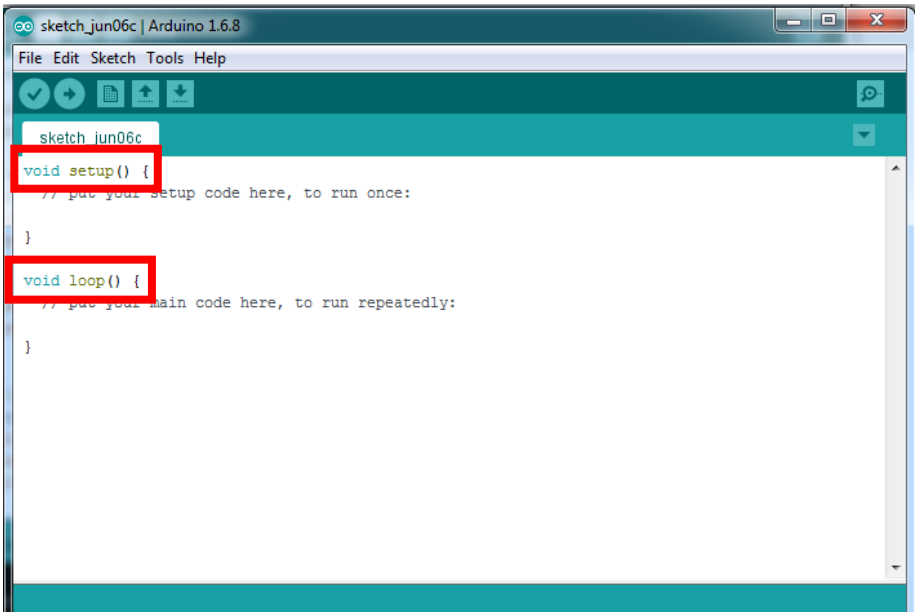
DIGITAL BRIDGE

Leçon 2

STRUCTURES ET INSTRUCTIONS POUR ECRIRE UN CODE ARDUINO

Lorsque l'on ouvre un croquis Arduino voici l'écran qui s'affiche :



void setup() { Cette partie détermine si un point de raccordement est une sortie (OUTPUT) ou une entrée (INPUT)

}

void loop() { Cette partie est une loop qui fonctionne en continue et permet à l'Arduino de fonctionner non-stop

}

IMPORTANT : Les accolades { } délimitent les blocs des fonctions void setup() et void loop()

int – Integer est utilisé pour définir un nombre entier. Par exemple une variable dénommée « *moteur_1* »

int moteur_1 = 10; donne à la variable moteur_1 la valeur de 10 et définit cette variable comme étant un nombre entier

pinMode – Arduino dispose de 13 points de raccordement qui peuvent être des points d'entrée ou de sortie des signaux. pinMode permet de définir le type de raccordement (entrée ou sortie)

Exemple : pinMode (moteur_1, OUTPUT) définit moteur_1 comme un point de sortie

pinMode (moteur_1, INPUT) définit moteur_1 comme un point d'entrée

digitalWrite – Envoie un signal digital (HIGH) au point de raccordement de sortie OUTPUT ou arrête l'envoi du signal (LOW)

Exemple : digitalWrite (moteur_1, HIGH) allume le point de sortie

digitalWrite (moteur_1, LOW) éteint le point de sortie

analogWrite – Envoie un signal analogue de 8 bits au point de raccordement de sortie OUTPUT. Ce signal varie de 0 à 5 volts et correspond à un signal PWM allant de 0 à 255. Ceci sera expliqué plus tard.

Exemple : analogWrite (LED_1, 255) envoie un signal 5V au point de sortie LED_1

analogWrite (LED_1, 127) envoie un signal 2,5V au point de sortie LED_

analogWrite (LED_1, 0) envoie un signal de 0V au point de sortie LED_1

digitalRead – Lis ou détecte le signal digital reçu au point de raccordement d'entrée INPUT (HIGH ou LOW, allumé ou éteint)

Exemple : digitalRead (alarme_1, signal_reçu) Le signal_reçu sera HIGH (allumé) si le point de raccordement est allumé et LOW s'il est éteint

analogRead – Lis ou détecte le signal analogue reçu au point de raccordement d'entrée INPUT. La carte Arduino transforme ce signal analogue en valeur de 8 bits (0 à 255)

Exemple : int Temperature = analogRead (sensorTemp) lis le voltage au point de raccordement INPUT que nous appelons « *sensorTemp* ». La variable « *Temperature* » prendra la valeur lue au capteur « *sensorTemp* ».

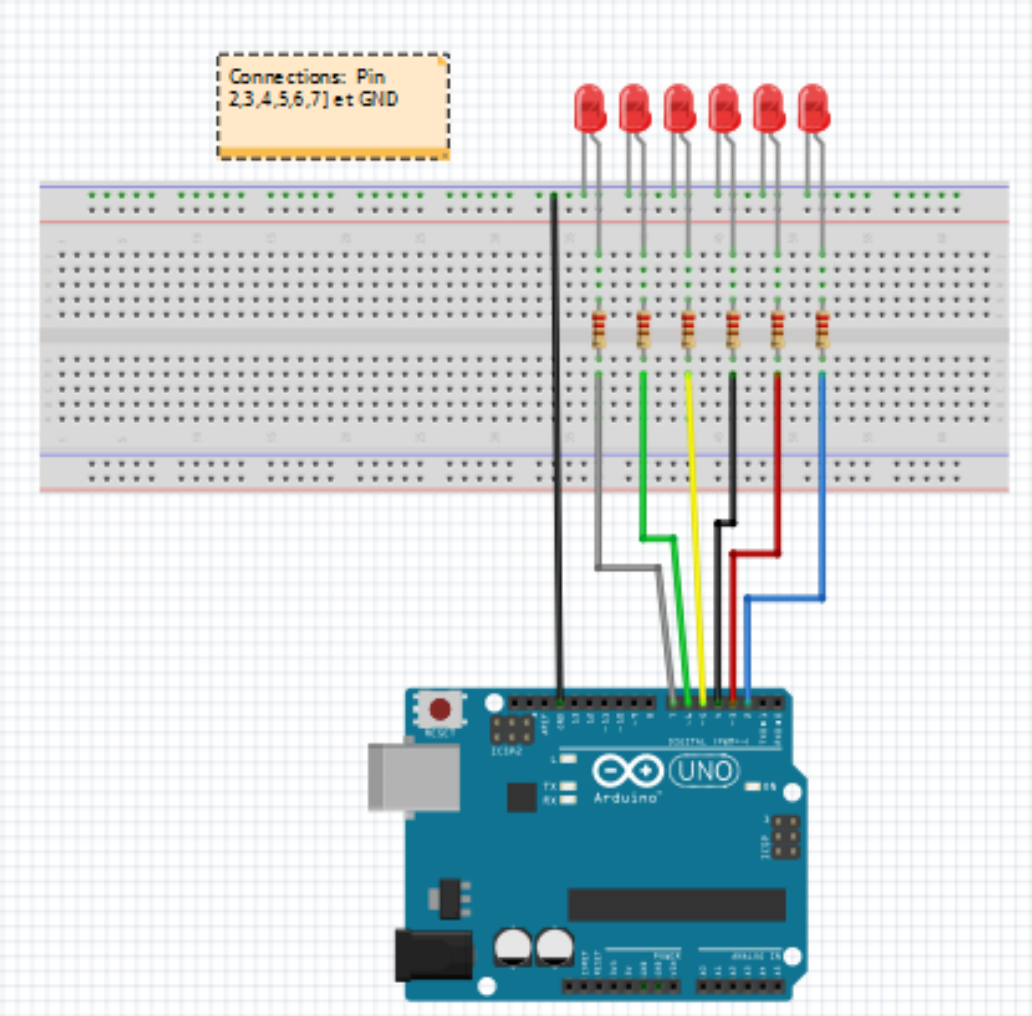
delay (temps en milliseconde) – Cette instruction permet de marquer une pause lors de l'exécution de la loop

Exemple : delay (1000) L'exécution de la loop prend une pause de 1000 millisecondes soit 1 seconde.

Exercices nº 1 : Matériel

- Carte Arduino
- (6) LED
- (7) câbles
- (7) résistances de 220
- (1) planche prototype

But : Recopier le code en respectant la syntaxe et utiliser les différents termes appris dans la leçon.




```
// Initialisation des points de raccordement
int timer = 100;  // definit le temps de la pause
int pin_1 = 2;
int pin_2 = 3;
int pin_3 = 4;
int pin_4 = 5;
int pin_5 = 6;
int pin_6 = 7;

void setup() {
  // Le code dans ce bloc ne fonctionne qu'une seule fois au debut:
pinMode(pin_1, OUTPUT);  // Definir les points de raccordement de sortie (OUTPUT)
pinMode(pin_2, OUTPUT);
pinMode(pin_3, OUTPUT);
pinMode(pin_4, OUTPUT);
pinMode(pin_5, OUTPUT);
pinMode(pin_6, OUTPUT);
}

void loop() {
  // Le code dans ce bloc fonction en continue (non-stop):
  digitalWrite(pin_1, HIGH);  //HIGH Allume la LED au point de raccordement 2
  delay(timer);             //Marque une pause qui correspond au temps de la variable timer
  digitalWrite(pin_1, LOW);   //LOW Eteint la LED au point de raccordement 2
```

```
  digitalWrite(pin_2, HIGH);
  delay(timer);
  digitalWrite(pin_2, LOW);

  digitalWrite(pin_3, HIGH);
  delay(timer);
  digitalWrite(pin_3, LOW);

  digitalWrite(pin_4, HIGH);
  delay(timer);
  digitalWrite(pin_4, LOW);

  digitalWrite(pin_5, HIGH);
  delay(timer);
  digitalWrite(pin_5, LOW);

  digitalWrite(pin_6, HIGH);
  delay(timer);
  digitalWrite(pin_6, LOW);
}
```