Algo Applications

Application de l'algorithmique à la programmation Arduino

Cours I - Installation et introduction à la programmation Arduino

ESME
INNOVATIVE ENGINEERING

Carte électronique et circuit imprimé

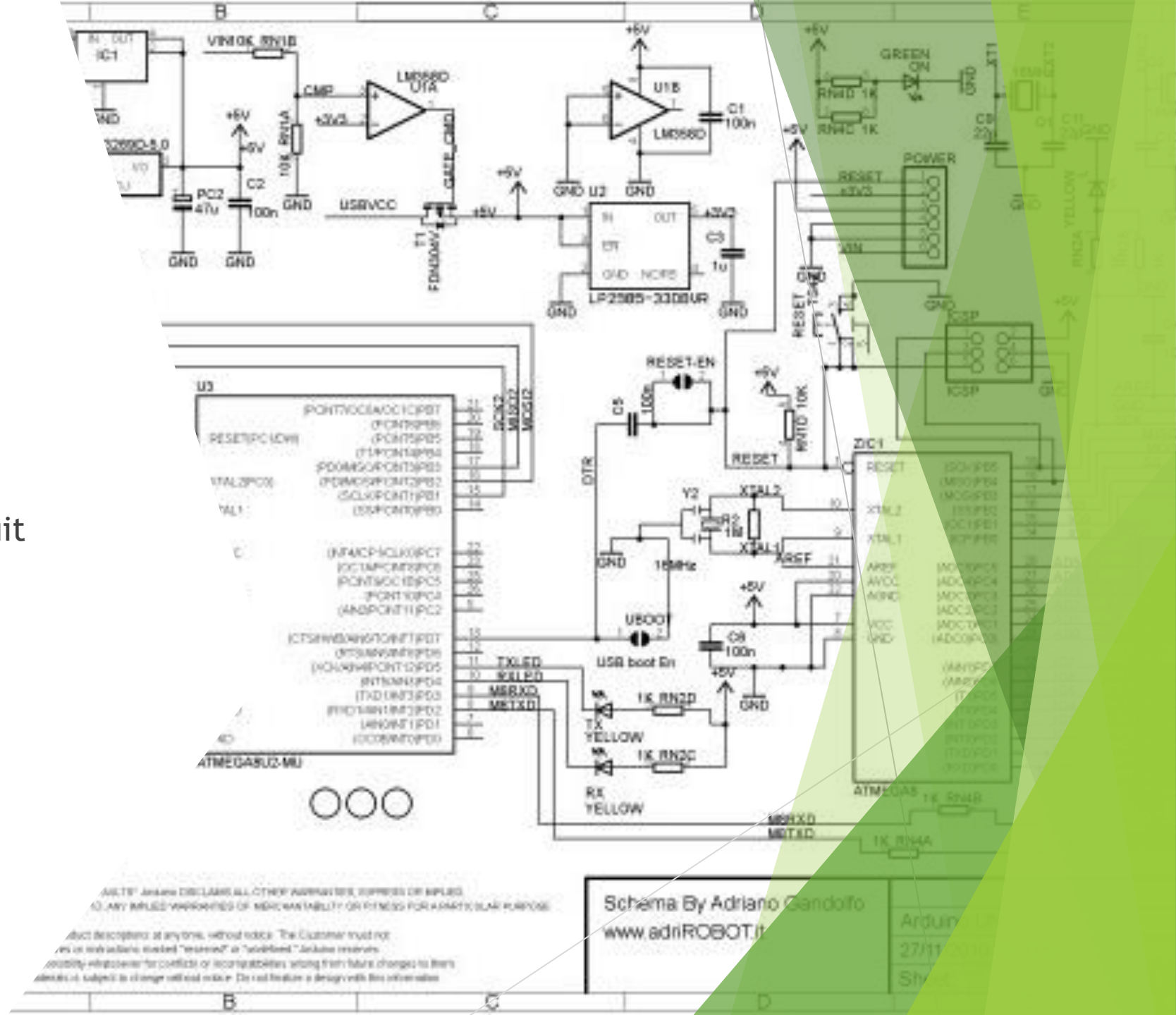
- **Un circuit imprimé** (ou PCB de l'anglais *printed circuit board*) est un **support**, en général une **plaque**, **permettant de maintenir et de relier électriquement un ensemble de composants électroniques entre eux**, dans le but de réaliser un circuit électronique complexe. On le désigne aussi par le terme de **carte électronique.**

- Une **carte électronique** permet la mise en relation électrique de composants électroniques.

- Chaque piste relie tel composant à tel autre, de façon à créer un **système électronique** qui fonctionne et qui réalise les opérations demandées.

Schéma électrique

- Evidemment, tous les composants d'une carte électronique ne sont pas forcément reliés entre eux.
- Le **câblage des composants** suit un **plan spécifique** à chaque carte électronique, ce plan se nomme le **schéma électronique**.



Arduino

- Une **carte électronique programmable** et un logiciel multiplateforme.

- Accessible à tout un chacun dans le but de créer facilement des **systèmes électroniques**.

Arduino

- **Arduino** est une marque qui couvre des **cartes électroniques** sur lesquelles se trouve un microcontrôleur.

- Les schémas de ces cartes électroniques sont publiés en **licence libre**. Cependant, certaines composantes, comme le **microcontrôleur** par exemple, ne sont pas sous licence libre.

- Licence libre, signifie que c'est une matériel dont les plans ont été rendus publics de façon que quiconque puisse les fabriquer, modifier, distribuer et utiliser.

- Le **microcontrôleur** peut être programmé pour analyser et produire des signaux électriques, de manière à effectuer des **tâches très diverses** comme la domotique (le contrôle des appareils domestiques — éclairage, chauffage…), le pilotage d'un robot, de l'informatique embarquée, etc.

Microcontrôleur

- Le ***Microcontrôleur*** (en notation abrégée **µc**, ou **uc** ou encore **MCU** en *anglais*) est un circuit capable **d'exécuter un programme**.

- Un microcontrôleur **rassemble les éléments essentiels d'un ordinateur** : processeur, mémoires (mémoire morte et mémoire vive), unités périphériques et interfaces d'entrées-sorties.

- Ils possèdent un **coût réduit** par rapport aux microprocesseurs polyvalents utilisés dans les ordinateurs personnels.

Carte programmable

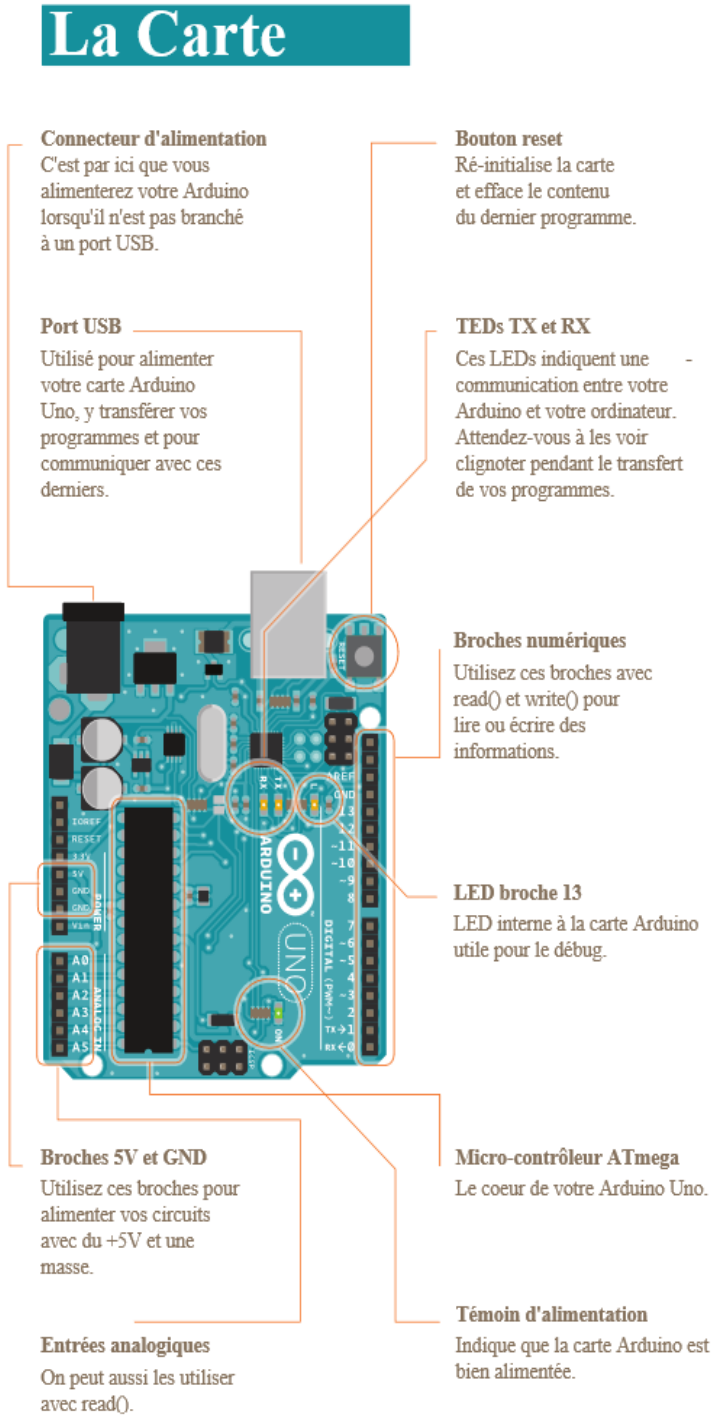
- La carte Arduino est une carte électronique *qui ne sait rien faire* sans qu'on lui dise *quoi faire*. Elle a besoin d'un **programme** pour fonctionner.
- Pour programmer la carte Arduino on utilise un logiciel spécialisé appelé Arduino IDE basé sur le langage C.
- Mais il est possible de programmer la carte à l'aide d'autres langages, par exemple le langage Python.
- La documentation pour programmer les **cartes Arduino** à l'aide du C est plutôt abondante sur internet de plus il existe des outils annexes comme le fait de programmer une carte via le navigateur, tous basés sur le C Arduino.
- C'est pourquoi nous l'avons choisi.

Applications possibles

- Voici une liste non exhaustive des applications possibles réalisées grâce à Arduino :
- contrôler des appareils domestiques
- donner une "intelligence" à un robot
- réaliser des jeux de lumières
- permettre à un ordinateur de communiquer avec la carte électronique et récupérer les informations issues de différents capteurs
- télécommander un appareil mobile (modélisme)
- etc.
- Il y a une infinité d'autres utilisations, vous pouvez simplement chercher sur votre moteur de recherche préféré ou sur *Youtube* le mot "Arduino" pour découvrir les milliers de projets réalisés avec !

Présentation de la carte

- Connecteur d’alimentation
- Bouton reset
- Port USB
- LEDs TX et RX
- Broches numériques
- LED broche 13
- Broche 5V et GND
- Micro-contrôleur Atmega
- Entrées analogiques
- Alimentation

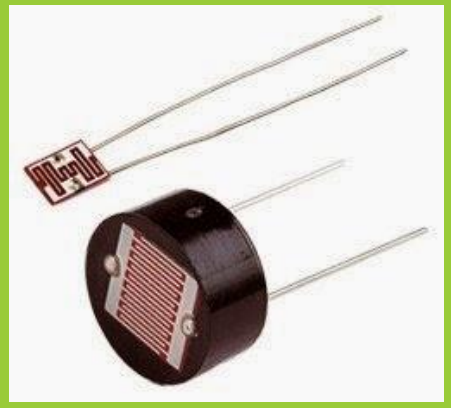


Communiquer avec la carte

- Si vous avez découvert l'électronique à l'école, on vous a sans doute appris à construire des circuits en utilisant des composants spécifiques.
- Ces circuits vendus en kits dédiés font parfaitement leur travail, mais ils ne peuvent pas faire grand-chose d'autre.
- C'est là qu'interviennent les microcontrôleurs. Ce sont de minuscules ordinateurs. Ils permettent à cette circuiterie de se comporter de manière bien plus sophistiquée et polyvalente.
- Le microcontrôleur est le cerveau de votre système, mais il a besoin de données pour sentir des choses ou pour agir dessus. Pour ce faire, il utilise **des entrées et des sorties.**

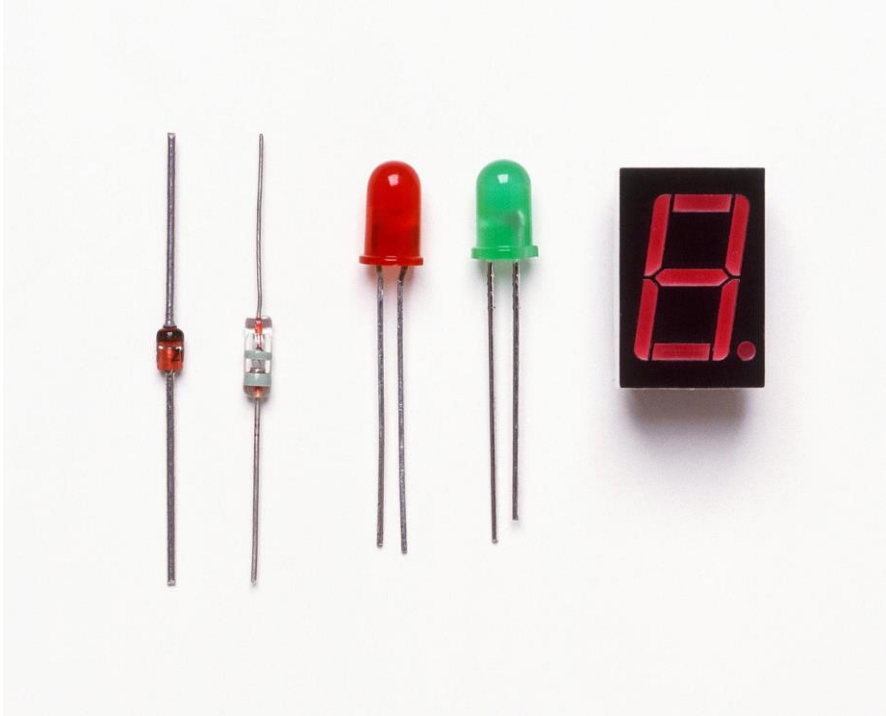
Les entrées

- Les entrées sont les organes des sens de votre Arduino. Elles l'informent sur ce qui se passe dans le monde réel. Sous sa forme la plus élémentaire, une entrée peut être un interrupteur ou bouton pressoir



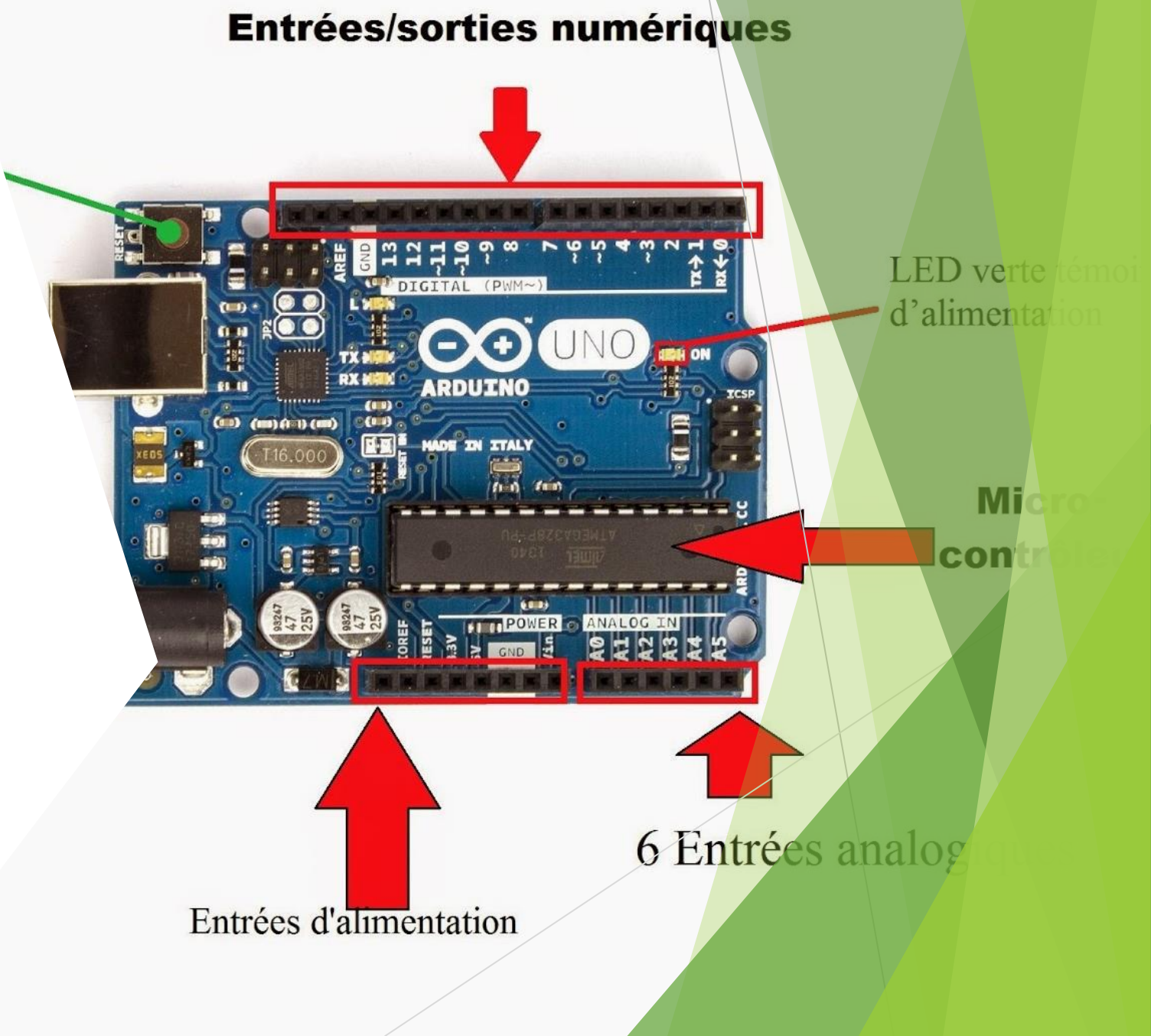
Les sorties

- Les sorties permettent à votre Arduino d'agir sur le monde réel d'une manière ou d'une autre. Elles témoignent d'un état ou donnent des informations à l'utilisateur. Sous sa forme la plus élémentaire, une sortie peut être représentée par une diode , un moteur ou un écran à cristaux liquides (LCD).

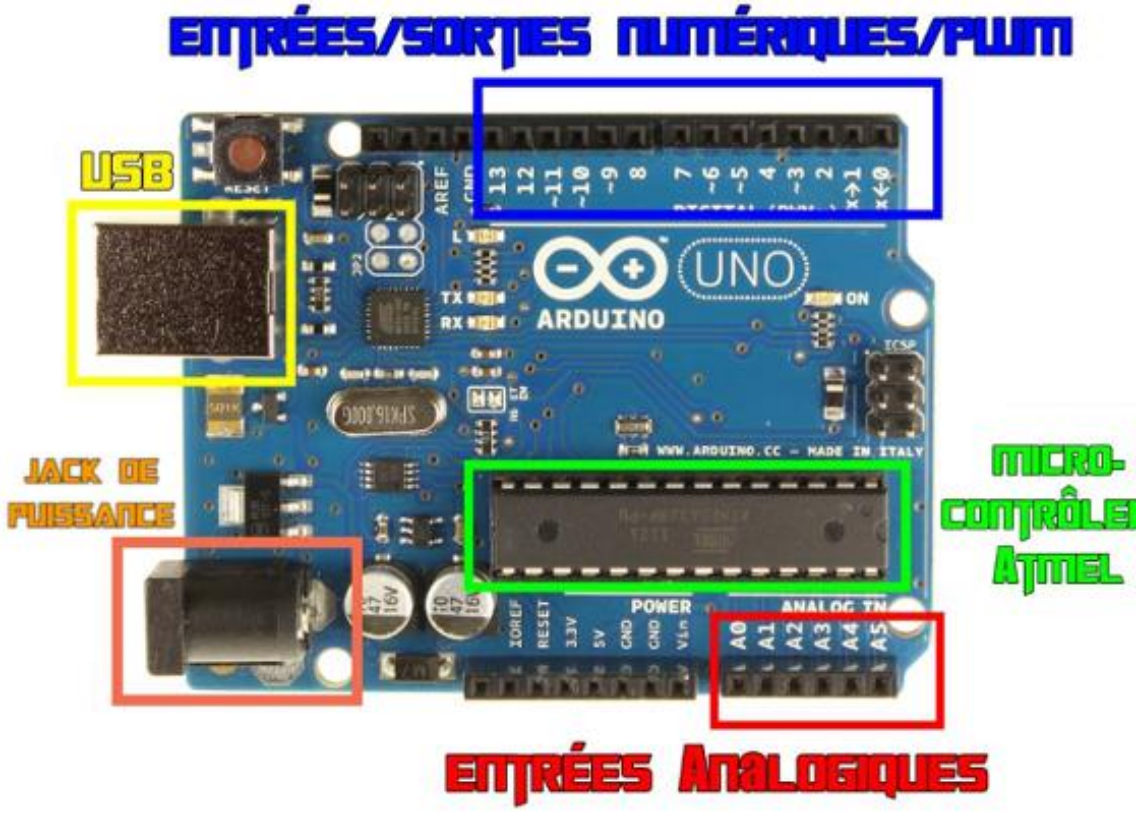


Broches numériques

- Les *broches numériques*, servent à recevoir des signaux numériques. Cela signifie que le signal ne possède que l'un des deux états haut ou bas. En termes électriques, cela signifie une tension de 0 volts ou de 5 volts, sans tension intermédiaire.
- Elles sont identifiées par le sigle *digital* et leur numéro va de 0 à 13.



Broches analogiques



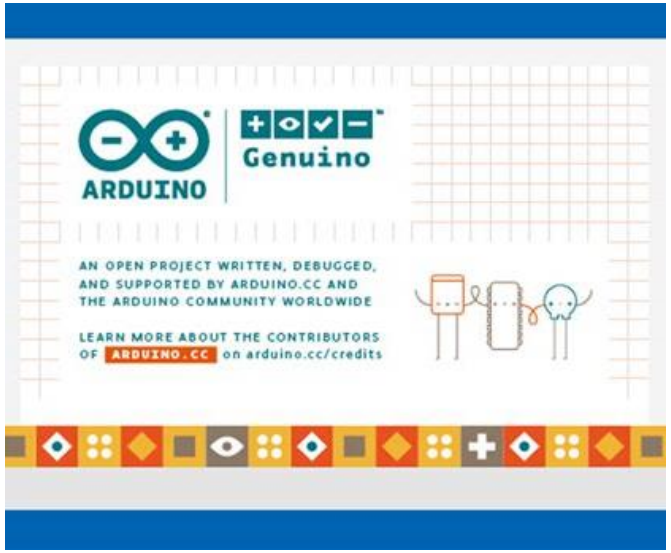
- Les *broches analogiques* servent à recevoir des valeurs analogiques. Une valeur est dite analogique lorsqu'elle peut prendre une valeur parmi toutes celles d'une plage prédéfinie. Dans le cas présent, la plage va de 0 à 5 volts, et la valeur peut se situer n'importe où entre les deux bornes : 0.1 volt, 1.2 volt, 4.9 volt, etc.

- Les broches analogiques prennent les valeurs en entrée (depuis la circuit vers la carte) pas en sortie. Nous verrons plus tard comment simuler les sorties analogiques.

Installer Arduino IDE

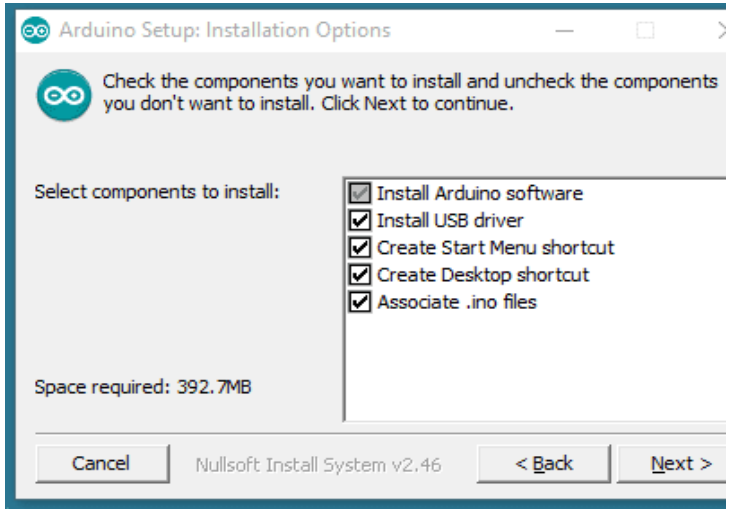
- Avant de commencer à travailler avec votre carte Arduino, vous devez installer un logiciel indispensable : **Arduino IDE.**

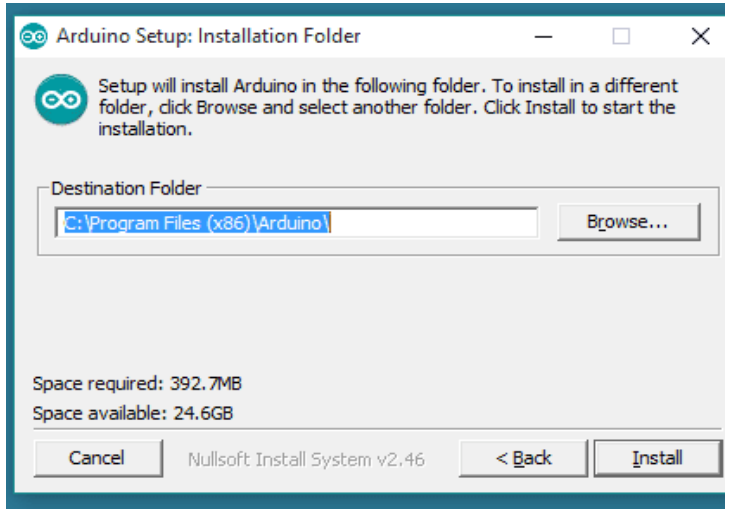
- Le logiciel Arduino vous permet d'écrire, de tester et de télécharger des programmes. On trouve des versions de ce logiciel Arduino pour Windows, Macintosh OS X et Linux.

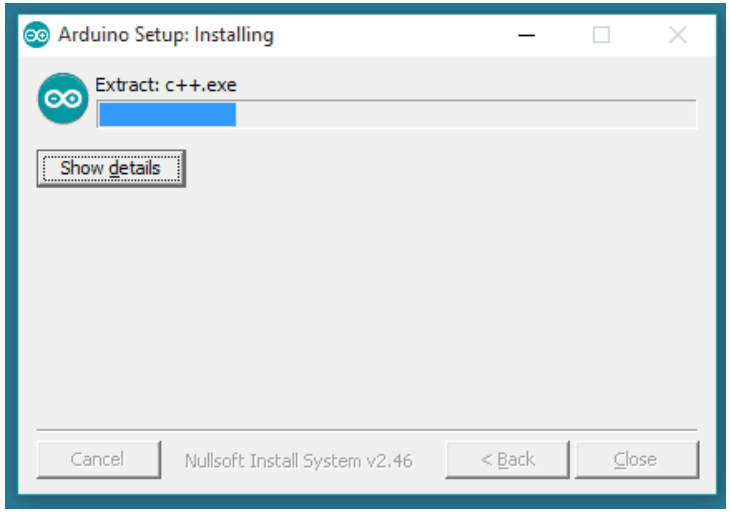


Installer Arduino IDE (Windows)

- Allez sur la page https://www.arduino.cc/en/software
- Cliquez sur la version qui correspond à votre système d'exploitation (Windows)
- Une fenêtre de téléchargement s'ouvre sur un fichier .zip. Téléchargez-le
- Plus d'infos sur https://www.arduino.cc/en/Guide/Windows







Installer Arduino (Windows)

- Quand le téléchargement se termine, autorisez l'installation de nouveaux drivers sur votre système (si on vous le demande) et sélectionnez les composants à installer (image1).
- Choisissez l'emplacement d'installation (ou laissez celui par défaut – image 2).

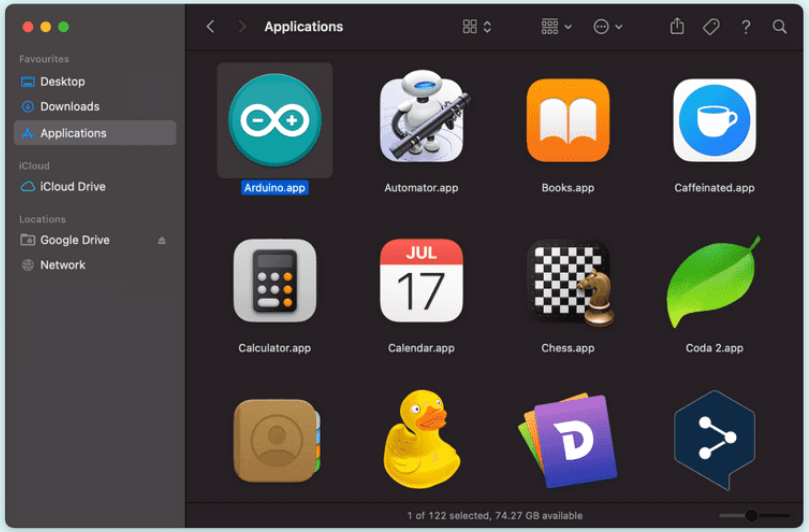
Installer Arduino IDE (Mac)

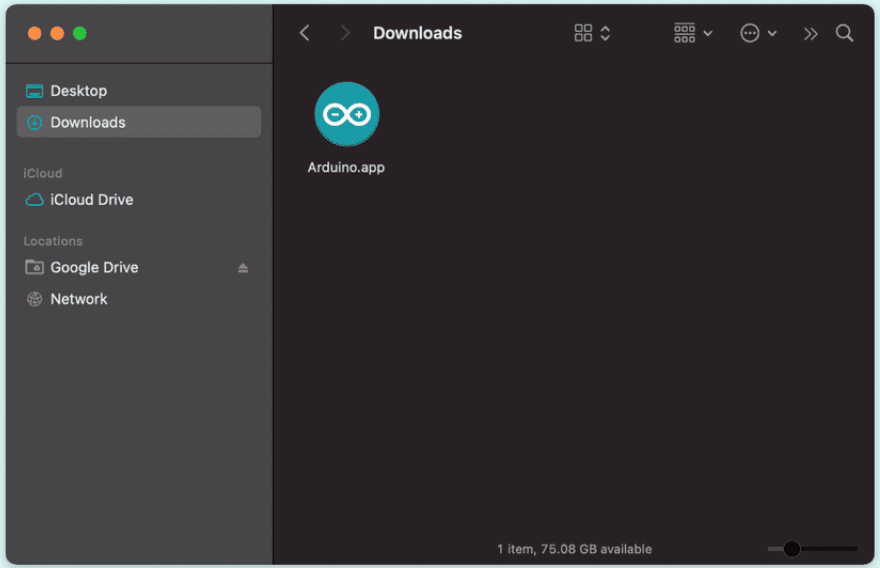
- Allez sur la page https://www.arduino.cc/en/software
- Cliquez sur la version qui correspond à votre système d'exploitation (Mac)
- Une fenêtre de téléchargement s'ouvre sur un fichier .zip. Téléchargez-le
- Plus d'infos sur https://www.arduino.cc/en/Guide/Windows

Installer Arduino (Mac)

- Le fichier est au format .zip. Si vous utilisez Safari, il sera extrait automatiquement.

- Copiez le fichier Arduino.app depuis le dossier « Téléchargements » vers le dossier « Applications »



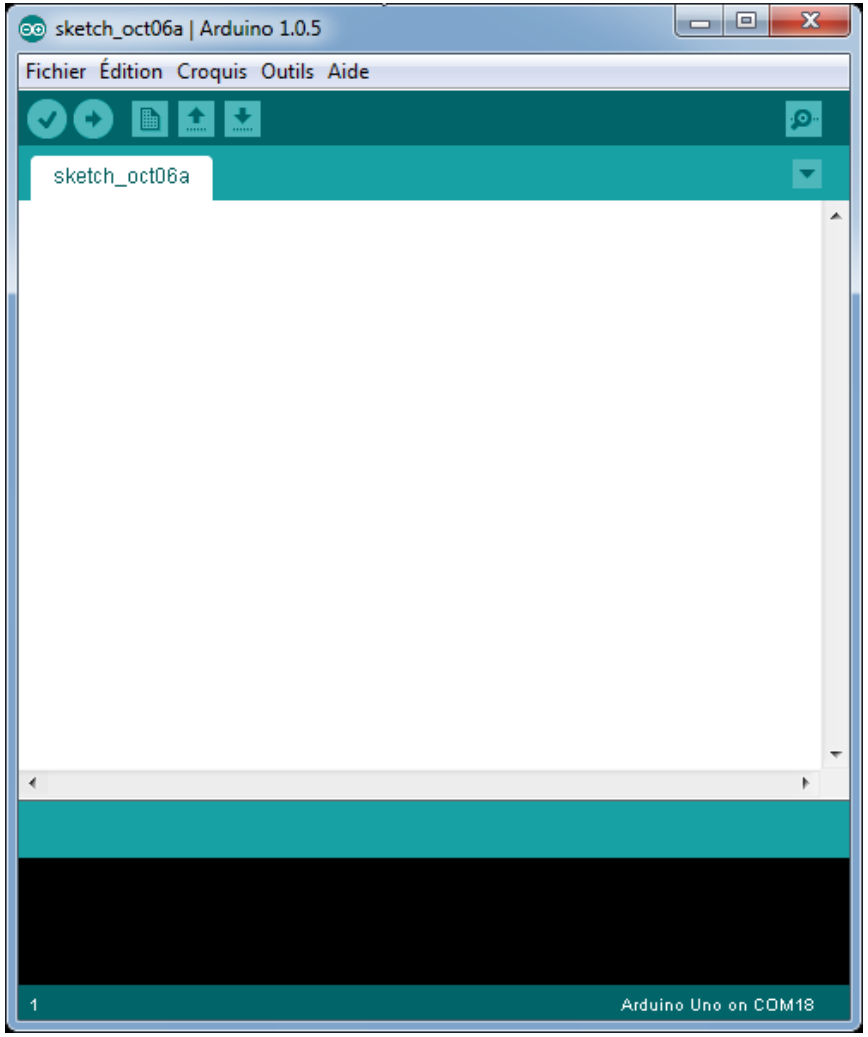


Installer Arduino IDE (Linux)

- Pour installer Arduino IDE sur Linux rendez vous sur https://playground.arduino.cc/Learning/Linux/

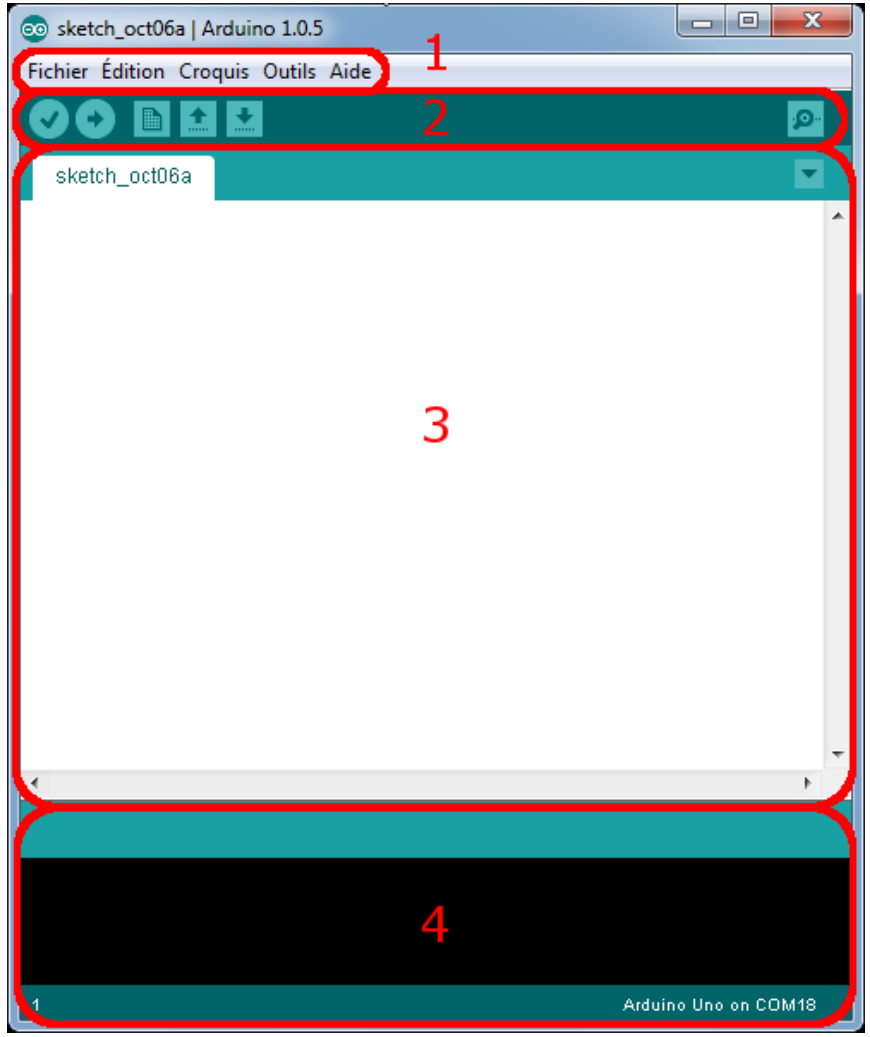
Présentation du logiciel

- Voici l’interface de départ du logiciel après lancement



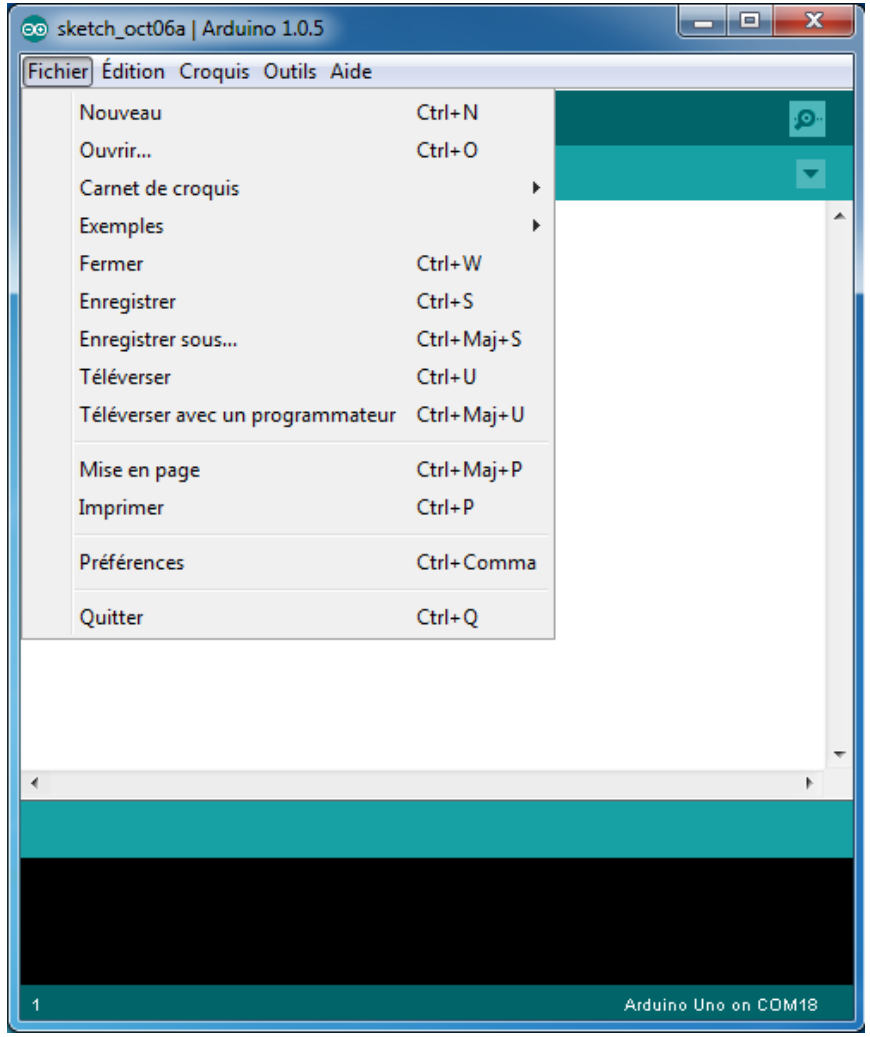
Interface du logiciel

- 1. Menu
- 2. Barre d’outils
- 3. Bloc dans lequel on programme
- 4. Console de débogage. C’est grâce à elle qu’on va pouvoir corriger les erreurs du programme.



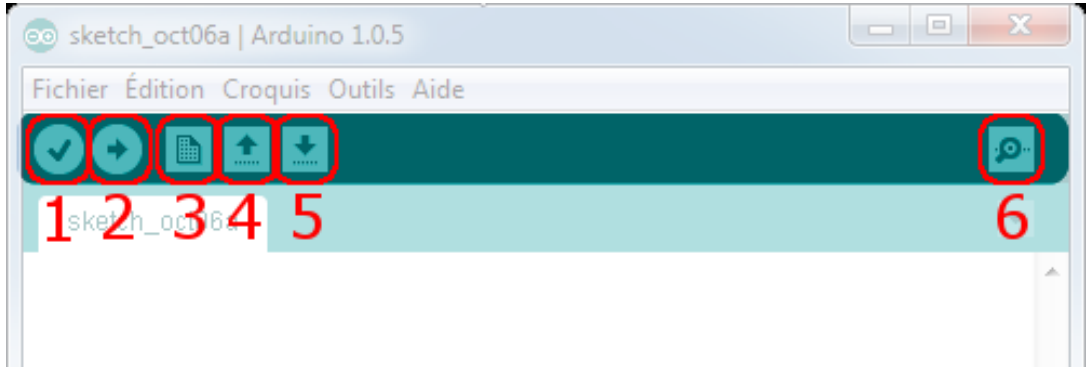
Le menu

- Le menu dispose d'un certain nombre d'options très utiles.
  - Carnet de croquis : Il regroupe tous les programmes que vous avez fait.
  - Exemples : Ce sont des exemples de programmes déjà écrits que vous pouvez directement utiliser.
  - Téléverser : Permet d'envoyer le programme sur la carte Arduino.
  - Téléverser avec un programmateur : Même chose mais avec l'aide d'un programmateur.
  - Préférences : Permet le réglage de certains paramètres.



Les boutons

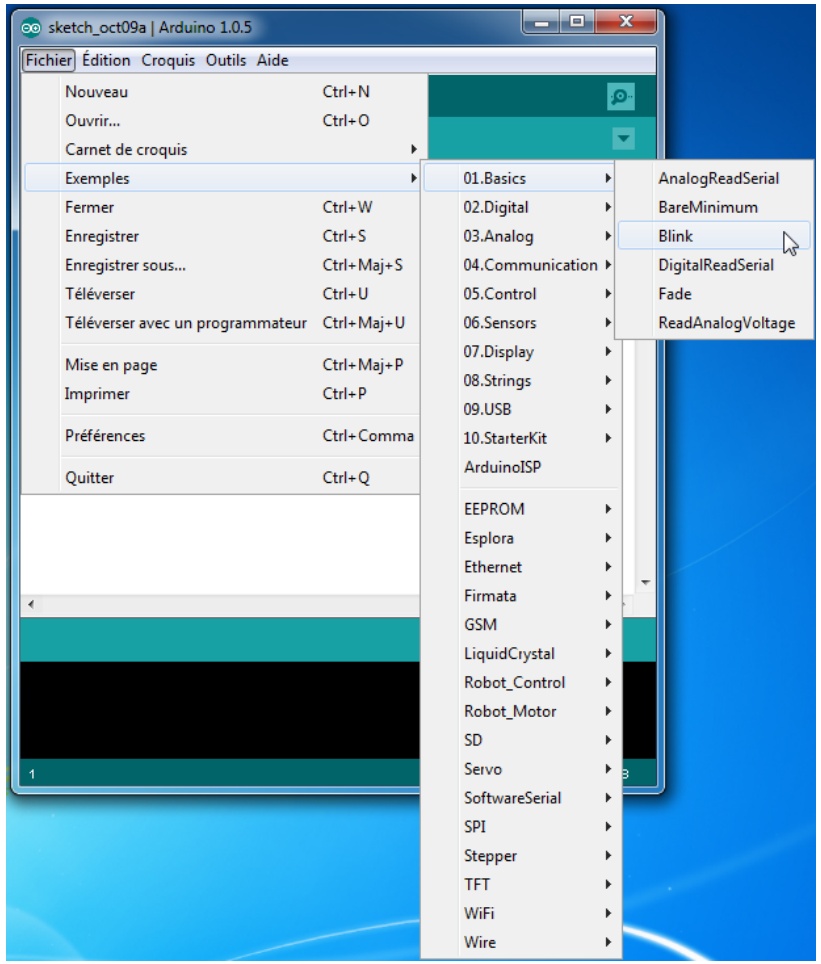
- Bouton 1 : Permet de vérifier le programme. Les erreurs peuvent être remontées dès l'appui sur ce bouton.
- Bouton 2 : Charge (téléverse) le programme dans la carte Arduino.
- Bouton 3 : Crée un nouveau fichier
- Bouton 4 : Ouvre un nouveau fichier
- Bouton 5 : Enregistre le fichier
- Bouton 6 : Ouvre le moniteur série (on verra plus tard à quoi il sert)



Tester le matériel

- Avant de programmer il faut tester le bon fonctionnement de la carte.

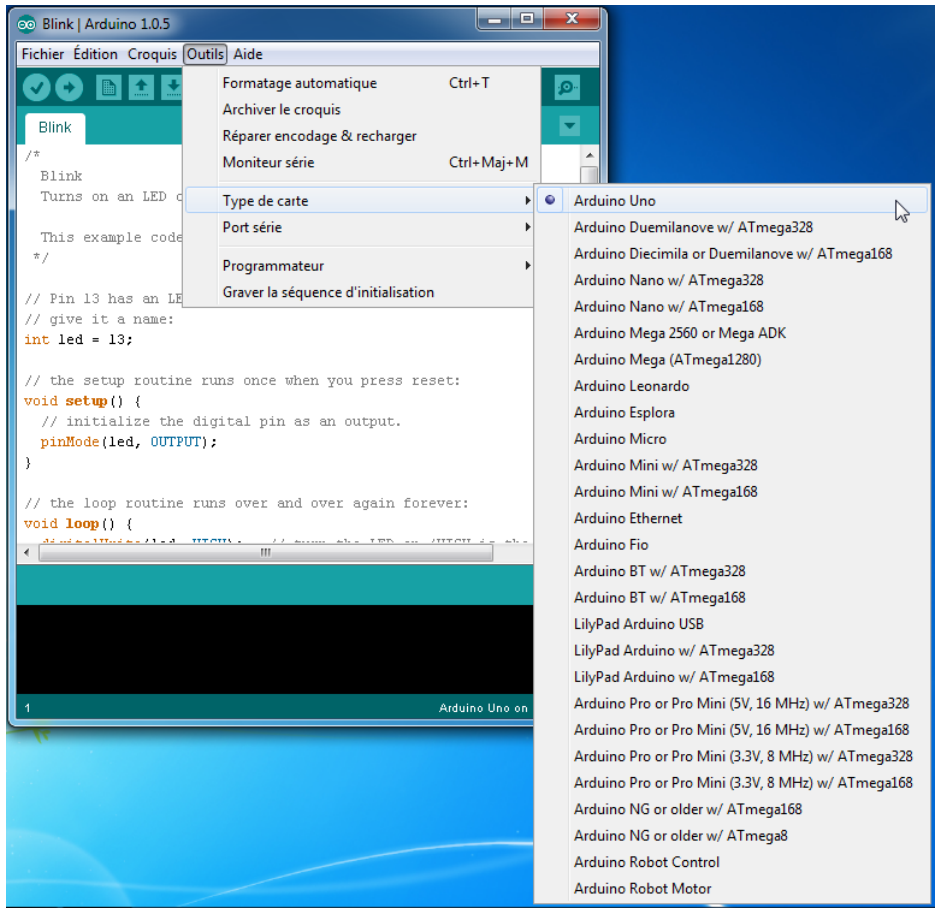
- le logiciel Arduino contient des exemples de programmes. Nous allons en utiliser un pour tester la carte.



Ouvrir un programme

- Nous allons choisir un exemple tout simple qui consiste à faire clignoter une LED. Son nom est *Blink* et vous le trouverez dans la catégorie Basics.

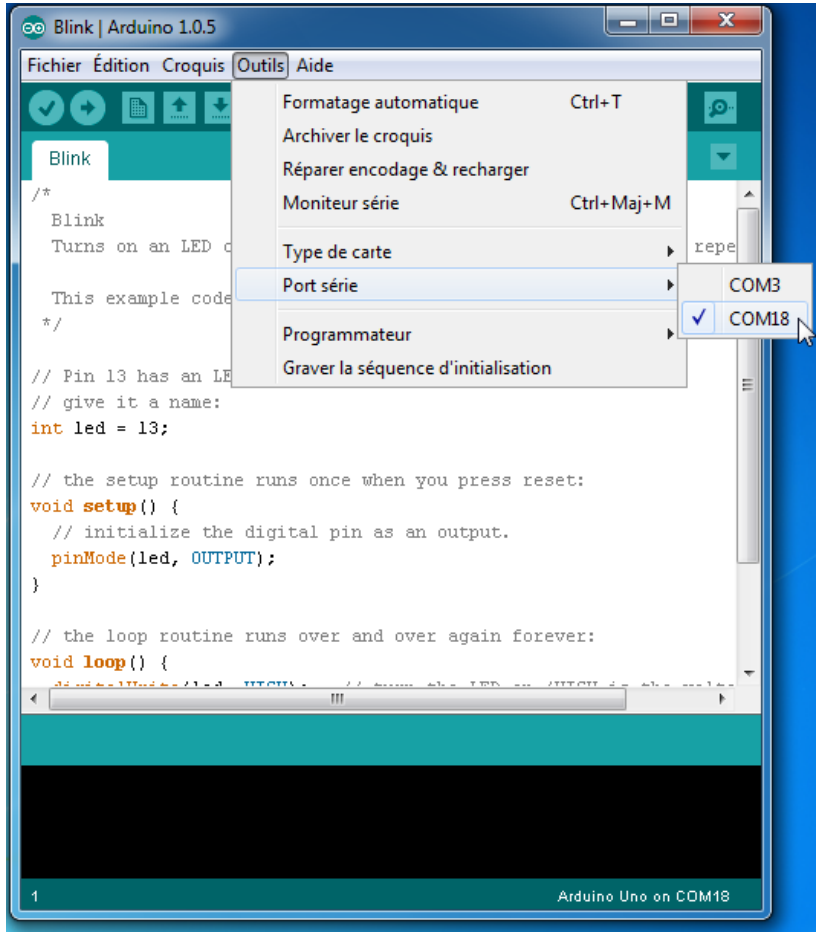
- Une fois que vous avez cliqué sur *Blink*, une nouvelle fenêtre va apparaître. Elle va contenir le programme *Blink*. Vous pouvez fermer l'ancienne fenêtre devenue inutile.



Choisir la bonne carte

- Avant d'envoyer le programme *Blink* vers la carte, il faut dire au logiciel quel est le nom de la carte et sur quel port elle est branchée. Le nom de votre carte est indiqué sur elle. s'agit de la carte "Uno".

- Allez dans le menu "Tools" ("outils" en français) puis dans "Board" ("carte" en français). Vérifiez que c'est bien le nom "Arduin Uno" qui est coché.

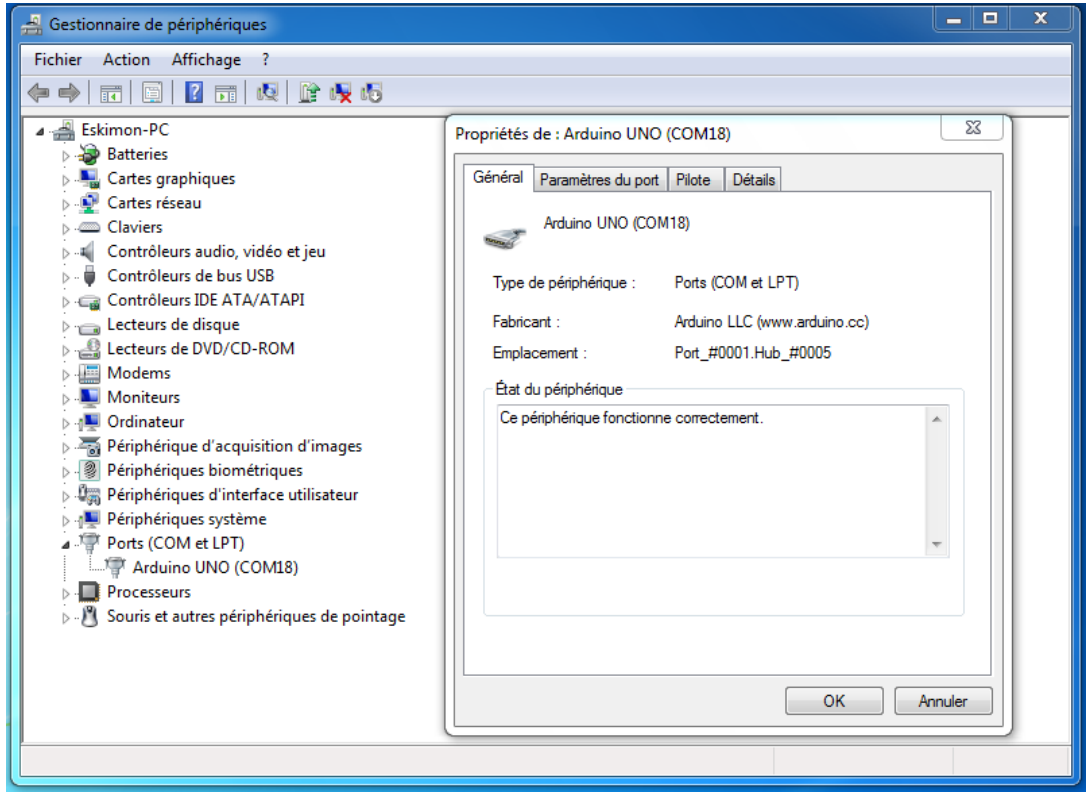


Choisir le bon port

- Allez dans le menu Outils, puis Port. Là, vous choisissez le port COMX, X étant le numéro du port qui est affiché.
- Si vous voyez plusieurs numéros s'afficher, en général prenez le plus grand possible.

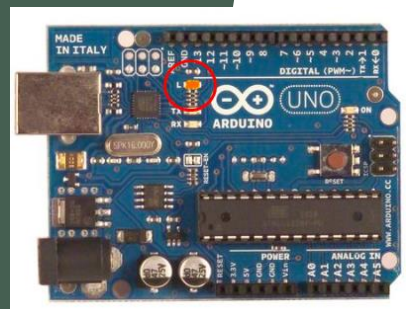
Le bon port

- En cas de doute sur votre numéro de port sur Windows vous pouvez aller dans le gestionnaire de périphérique qui se trouve dans le panneau de configuration. Regardez à la ligne « Ports (COM et LPT) » et là, vous devriez avoir Arduino Uno (COMX).



Charger le programme Blink

- Maintenant, il va falloir envoyer le programme dans la carte. Pour ce faire, il suffit de cliquer sur le bouton Téléverser, en jaune orangé sur la photo 1.
- Vous verrez tout d'abord le message "Téléversement…" pour vous informer que le programme est en train d'être compilé en langage machine avant d'être envoyé.
- Ensuite le message: "Téléversement terminé" signale que le programme à bien été chargé dans la carte. Si votre matériel fonctionne, vous devriez avoir une LED sur la carte qui clignote (en orange).



Quelques éléments de syntaxe Arduino

- Lorsque le logiciel Arduino lit un croquis, il l'exécute très rapidement ligne après ligne, dans l'ordre. La meilleure manière de comprendre le code est de le passer en revue de la même manière, mais très lentement.

- Arduino utilise le langage de programmation C, l'un des langages les plus utilisés de tous les temps. C'est un langage extrêmement puissant et versatile, mais il faut quelque temps pour s'y faire…

Dans les arcanes de Blink

- Une fois le croquis ouvert, vous devriez visualiser quelque chose de cet ordre :

```
void setup() {
    // initialize digital pin LED_BUILTIN as an output.
    pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(LED_BUILTIN, HIGH);
    delay(1000);
    digitalWrite(LED_BUILTIN, LOW);
    delay(1000);
}
```

Dans les arcanes de Blink (2)

- On y trouve plusieurs éléments courants des langages de programmation
  - Des commentaires
  - Une fonction **setup()**
  - Une fonction **loop()**

- D'autres éléments ont été passés sous silence comme les déclarations de variables. Examinons tout ceci en détail.

Les commentaires

- Notez comment les lignes de code sont comprises entre les symboles /* et */. Ces symboles délimitent le début et la fin d'un commentaire sur plusieurs lignes, ou bloc de commentaires.

- Dans l'exemple de Blink, le commentaire vous indique le nom du croquis, ce qu'il fait (allumer la LED durant une seconde, l'éteindre une seconde, et répéter), et contient aussi une note expliquant que le code est dans le domaine public.

- On peut aussi faire les commentaires sur une ligne. Pour ce faire, on utilise le // (voir code précédent).

Les variables

- Une variable est un nom qui symbolise une adresse numérique dans la mémoire du processeur. Cette adresse correspond à un contenu qui est une valeur. Ce contenu peut varier selon la manière dont le programme les utilise.

- En C, vous pouvez déclarer le type, le nom et la valeur de la variable avant le reste du code, un peu comme vous feriez l'inventaire préalable des ingrédients d'une recette.

```
int led = 13
```

Les fonctions

- Les deux sections qui suivent sont des fonctions qui commencent par le mot-clé ***void*** : **void setup** et **void loop**. Une fonction est un bloc de plusieurs lignes de code qui effectue une tâche, laquelle est souvent répétitive.

- Le mot ***void*** est utilisé quand la fonction ne retourne aucune valeur, et le mot qui suit est le nom de la fonction en question.

- Les deux déclaration de fonctions **void setup** et **void loop** doivent figurer dans tout croquis Arduino ; c'est le minimum requis pour un téléversement.

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

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

Les fonctions (2)

- Le corps des fonctions **setup** et **loop** se trouve entre les deux accolades { et }. Chaque fonction doit comporter cette paire d'accolades. Elles délimitent le début et la fin du corps de la fonction
  - Un peu comme ce que fait l'indentation en langage Python…

Setup

- **setup** est la première fonction qui sera exécutée par l'Arduino. Son objet est de configurer l'Arduino, en assignant des valeurs et des propriétés à la carte qui ne changeront pas pendant le fonctionnement.

```
void setup() {
    // initialize digital pin LED_BUILTIN as an output.
    pinMode(LED_BUILTIN, OUTPUT);
}
```

- La constante **LED_BUILTIN** fait référence à une numéro de LED spécial pour la Arduino uno : la numéro 13

Loop

- **loop** est aussi une fonction, mais au lieu de n'être exécutée qu'une fois comme setup(), elle est exécutée encore et encore jusqu'à ce que vous pressiez le bouton Reset sur la carte Arduino ou que vous coupiez le courant. Voici le corps de la fonction :

```
void loop() {
    digitalWrite(LED_BUILTIN, HIGH);
    // turn the LED on (HIGH is the voltage level)

    delay(1000); // wait for a second

    digitalWrite(LED_BUILTIN, LOW);
    // turn the LED off by making the voltage LOW

    delay(1000); // wait for a second
}
```

digitalWrite()

- La fonction ***digitalWrite()*** définit un état électrique sur la broche. Les broches numériques n'ont que deux états : haut (HIGH) ou bas (LOW), ce qui correspond au niveau de la tension qui leur est appliquée en référence à la tension dans la carte.

- la valeur HIGH est équivalente à une tension de 5 V, et la valeur LOW est équivalente à une tension de 0 V.

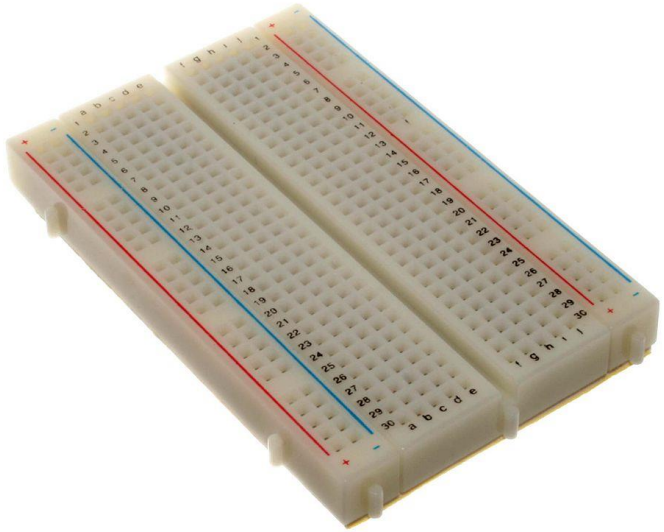
- La fonction attend deux paramètres pour démarrer :
  - pin : le numéro de la broche à faire varier ;
  - value : une des deux valeurs binaires possibles, soit HIGH, soit LOW.

delay()

- Cette fonction arrête le programme durant un certain temps, **indiqué en millisecondes.**

- Dans le cas du code ***Blink***, la valeur est de 1000 millisecondes, ce qui correspond donc à une seconde. Durant cet intervalle, rien ne se passe. Votre Arduino attend seulement l'expiration du délai.

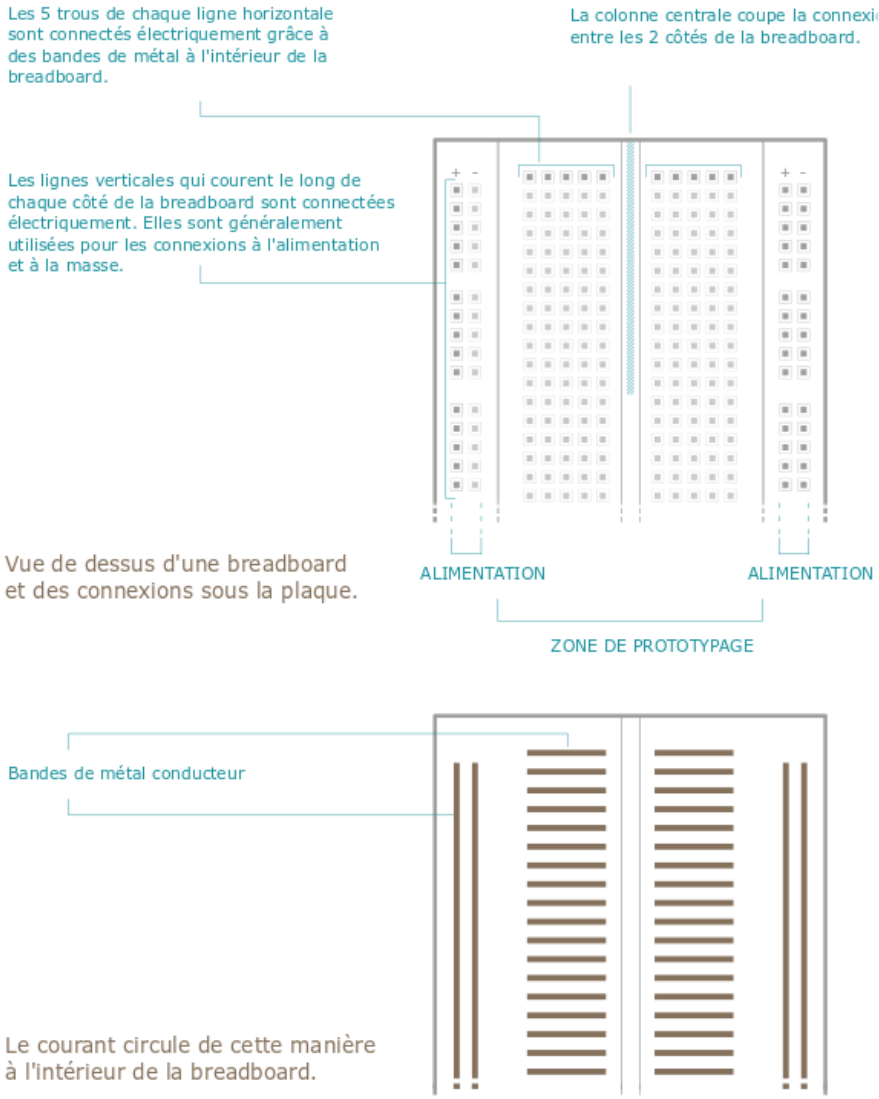
Montage



- Pour tester notre programme sur des LED branchées à des bornes digitales il faut monter un circuit électronique.
- Les slides suivant s'attachent à expliquer comment monter des composants sur la plaque d'essai (surnommée la *breadboard*)

A propos des circuits

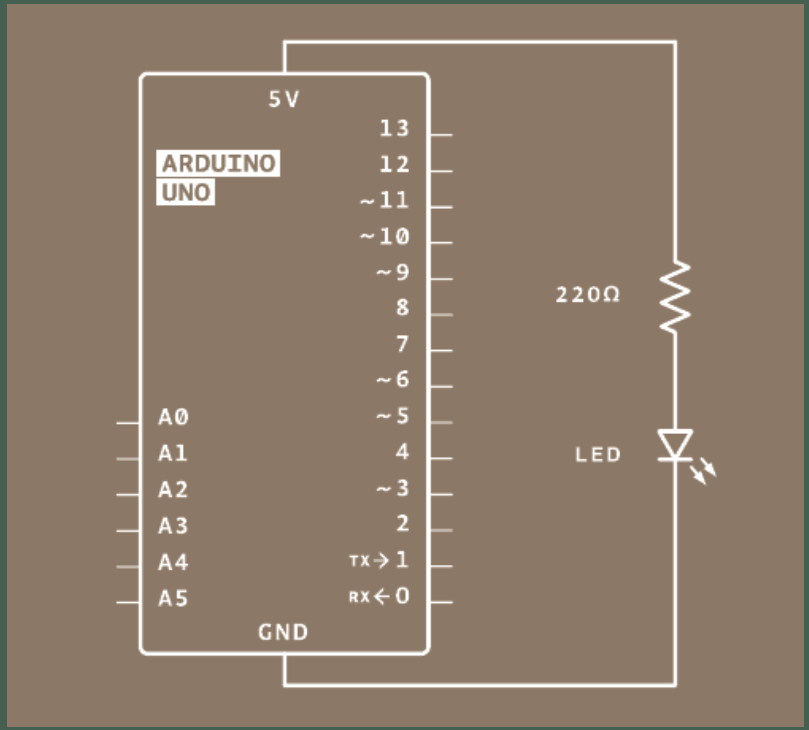
- Il faut qu'il y ait un chemin complet depuis la source d'énergie (5V) jusqu'au point de moindre énergie (masse identifiée par GND). Sans cela le circuit ne marchera pas.
- Toute l'énergie électrique d'un circuit est consommée par ses composants. Elle est alors convertie en un autre type d'énergie : lumière, chaleur, son etc.
- Le courant électrique essaiera toujours d'emprunter le chemin ayant la moindre résistance. En reliant l'alimentation directement à la masse, sans résistance, vous faîtes un court-circuit, le courant n'est plus limité et l'énergie électrique se transforme en chaleur, étincelles ou en explosion : c'est très dangereux !

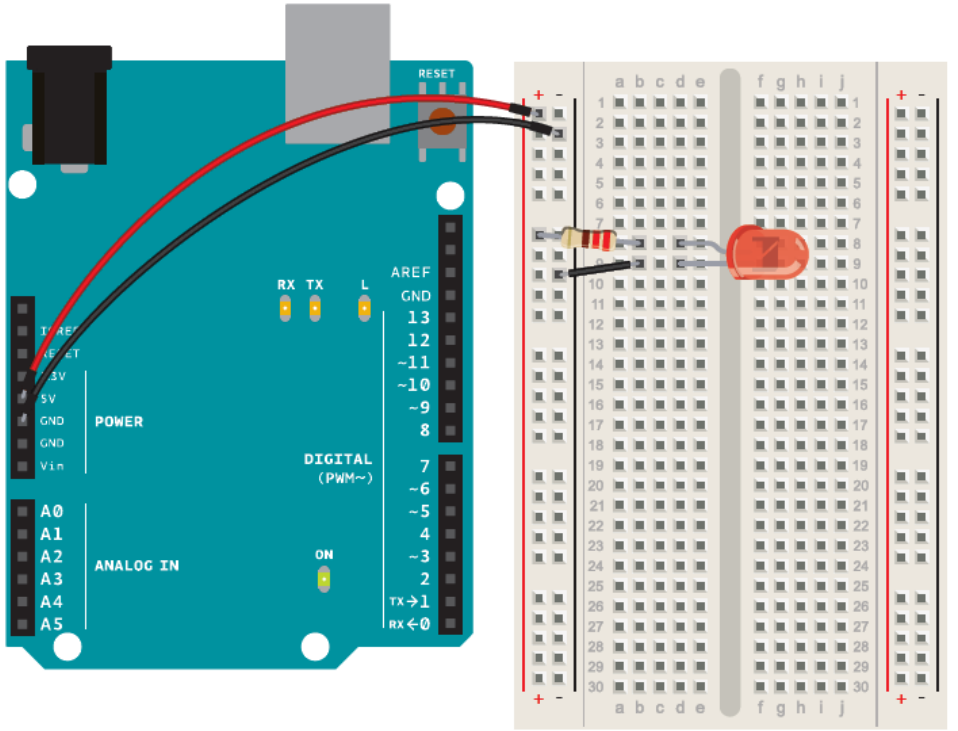


Monter un circuit électronique

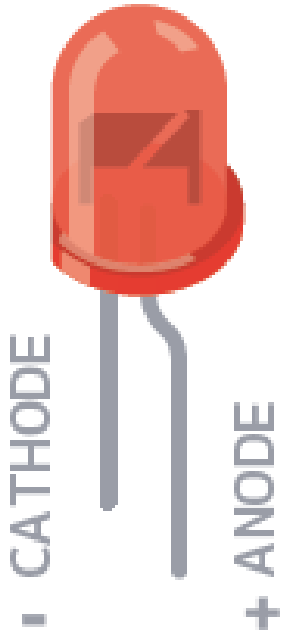
- La plaque d’essai est l’endroit où vous constituez vos circuits.
- Les lignes horizontales et verticales de la plaque conduisent l’électricité via des petits connecteurs en métal situés sous le plastique et accessibles via les trous.

Illustration





La Diode Electro Luminescente



- Une LED (DEL en français) est un composant qui convertit l'énergie électrique en lumière. Les LED sont polarisées, ça veut dire qu'elles n'autorisent le passage du courant que dans un seul sens. La patte la plus longue est appelée **l'anode**. Elle transmet l'énergie reçue. La patte la plus courte est la **cathode** et est reliée à la masse.

La résistance

- Une résistance est un composant qui s’oppose à la circulation de l’énergie électrique. Elle convertit une partie de l’énergie électrique en chaleur. Elles accompagnent toujours les LED.
- La résistance électrique traduit donc la propriété d'un composant à s'opposer au passage d'un courant. Son unité de mesure est **l’ohm.** La lecture d’un code couleur dessiné sur la résistance permet de connaître sa valeur de résistance.
- Pour plus d’informations sur le code couleur cliquez ici :

https://fr.wikipedia.org/wiki/CEI_60757

Autre exemple