Librairie logicielle

Librairies, ça vient de l'anglais "Libraries", qui signifie **Bibliothèques**. L'idée est d'avoir sous la main un rassemblement de morceaux de code, classés par thématiques, que vous invoquez à la demande.

Les librairies pour Arduino sont nombreuses et abordent la plupart des besoins courants. On trouve ainsi les bibliothèques standards, pour par exemple gérer le Wifi, les écrans à cristaux liquide, utiliser simplement une carte SD, ou encore des moteurs. Pour l'occasion, nous allons nous intéresser à des moteurs un peu particulier que l'on retrouve dans le monde du modélisme : les servo-moteurs.

Ouvrons l'exemple « sweep » que l'on trouve dans les exemples du dossier *Servo* : *Fichier → Exemples → Servo> → Sweep*.

Détaillons maintenant les nouvelles instructions présentes dans ce programme.

Après quelques lignes de commentaires, nous trouvons une instruction particulière :

```
#include<Servo.h>
```

Voilà, vous venez de charger la bibliothèque et obtenez du même coup la boîte à outils correspondante.

À partir de là, vous pouvez créer des objets en partant du « moule » `Servo`.

Voilà un servomoteur, que nous appelons ici `myservo`.

```
Servo myservo;
```

On déclare ensuite une variable `pos`, pour stocker une position au cours du programme :

```
int pos = 0;
```

Il est temps de passer au bloc setup. Il suffit d'attacher notre servo fraîchement créé à la broche 9 :

```
myservo.attach(9);
```

La méthode `attach()` est disponible pour les objets de type Servo. La bibliothèque de code gère le reste pour vous.

Les méthodes disponibles sur les objets de type Servo sont assez explicites :

`attach()`, `write()`, `writeMicroseconds()`, `read()`, `attached()`, `detach()`.

La **boucle principale** commence par... faire une boucle !

```
For (int  pos = 0; pos < 180; pos +=1) {
  myservo.write(pos);
  delay(15);
  }
```

Cette “boucle” à pour but de faire varier la position cible demandée au moteur. Autrement dit, elle sert uniquement à faire varier la valeur de `pos` du minimum au maximum. En français, cela donne

quelque chose comme « Pour une valeur `pos`ition allant de 0 à 179 et une marche à la fois, demande au servo `myservo` d'aller en position `pos` »

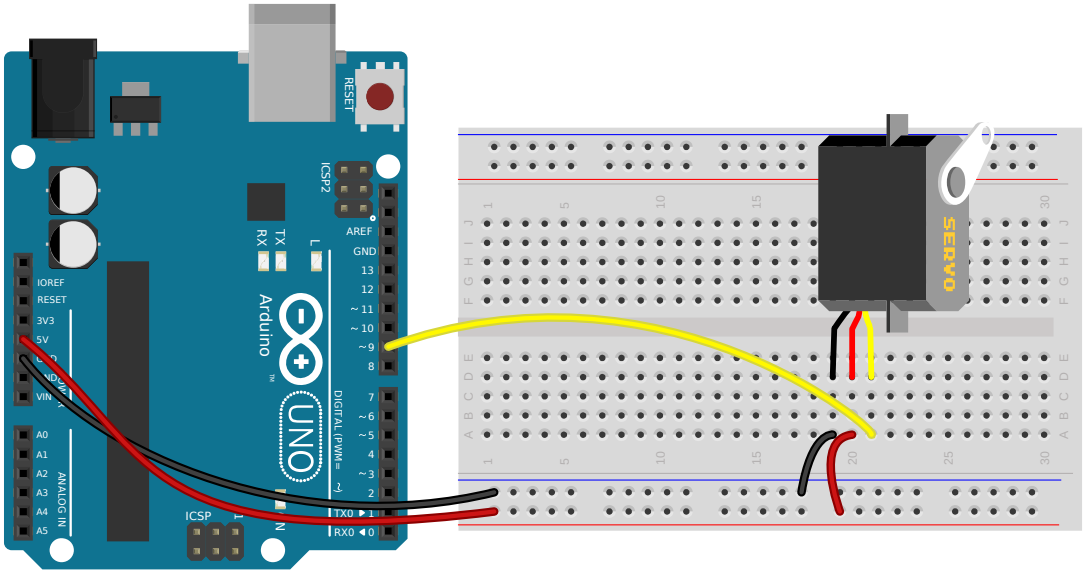
Un `pos += 4` vous aurait fait monter l'escalier 4 à 4

On laisse quelques millisecondes au moteur afin qu'il ait le temps d'aller une marche plus loin, concrètement de tourner de 1°.

C'est reparti pour un tour, cette fois-ci dans l'autre sens. Au lieu d'incrémenter la valeur de +1 à chaque passage de bouclette, on la décrémente ici de -1 :

```
for (int pos = 180; pos >= 1; pos -= 1) {
  myservo.write(pos);
  delay(15);
  }
```

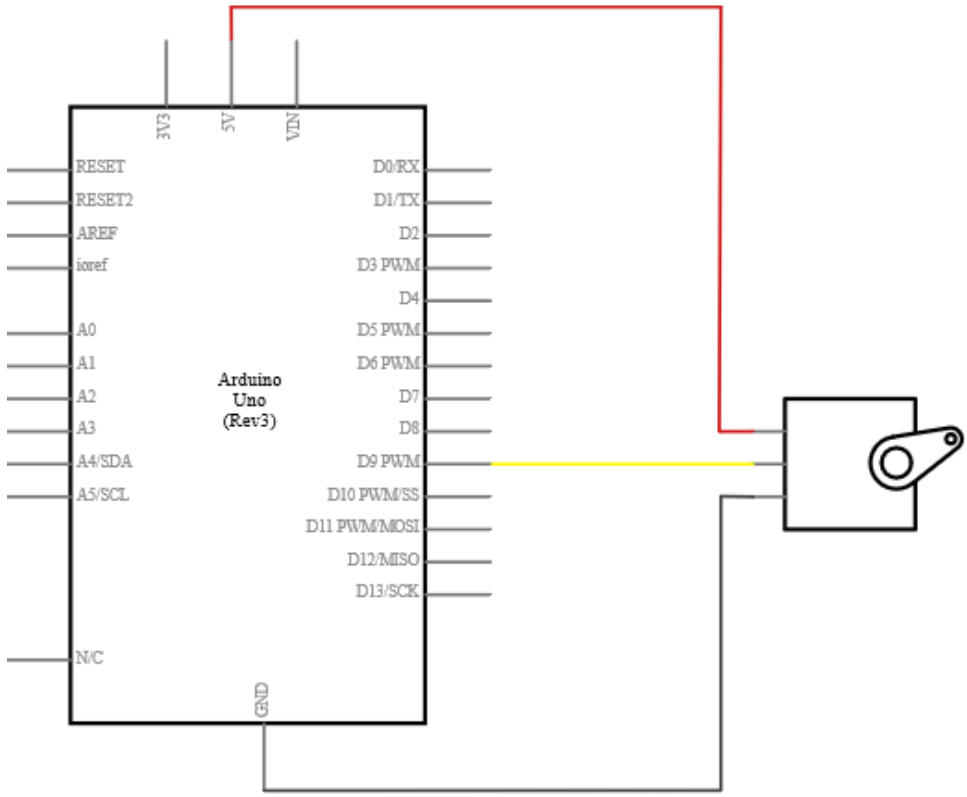
Montage



Pour le réaliser, vous aurez besoin de :

- Un Arduino
- Un câble USB
- Des fils de prototypage
- Une platine de prototypage
- Un servomoteur

Schéma



Code

```
#include <Servo.h>

const int servomoteur = 9;

Servo myservo;  // créer un objet appelé myservo à partir du module Servo

void setup()
{
  // attacher notre objet myservo au servomoteur branché sur la broche 9
  myservo.attach(servomoteur);
}

void loop()
{
  for (int pos = 0; pos < 180; pos += 1) {  // aller de 0° à 180°
    myservo.write(pos); // aller à la position stocké dans 'pos'
    delay(15);          // attendre 15ms que le servomoteur se rende à 'pos'
    }

  for(int pos = 180; pos >=1 ; pos -= 1) {  // aller de 180° à 0°
    myservo.write(pos); // aller à la position stocké dans 'pos'
    delay(15);          // attendre 15ms que le servomoteur se rende à 'pos'
    }
}
```