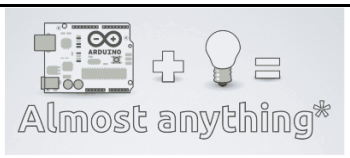
Arduino

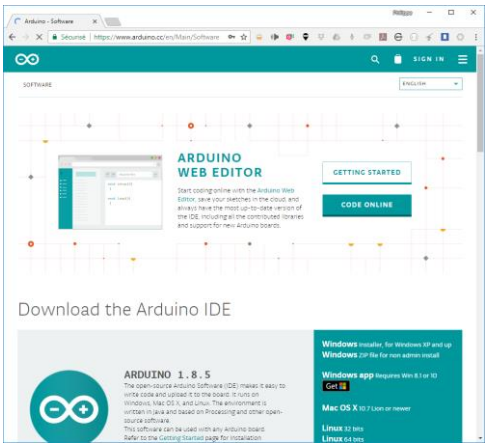
Ph. Truillet

Avril 2018 – v. 1.2



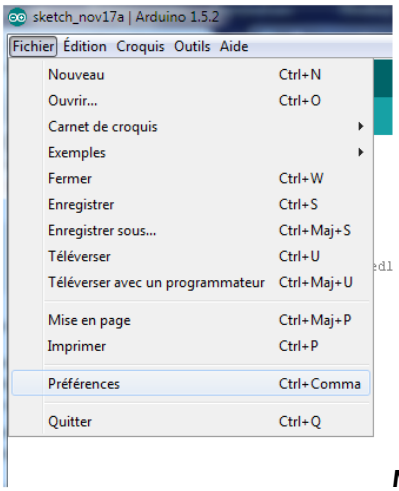
0. déroulement des travaux pratiques

1. Avant de commencer, il faut installer sur le poste de travail l'environnement de travail (IDE) arduino (**http://www.arduino.cc**). **Supposons que l'IDE soit installé sur le répertoire C:/langages)**
   Dans le cas contraire, il faut :
   - *Télécharger le logiciel (version 1.8.5)*
   - *Créer deux répertoires sur* ***C:/ langages*** *et sur C:/****dev***
   - *Décompresser l'archive téléchargée dans* ***C:/langages/arduino-1.x.x*** *(où x.x représente le numéro de version)*



Téléchargement de l'IDE Arduino (v. 1.8.5)

1. Brancher sur un port usb une plaque arduino et installer le pilote (driver) si nécessaire [le chemin vers le pilote est dans **C:/langages/arduino-1.x.x/drivers**]
2. Lancer l'IDE arduino en cliquant sur **arduino.exe** dans son répertoire
3. **Dans le répertoire dev, créer un répertoire arduino**
4. Modifier dans **Fichier | Préférences** l'emplacement du carnet de croquis vers **L:/dev/arduino**. Appuyer sur ok et redémarrer **arduino.exe**



Modification des préférences (emplacement des réalisations)

5. **Nous pouvons commencer à travailler** ☺ Nous installerons plus tard de nouvelles librairies et de nouveaux langages (Processing.org par exemple) pour contrôler arduino et le logiciel (Fritzing) pour se rappeler de nos montages arduino facilement.

1. introduction

1.1 Histoire d'arduino

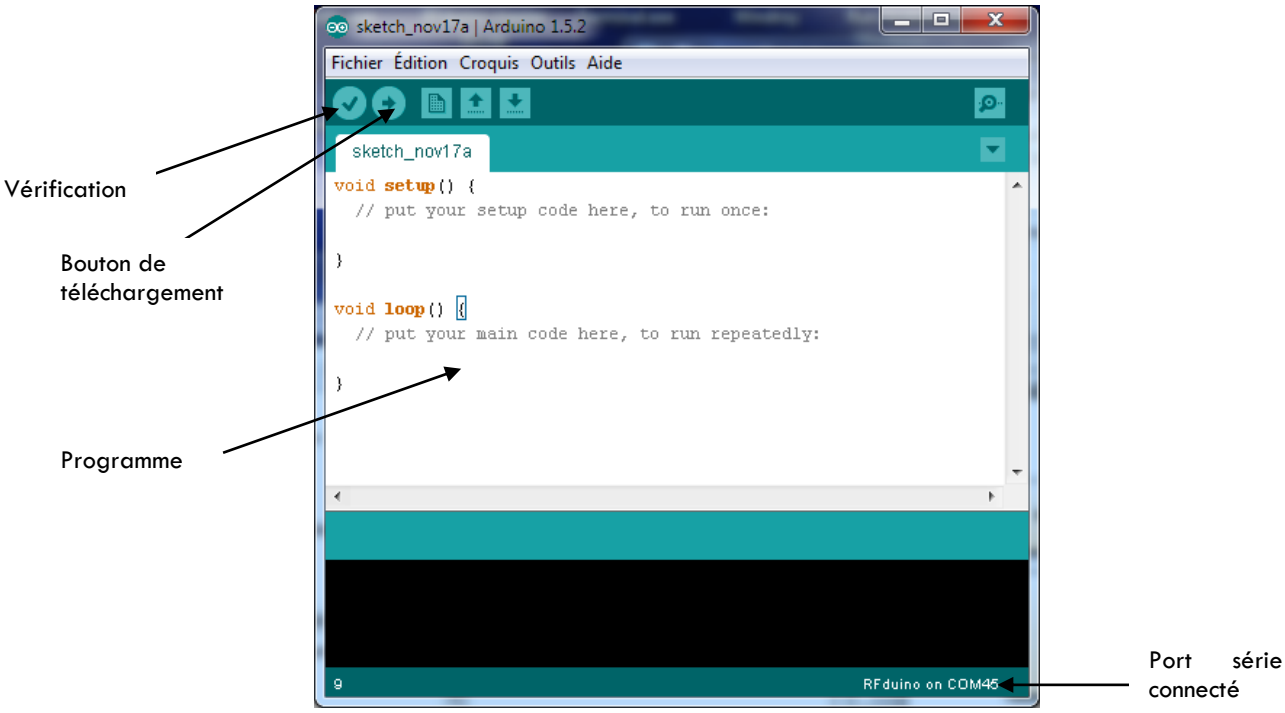
(**http://spectrum.ieee.org/geek-life/hands-on/the-making-of-arduino/0**)

C'est à Ivrea en Italie, terres du roi Arduin (Arduino en italien) vers l'an mil que commence l'histoire de cette plateforme électronique. Créé en 2005 comme outil pour les étudiants de l'Interaction Design Institute d'Ivrea, Arduino est devenu en moins de 8 ans le projet de plus influent de l'électronique moderne.
Sous licence Creative Commons (les plans sont libres), arduino peut permettre d'effectuer des tâches extrêmement diversifiées comme des tâches domotique ou robotique.

Il existe de nombreux matériels compatibles Arduino comme Freeduino, (**http://www.freeduino.org**), Educaduino (**educaduino.fr**), …

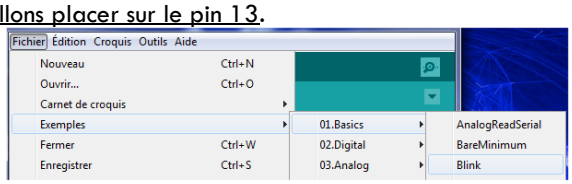
1.2 IDE Arduino

L'IDE (Environnement de Développement) permet de préparer ses programmes (appelés « *sketch* »), vérifier la syntaxe et télécharger le programme sur la plateforme arduino.



Vue de l'IDE Arduino

Il y a plusieurs manières de programmer pour arduino. La première est d'utiliser le langage (la syntaxe est proche du langage C).
Prenons un exemple. Ouvrons le programme « **blink** » situé dans « **Fichier | Exemples | 01.Basics | Blink** ». Le programme va faire clignoter une led que nous allons placer sur le pin 13.



Ouverture de l'exemple « blink »

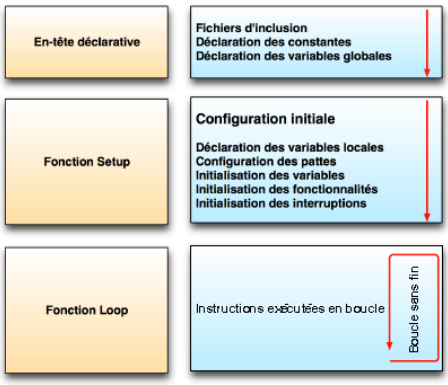
Le programme arduino est divisé en deux parties (fonctions) par défaut :

- **setup()** qui initialise un certain nombre de valeurs
- **loop()** qui correspond à une boucle sans fin et qui exécute le code contenu dans la fonction.

initialisation

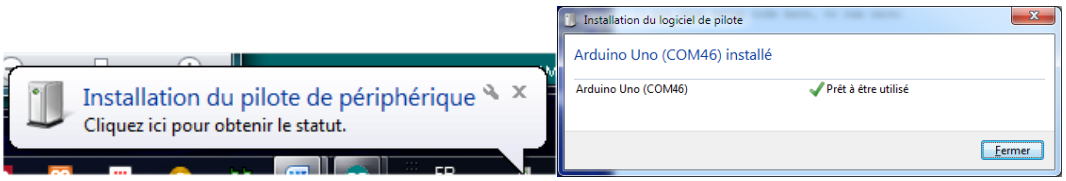
Code du programme : allumer et éteindre la led

```
Blink
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```



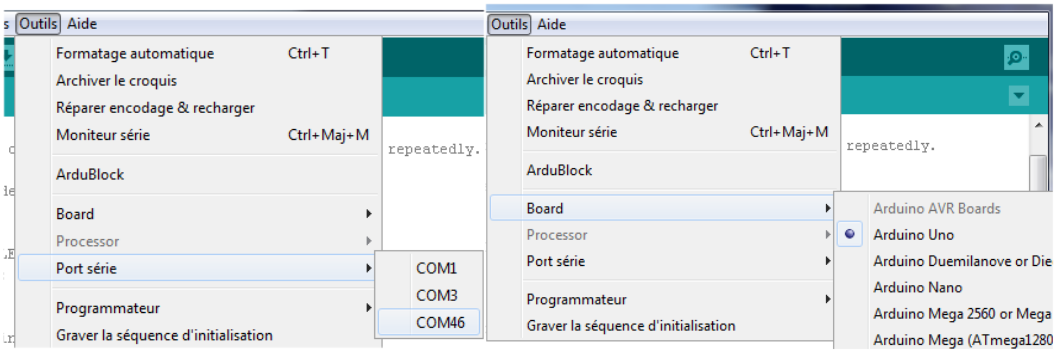
Exemple de clignotement de led

Mettre une led sur la plaque arduino : grande patte (+) sur le pin 13[1], petite sur le pin GND (-)
Branchez la plaque arduino : à la première utilisation, le pilote de périphérique s'installe. Attendez qu'un port série soit affecté à notre plaque



Installation du pilote de périphérique

Toujours lors de la première utilisation, il faut configurer la version de votre plaque et le numéro de port série. Pour ce faire, ouvrer dans le menu « **Outils** » puis successivement « **Board** » pour configurer le type de votre plaque arduino et « **Port Série** » pour le numéro de port (choisissez le numéro de port préalablement donné lors de l'installation du pilote)



Configuration de la plaque et du port série

Vérifiez le programme en appuyant sur le bouton 

puis téléchargez votre code en appuyant sur le bouton 

. **C'est prêt !** La led devrait clignoter …

[1] Le pin 13 possède une résistance interne de 220 ohms

1.4 Liens

Ce rapide aperçu laisse entrevoir de très nombreuses possibilités simples à mettre en œuvre : contrôle de capteurs, d'effecteurs (led, buzzer, moteurs, …), contrôle à distance sans fil, etc, etc.

Pour aller plus loin, le mieux reste de lire et de partager ses expériences ! Il existe de très nombreux sites accessibles via votre moteur de recherche préféré ! Parmi ceux-ci, citons :

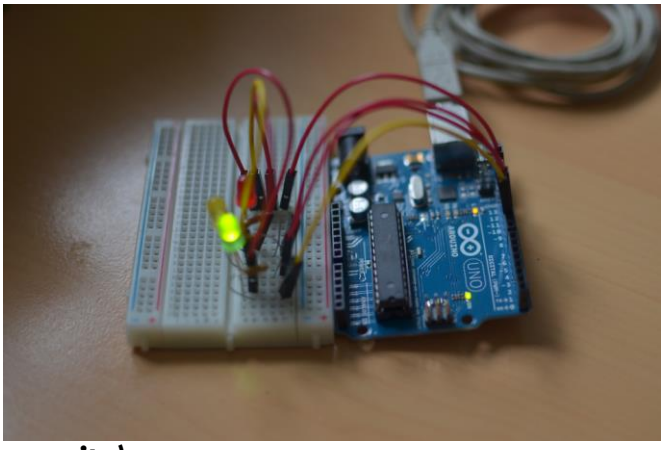
- **http://arduino.cc/en/Reference/HomePage** : la page de référence du langage

2. A vous de jouer !!!

2.1 une led comme capteur de lumière

Nous allons utiliser une led infrarouge pour allumer un éteindre une led branchée sur le pin 13.
Branchez la led infrarouge sur l'entrée analogique A0 et GND.
Ecrire un programme qui lit une valeur sur l'entrée A0 et allume la led « 13 » si une valeur de seuil est dépassée.

2.2 feux tricolores



Un feu tricolore peut être dans l'état rouge, orange, vert. Dans le cas classique, le feu est successivement vert, puis orange, puis rouge, puis de nouveau vert, etc.
Dans ce cas, il reste 6 secondes au vert, 1 seconde à l'orange, et 7 secondes au rouge (histoire de ne pas attendre trop longtemps ☺)

Déterminez le montage électronique (**Attention** : pensez à utiliser une résistance de 220 ohm si vous utilisez un autre pin que le pin 13), codez et testez !
Nota : vous pouvez utiliser le logiciel Fritzing (**http://fritzing.org/download**) pour dessiner et sauver votre montage !

2.3 changement d'état via un bouton physique (feux … suite)

Modifiez votre montage et votre code de telle manière que l'appui sur un bouton physique permette de changer de l'état nominal à l'état clignotant (le feu orange clignote à une fréquence de clignotement toutes les deux secondes) et inversement.

2.4 contrôle par le PC (feux … encore)

Modifiez encore le résultat de telle sorte qu'un envoi de données par la liaison série (un caractère par exemple) à l'arduino permette ce changement d'état (contrôle par le PC).

3. Allez plus loin … sans fil et en codant une interface sur le PC

Nous pouvons aller plus loin en contrôlant par un réseau (wifi, bluetooth, zigbee, …) et en codant des interfaces sur le PC permettant de communiquer avec arduino

Pour ce faire, vous aurez sans doute à télécharger les logiciels suivants :

- **X-CTU** : **http://www.digi.com/support/productdetail?pid=3352&osvid=57&type=utilities**
  Ce logiciel permet de configurer les puces sans fils Zigbee

- **Processing.org** : **https://processing.org/download/?processing**
  Processing est le langage qui a inspiré Arduino. Même IDE, syntaxe proche (même si Processing est issu du langage Java), Arduino et Processing forment une combinaison « magique » !

En partant de l'exemple à voir ici : **http://www.irit.fr/~Philippe.Truillet/zilab/arduino.html** (*Allumer des leds en utilisant une liaison Zigbee*), développer un système sans fil de gestion de feux tricolores (une intersection par exemple).