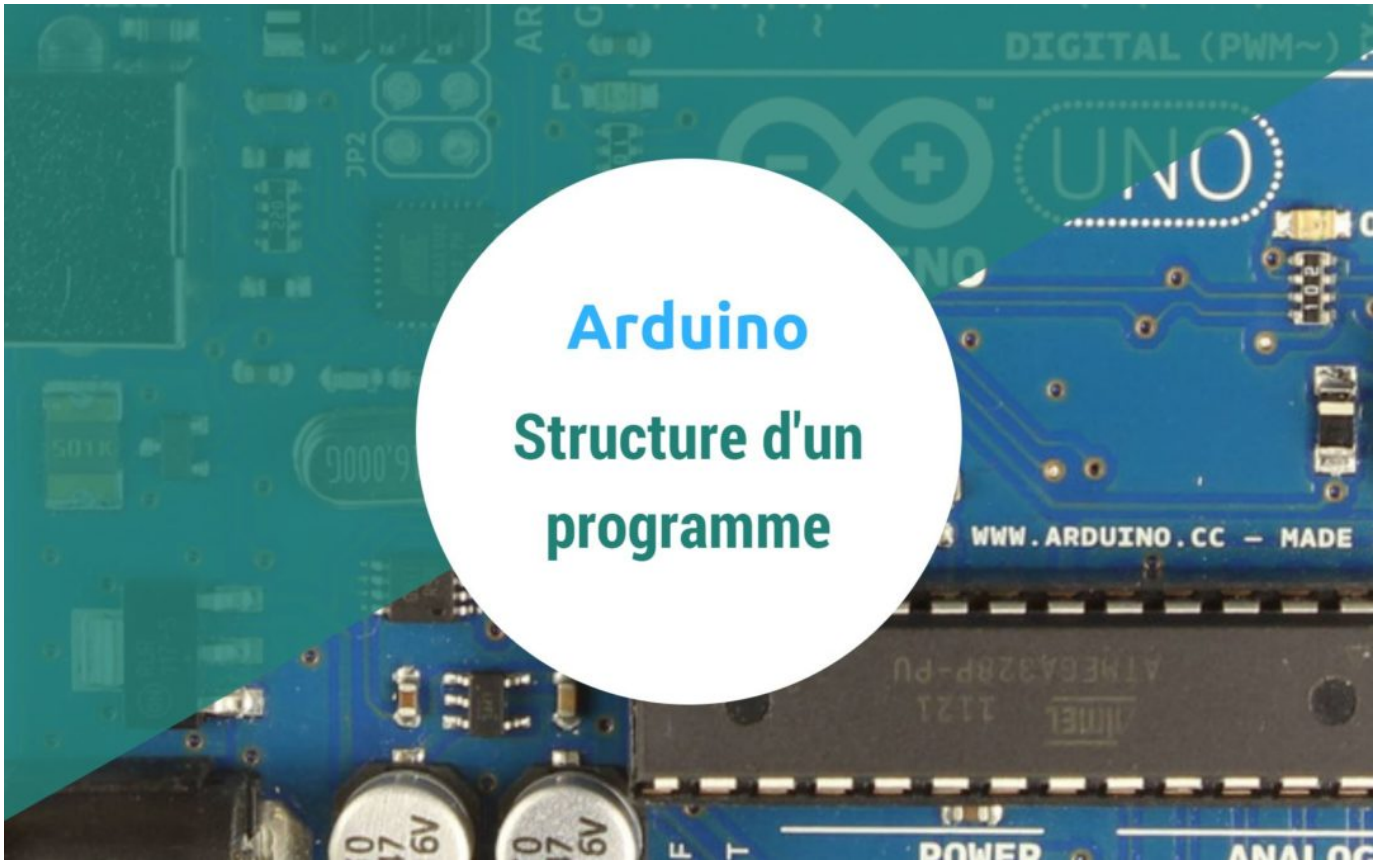
Arduino – Structure d’un programme



Je vais vous expliquer la structure d’un programme Arduino. Cette structure devra être respectée quel que soit votre microcontrôleur Arduino, et quel que soit votre programme.

Structure d’un programme Arduino

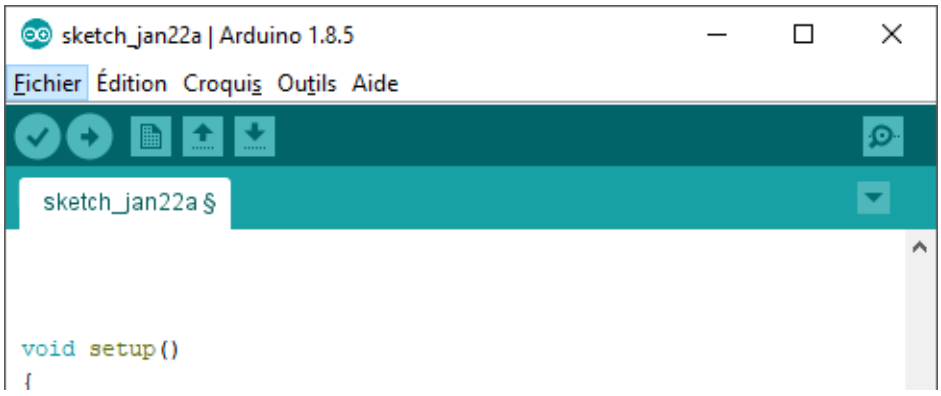
Les programmes Arduino peuvent être divisés en 3 parties :

- la **Structure**
- les **Valeurs** (variables et constantes)
- les **Fonctions**.

Nous allons parler de la **Structure** qui correspond au **squelette de notre programme Arduino**.

La Structure d’un programme Arduino est **obligatoirement** composée de ces deux fonctions principales :

- La fonction **setup()**
- La fonction **loop()**



```
}

void loop()
{

}
```

14 Arduino/Genuino Uno sur COM3

***Structure** de base d'un programme Arduino.*

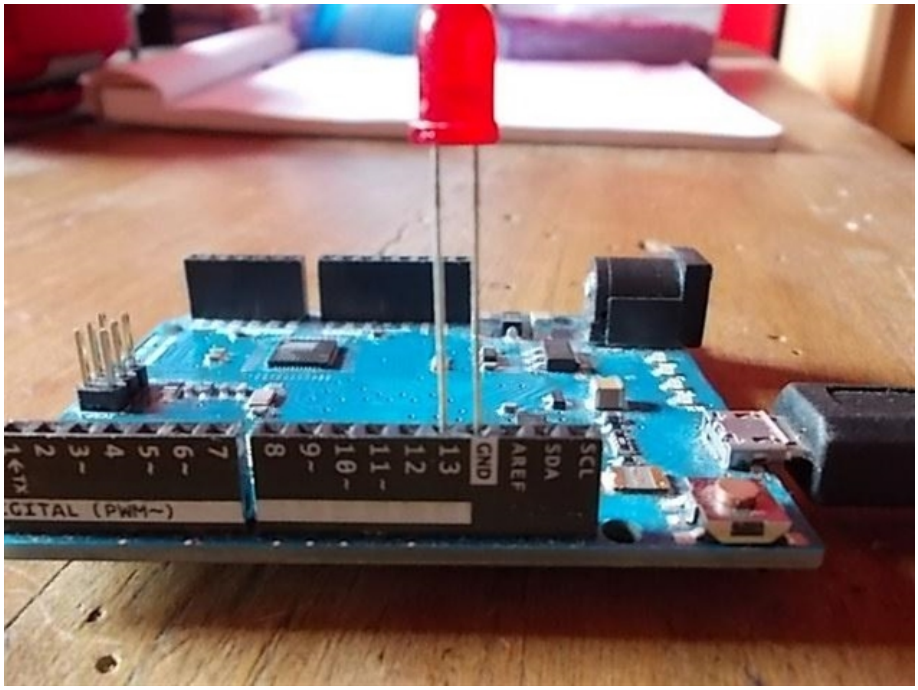
Ces deux fonctions sont obligatoires, même vides, pour que le programme soit transféré à la carte Arduino.

La fonction setup()

Il s'agit de la **fonction d'initialisation** de votre Arduino. C'est la première fonction qui va être lancée au démarrage de l'Arduino. C'est donc ici que l'on va paramétrer nos différents objets, nos protocoles de communications, la configuration de nos entrées/sorties, etc…

Ce qu'il faut retenir simplement ici, c'est que **setup() se lance en premier au démarrage ou au redémarrage (*RESET*) de votre carte Arduino**.

Dans l'exemple suivant, nous allons faire clignoter une LED montée sur la carte Arduino



Blink §

```
void setup() {

  // Initialisation de la broche 13 de l'Arduino en sortie (OUTPUT)
   pinMode(13, OUTPUT);
}

// La fonction "LOOP" va allumer une led pendant 1 seconde puis éteindre la led pendant 2 secondes
void loop() {

  digitalWrite(13, HIGH);
  delay(1000);

  digitalWrite(13, LOW);
  delay(2000);
}
```

15 Arduino Mega 2560 or Mega ADK on COM13

Programme « Blink » - Clignotement d'une LED

La fonction loop()

La fonction **loop()** est la **2ème fonction a être lancée sur votre Arduino**, juste après la fonction setup(). Cette fonction a la particularité, comme son nom l'indique, d'être **une boucle** (***loop*** en anglais). En effet, cette fonction reboucle sur elle-même. Lorsque la dernière ligne de la fonction loop() sera atteinte par votre programme, ce dernier remontera à la première ligne, à la manière de cet exemple ci-dessous :

Blink | Arduino 1.0.6

Fichier Édition Croquis Outils Aide

Blink §

```
void setup() {

  // Initialisation de la broche 13 de l'Arduino en sortie (OUTPUT)
   pinMode(13, OUTPUT);
}

// La fonction "LOOP" va allumer une led pendant 1 seconde puis éteindre la led pendant 2 secondes
void loop() {

  digitalWrite(13, HIGH);
  delay(1000);

  digitalWrite(13, LOW);
  delay(2000);
}
```

Boucle du programme Blink qui consiste à faire clignoter une LED indéfiniment.

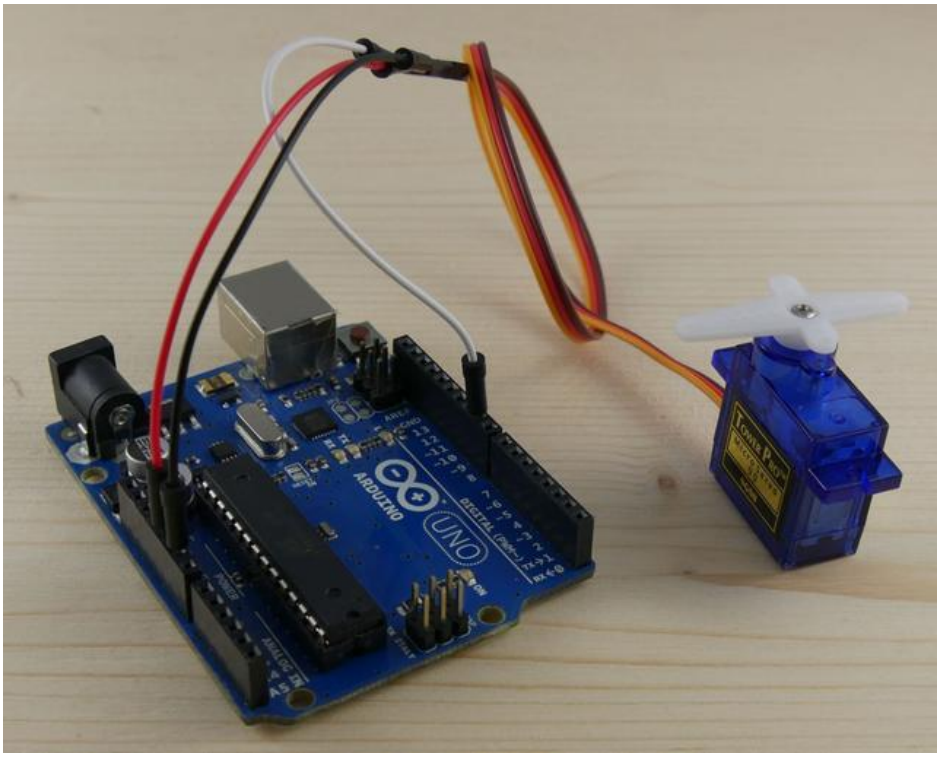
Et en dehors de ces fonctions ?

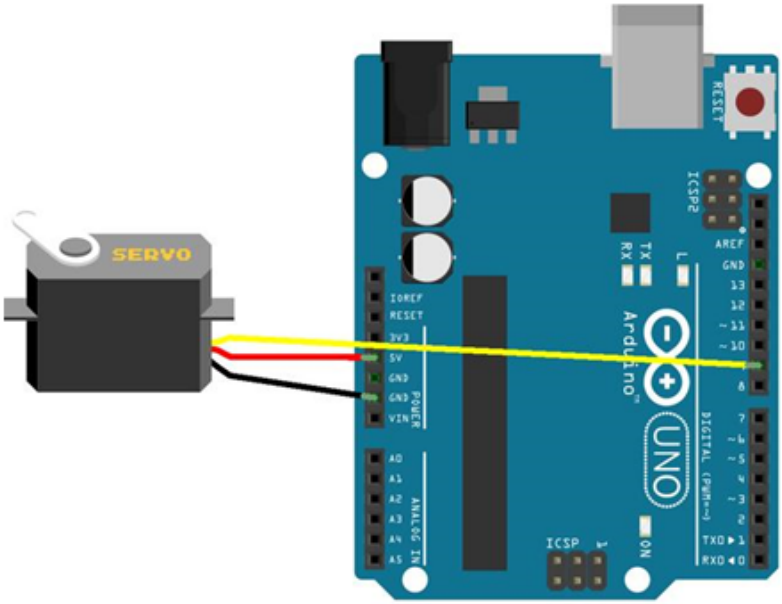
En dehors de ces fonctions, on peut déclarer nos variables, d'autres fonctions, des librairies additionnelles…

Exemple de programmation un servomoteur

Montage

Dans l'exemple suivant, nous allons faire varier un servomoteur entre 0° et 180 ° montée sur la carte Arduino





Câblage d'un servomoteur

Programme

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.

Il suffit d'importer la bibliothèque en ajoutant la ligne en début de programme : « **#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.

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

Sweep | Arduino 1.0.6

Fichier Édition Croquis Outils Aide

Sweep §

```
// Insertion de la librairie pour servomoteurs
#include <Servo.h>
// Déclaration d'un servomoteur
Servo myservo;
// Initialisation d'une variable position d'un servomoteur
int pos;

void setup()
{
  // Attibution d'une broche Arduino à un servomoteur
  myservo.attach(9);
}
void loop()
{
  // Variation de 0° à 180° d'un servomoteur
  for(pos = 0; pos <= 180; pos += 1)
  {
    myservo.write(pos);
    delay(15);
  }
    // Variation de 180° à 0° d'un servomoteur
  }
  for(pos = 180; pos>=0; pos-=1)
  {
    myservo.write(pos);
    delay(15);
  }
}
```

Formatage automatique terminé.

12 Arduino Mega 2560 or Mega ADK on COM13

Rotation d'un servomoteur de sur une plage de 0° à 180°.