Arduino PR 11

Les fonctions

Présentation

Nous allons un peu transformer le programme Blink pur qu'il soit plus facile à utiliser pour nous. Nous allons écrire nos propres fonctions.

Une **fonction** en informatique c'est une suite d'instructions qui réalise un calcul ou une tâche.

Les fonctions permettent de regrouper des instructions que l'on utilise souvent et de les réutiliser plus facilement. Elles permettent aussi de découper un problème compliqué en éléments plus simples.

Nous avons déjà utilisé des fonctions :

- `pinMode(13, OUTPUT)`
- `digitalWrite(13, HIGH)`
- `delay(1000)` attend 1s (1000 ms)

La définition d'une fonction comporte des **paramètres d'entrée**, par exemple 13 ou HIGH. La fonction va les utiliser pour réaliser sa tâche.

Elle peut retourner une **valeur de sortie.**

Par exemple si j'écris

```
int produit = multiplie(2, 3)
```

`produit` aura la valeur 6 après l'appel de la fonction.

Le mot qui est devant `produit` s'appelle le **type** et `produit` s'appelle une **variable**. On ne connait pas ça valeur. Ce sera le résultat de l'exécution de la fonction.

Il existe plusieurs types : `int` indique que c'est un nombre entier. Il existe d'autres type comme `float` pour les nombre décimaux, ou `char` pour les caractères.

Les types sont aussi nécessaires pour la définition de la fonction.
Par exemple, la fonction qui multiplie deux nombres entiers s'écrirait comme ça :

```
int multiplie(int a, int b) {
    return a * b;
}
```

Tu peux maintenant écrire des fonctions plus simples à utiliser comme allumeLED(), eteintLED et attend pour éviter de calculer les milli-secondes.

```
// the loop function runs over and over again forever
void loop() {
  allumeLED();
  attend(1);               // wait for a second
  eteintLED();
  attend(1);               // wait for a second
}

void allumeLED() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
}

void eteintLED() {
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
}

void attend(int temps) {
  delay(temps * 1000);    // attends le nombre de secondes demandees
}
```