



sketch\_feb08a

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

# Introduction à Arduino

<http://www.arduino.cc>

Avril 2017

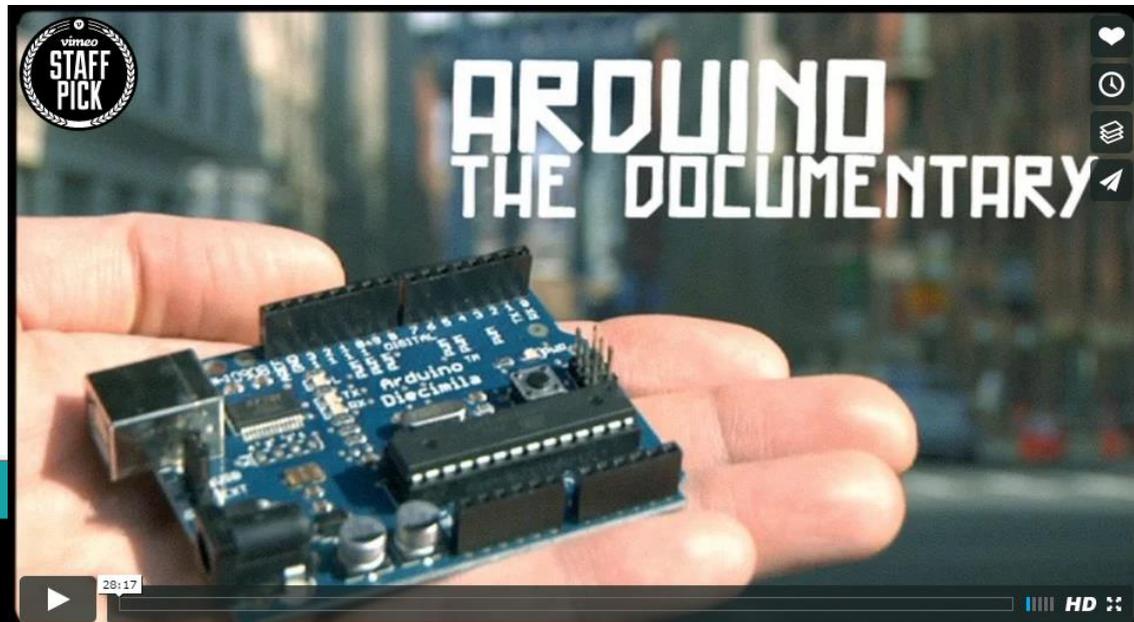
# Une histoire d'Arduino ...

sketch\_feb08a

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

<https://vimeo.com/18539129>

<https://arduinohistory.github.io>



# Historique

sketch\_feb08a

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

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

## Design by Numbers

<http://dbn.media.mit.edu>  
 Date : 1999-2001  
 Lieu : MIT Media Lab  
 John Maeda



## Processing



<http://www.processing.org>  
 Date : Printemps 2001  
 Lieu : MIT Media Lab  
 Ben Fry / Casey Reas



Processing 3



p5.js



## Wiring



<http://wiring.org.co>  
 Date : 2003  
 Lieu : IDII  
 Hernando Barragán

## Arduino



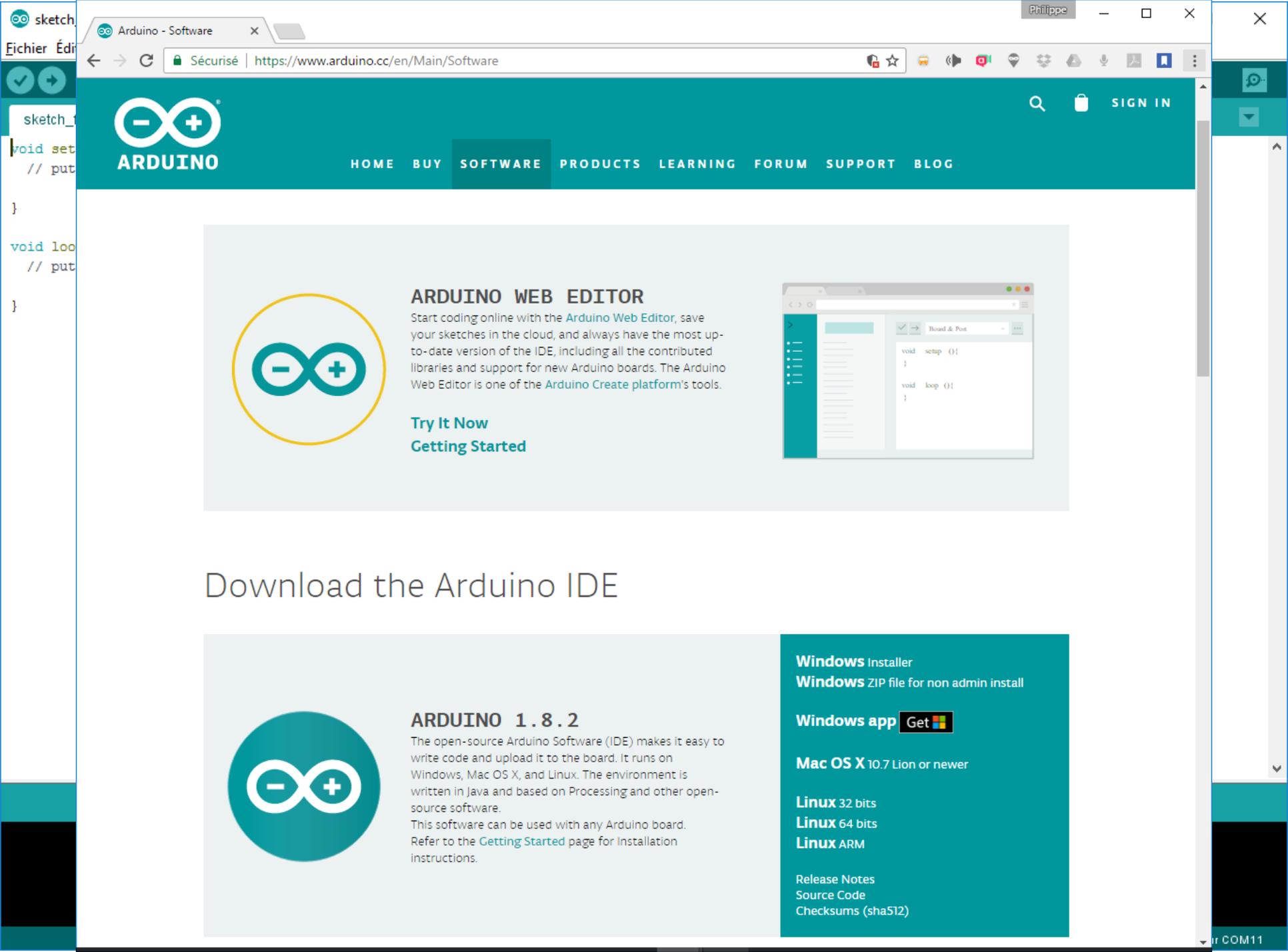
<http://www.arduino.cc>  
 Date : 2005  
 Lieu : IDII  
 Massimo Banzi



Genuino



ARDUINO



### ARDUINO WEB EDITOR

Start coding online with the [Arduino Web Editor](#), save your sketches in the cloud, and always have the most up-to-date version of the IDE, including all the contributed libraries and support for new Arduino boards. The Arduino Web Editor is one of the [Arduino Create platform's](#) tools.

[Try It Now](#)  
[Getting Started](#)



## Download the Arduino IDE



### ARDUINO 1.8.2

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board. Refer to the [Getting Started](#) page for installation instructions.

**Windows** Installer  
**Windows** ZIP file for non admin install

**Windows app**  [Get](#)

**Mac OS X** 10.7 Lion or newer

**Linux** 32 bits  
**Linux** 64 bits  
**Linux** ARM

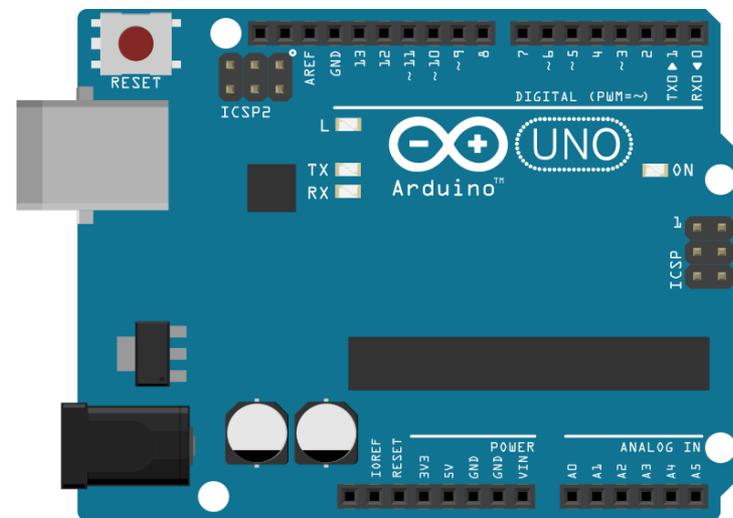
[Release Notes](#)  
[Source Code](#)  
[Checksums \(sha512\)](#)

# La carte générale ...

sketch\_feb08a

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

- Des entrées/sorties numériques
- Des entrées analogiques (A)
- ...



# Avantages

sketch\_feb08a

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

## Les « + »

- Prototypage rapide et simple d'objets physiques interactifs !
- Peu cher (suivant les cartes), logiciel et matériel open-source (et donc possibilité de clones !)
- Environnement de programmation simple

# Avantages

sketch\_feb08a

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

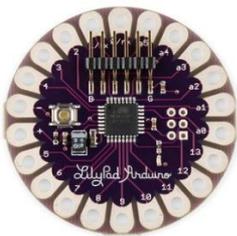
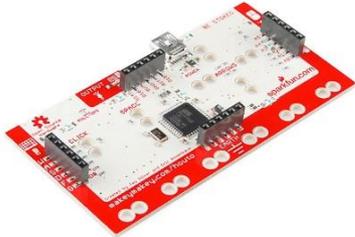
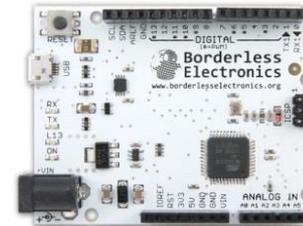
- Multiplateforme (Windows, MacOS, linux)
- Nombreuses librairies
- Des « *shields* » connectables pour augmenter les possibilités (ethernet, GPS, afficheur graphique, ...)

# Qu'est ce qu'Arduino ?

sketch\_feb08a

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

De multiples versions disponibles



... plein d'autres !

# Qu'est ce qu'Arduino ?

sketch\_feb08a

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

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

Arduino est « un langage commun » indépendant des langages bas-niveau permettant de prototyper rapidement des applications physiques.

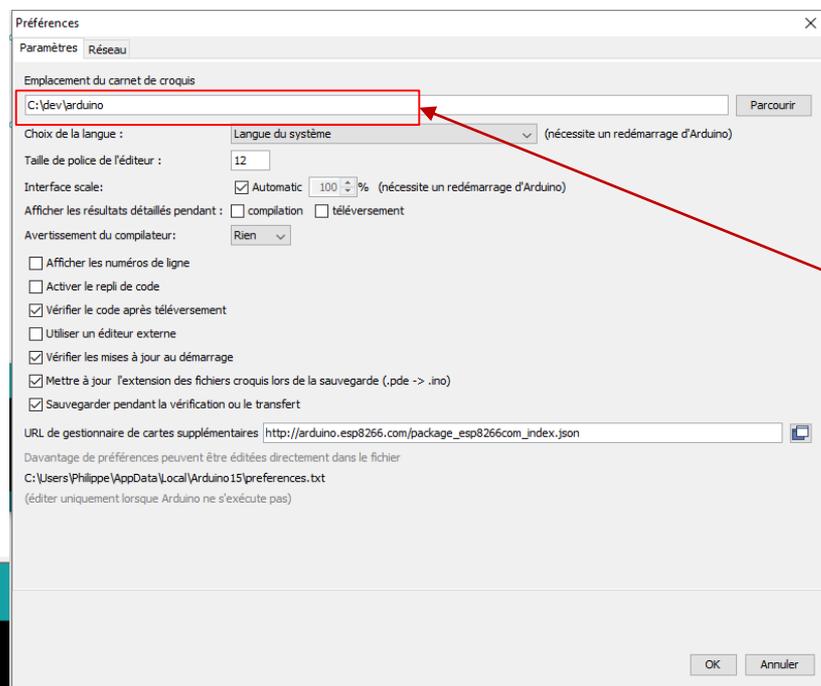
La base du programme Arduino est le « *sketch* » (programme, prototype)  
L'extension est le « **.ino** »

# Structure

sketch\_feb08a

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

- Les « sketches » (programmes) sont localisés dans le répertoire « préférences »



sketch\_may09a | Arduino 1.6.8

Fichier Édition Croquis Outils Aide

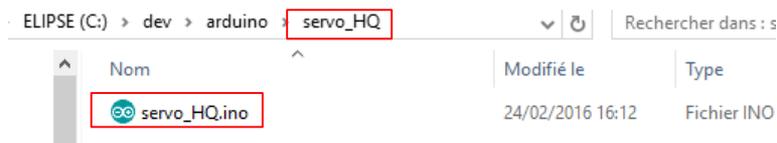
Nouveau	Ctrl+N
Ouvrir...	Ctrl+O
Ouvert récemment	>
Carnet de croquis	>
Exemples	>
Fermer	Ctrl+W
Enregistrer	Ctrl+S
Enregistrer sous...	Ctrl+Maj+S
Mise en page	Ctrl+Maj+P
Imprimer	Ctrl+P
Préférences	Ctrl+Virgule
Quitter	Ctrl+Q

# Structure

sketch\_feb08a

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

- un sketch est composé de :
  - Au moins un fichier « **.ino** » (cela peut être plus – un par classe objet).  
Le fichier principal doit avoir le même nom que le répertoire du sketch



# Deux fonctions basiques

sketch\_feb08a

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

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

- **setup** : exécuté une seule fois au démarrage – permet d’initialiser les variables du programme

```
void setup() {  
  Serial.begin(9600);  
  Serial.println("16 channel Servo test!");  
  
  pwm.begin();  
  pwm.setPWMPfreq(60); // Analog servos run at ~60 Hz updates  
  yield();  
}
```

- **loop** : c’est la boucle de traitement des capteurs exécutée « à l’infini » (mainloop)



sketch\_feb08a

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

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

Arduino - Reference x

arduino.cc/en/Reference/HomePage

Buy Download Products Learning Forum Support Blog LOG IN SIGN UP

## Structure

- setup()
- loop()

### Control Structures

- if
- if...else
- for
- switch case
- while
- do...while
- break
- continue
- return
- goto

### Further Syntax

- ; (semicolon)
- {} (curly braces)
- // (single line comment)
- /\* \*/ (multi-line comment)
- #define
- #include

### Arithmetic Operators

- = (assignment operator)
- + (addition)
- - (subtraction)
- \* (multiplication)
- / (division)
- % (modulo)

## Variables

### Constants

- HIGH | LOW
- INPUT | OUTPUT | INPUT\_PULLUP
- LED\_BUILTIN
- true | false
- integer constants
- floating point constants

### Data Types

- void
- boolean
- char
- unsigned char
- byte
- int
- unsigned int
- word
- long
- unsigned long
- short
- float
- double
- string - char array
- String - object
- array

### Conversion

- char()
- byte()

## Functions

### Digital I/O

- pinMode()
- digitalWrite()
- digitalRead()

### Analog I/O

- analogReference()
- analogRead()
- analogWrite() - PWM

### Due only

- analogReadResolution()
- analogWriteResolution()

### Advanced I/O

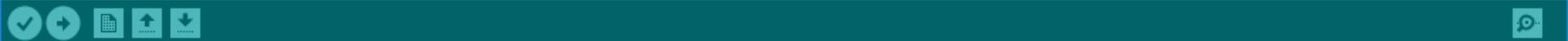
- tone()
- noTone()
- shiftOut()
- shiftIn()
- pulseIn()

### Time

- millis()
- micros()
- delay()
- delayMicroseconds()

### Math

- min()
- max()



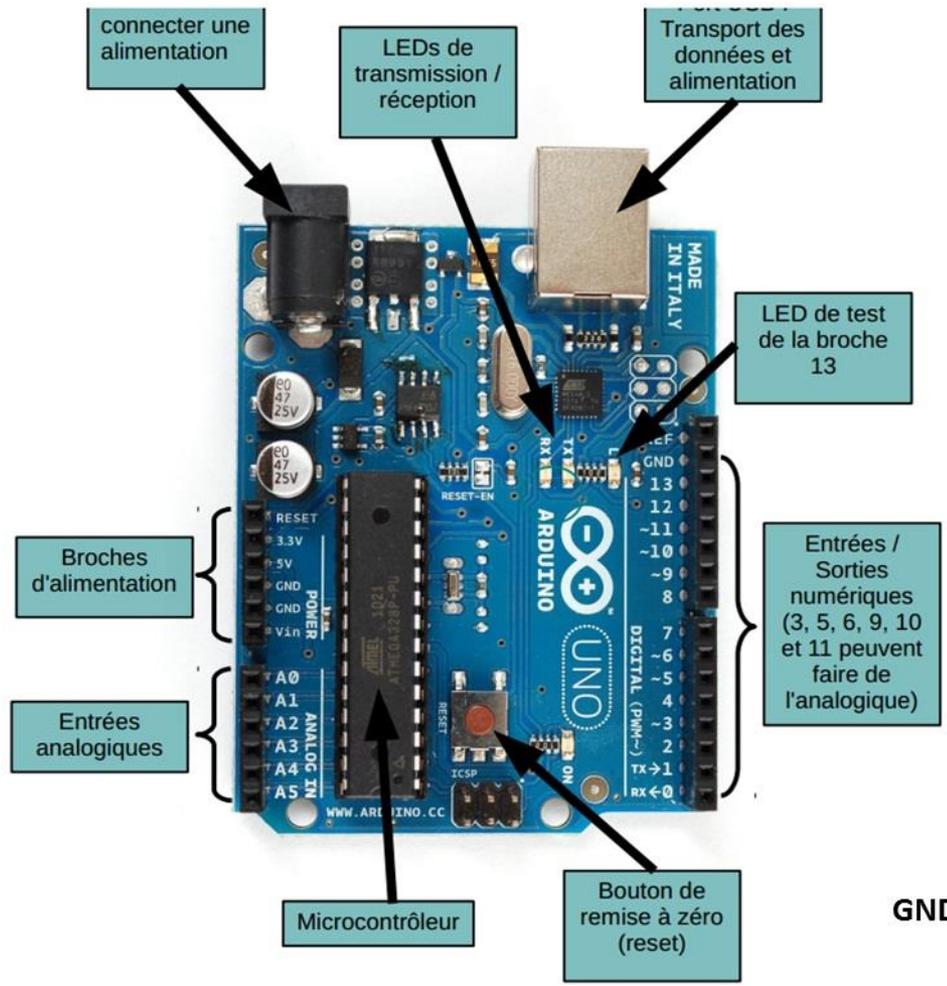
sketch\_feb08a

```

void setup() {
  // put your setup c
}

void loop() {
  // put your main co
}

```



GND = Masse ou -

# Un premier exemple

sketch\_feb08a

```
void setup() {  
  // put your setup code here,  
  
}  
  
void loop() {  
  // put your main code here, t  
  
}
```

Blink | Arduino 1.6.7

Fichier Édition Croquis Outils Aide

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.  
// Pin 11 has the LED on Teensy 2.0  
// Pin 6 has the LED on Teensy++ 2.0  
// Pin 13 has the LED on Teensy 3.0  
// 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  
}
```

# « A ne pas oublier »

sketch\_feb08a

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

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

- Outils | Type de carte >> type de la carte utilisée
- Outils | Port >> port série utilisé par la carte



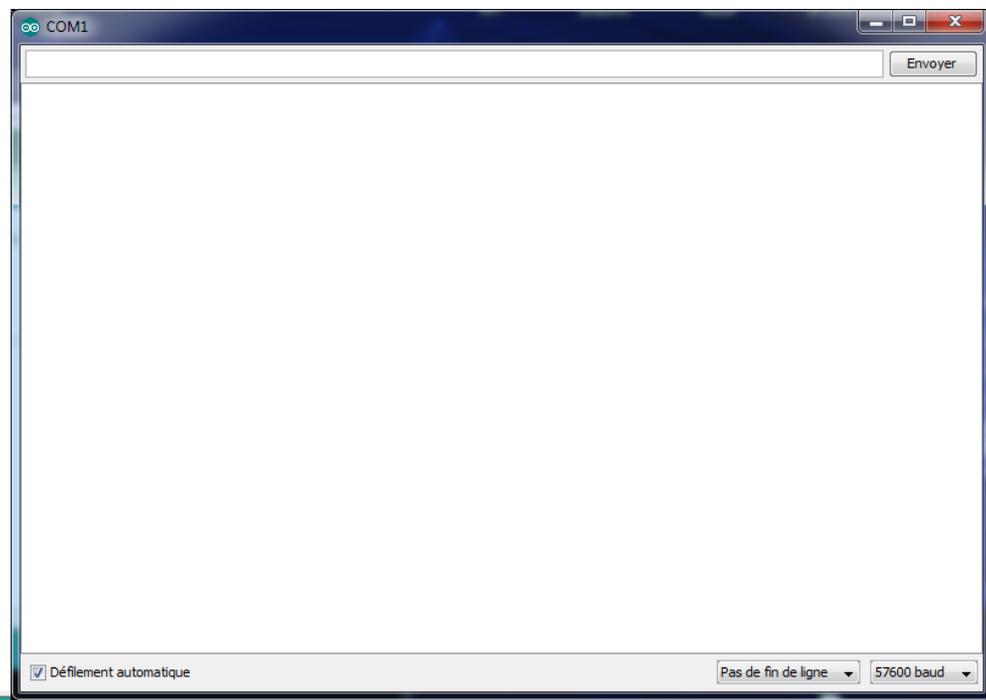
# « Astuces »



sketch\_feb08a

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

- Outils | Moniteur série





# au montage : Fritzing

sketch\_feb08a

```

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

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

```

- <http://fritzing.org>

The screenshot shows the Fritzing software interface with a breadboard circuit. The circuit includes an Arduino Uno microcontroller board, a yellow battery (E603450 7B20 E7B02-D60-1 + 1100mAh 3.7V), a black sensor module, and a red temperature sensor module. Wires connect the components on the breadboard. The software interface includes a menu bar (Fichier, Édition, Composant, Vue, Fenêtre, Routage, Aide), a toolbar with icons for Page d'accueil, Platine d'essai, Vue schématique, Circuit imprimé, and Code, and a right-hand panel with 'Composants' and 'Inspecteur' sections. A status bar at the bottom indicates '5 des 8 réseaux ont été routé - 3 connecteur(s) doivent être reliés' and has a 'Partager' button.

# Un simulateur en ligne : AutoDesk

sketch\_feb08a

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

- <https://123d.circuits.io>

```
1 // Pin 13 has an LED connected on most Arduino boards.  
2 // give it a name:  
3 int led = 13;  
4  
5 // the setup routine runs once when you press reset:  
6 void setup() {  
7 // initialize the digital pin as an output.  
8 pinMode(led, OUTPUT);  
9 }  
10  
11 // the loop routine runs over and over again forever:  
12 void loop() {  
13 digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)  
14 delay(100); // wait for a second  
15 digitalWrite(led, LOW); // turn the LED off by making the voltage LOW  
16 delay(100); // wait for a second  
17 }
```

# Exercices

sketch\_feb08a

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

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

- Utiliser une librairie externe → capteur ultrason
  - [https://bitbucket.org/teckel12/arduino-new-ping/downloads/NewPing\\_v1.8.zip](https://bitbucket.org/teckel12/arduino-new-ping/downloads/NewPing_v1.8.zip)
- Utiliser des capteurs analogiques avec émission/réception sur la liaison série
- Mini-projet incrémental (feux tricolores)
- Construire une interface de commande avec Processing