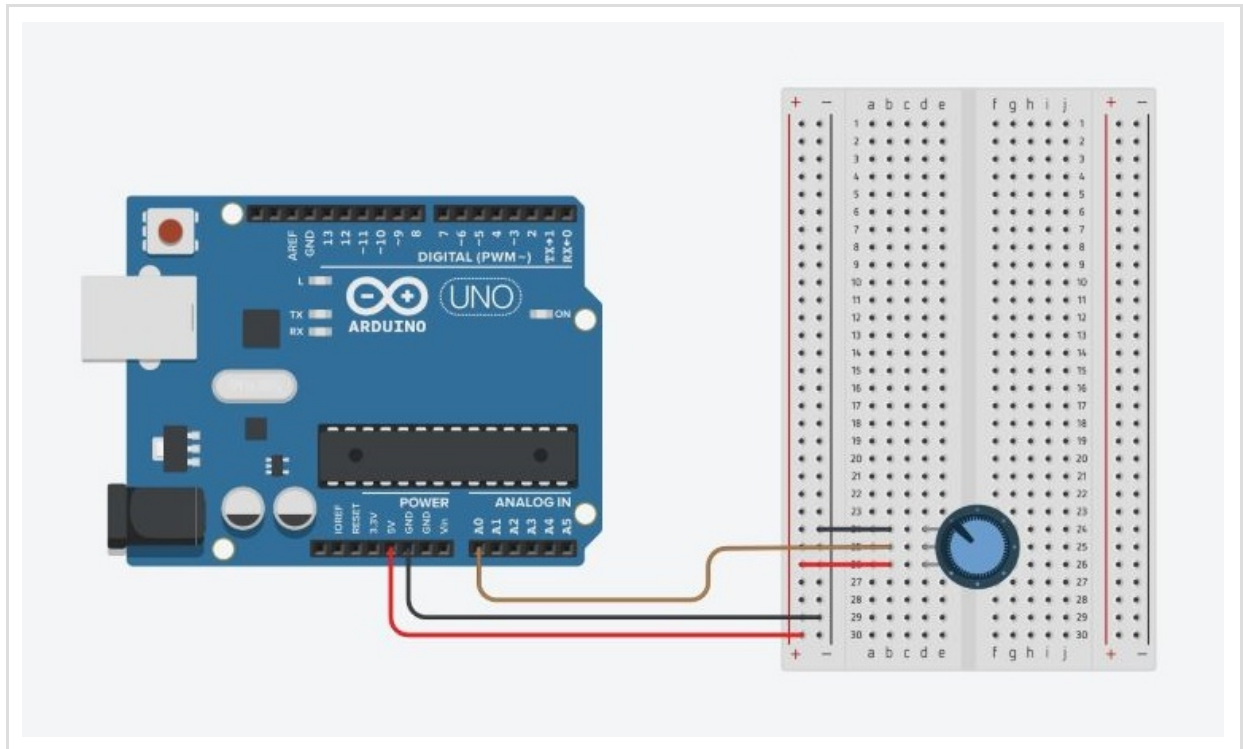


# Potentiomètre

## (La conversion analogique numérique)



### . Liste des composants :

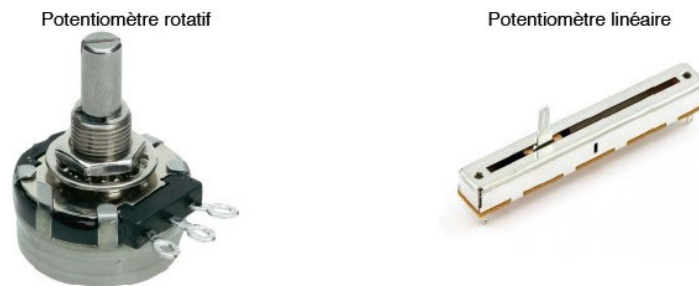
- . 1 potentiomètre de 10 k $\Omega$
- . 1 plaque d'essai
- . Fils de connexion

### . Objectif

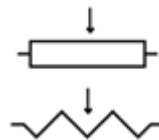
L'objectif de cette activité est l'apprentissage du principe de fonctionnement des entrées analogiques de l'Arduino et de l'utilisation du moniteur série.

## . Le potentiomètre

Le potentiomètre est une résistance variable. C'est le bouton de réglage du volume que l'on retrouve sur une radio. La plupart des potentiomètres sont soit rotatifs, soit linéaires.

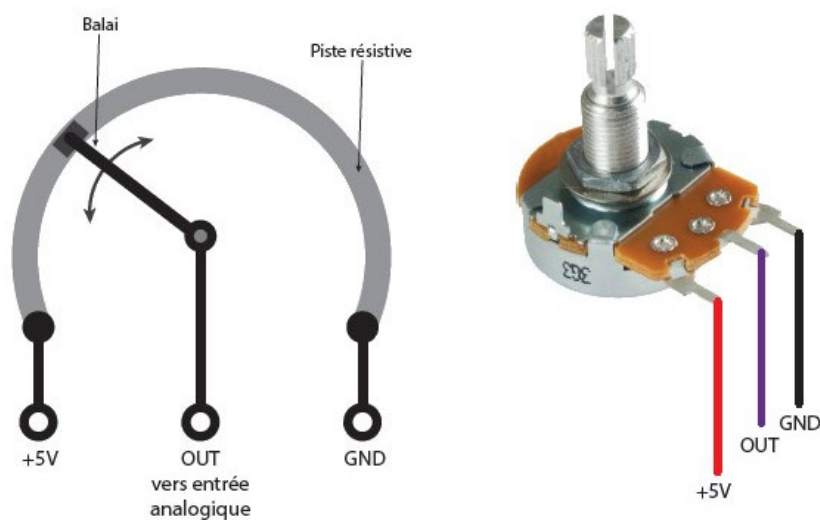


Voici les symboles électroniques (européen dessus et américain dessous) du potentiomètre :



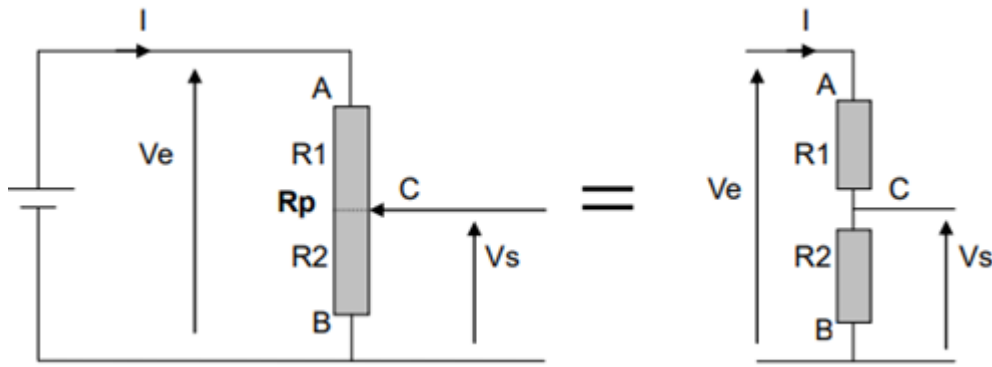
Comme toute résistance, le potentiomètre modifie la tension d'un circuit. On va donc l'utiliser principalement comme entrée (input) sur une broche analogique (A0 à A5) de l'Arduino.

Les potentiomètres ont en général trois broches :



Les broches extérieures se connectent sur l'alimentation +5V et sur la masse, alors que la broche centrale envoie le signal sur une broche d'entrée analogique de l'Arduino, comme sur le circuit ci-dessus (broche A0).

Dans ce montage, le potentiomètre de résistance totale  $R_p$  est utilisé en pont diviseur de tension :



On a :  $V_e = (R_1 + R_2) I$

$$I = \frac{V_e}{(R_1 + R_2)}$$

$$\text{Et : } V_s = R_2 I = \frac{R_2}{(R_1 + R_2)} V_e = \frac{R_2}{R_p} V_e \quad (\text{car } R_p = R_1 + R_2)$$

On peut remplacer le rapport  $R_2 / R_p$  par la position du curseur comprise entre 0 (position B) et 1 (position A).

Dans ce cas, la relation devient :

$$V_s = \alpha V_e \quad (\text{avec } \alpha \text{ la position du curseur : } 0 \leq \alpha \leq 1)$$

$V_s$ , qui est la tension appliquée sur une entrée analogique de l'Arduino, varie donc entre 0 et +5V en fonction de la position du curseur du potentiomètre.

De façon à limiter le courant dans le circuit à 0,5 mA, on utilise généralement des potentiomètres de **10 kΩ**.

## . Gestion des entrées analogiques

Pour lire la valeur de la tension d'une entrée analogique, on utilise la fonction :

**analogRead()**

. Syntaxe :

**analogRead(broche\_analogique)**

. Paramètres :

**broche\_analogique** : le numéro de la broche sur laquelle il faut convertir la tension analogique appliquée (0 à 5 sur la plupart des cartes Arduino)

. Valeur retournée :

valeur int (0 à 1023) correspondant au résultat de la mesure effectuée

. Remarque :

Il n'est pas nécessaire de déclarer les broches A0 à A5 en entrée avec la fonction **pinMode()** pour lire la tension appliquée sur celles-ci.

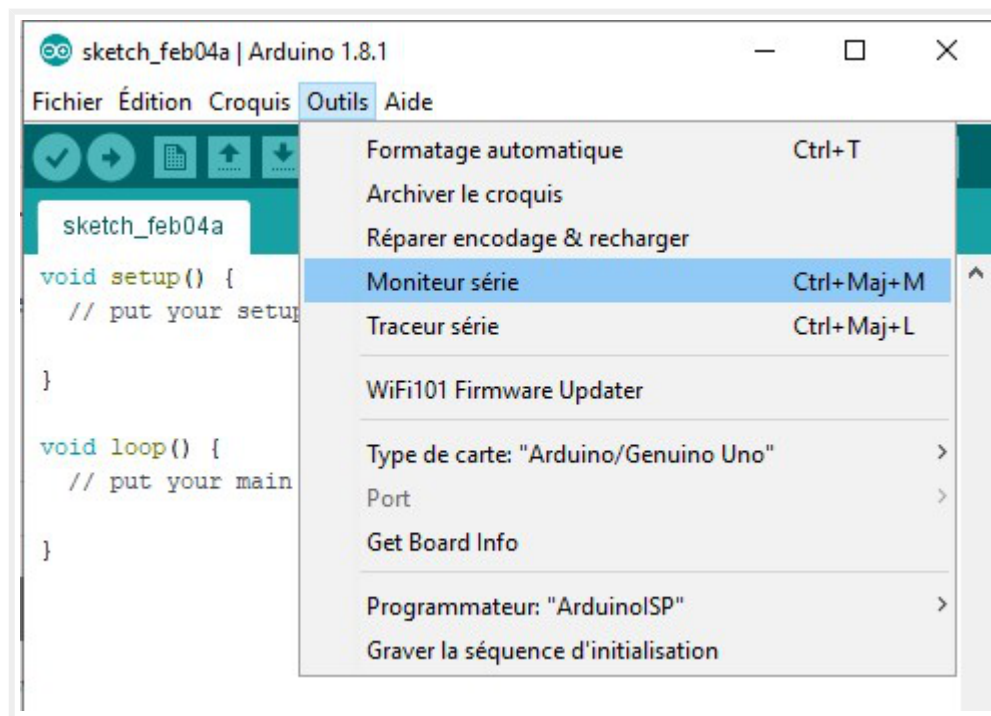
## . Le moniteur Série

Le logiciel Arduino IDE dispose d'un moniteur série, qui permet de recevoir et d'envoyer des informations via une liaison série. Il est particulièrement intéressant pour les tests des programmes puisqu'il permet d'afficher les valeurs des variables, les états logiques des entrées et des sorties de l'Arduino, etc...

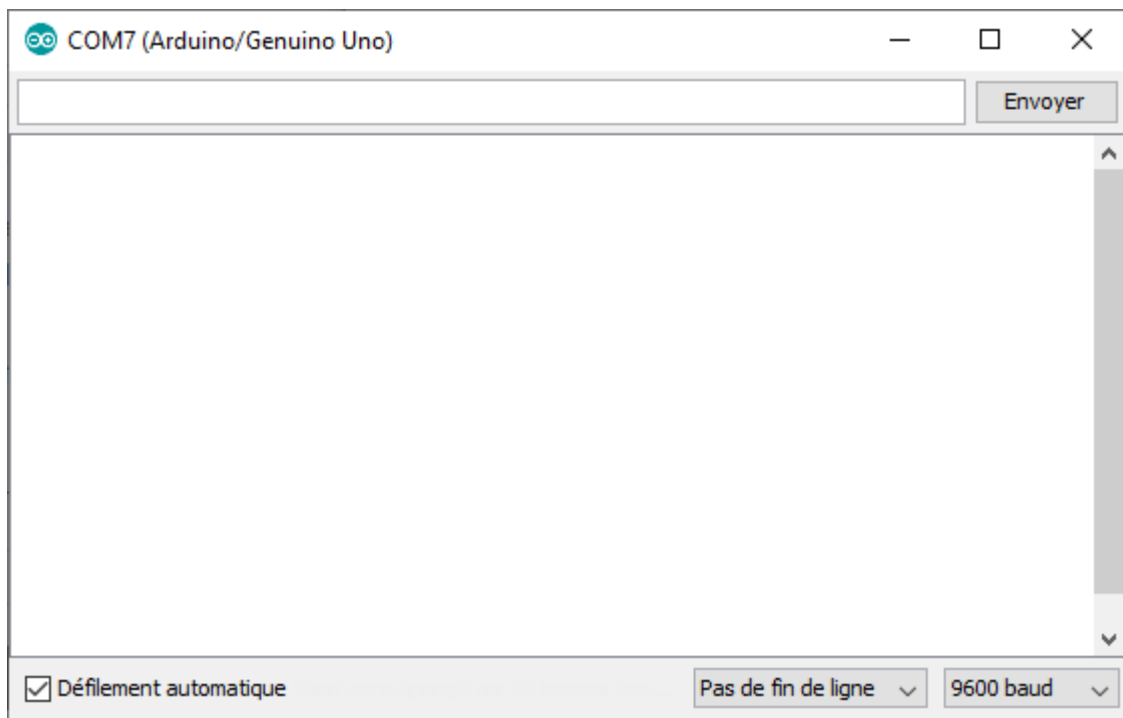
Le moniteur série est accessible depuis le logiciel Arduino en cliquant sur la loupe en haut à droite de la fenêtre du logiciel:



Ou à partir du menu **Outils/Moniteur série** :



Une nouvelle fenêtre s'ouvre alors :



La fenêtre est composée de deux zones blanches d'exploitation et de plusieurs commandes :

- la grande zone au centre est la zone d'affichage des données reçues par le moniteur et donc envoyées par l'Arduino,
- la petite zone rectangulaire, en haut est une zone de saisie des données à envoyer à l'Arduino à l'aide du bouton « Envoyer » qui se trouve à sa droite. C'est le programme du microcontrôleur qui génère la demande d'envoi de données par l'utilisateur et qui s'occupe de traiter les informations reçues,
- la case cochée « Défilement automatique » permet d'arrêter le défilement des données retournées par l'Arduino si on la décoche,
- une liste déroulante permettant le réglage du mode de défilement des informations (Pas de fin de ligne, Nouvelle ligne, ...),
- une liste déroulante permettant de régler le débit de transmission des données par le port série en bauds.

### **Remarques :**

- Le baud est une unité de mesure utilisée dans le domaine des télécommunications en général, et dans le domaine informatique en particulier. Le baud est l'unité de mesure du nombre de symboles transmissibles par seconde.

Le terme « baud » provient du patronyme d'Émile Baudot, l'inventeur du code Baudot utilisé en télégraphie.

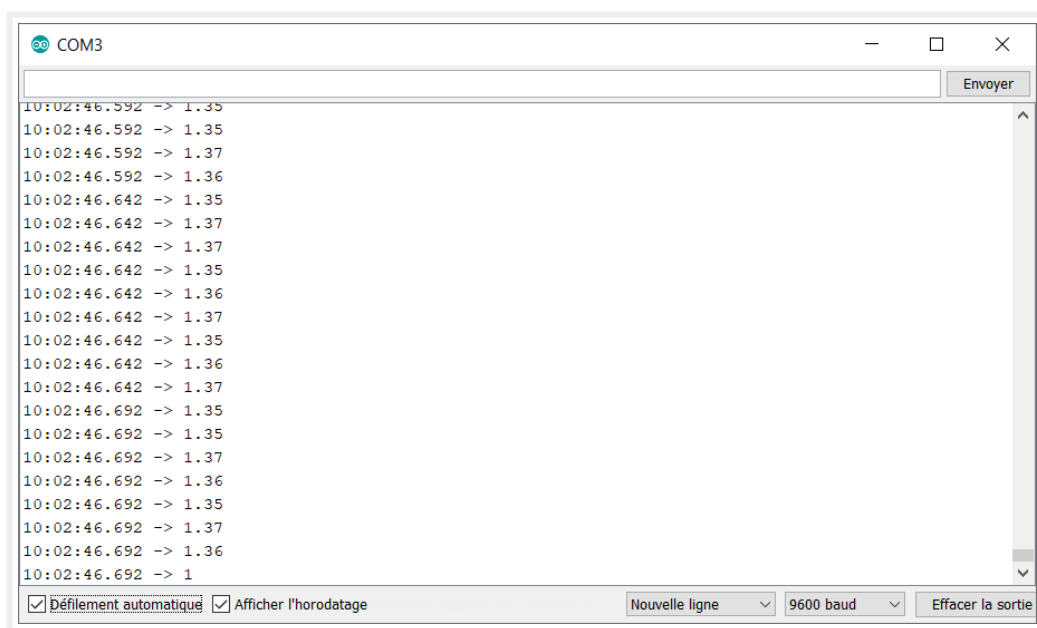
Il ne faut pas confondre le baud avec le bps ou bit par seconde, ce dernier étant l'unité de mesure du nombre d'information effectivement transmise par seconde. Il est en effet souvent possible de transmettre plusieurs bits par symbole. La mesure en bps de la vitesse de transmission est alors supérieure à la mesure en baud.

- La valeur par défaut est de 9600 bauds. La carte Arduino peut monter jusqu'à une cadence de 115200 bauds. Mais un débit de communication de 9600 caractères (chaque caractère étant codé sur 8 bits, soit 1 octet) par seconde (**9600 bauds**) est généralement suffisant.

- Dans les dernières versions du logiciel **Arduino IDE**, deux nouvelles commandes ont été ajoutées au moniteur série :

. "Afficher l'horodatage " (affiche l'heure de la transmission des données)

. "Effacer la sortie " (efface la zone d'affichage des données reçues par le moniteur série)



Pour utiliser le moniteur série, le programme téléversé dans la mémoire de l'Arduino doit établir la liaison série.

Pour cela, dans la fonction `Setup()` du programme, on utilise la fonction **`begin()`** de la classe « **Serial** » :

```
void setup() {  
  Serial.begin(9600);  
}
```

Le port série de la carte Arduino est alors configuré à 9600 bauds.

### **Important :**

- Le microcontrôleur et le moniteur série doivent être configurés sur le même débit de transmission.
- Le programme du microcontrôleur est réinitialisé à l'ouverture du moniteur série.

La classe « **Serial** » regroupe toutes les fonctions utiles à la gestion d'un port série. Son travail est de gérer le traitement des données d'une communication série entre le moniteur série et le périphérique Arduino.

### Les principales fonctions de la classe Serial :

- . **begin();** (Configure la vitesse de transmission du port série)
- . **print();** (Envoie une donnée sous forme de chaîne de caractères sur le port série)
- . **println();** (Envoie une donnée sur le port série et fait un saut à la ligne)
- . **write();** (Ecrit des données binaires sur le port série. Ces données sont envoyées comme une série d'octets)
- . **read();** (Lis les données contenues dans la mémoire tampon ou « buffer » du port série)
- . **flush();** (Vide la mémoire tampon de la liaison série)
- . **available();** (Donne le nombre d'octets ou caractères disponible pour lecture dans la file d'attente ou « buffer » du port série)

Une fois la liaison série établie, il est possible d'envoyer des données depuis la carte Arduino vers le moniteur série avec la fonction "**print()**" de la classe "**Serial**".

On peut envoyer différents types d'informations :

. Envoie d'une chaîne de caractères, sans saut de ligne à la fin :

```
Serial.print("Test");
```

. Envoie d'une chaîne de caractères, avec saut de ligne à la fin :

```
Serial.println("Test");
```

. Envoie de la valeur d'une variable, sans saut de ligne à la fin :

```
int a = 3;
```

```
Serial.print(a);
```

### . [Le programme](#)

Dans [le programme de cette activité](#), on demande à l'Arduino de lire la valeur de l'entrée analogique A0 et d'afficher le résultat en numérique et en analogique dans le moniteur série si la valeur lue est différente de la précédente (écart de plus d'une unité entre les valeurs lues)

## Potentiometre

```
// Déclaration des constantes et variables

const int PinPOT = 0;
int ValPot = 0;
int OldValPot =0;
float Tension=0.00;

// Initialisation des entrées et sorties

void setup() {
  Serial.begin(9600);
  Serial.println("Valeur A0 ; Tension (V):");
}

// Fonction principale en boucle

void loop() {
  ValPot = analogRead(PinPOT);
  if (abs(ValPot - OldValPot)>=2) {
    Serial.print(ValPot);
    Serial.print(" ; ");
    Tension=ValPot*5.00/1023;
    Serial.println(Tension);
    OldValPot = ValPot;
  }
  delay(100);
}
```

### Déroulement du programme :

– Déclaration des constantes et variables :

- . **const int PinPOT = 0** (constante nombre entier correspondant au n° de la broche sur laquelle le potentiomètre est connecté)
- . **int ValPot = 0** (variable nombre entier pour stocker la valeur de la broche du potentiomètre)
- . **int OldValPot = 0** (variable nombre entier pour stocker la valeur précédente de la broche du potentiomètre)
- . **float Tension = 0.00** (variable nombre décimal pour le calcul de la tension en V appliquée sur la broche du potentiomètre)

– Initialisation des entrées et sorties :

- . La liaison série est initialisée à 9600 bauds : **Serial.begin(9600)**

– Fonction principale en boucle :



- . Lecture de la valeur de la tension de l'entrée analogique du potentiomètre : **ValPot = analogRead(PinPot)**
- . Affichage de la valeur lue dans le moniteur série si celle-ci est différente de la valeur précédente
- . Calcul et affichage de la tension appliquée sur A0 en V
- . Stockage de la valeur lue dans la variable OldValPot
- . Attente de 100 ms avant une nouvelle mesure

**Résultat dans le moniteur série :**

