Clignotement- Arduino

1. Objectifs :

- Se familiariser avec une carte de type Arduino UNO
- Réaliser son premier programme Arduino!

2. Matériel :

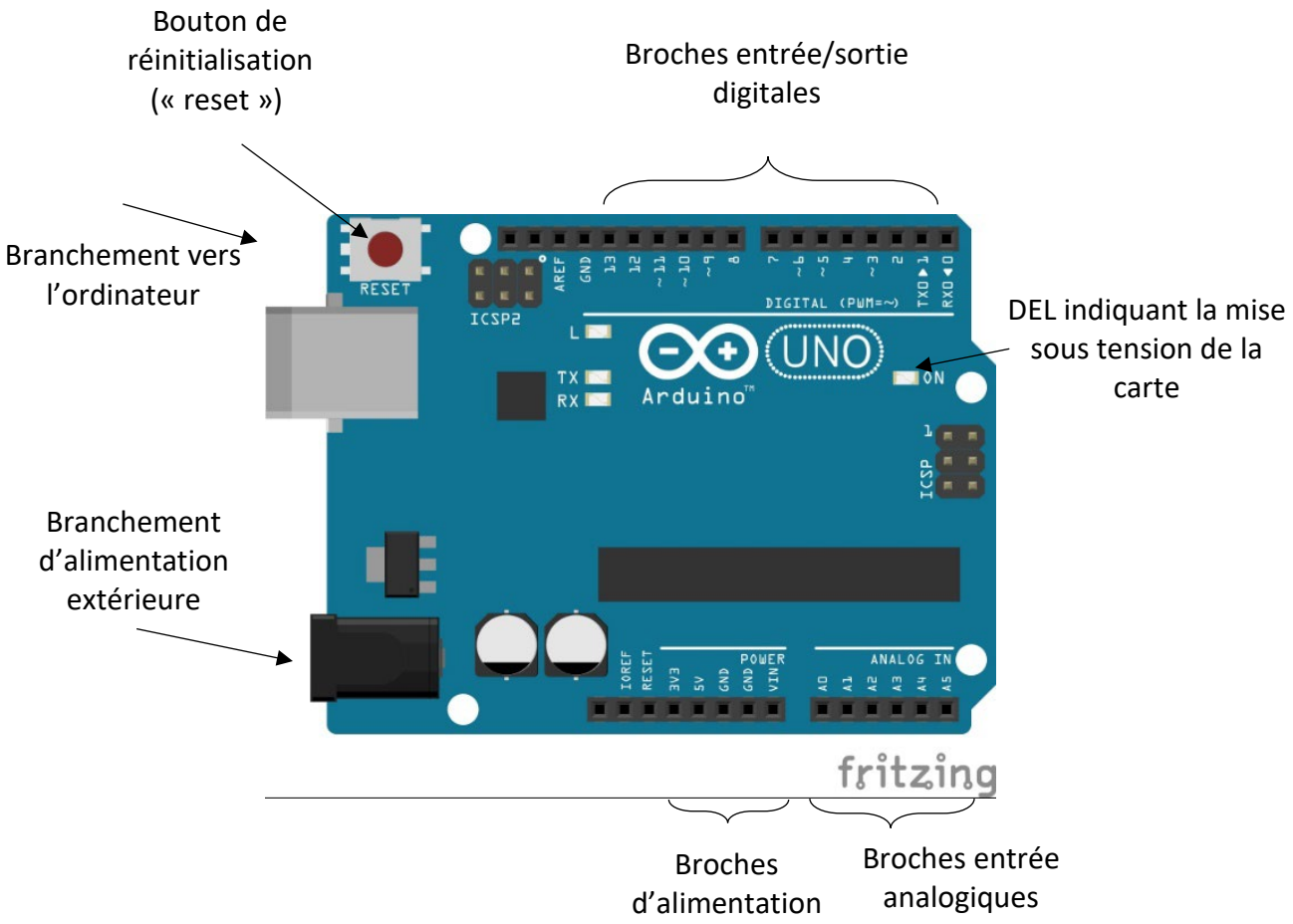
- Arduino UNO ou Bluno
- Résistance de 220 Ω
- DEL de la couleur désirée
- Ordinateur + fil de branchement pour la carte
- Quelques fils
- Platine d'essai (« breadboard »)

3. Cartes de type Arduino

Les cartes Arduino ou compatibles avec Arduino sont des cartes sur lesquelles on retrouve un microcontrôleur permettant de programmer les actions des différentes composantes branchées aux entrées et sorties de la carte. Il est ainsi possible de réaliser un programme permettant d'allumer une ampoule (signal de sortie) lorsque la température (signal d'entrée) dépasse une certaine valeur.

Voici les différentes parties de base d'une carte de type Arduino UNO®. Il y a d'autres fonctionnalités qui seront vues plus tard au besoin.

Les broches identifiées « GND » servent de mise à la terre pour le retour du courant. Elles jouent toutes le même rôle et peuvent être interchangées sans problème.
Pour l'instant, l'alimentation électrique sera fournie par le branchement vers l'ordinateur.

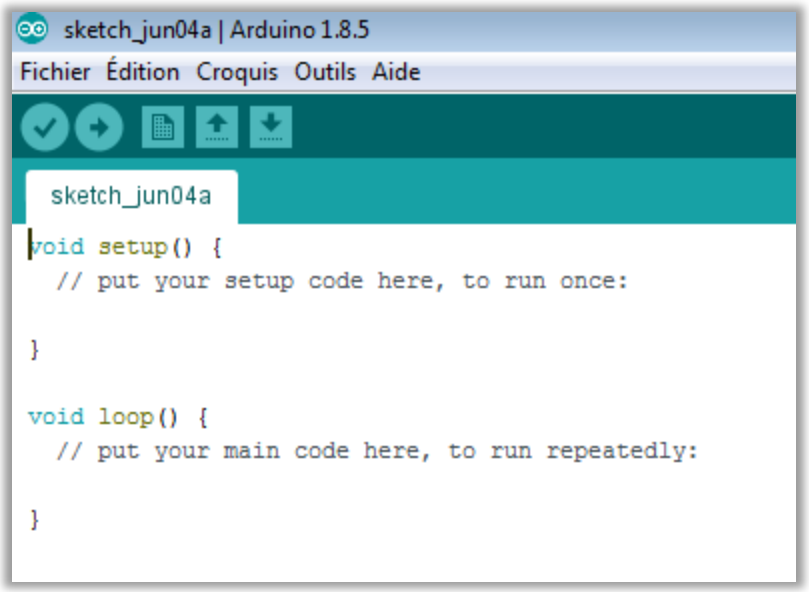


4. Ce que fait le programme

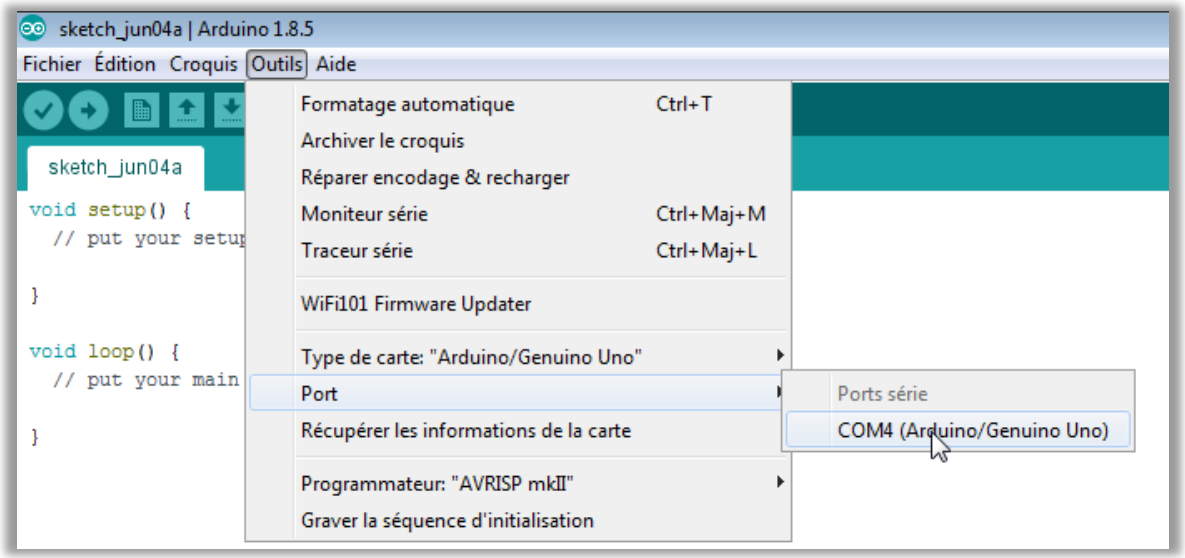
Ce programme permet de faire clignoter une diode électroluminescente (DEL) : Allumée pendant 1 s et éteinte pendant 1 s, et ainsi de suite.

5. Étapes de réalisation

1. Brancher la carte Arduino UNO dans l'ordinateur avec le fil approprié. La DEL (verte) d'alimentation de la carte devrait être allumée.
2. Ouvrir le logiciel Arduino. Il est possible d'utiliser la version en ligne disponible au www.arduino.cc
3. Un nouveau « sketch » (nom des programmes pour Arduino) devrait apparaître.



4. On doit ensuite s'assurer que le bon type de carte et le bon port de communication sont utilisés. Dans l'onglet *Outils*, modifier le *Type de carte* dans la liste déroulante si « Arduino/Genuino UNO » n'est pas sélectionné.

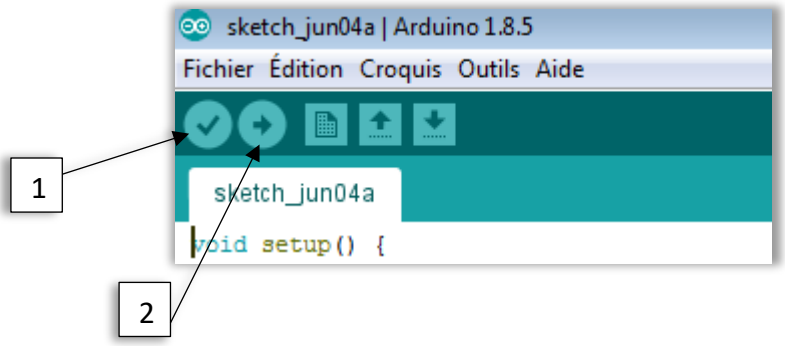


5. Dans l'onglet *Outils*, dans *Port*, sélectionner celui qui porte le nom « Arduino ».

6. Réaliser le circuit demandé (voir plus bas).
    a. La carte Arduino doit rester branchée à l'ordinateur.
    b. La patte plus longue de la DEL se branche du côté de la résistance.
    c. Les fils n'ont pas à être de la couleur qui est indiquée sur le schéma.

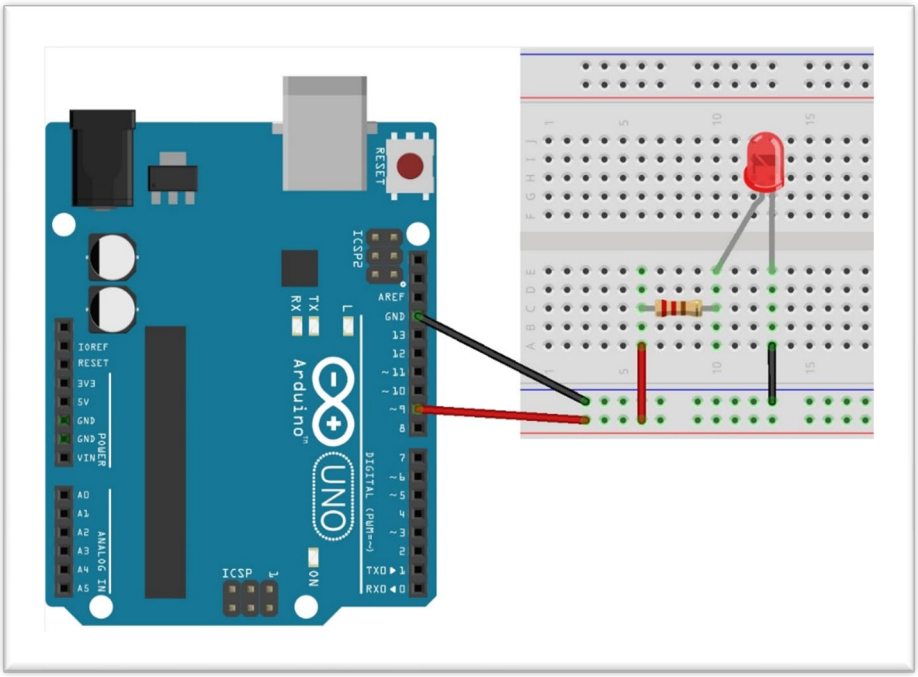
7. Copier ou écrire le code dans le sketch.

8. Compiler (1) et téléverser (2) le programme dans la carte. S'il y a des erreurs, elles seront indiquées en orange/rouge en bas de la fenêtre du programme.



9. Observer le résultat !

6. Schéma du circuit



7. Code

```
int DELpin =  9;

void setup() {
     pinMode(DELpin, OUTPUT);
}

void loop() {
     digitalWrite(DELpin, HIGH);
     delay(1000);
     digitalWrite(DELpin, LOW);
     delay(1000);
}
```

8. Fonctions utilisées

```
int DELpin =  9;
```

Dans un programme, il est utile de définir des variables qui portent un nom précis. Par exemple, ici, une variable du nom *DELpin* a été définie. On lui assigne ensuite la valeur de « 9 » car c'est dans cette broche (*pin*) qu'est branché le circuit et qui contrôlera la DEL.

Type de variables :

Toute variable qui sera utilisée dans un programme doit être déclarée avant sa première utilisation. On doit utiliser un mot qui indique son ***type,*** suivi de son nom. Le type n'est pas répété lors des utilisations de la variable. On peut lui donner ou non une valeur à ce moment.

Dans le code utilisé ici, le mot ***int*** fait référence à un nombre entier (« **int**eger »). La valeur de *DELpin* ne peut donc pas être décimale. Si l'on tente de lui donner une valeur décimale, seul l'entier sera retenu. La valeur n'est donc pas arrondie.

On écrit le mot spécifiant le type de variable uniquement lors de la déclaration de celle-ci.

Nom des variables

- Les noms des variables sont sensibles à la casse : DELpin, delPin, DelPin sont donc trois variables différentes.
- On n'utilise pas d'accents ou de symboles (é, è, ç, û, #, …).
- Les espaces sont aussi interdits. Pour écrire plusieurs mots, on utilise le soulignement (_) ou encore on met une majuscule à chaque début de nouveau mot.
- Les chiffres sont permis, mais un nom ne peut pas être uniquement un nombre.
- Il est toujours mieux de choisir des noms de variables qui permettent de savoir facilement à quoi elles font références.

Portée des variables

Si une variable est définie au début du programme, elle est définie partout dans le programme.

Si une variable est définie à l'intérieur d'un bloc délimité par des accolades { }, elle existe dans ce bloc uniquement.

À quoi ça sert?

L'avantage de définir la variable *DELpin* au début au lieu d'écrire la valeur « 9 » à chaque fois qu'elle est utilisée est que si l'on veut modifier la valeur de la broche reliée au circuit, on change seulement la valeur une fois en haut du programme.

Défi :

Changer le nom de la variable *DELpin* pour autre chose et faire fonctionner le programme correctement.

```
void setup() {
}
```

Ceci est l'un des deux fonctions de base de tout programme Arduino. Elle est toujours présente, sinon le programme ne se compile pas et il n'y en a qu'une seule.
La fonction *setup* est exécutée une seule fois par la carte Arduino au départ ou encore à toutes les fois où l'on appuie sur le bouton de réinitialisation (« reset »).

À quoi ça sert?

On utilise la fonction *setup* pour initialiser les valeurs de certaines variables, initialiser le mode des différentes broches de la carte qui sont utilisées. Ces commandes ne seront effectuées qu'une seule fois.

Attention : les variables déclarées dans la fonction *setup* n'existeront pas dans la fonction *loop* qui suit.

En général, on se sert d'une fonction pour rassembler une série de commandes qui est répétée à quelques reprises dans le programme. Cela permet de simplifier les choses.

Fonctions

En programmation, une fonction est un bloc de commandes qui transforme une entrée en une sortie. Dans le langage utilisé avec Arduino, elles ont la forme suivante :

```
type nomDeFonction(paramètres d'entrée) {
}
```

Le ***type*** indique le type de la valeur retournée par la fonction. Pour la fonction *setup*, il n'y a pas de valeur retournée. C'est pourquoi elle est de type ***void*** (vide, néant)***.***

La fonction *setup* n'a pas de paramètres d'entrée, la parenthèse est donc vide. Elle doit toutefois être présente.

Toutes les commandes qui sont incluses à l'intérieur des accolades seront effectuées une seule fois lors de la mise sous tension de la carte Arduino ou en appuyant sur le bouton « reset ».

```
pinMode(nomDePin, OUTPUT);
```

La fonction *pinMode* fait partie des fonctions de base en programmation Arduino. Elle sert à configurer une broche en mode *Entrée*, « INPUT », ou *Sortie*, «OUTPUT».

En mode Entrée, il sera possible de lire des valeurs (signaux) provenant de la broche. Typiquement, on se sert de ce mode pour faire la lecture de valeurs données par un capteur. Par défaut, les broches sont configurées en mode *INPUT*. Il n'est donc pas nécessaire d'utiliser la fonction *pinMode* pour configurer une broche en mode INPUT. Cela permet, par contre, de connaître l'utilisation de la broche qui sera fait dans le programme.

En mode Sortie, on peut envoyer un signal par la broche à un moteur, une DEL, … Le signal consiste à envoyer du courant électrique par la broche. Les valeurs de courant permises peuvent aller jusqu'à 40 mA. Par contre, ce n'est généralement pas indiqué d'alimenter un moteur de cette façon.

```
void loop() {
}
```

La fonction *loop* est similaire à la fonction setup. Les deux différences sont qu'elle sera exécutée après la fonction setup et que le bloc de commandes sera répété à l'infini.

```
digitalWrite(nomDePin, HIGH);  et digitalWrite(nomDePin, LOW);
```

La fonction *digitalWrite* permet d'envoyer un signal électrique à la broche *nomDePin*. La tension électrique fournie par la broche sera d'environ 5V lorsque la valeur *HIGH* est utilisée et de 0V pour la valeur *LOW*.
La tension électrique ne change pas tant que la fonction n'est pas utilisée pour la modifier.

```
delay(temps);
```

Cette fonction introduit un délai d'une durée *temps* dans le programme. Le temps d'attente doit être donné en millisecondes.

Défis :

- Changer la broche (PIN) de branchement de la DEL et ajuster le programme en conséquence.
- Modifier le programme pour faire d'autres rythmes de clignotement!
- Faire clignoter la DEL une seule fois! (Deux façons possibles)
- Réussir à faire clignoter la DEL pour obtenir le code morse de S.O.S.!