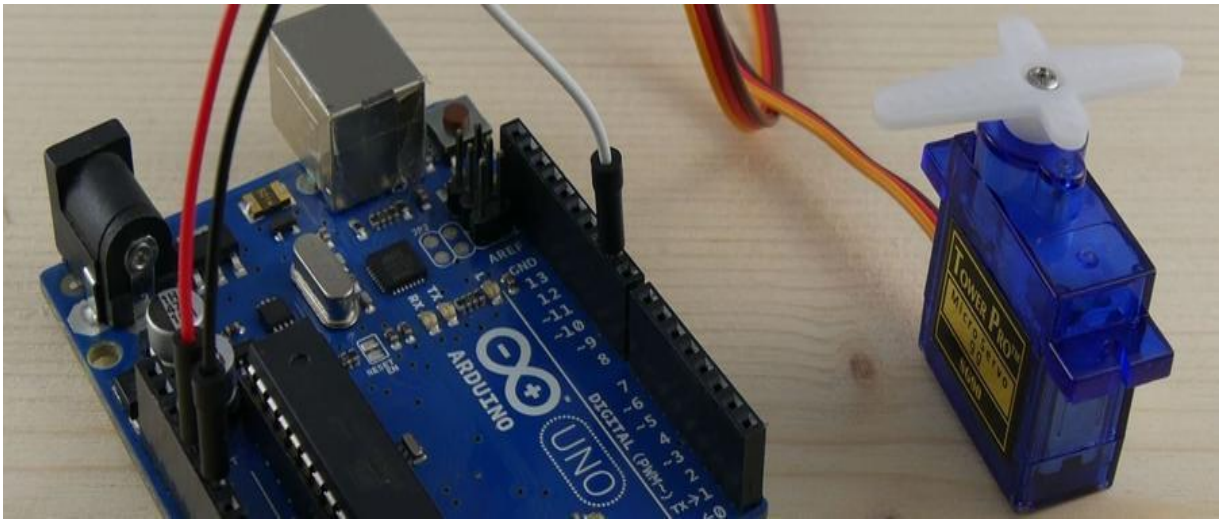
Contrôler un servomoteur avec une carte Arduino / Genuino



Dans ce tutoriel, nous allons apprendre ensemble à utiliser des servomoteurs de modélisme avec une carte Arduino / Genuino. Nous étudierons la fonction d'un servomoteur et nous ferons quelques tests avec un modèle classique de servomoteur. En bonus, nous verrons comment tirer profit de toutes les fonctionnalités offertes par la bibliothèque Arduino "Servo".

Sommaire

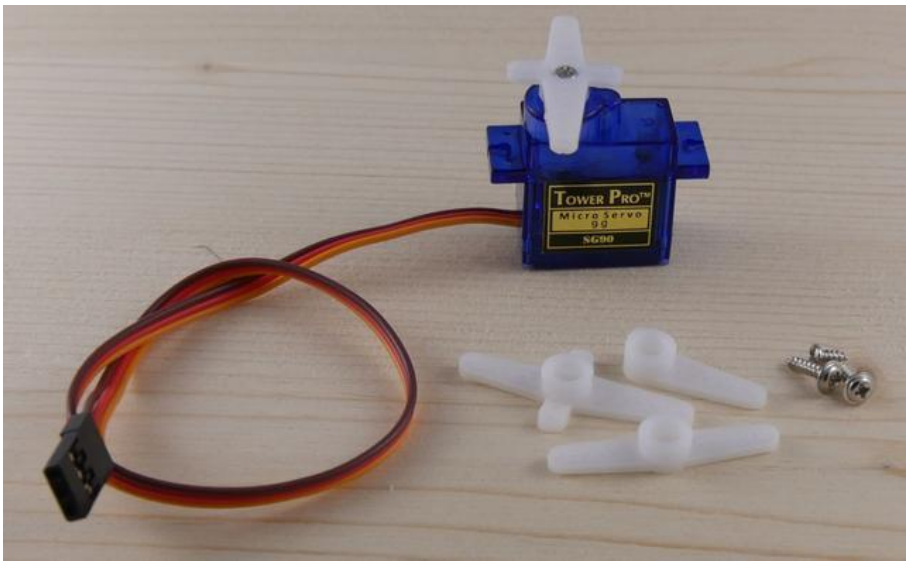
Bonjour à toutes et à tous !

Dans ce tutoriel, on va s'intéresser aux servomoteurs et à l'utilisation de servomoteurs avec une carte Arduino / Genuino.

Les servomoteurs sont des moteurs un peu particuliers, qui peuvent tourner avec une liberté d'environ 180° et garder de manière relativement précise l'angle de rotation que l'on souhaite obtenir.

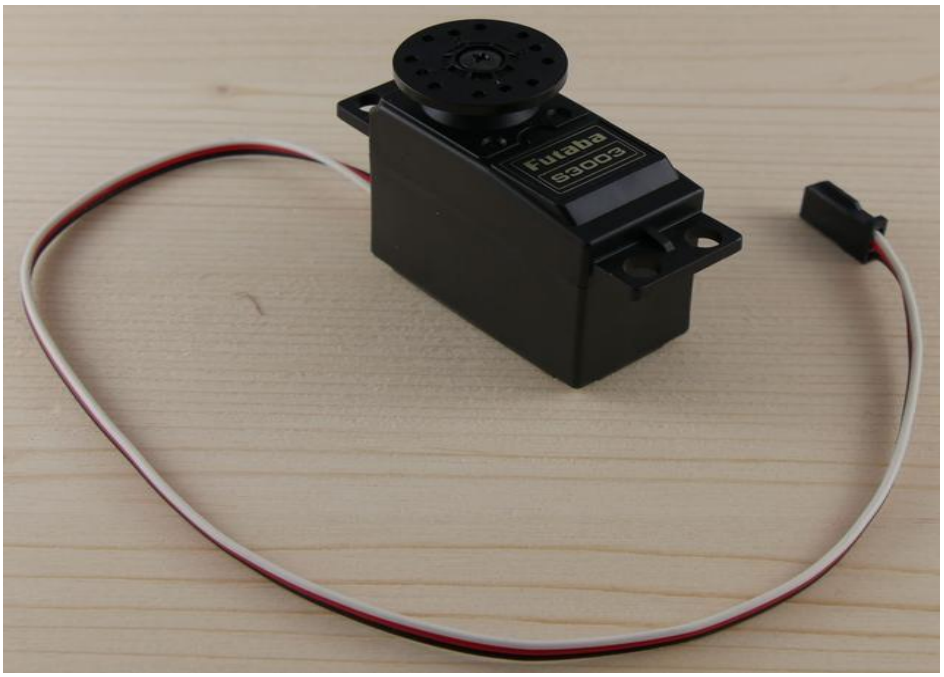
On utilise des servomoteurs couramment en modélisme pour contrôler des systèmes mécaniques (gouverne d'avion, accélérateur de moteur thermique, etc.). Les servomoteurs sont aussi couramment utilisés en robotique pour faire des mini-robots, des actionneurs ou des indicateurs rotatifs.

Principe de fonctionnement et de contrôle d'un servomoteur



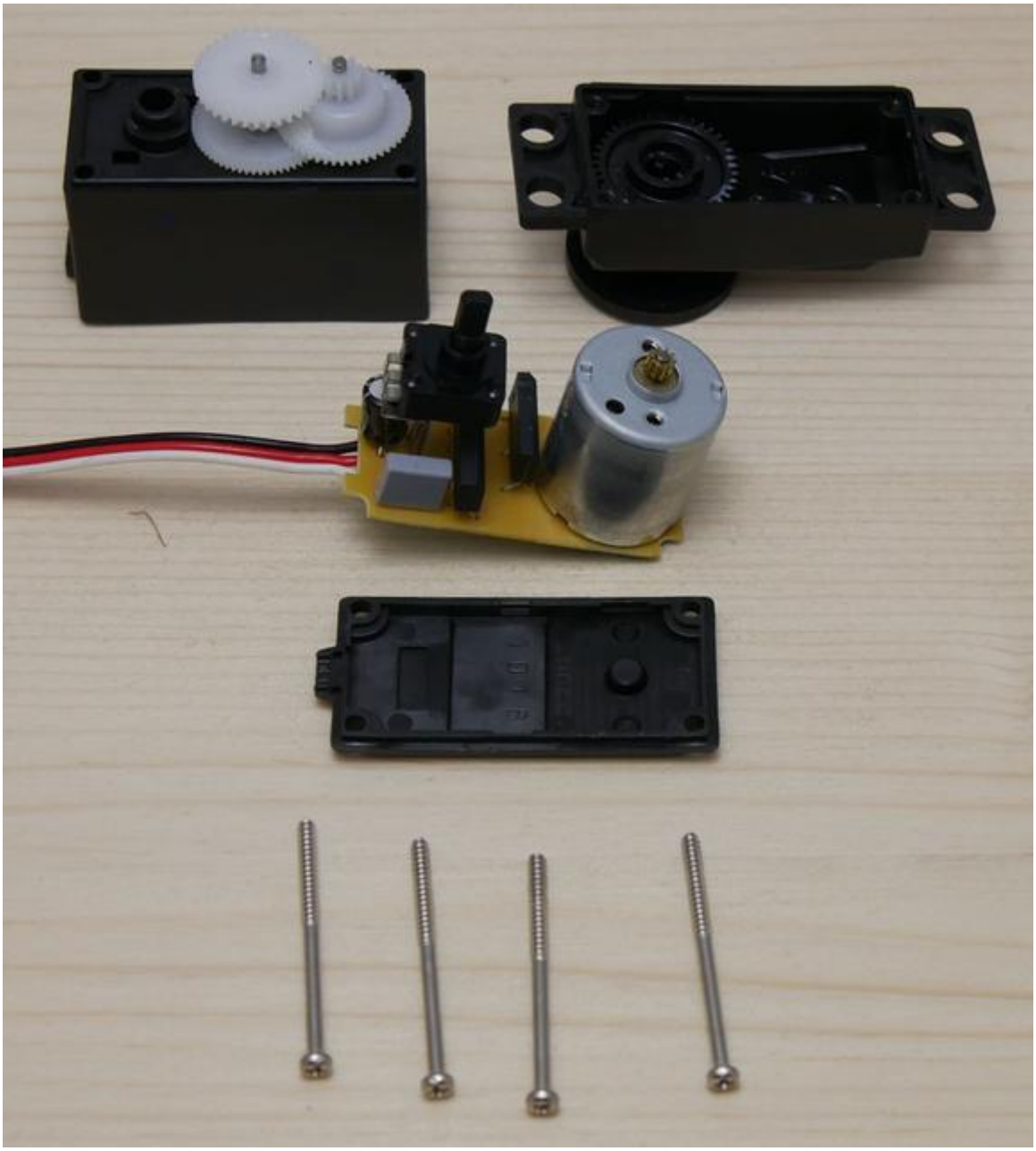
Servomoteur chinois "9 grammes"

Un servomoteur de modélisme se présente sous la forme d'un petit rectangle avec deux languettes sur les côtés pour la fixation et un axe décentré avec un bras (interchangeable) pour la liaison mécanique.



Servomoteur Futuba S3003

Il existe divers types de servomoteurs, de taille, poids et couple (force) différents. La photographie ci-dessus présente un servomoteur très classique en modélisme : le Futuba S3003. Un peu plus bas dans l'article, on utilisera un autre servomoteur, communément appelé "servomoteur 9 grammes", par souci de consommation électrique.



Vue éclatée d'un servomoteur

Le fonctionnement interne d'un servomoteur est assez basique.

Un petit circuit électronique permet de contrôler un moteur à courant continu en fonction de la position d'un potentiomètre intégré au servomoteur.

La sortie du moteur à courant continu est reliée mécaniquement à une série d'engrenages qui augmente la force (le couple) du servomoteur en réduisant la vitesse de rotation de celui-ci.

Quand le moteur tourne, les engrenages s'animent, le bras bouge et entraine avec lui le potentiomètre. Le circuit électronique ajuste continuellement la vitesse du moteur pour que le potentiomètre (et par extension le bras) reste toujours au même endroit.

Il suffit de donner une consigne au servomoteur ("reste à 45°" par exemple) et le servomoteur fera son maximum pour rester au plus près de cette consigne.

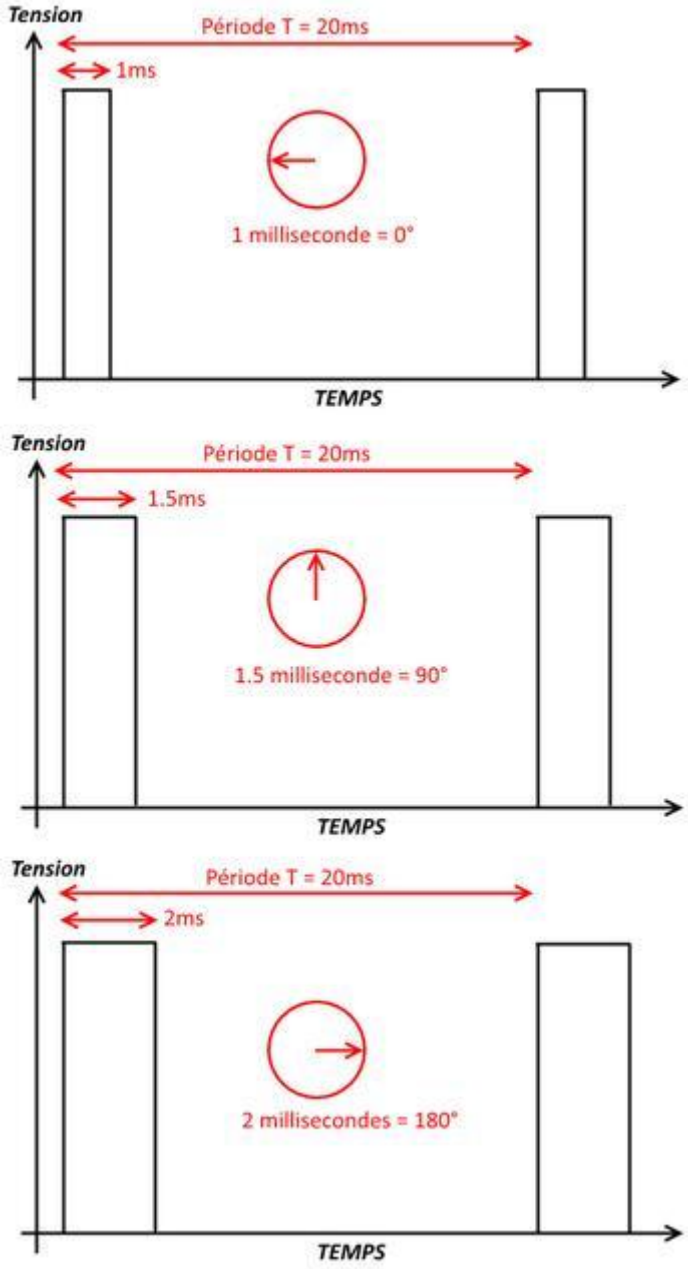


Illustration du signal de contrôle

Cette consigne est transmise au moyen d'un signal numérique, d'une impulsion pour être précis.

Pour que le servomoteur reste à une position donnée, il faut transmettre toutes les 20 millisecondes (soit à une fréquence de 50Hz) une impulsion d'une longueur comprise entre 1 et 2 millisecondes.

- Une impulsion de 1 milliseconde correspond à un angle de 0°.

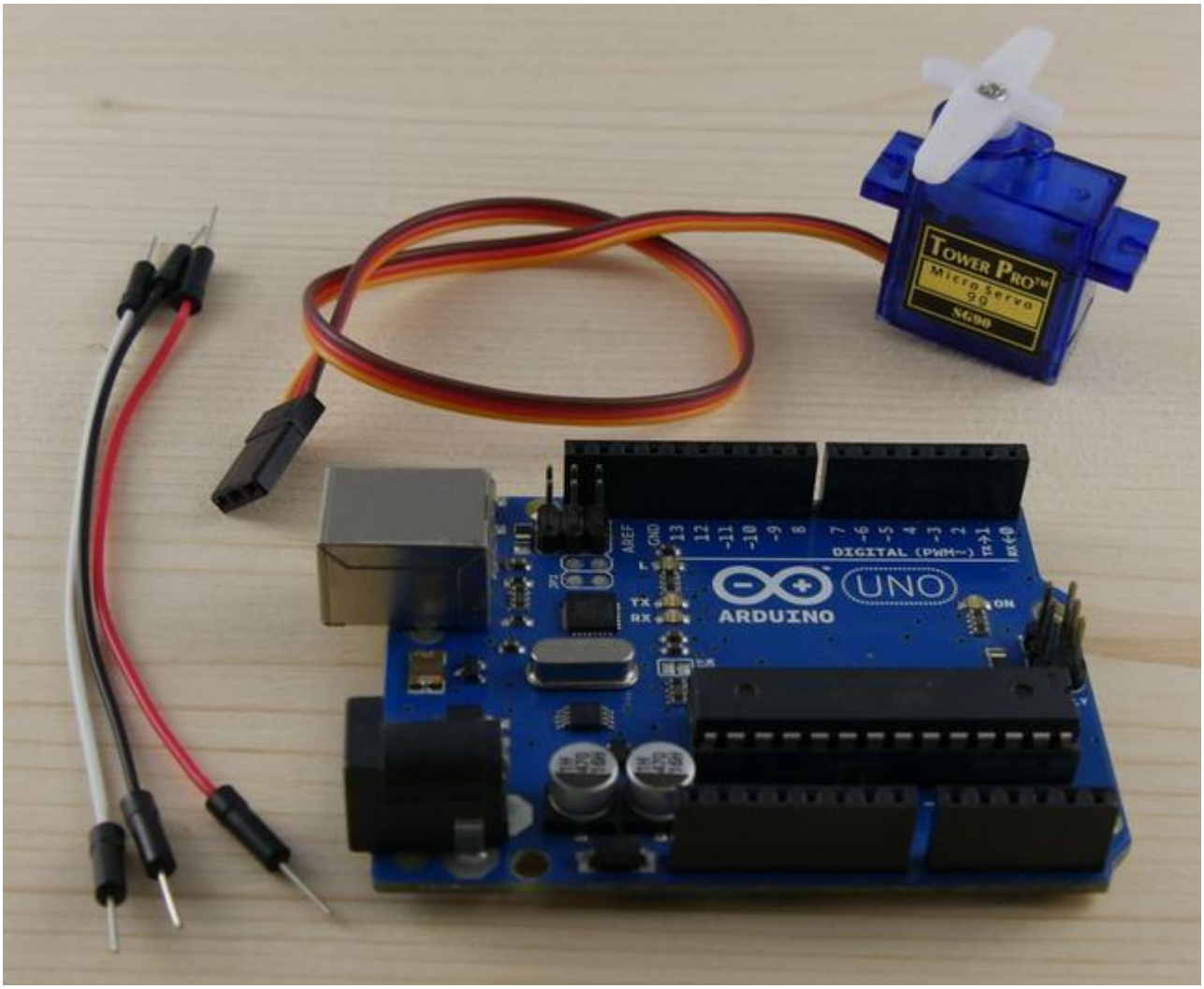
- Une impulsion de 2 millisecondes correspond à un angle de 180°.
- En envoyant une impulsion d'une longueur intermédiaire, on obtient des angles différents, 90° avec une impulsion de 1.5 milliseconde par exemple.

N.B. La plupart des servomoteurs fonctionnent en 5 volts, mais certains fonctionnent en 3.3 volts. Pensez à bien lire la documentation du servomoteur avant de l'utiliser.

Utiliser un servomoteur avec une carte Arduino / Genuino

Dans ce chapitre, nous allons mettre en oeuvre un petit servomoteur "9 grammes" avec une carte Arduino / Genuino.

Le montage

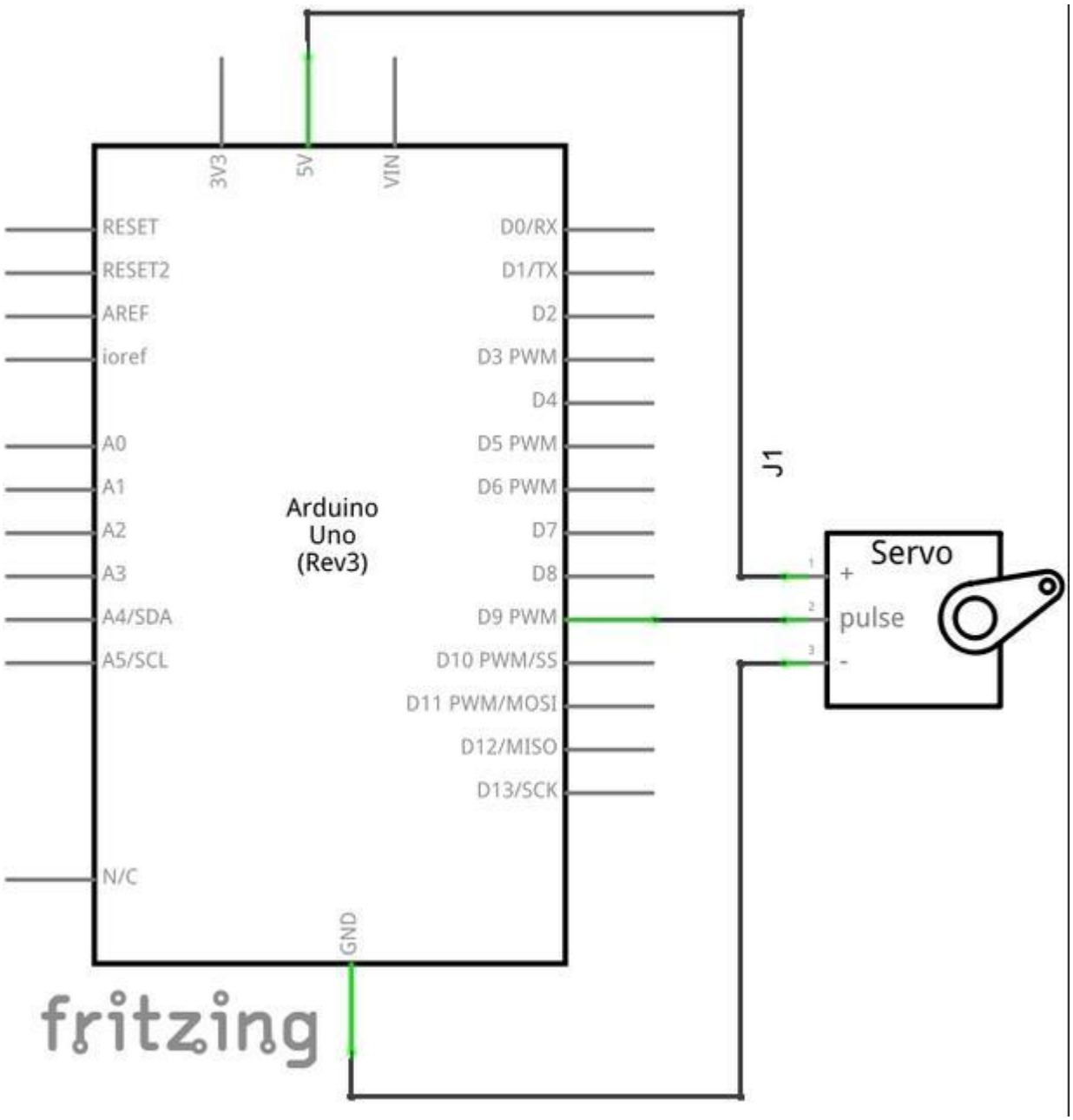


Matériel nécessaire

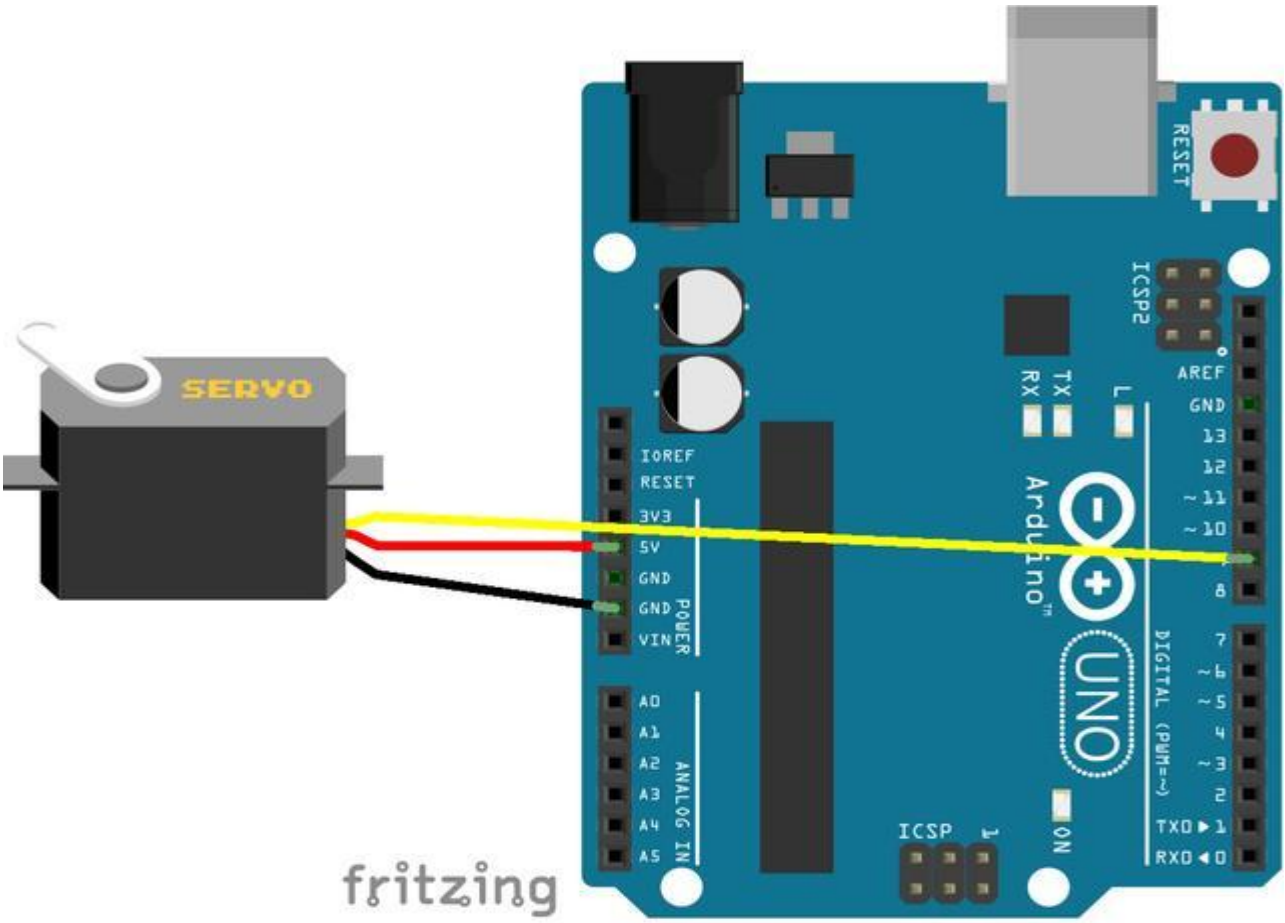
Pour réaliser ce montage, il va nous falloir :

- Une carte Arduino UNO (et son câble USB),
- Un servomoteur "9 grammes" ou assimilé,

- Des fils pour câbler notre servomoteur.

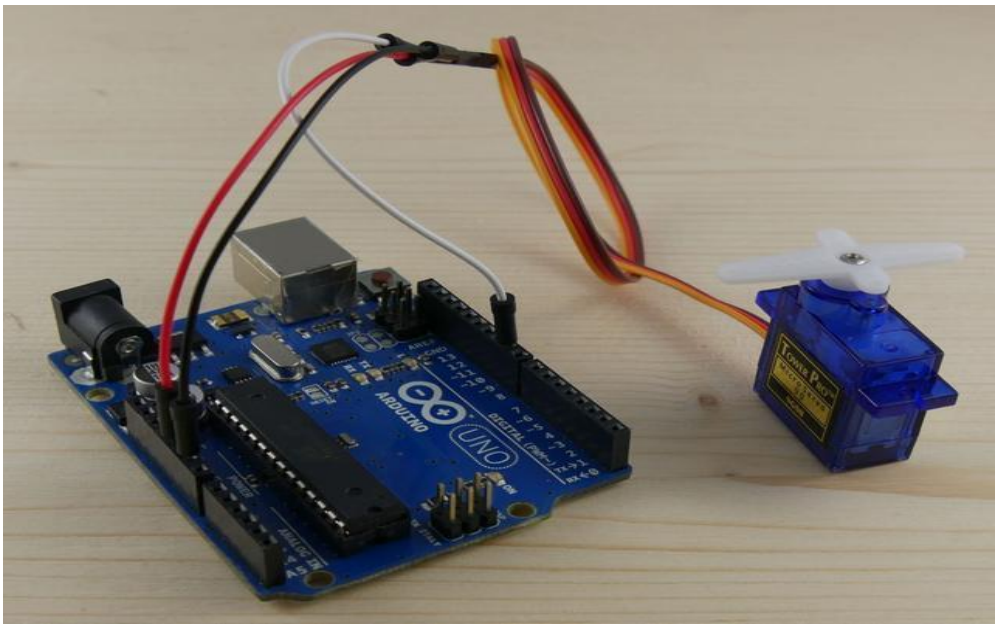


Vue schématique du montage



Vue prototypage du montage

Le câblage est relativement simple : fil rouge du servomoteur sur la broche 5V de la carte Arduino, fil noir sur la broche GND et fil blanc (ou jaune en fonction des constructeurs) sur la broche D9 de la carte Arduino.



Le montage fini

Dans le cas de mon servomoteur, le constructeur (chinois) s'est fait plaisir au niveau des couleurs. J'ai donc un fil marron à la place du noir et un fil orange à la place du rouge. Dans le doute, toujours consulter la documentation constructeur

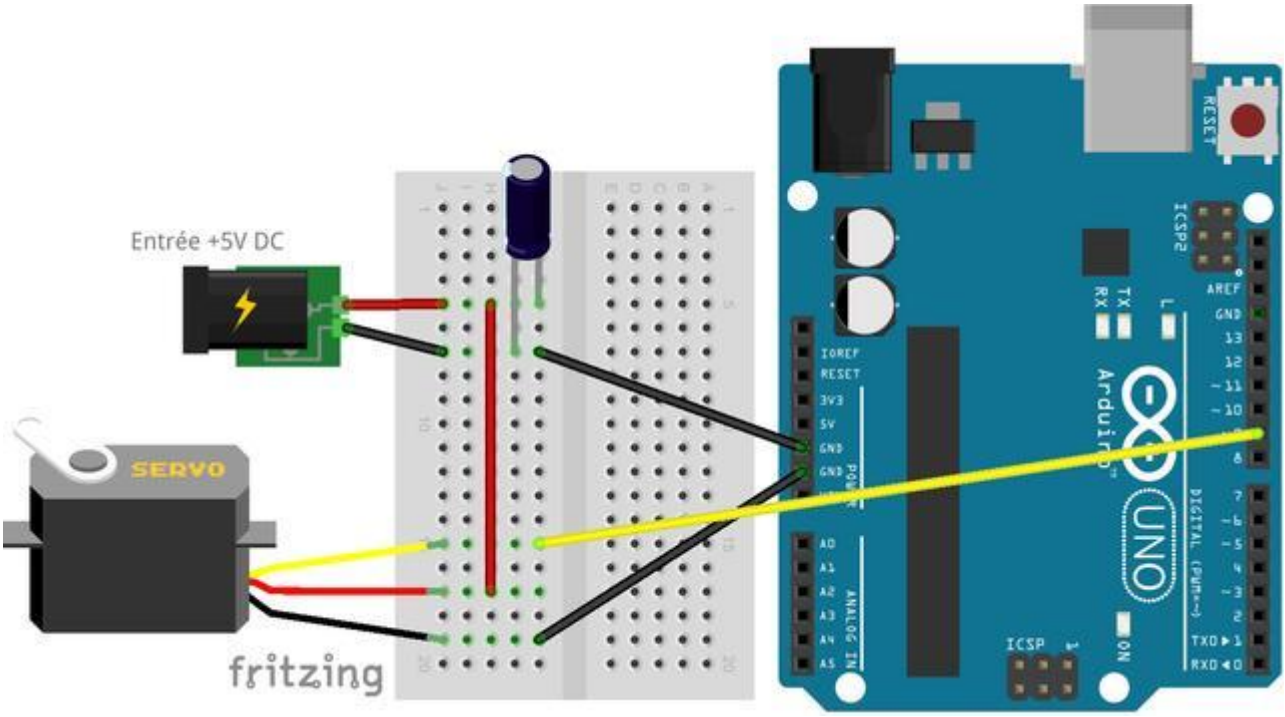
PS Avec un servomoteur standard, l'alimentation est toujours sur le fil du milieu.

Attention à la consommation électrique du servomoteur

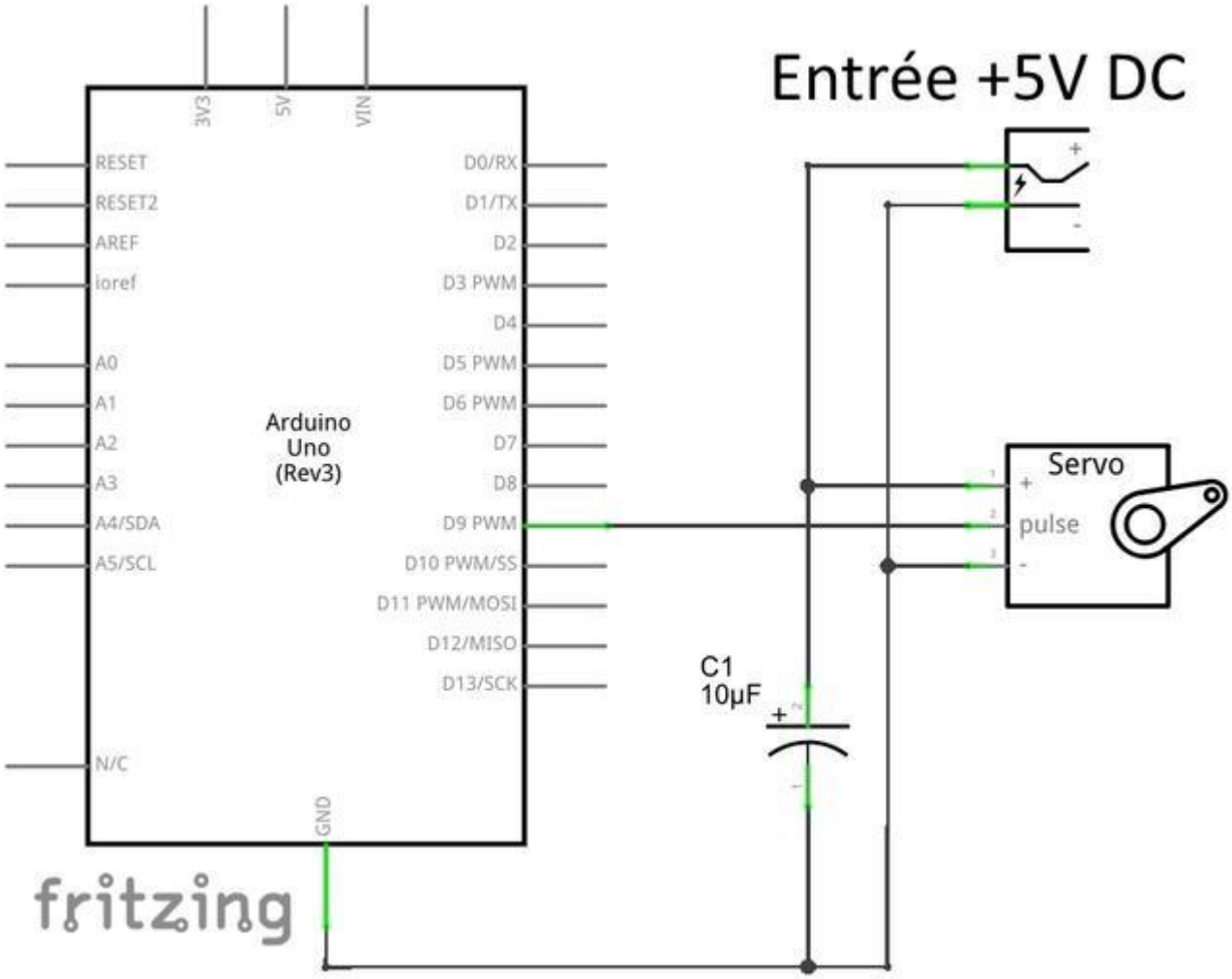
Un problème récurrent avec le montage ci-dessus est la perte de la communication USB et / ou le redémarrage intempestif de la carte Arduino.

Cela est dû à la consommation électrique du servomoteur. Un servomoteur consomme beaucoup de courant, parfois trop pour une simple carte Arduino. Cette surconsommation déclenche le fusible électronique des cartes Arduino et a pour effet de les faire redémarrer.

Il n'existe qu'une seule solution à ce problème : câbler l'alimentation des servomoteurs à une alimentation 5 volts dédiée.



Vue prototypage du montage (avec une alimentation +5V DC externe)



Vue schématique du montage (avec une alimentation +5V DC externe)

Le câblage d'un ou plusieurs servomoteurs sur une alimentation externe est relativement simple.

Le "+" de chaque servomoteur doit être relié au "+" de l'alimentation externe. Il en va de même pour les "-" de chaque servomoteur qui doivent être reliés au "-" de l'alimentation. La broche `GND` de la carte Arduino doit elle aussi être reliée au "-" de l'alimentation.

N.B. La masse de l'alimentation et des servomoteurs doit toujours être reliée à la masse de la carte Arduino. Si la masse de la carte Arduino et des servomoteurs n'est pas commune, rien ne fonctionnera.

De plus, avec de gros servomoteurs comme le Futuba S3003, il est souvent nécessaire de câbler un condensateur de 10 ~ 100µF en parallèle de l'alimentation des servomoteurs, pour éviter les oscillations intempestives du bras.

Attention lors du câblage du condensateur, les condensateurs chimiques (ceux en forme de cylindre) sont des composants polarisés. Le "-" d'un condensateur chimique est matérialisé par une bande blanche sur le côté du condensateur.

N.B. Evitez d'inverser le "+" et le "-", les condensateurs n'aiment pas du tout ça et explosent quand cela arrive. La fumée qui s'en échappe n'est pas très bonne pour la santé, mais surtout, l'odeur est abominable.

Le code

Pour faire bouger notre servomoteur, nous allons devoir utiliser une bibliothèque de code, nommée "Servo". Celle-ci est fournie de base avec l'environnement de développement Arduino.

La bibliothèque Servo permet de contrôler jusqu'à 12 servomoteurs simultanément avec une carte Arduino UNO et 48 avec une carte Arduino Mega.

Avec une carte Arduino UNO, l'utilisation de la bibliothèque Servo rend inutilisable les broches `D9` et `D10` en PWM avec `analogWrite()`.

Avec une carte Arduino Mega, 11 servomoteurs peuvent être utilisés simultanément sans poser de problèmes particuliers. A partir de 12 servomoteurs, les broches `D11` et `D12` ne pourront plus être utilisées en PWM avec `analogWrite()`.

Mise en oeuvre de la bibliothèque Servo

La bibliothèque Servo est fournie de base avec l'environnement de développement Arduino. Il n'y a donc pas d'installation à prévoir.

Il suffit d'importer la bibliothèque en ajoutant cette ligne en début de programme pour l'utiliser :

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

La bibliothèque Servo est une bibliothèque orientée object, cela signifie qu'elle fonctionne en assignant une variable (un objet) à chaque servomoteur que l'on souhaite utiliser.

Pour créer un objet Servo, il suffit de déclarer une variable (globale) de type "Servo", exemple :

```
Servo monServomoteur;
```

Si vous souhaitez utiliser plusieurs servomoteurs simultanément, il faut créer plusieurs variables, une par servomoteur.

PS Un exemple de code complet est disponible en fin de chapitre

Initialisation de la bibliothèque Servo

L'initialisation de la bibliothèque Servo se fait au moyen de la fonction attach() de chaque objet de type Servo. Exemple :

```
void setup() {
  monServomoteur.attach(9);
}
```

```
Servo.attach(int broche);
```

La fonction attach() prend en argument un unique paramètre obligatoire correspond au numéro de broche sur laquelle le servomoteur est câblé.

```
Servo.attach(int broche, unsigned long min, unsigned long max);
```

La fonction `attach()` peut prendre en plus deux paramètres optionnels, "min" et "max", correspondant respectivement à la durée minimum et maximum en microsecondes de l'impulsion de contrôle vu en début d'article.

Les durées par défaut sont de 544µs pour 0° ~ 240µs pour 180°. Ces valeurs par défaut sont conçues pour fonctionner avec la grande majorité des servomoteurs du commerce.

Compatibilité avec les versions Arduino 0016 et inférieure

Si vous utilisez une vieille version du logiciel Arduino, version 0016 ou inférieur, la bibliothèque Servo ne permet dans ce cas que d'utiliser 2 servomoteurs, sur les broches `D9` et `D10` uniquement.

Cette limitation (assez contraignante) n'existe que dans ces versions. Il suffit d'utiliser une version plus récente pour ne plus avoir de problème.

Si plus tard dans votre code vous souhaitez vérifier qu'un objet Servo est bien attaché à une broche, vous pouvez utiliser la fonction attached().

```
bool Servo.attached();
```

La fonction attached() retourne un booléen, Vrai (`true = 1`) si l'objet Servo est attaché à une broche, Faux (`false = 0`) dans le cas contraire.

Et si vous souhaitez détacher un objet Servo d'une broche, il suffit d'appeler la fonction detach().

```
Servo.detach();
```

La fonction detach() ne prend aucun paramètre et ne retourne aucune valeur. Il suffit de l'appeler pour détacher l'objet Servo de sa broche.

N.B. Si vous détachez tous les objets Servo de leurs broches respectives, la bibliothèque Servo libère automatiquement les broches `D9` et `D10` (ou `D11` et `D12` sur Mega) qui redeviennent utilisables avec `analogWrite()`.

Modification de l'angle du servomoteur

Pour modifier l'angle du bras du servomoteur, il existe deux solutions : write() et writeMicroseconds().

```
Servo.write(int angle);
```

La fonction write() permet de modifier l'angle du bras du servomoteur en donnant en paramètre l'angle en question, sous la forme d'un nombre entier compris entre 0° et 180°.

```
int Servo.read();
```

Si par la suite, vous voulez récupérer la dernière valeur d'angle affectée avec la fonction `write()`, il suffit d'appeler la fonction read() qui retourne la dernière valeur connue.

N.B. L'amplitude de rotation disponible dépend du modèle de servomoteur. Certains servomoteurs peuvent faire un 0° ~ 180° sans problème, quand d'autre arrive à peine à tourner de 60°. Pensez à bien lire les spécifications techniques

```
1Servo.writeMicroseconds(unsigned long us);
```

La fonction writeMicroseconds() permet de modifier l'angle du bras du servomoteur en donnant en paramètre la durée de l'impulsion à transmettre au servomoteur.

Avec un servomoteur standard, une valeur de 1000µs donnera un angle de 0° et une valeur de 2000µs donnera un angle de 180°. Cependant, avec certains servomoteurs, ces valeurs peuvent aller jusqu'à une plage de l'ordre de 700 ~ 2300µs pour 0 ~ 180°.

Cette fonction permet de choisir précisément l'angle du bras en fonction du servomoteur utilisé. Il est cependant nécessaire de faire des tests aux préalables pour déterminer les limites du bras.

C'est quoi ce bruit ?

Quand on demande à un servomoteur d'aller au-delà de ses limites fonctionnelles (en lui demandant de faire un 360° par exemple), on entend en général un bruit aigu.

Si le servomoteur émet un bruit aigu / inhabituel, c'est que vous lui demandez l'impossible

Bloquer le bras d'un servomoteur dans une position "en force" n'est pas très bon et finira par le tuer. Cela induit aussi une très forte consommation de courant.

Exemple : Sweep

Pour illustrer l'utilisation des fonctions décrites ci-dessus, voici un petit exemple de code nommé "Sweep".

Son but est simple : faire tourner le bras du servomoteur dans un sens puis dans l'autre, indéfiniment.

Exemple Sweep utilisant `Servo.write()` avec commentaires :

```
/**
 * Exemple de code pour un servomoteur, il fait faire des va-et-vient à
la tête du servomoteur.
 */


 /* Inclut la lib Servo pour manipuler le servomoteur */
#include <Servo.h>


/* Créer un objet Servo pour contrôler le servomoteur */
Servo monServomoteur;


void setup() {


  // Attache le servomoteur à la broche D9
  monServomoteur.attach(9);
}


void loop() {


  // Fait bouger le bras de 0° à 180°
  for (int position = 0; position <= 180; position++) {
    monServomoteur.write(position);
    delay(15);
  }


  // Fait bouger le bras de 180° à 10°
  for (int position = 180; position >= 0; position--) {
    monServomoteur.write(position);
    delay(15);
  }
```

```
}
```

```
/**
 * Exemple de code pour un servomoteur, il fait faire des va-et-vient à la tête du servomoteur.
 */

 /* Inclut la lib Servo pour manipuler le servomoteur */
#include <Servo.h>

/* Créer un objet Servo pour contrôler le servomoteur */
Servo monServomoteur;

void setup() {

  // Attache le servomoteur à la broche D9
  monServomoteur.attach(9);
}

void loop() {

  // Fait bouger le bras de 0° à 180°
  for (unsigned long position = 1000; position <= 2000; position += 5)
{
    monServomoteur.writeMicroseconds(position);
    delay(15);
  }

  // Fait bouger le bras de 180° à 10°
  for (unsigned long position = 2000; position >= 1000; position -= 5)
{
```

```
    monServomoteur.writeMicroseconds(position);
    delay(15);
  }
}
```

Conclusion

Ce tutoriel est désormais terminé.

Si ce tutoriel vous a plu, n'hésitez pas à le commenter sur le forum, à le diffuser sur les réseaux sociaux et à soutenir le site si cela vous fait plaisir.