre 0 et 1024 (10 bits, par défaut)

Potentiomètre



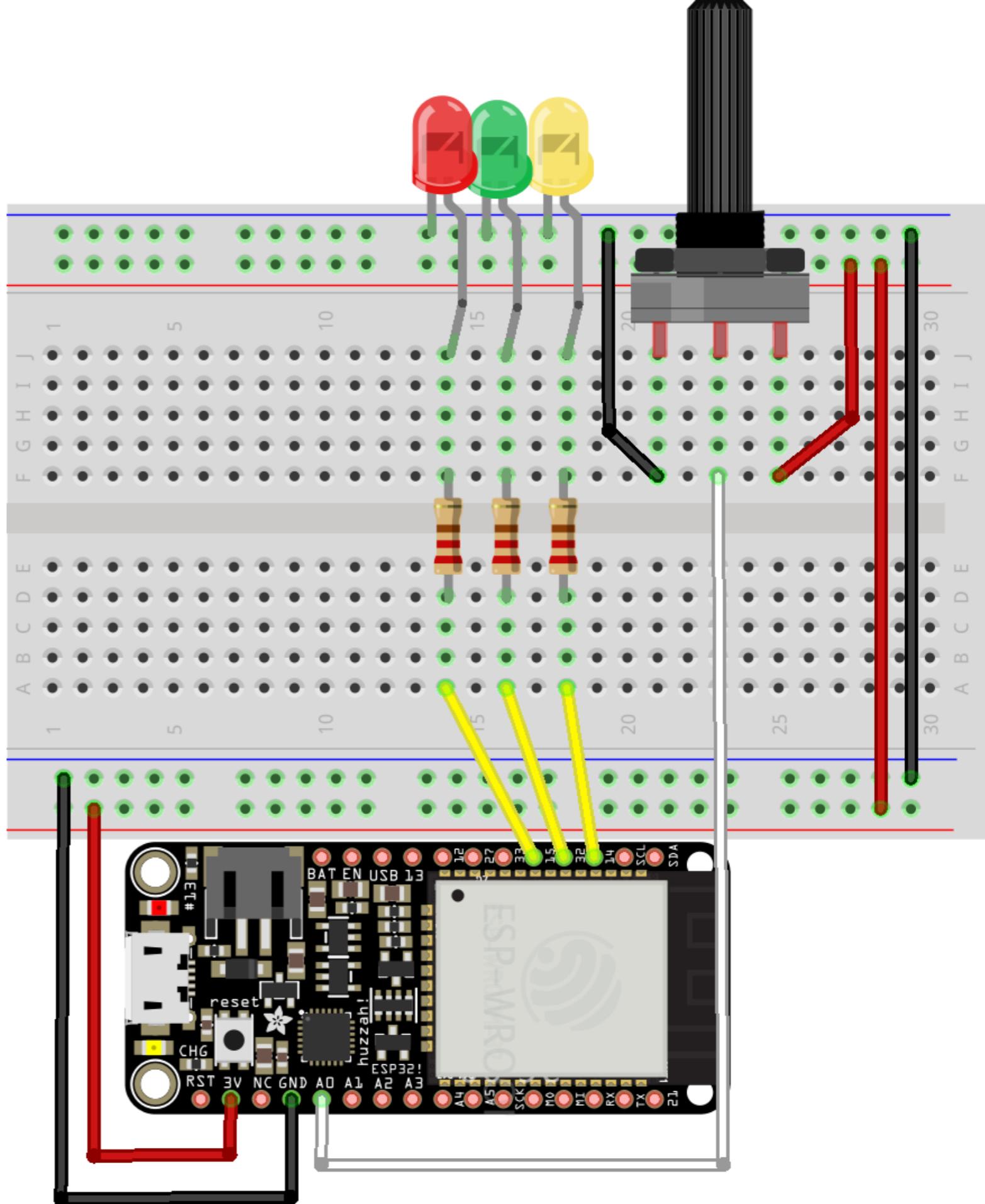
La librairie Serial

- Permet de faire communiquer la carte Arduino et l'ordinateur via l'USB
- dans **setup()** :
 - **Serial.begin(*rate*);**
 - initialise la communication à une vitesse de ***rate*** bits/seconde
- dans **loop()**
 - `Serial.println("texte ou variable");`
 - Affiche un texte ou la valeur d'une variable dans le moniteur série

Exercices

- Utiliser le potentiomètre pour faire varier la vitesse de clignotement d'une LED

Potentiomètre + LEDs

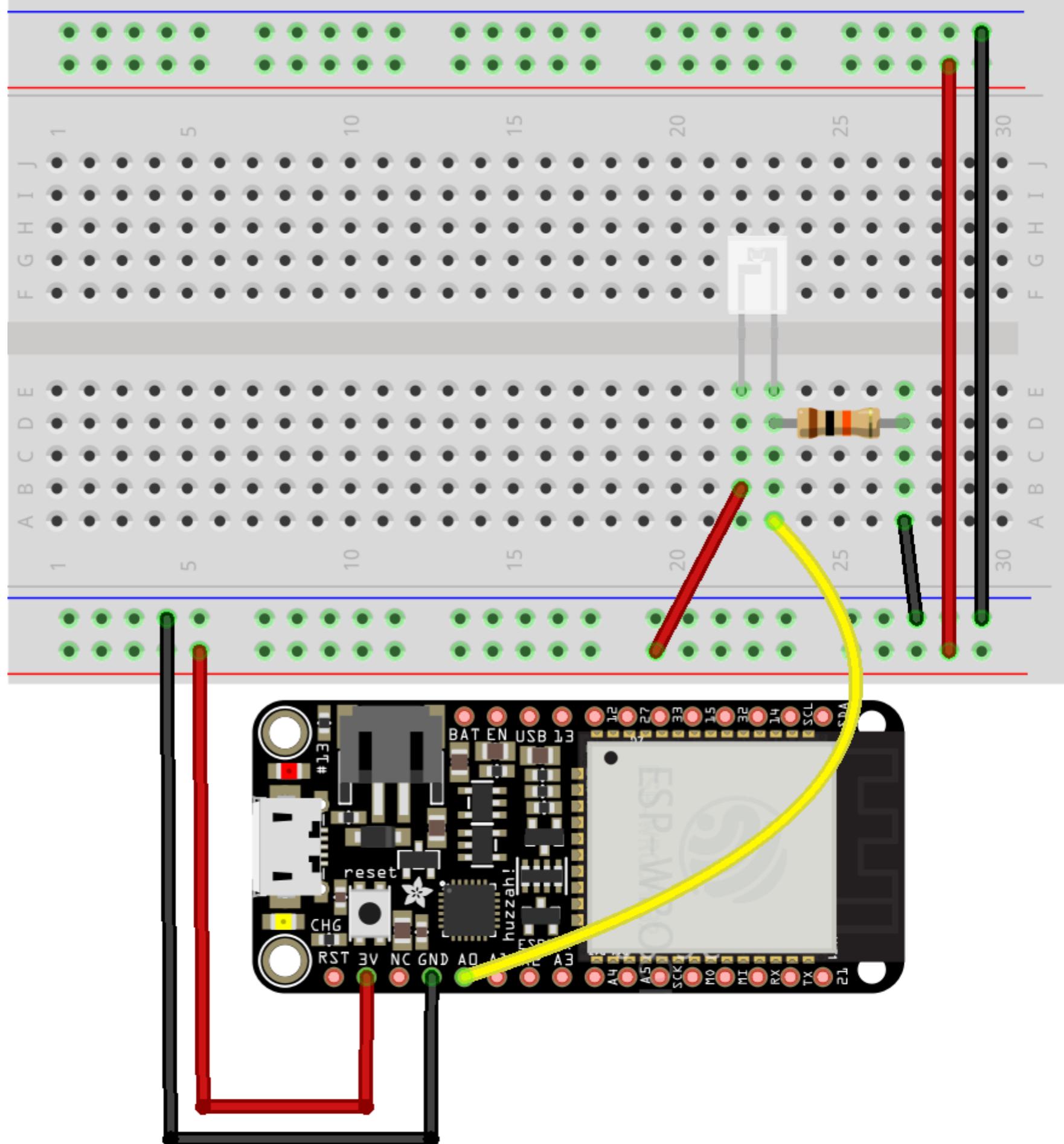


fritzing

Exercices

- Utiliser le potentiomètre pour allumer plus ou moins de LEDs

Phototransistor avec pont diviseur de tension



Exemples

- La fonction **map**
- La calibration