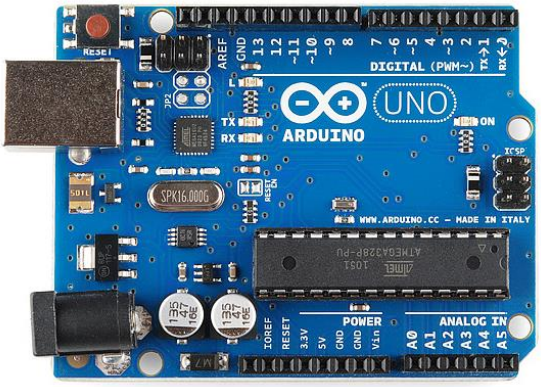
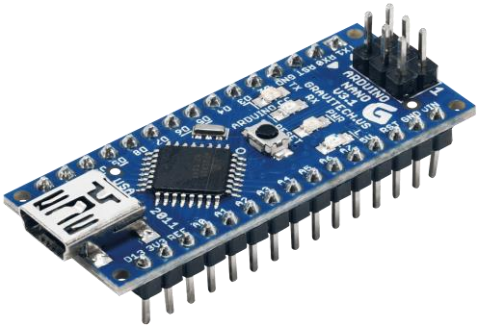
ARDUINO : LED

Arduino est une carte de prototypage open-source, c'est-à-dire qu'elle permet de réaliser des prototypes, d'essayer des programmes, de le modifier, de les améliorer, de les re-tester,… Avant de passer à la manipulation, un peu de présentation pour apprendre à bien se connaitre.

Le fait qu'elle soit open-source permet à la communauté d'améliorer notamment les logiciels qui nous permettrons d'interagir avec ces cartes.



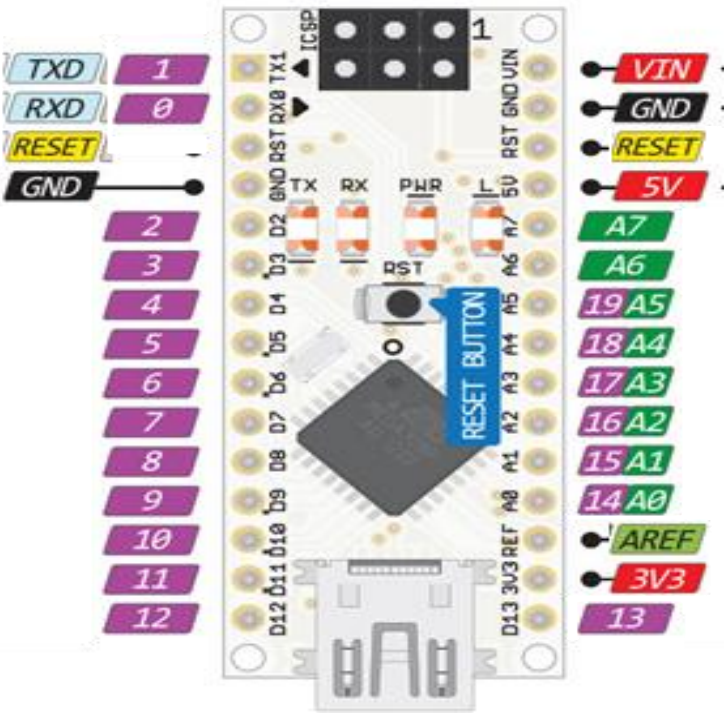
Il existe différentes sortes de cartes que l'on choisira en fonction du projet à réaliser (Consommation énergétique, encombrement, nombre de connexions, de capteurs…). Les plus courantes sont les Arduino UNO qui sont un bon compromis entre les Nano et les Mega en passant par la mini.



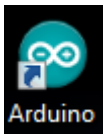
Nous travaillerons dans cet exemple et très souvent au labo avec des cartes Arduino Nano. Elles possèdent le même nombre de connexions (Pins) que le UNO mais sont plus facilement connectables sur une plaque de prototypage (aussi appelée BreadBoard).

Chacune des pins (D2 à D13 et A0 à A7) peut être utilisée comme une entrée (mesure d'un capteur) ou comme sortie (allumage d'une lampe). Les pins commençant par D (digital) sont pins numériques, c'est-à-dire des pins « tout ou rien » alors que les pins commençant par A sont des pins dites analogiques, autrement dit, elles peuvent dire si le signal d'entrée est plus ou moins « fort ».

Pour ce qui est des autres pins et du fonctionnement particulier de chacune d'entre-elles nous le verrons au fur et à mesure des fiches d'initiation.

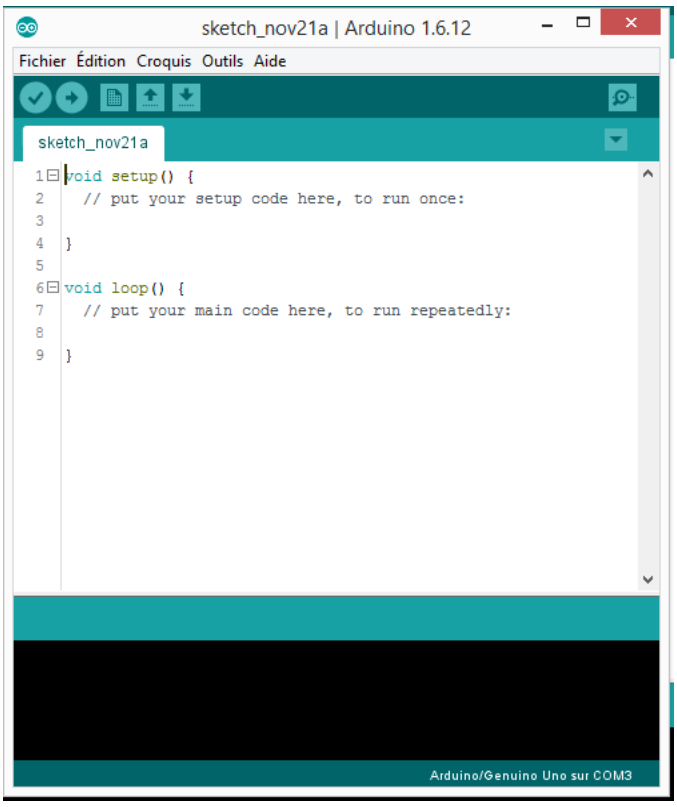


Maintenant que nous avons parlé du côté matériel (HardWare) il est nécessaire de parler du côté logiciel (SoftWare) puisque c'est lui qui va nous permettre de programmer et d'envoyer nos programmes dans l'arduino via son câble USB.



Lancez le programme situé sur le bureau et commençons à nous familiarisé au logiciel. Le software se met alors à se charger :

Vous arrivez à présent sur l'interface de programmation de l'arduino. Elle se décompose en plusieurs parties mais dans un premier temps c'est la structure du programme qui retiendra notre attention.




```
1  void setup() {
2    // put your setup code here, to run once:
```

Cette fonction du code correspond à l'initialisation, elle ne sera réalisée qu'une seule fois au démarrage de votre microcontrôleur. Nous pourrons par exemple y déclarer l'utilisation des pins de l'Arduino. Elle commence à l'ouverture de l'accolade et de terminera par une fermeture d'accolade.

Pour ce qui est de l'utilisation de bibliothèques particulières comme l'utilisation d'un afficheur, d'un thermomètre(…) mais aussi pour les variables ou les constantes, il est de coutume de les déclarer avant cette fonction.

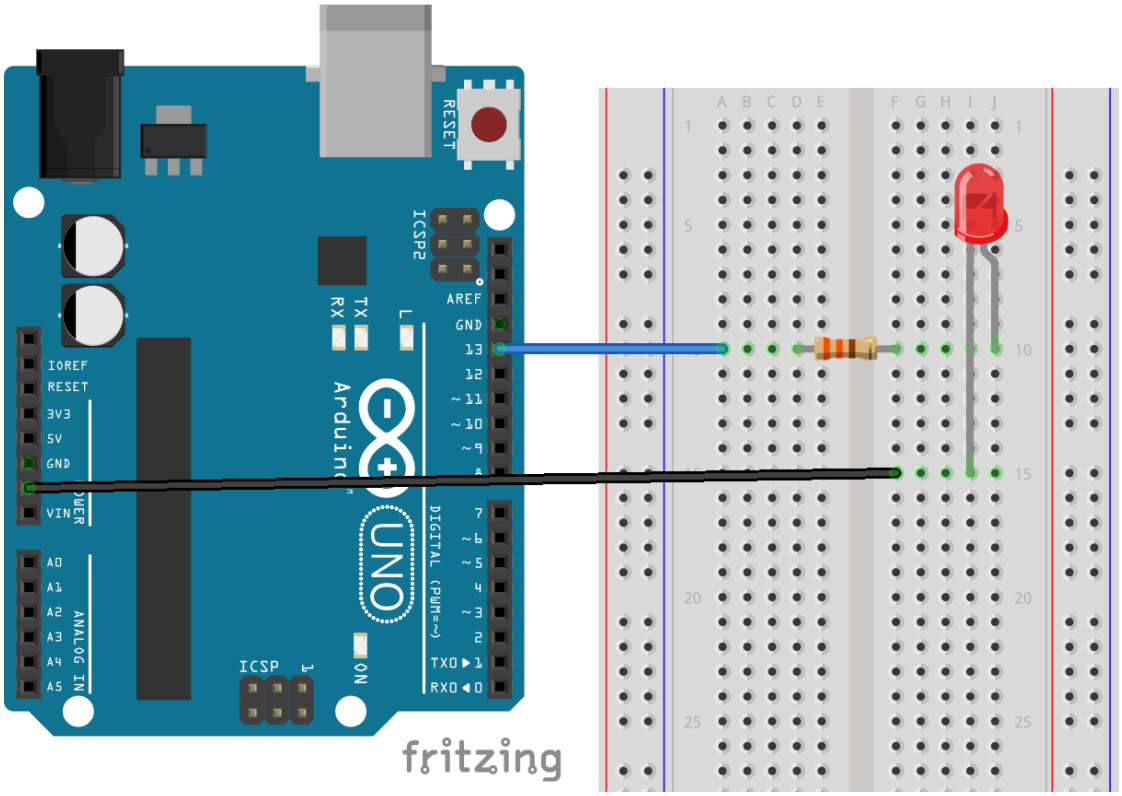
En dessous de cette fonction, nous pouvons voir la fonction loop qui elle est une boucle infinie :

```
6  void loop() {
7    // put your main code here, to run repeatedly:
```

Cette fonction est limitée également par des accolades et ne s'arrêtera jamais ! Le microcontrôleur la parcourt, réalise ce que lui est demandé puis recommence.

Nous allons à présent commencer par un exemple simple, le clignotement d'une LED (ou diode électroluminescente). La LED est polarisée, c'est-à-dire qu'elle possède une patte à brancher du côté positif (+ cathode) et une autre côté négatif (- anode). Sauf cas particulier, la patte la plus longue est la cathode. Pour éviter tout problème, il est préférable de ne pas brancher directement une LED à l'Arduino mais de rajouter une petite résistance pour limiter la quantité de courant qui pourra traverser.

La théorie étant faite, passons à la pratique. Commençons par relier la cathode de la LED à la pin 13 de l'Arduino et l'anode à une patte de la résistance (non polarisée). La patte de la résistance sera elle reliée à la pin GND de l'Arduino qui signifie Ground (terre) et qui n'est rien d'autre que le pôle négatif du microcontrôleur. Vous obtenez le montage ci-dessous :



Avant de brancher l'Arduino à l'ordinateur, il faut à présent préparer le programme.

```
1  int led = 13;  //Pin de la borne positive de la LED
```

Pour débuter, une bonne habitude est définir les constante. Ici nous allons attribuer un nom original de variable à notre diode : led. Cette variable contient la pin sur laquelle est branchée la Led, ici la borne 13.

Remarque : Vous devez finir chacune des lignes par un point virgule sans quoi vous rencontrerez des problèmes lors de l'envoi du programme sur le microcontrôleur.

```
3  void setup() {
4    pinMode(led, OUTPUT);
5  }
```

Ensuite nous allons initialiser le mode de cette pin. Comme la pin commandera la sortie, il faut la définir comme étant une sortie (OUTPUT).

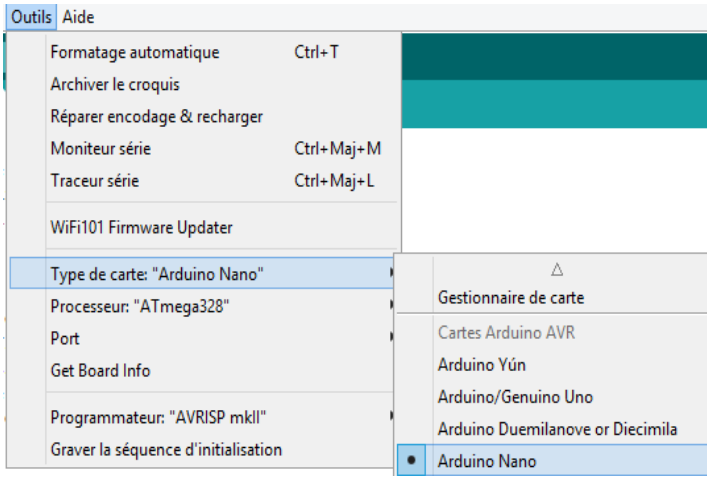
```
7  void loop() {
8    digitalWrite(led, HIGH);
9    delay(2000);
10   digitalWrite(led, LOW);
11   delay(2000);
12 }
```

Passons au corps du programme, à la fameuse boucle infinie. Pour mettre notre pin au niveau haut, c'est-à-dire allumée, il faut dire à l'Arduino « d'écrire » sur la pin digital correspondant à notre Led la valeur haute (HIGH). Si on alterne niveau haut et niveau bas, la fréquence sera telle qu'on ne la verra pas clignoter, pour cela nous rajoutons un « delay » de 2000 millisecondes (2 secondes) durant lequel elle restera allumer avant de s'éteindre à nouveau deux secondes avant que la boucle ne recommence et que la led s'allume à nouveau.

Envoyer le programme vers l'Arduino :

Votre programme fini, vérifiez que le programme reconnait bien votre carte Arduino, pour cela rendez-vous dans Outils puis choisissez votre carte. Vérifiez également que le Port est bien différent du COM1.

Pour envoyer votre programme vous n'avez plus qu'à le téléverser, pour cela repérez les deux ronds en haut à gauche et appuyez sur : 



Erreur à ne pas commettre : n'oubliez pas le second délais sans quoi votre Led s'éteindra mais la boucle recommançant instantannéement elle se rallumera sans que vous ne l'ayez vu s'éteindre.

```
7  void loop() {
8    digitalWrite(led, HIGH);
9    delay(2000);
10   digitalWrite(led, LOW);
11 }
```

Remarque : La pin 13 n'a pas été choisie au hasard, elle possède déjà une mini-led qui sert de témoin. Vous pouvez voir directement sur l'Arduino une diode s'allumer et s'éteindre au même rythme que votre diode.

D E F I : Réaliser un feu rouge de signalisation routière où le rouge restera allumé 10 secondes avant de passer au vert pendant également 10 secondes puis au orange pendant seulement 3 secondes.