Programmer une carte Arduino

C++

Il existe de nombreux **langages de programmation** (C, Python, JavaScript). Pour programmer une carte Arduino, on utilise le **C++**.

Ce langage nécessite une étape de **compilation** : transformer la syntaxe compréhensible par un humain, écrite en C++, en une syntaxe compréhensible par le processeur, en langage binaire (suite de 0 et 1). On appelle "**exécutable**", le résultat de la compilation.

Pour cela, on utilise l'**IDE Arduino** : un logiciel qui permet d'écrire du code, de le vérifier et le compiler, et de le transférer sur une carte Arduino.

Un programme Arduino se compose de 2 fonctions principales : setup() et loop().

setup()

Sert à **initialiser les broches** utilisées comme étant des **entrées** ou des **sorties** dans le programme.

Ici, on utilise en particulier la fonction **pinMode()**.

loop()

Ce code **tourne en boucle** quand la carte reçoit du courant.
C'est ici que l'on écrit **l'algorithme** : **la logique du programme**, la suite d'instructions que doit suivre le processeur pour réaliser les tâches que l'on souhaite.

On y utilise, par exemple, les fonctions **digitalWrite()**, **digitalRead()**, **analogWrite()**, **analogRead()**, **delay()**.

Exemple : faire clignoter une LED

La LED est branchée sur la broche numérique n° 4. Elle s'allume pendant 1 seconde puis s'éteint pendant 1 seconde.

```
/*
    Commentaire sur plusieurs ligne
    que l'on peut supprimer
*/

void setup() {
    pinMode(4, OUTPUT);   // LED branchée en D4
}

void loop() {
    digitalWrite(4, HIGH); // Allume la LED
    delay(1000);           // Attend 1000 ms = 1sec
    digitalWrite(4, LOW); // Éteint la LED
    delay(1000);           // Attend 1000 ms
}
```

A savoir

Pour rendre le code compréhensible par quelqu'un d'autre, on peut y ajouter des commentaires qui ne seront pas visible par le processeur.

Penser à bien écrire son code à l'intérieur des accolades { } des fonctions loop() et setup().

Ne pas modifier ni supprimer les lignes en gras sur l'exemple. Le programme ne fonctionnerait plus.

Les fonctions et valeurs à connaître

Dans setup()

pinMode(pinNb, mode)
Déclarer le mode de partage d'information d'un composant.

pinNb = le numéro de la broche où est branché le composant
mode = **INPUT** ou **OUTPUT**

```
ex. : pinMode(4, OUTPUT);
```

INPUT : La broche sert d'entrée, le processeur reçoit un signal du composant (**lecture**).

> Permet d'utiliser la fonction digitalRead() dans loop().

ex : un bouton dont on voudra savoir s'il est appuyé ou non

OUTPUT : La broche sert de sortie, le processeur envoie un signal au composant (**écriture**).

> Permet d'utiliser la fonction digitalWrite() ou analogWrite() selon la broche, dans loop().

ex : une LED que l'on voudra allumer ou éteindre

Dans loop()

digitalWrite(pinNb, valeur)
Envoyer un signal binaire au composant.

> S'utilise avec les broches numériques 0 à 13, mais aussi avec les broches analogiques A0 à A5. Ces broches doivent être définies comme OUTPUT dans setup().

pinNb *= le numéro de la broche numérique où est branchée le composant ;*
valeur *= valeur binaire **HIGH** ou **LOW***

```
ex. : digitalWrite(4, HIGH);
```

digitalRead(pinNb)
Lire le signal binaire émis par un composant.

La fonction renvoie une valeur binaire telle que **HIGH** ou **LOW**.

> S'utilise avec les broches numériques 0 à 13, mais aussi avec les broches analogiques A0 à A5. Ces broches doivent être définies comme INPUT dans setup().

```
ex. : int mode = digitalRead(7);
```

HIGH : la tension est au maximum (5 volts)
LOW : la tension est nulle (0 volts)

Penser à mettre un point virgule à la fin de chaque instruction.

analogWrite(pinNb, valeur)
Envoyer un signal analogique au composant.

> S'utilise avec les broches numériques ayant le symbole PWM~ (3, 5, 6, 9, 10, 11). Ces broches doivent avoir été définies comme OUTPUT dans setup().

pinNb = le numéro de la broche analogique où est branchée le composant ;
valeur = nombre entier compris **entre 0 et 255**

```
ex. : analogWrite(5, 128);
```

analogRead(pinNb)
Lire le signal analogique émis par un composant.

La fonction renvoie un nombre entier compris **entre 0 et 1023**.

> S'utilise avec les broches analogiques A0 à A5.

```
ex. : int valeur = analogRead(A2);
```

Autre fonction utile :

delay(durée)
Met en pause le programme pendant une certaine durée.

durée = temps de pause exprimé en milliseconde (1 seconde = 1000 millisecondes)