

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

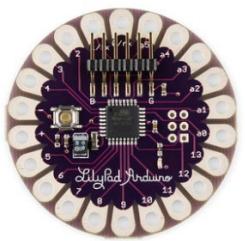
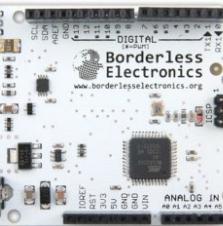
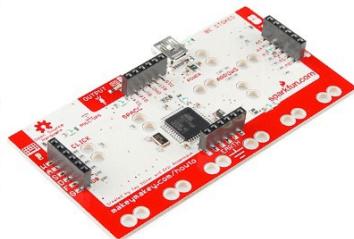
<https://www.arduino.cc>

Février 2026

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:  
}
```

C'est d'abord du matériel !



avec plein de versions différentes !

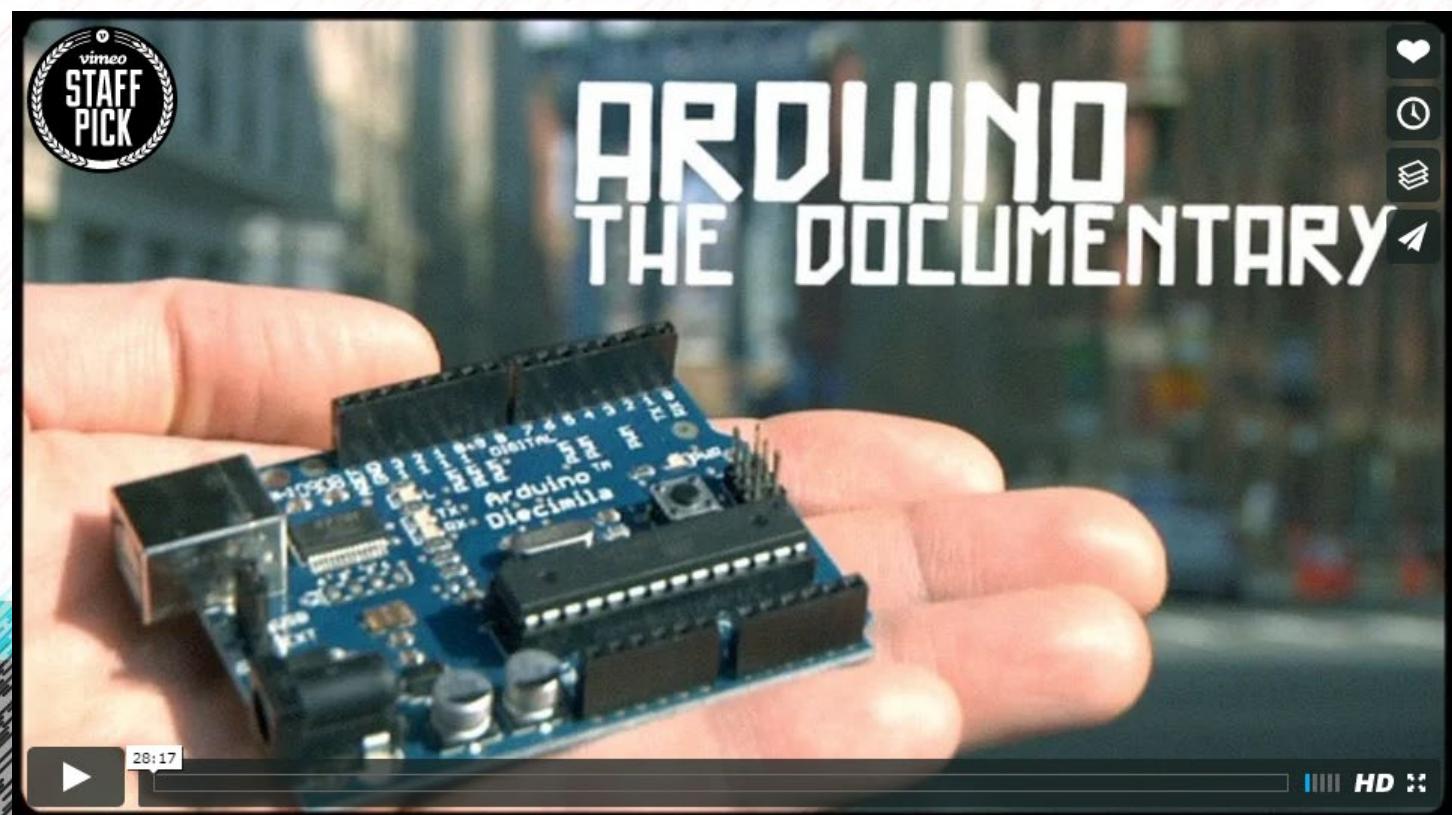
Une histoire d'Arduino ...

sketch_feb08a

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

https://youtu.be/D4D1WhA_mi8

<https://arduinohistory.github.io>



Historique

sketch_feb08a

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

Design by Numbers<http://dbn.media.mit.edu>*Date : 1999-2001**Lieu : MIT Media Lab*

John Maeda

**Visible Language Workshop**<http://museum.mit.edu/150/115>*Date : 1975**Lieu : MIT
Muriel Cooper***Processing**<https://www.processing.org>*Date : Printemps 2001**Lieu : MIT Media Lab*

Ben Fry / Casey Reas

**Processing 4**

p5.js

**Wiring**<http://wiring.org.co>*Date : 2003**Lieu : IDII*

Hernando Barragán

**Arduino**<https://www.arduino.cc>*Date : 2005**Lieu : IDII*

Massimo Banzi



Qualcomm



Arduino, c'est aussi du code ...

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

- Une syntaxe simple
- du codage bas niveau (C/C++)
- Des possibilités d'extension via des librairies

IDE – Environnement de dév.

The screenshot shows a web browser window displaying the Arduino IDE 2.0 download page. The URL in the address bar is <https://www.arduino.cc/en/software/#ide>. The page features a large teal header with the Arduino logo and the text "Arduino IDE 2.3.7". Below the header, there is a section titled "Release notes" which describes the new major release as faster and more powerful, featuring a modern editor, responsive interface, autocompletion, code navigation, and a live debugger. It links to the "Arduino IDE 2.0 documentation". A "DOWNLOAD" button is visible, along with a dropdown menu for operating system selection. A "Nightly Builds" section is also present, featuring a moon icon and a link to download a preview version. At the bottom, a note states that the Arduino IDE 2.0 is open source and its source code is hosted on GitHub.

sketch_feb08a

```
void setup() {  
 // put your setup code here, to run  
 // once the sketch starts  
}  
  
void loop() {  
 // put your main loop code here  
}
```

arduino.cc/en/software#ide

https://www.arduino.cc/en/software/#ide

Arduino IDE 2.3.7

Release notes

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger. For more details, check the [Arduino IDE 2.0 documentation](#).

Windows Win 10 or newer (64-bit)

DOWNLOAD

Nightly Builds

Download a preview of the incoming release with the most updated features and bugfixes.

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

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, RPi)
- Nombreuses librairies disponibles
- Des « *shields* » connectables pour augmenter les possibilités (ethernet, GPS, afficheur graphique, ...)

Avantages

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

Les « + »

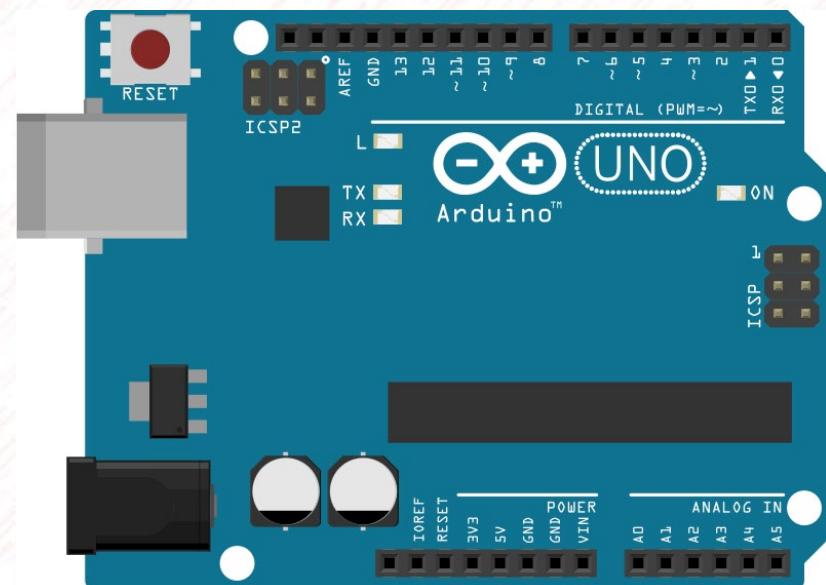
- Prototypage rapide et simple d'objets physiques interactifs !
- Peu cher (suivant les cartes), logiciel et matériel open-source (et donc possibilité de clones !)
- Environnement de programmation simple

La carte arduino UNO ...

```
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)
- ...

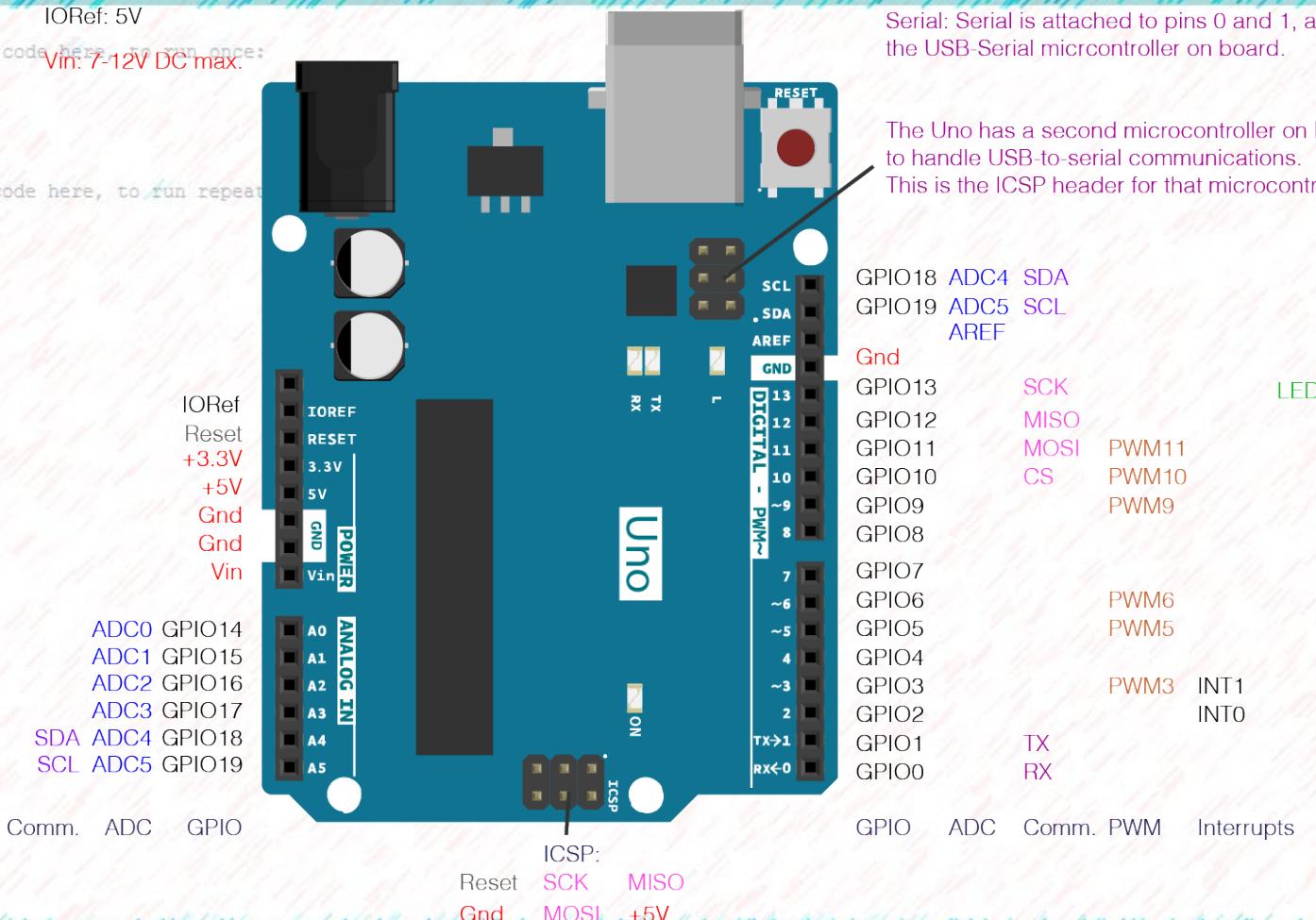


arduino UNO pinout

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
  Vin: 7-12V DC max.

}

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

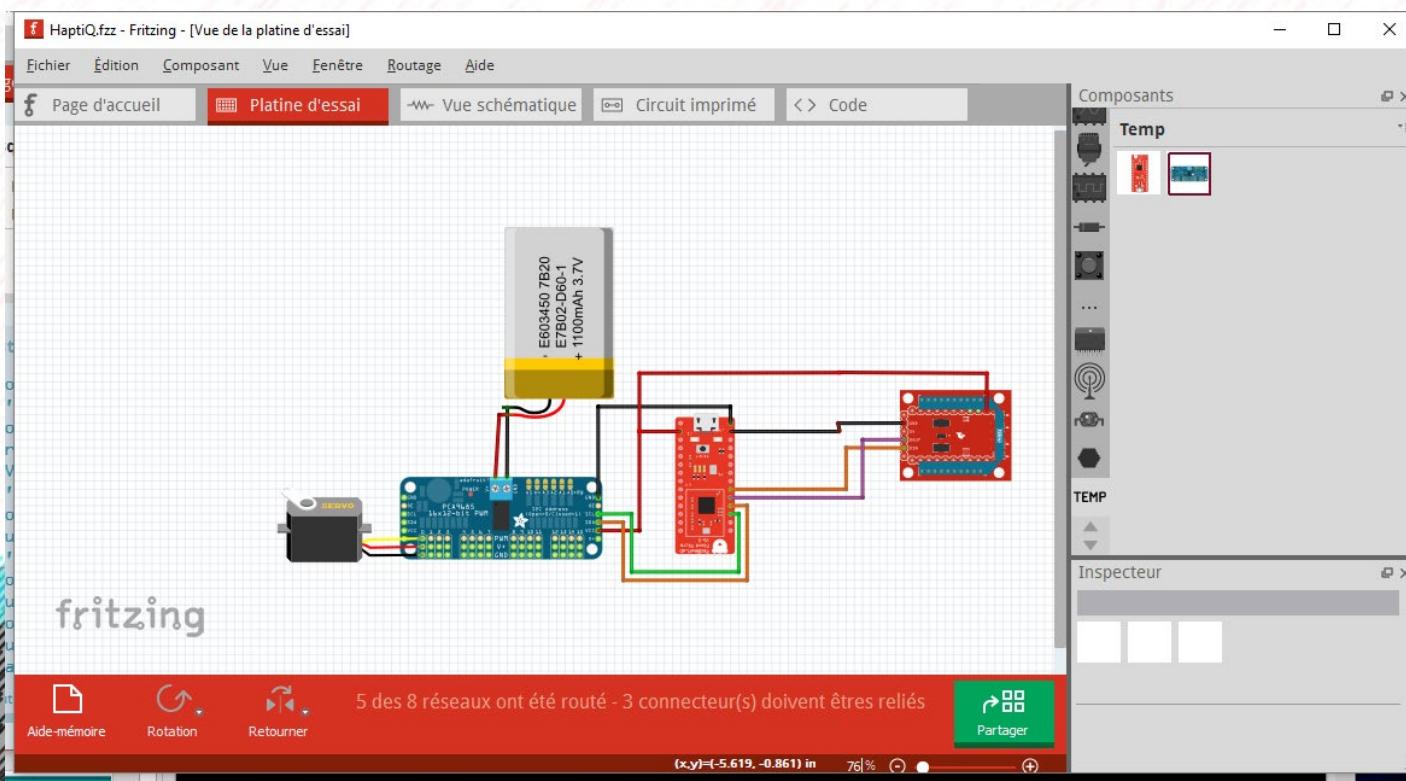


Un outil d'aide au montage

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

- **Fritzing** - <https://fritzing.org> (**payant depuis 2019**)
<https://www.softpedia.com/get/Science-CAD/Fritzing.shtml#download>



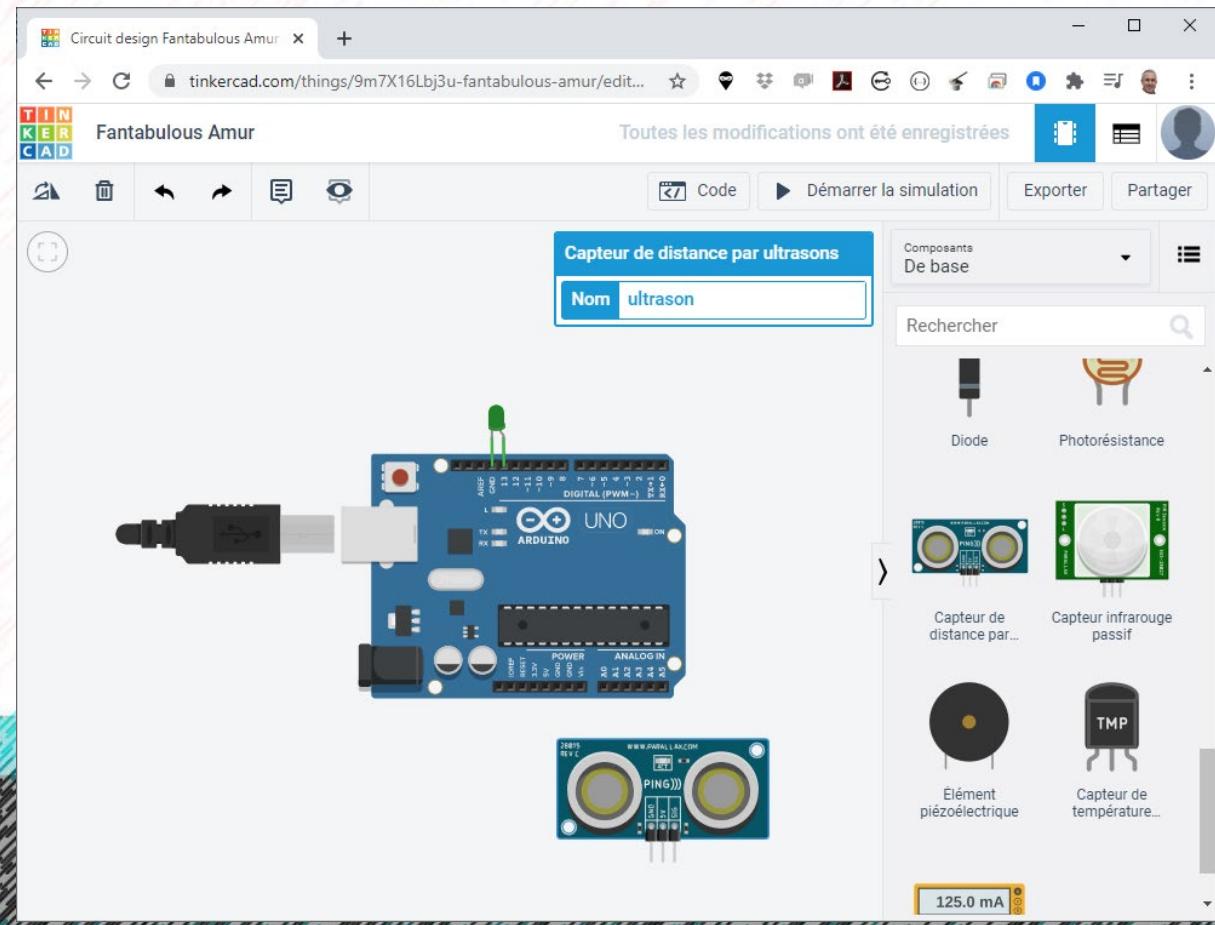
Un simulateur en ligne

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

- Choisir

Circuits



Programmation 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** (syntaxe C++)
indépendant des langages bas-niveau permettant de
prototyper rapidement des applications physiques.

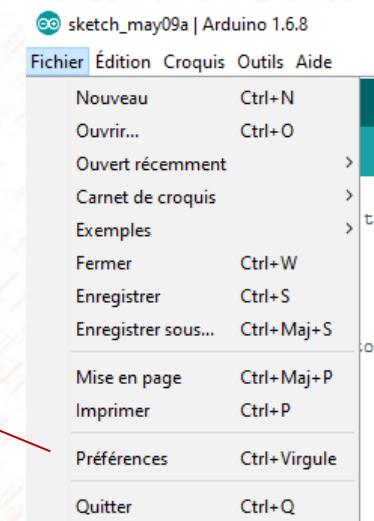
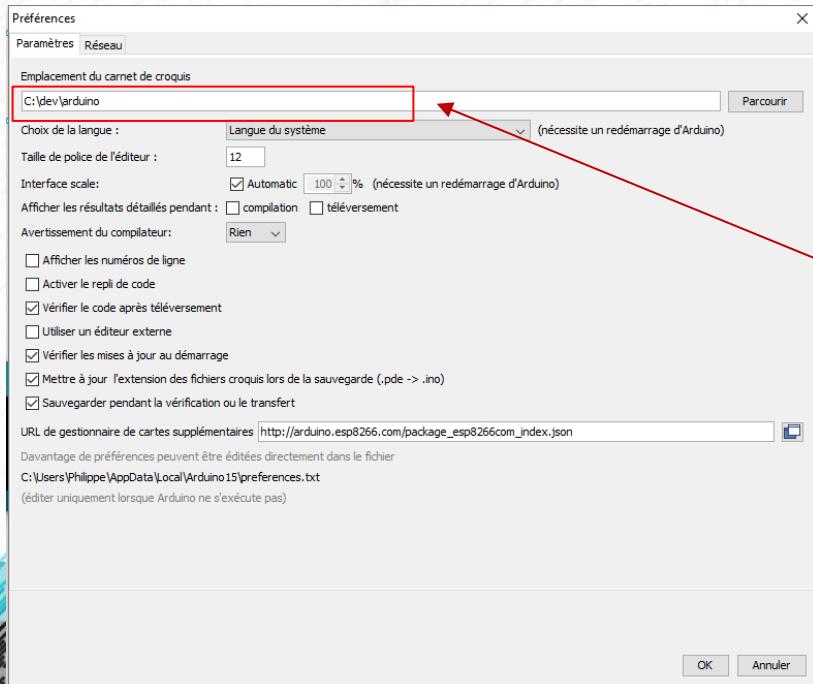
La base du programme arduino est le « *sketch* »
(programme, prototype)
L'extension est le « **.ino** »

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 »



Structure

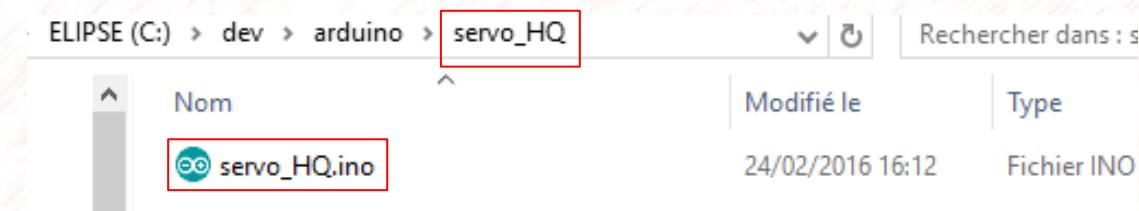
```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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.setPWMFreq(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

Arduino - Reference

arduino.cc/en/Reference/HomePage

Philippe

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()`

Un premier exemple

sketch_feb08a

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

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  
}
```

Exercices de chauffe

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

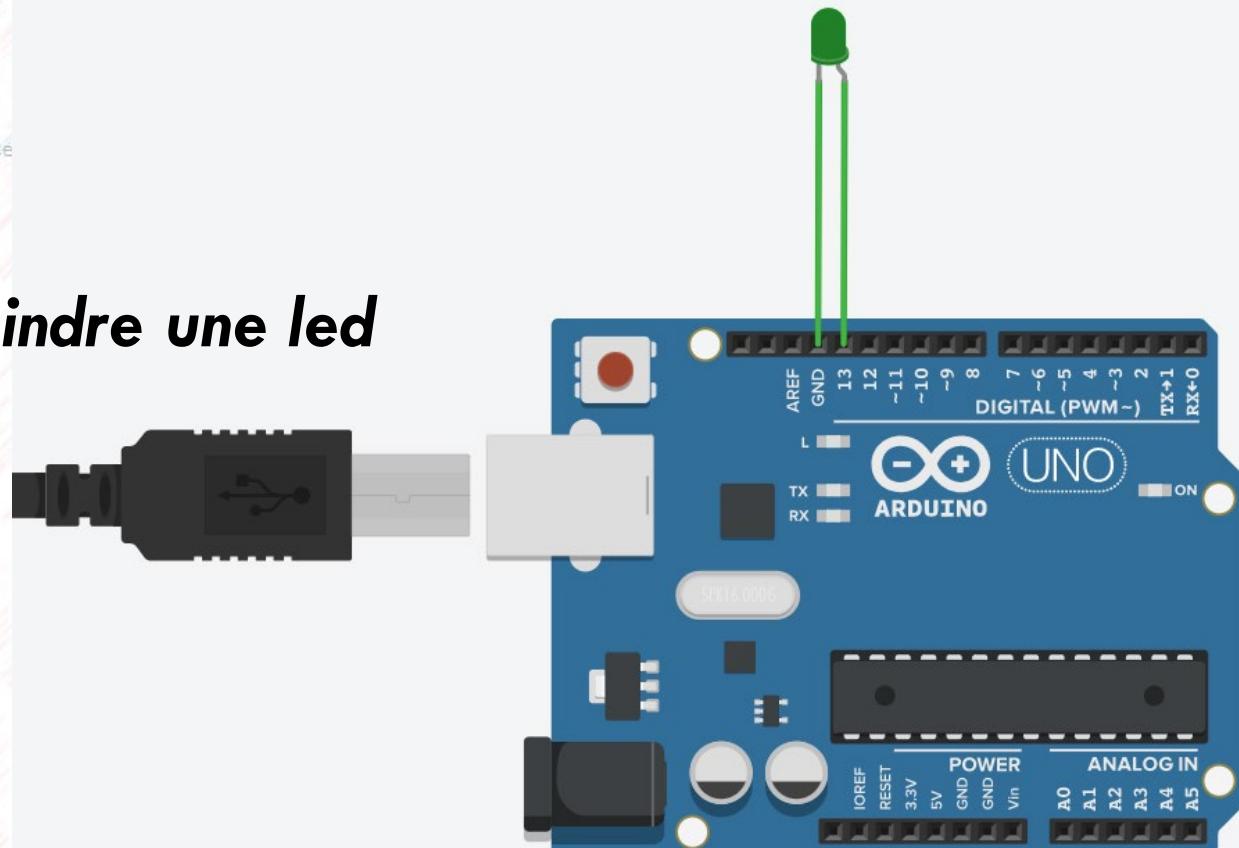
- Des leds
- Un capteur de distance

Exercices de démarrage

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

- Allumer/Eteindre une led



Exercices de démarrage

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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



- **Allumer/Eteindre une led**

Ouvrir Fichier | Exemples | 01.Basics | Blink
LED_BUILTIN → Pin 13 sur l'Arduino UNO

→ Modifier la durée du clignotement

Exercices de démarrage

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

→ Modifier le programme et envoyer l'état de la LED sur la liaison série

```
Serial.begin(rapidite_modulation)
Serial.println()
```

→ Modifier le programme pour piloter l'état de la LED depuis le PC

Liaison série

sketch_feb08a

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

- **interruption**

```
void serialEvent() { // instructions}
```

!\\ Ne fonctionne pas pour tous les arduino

Exercices de démarrage

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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



- Un capteur de distance
- Utiliser une librairie externe → capteur ultrason HC-SR04
 - https://bitbucket.org/teckell12/arduino-new-ping/downloads/NewPing_v1.9.7.zip (ou via la bibliothèque)

Please Notice This



Pour être utilisable sur ESP32, il faut le modèle **HC-SR04P** ou modifier le capteur (<https://www.instructables.com/Modify-Ultrasonic-Sensors-for-3-Volts-Logic-prepar/>)

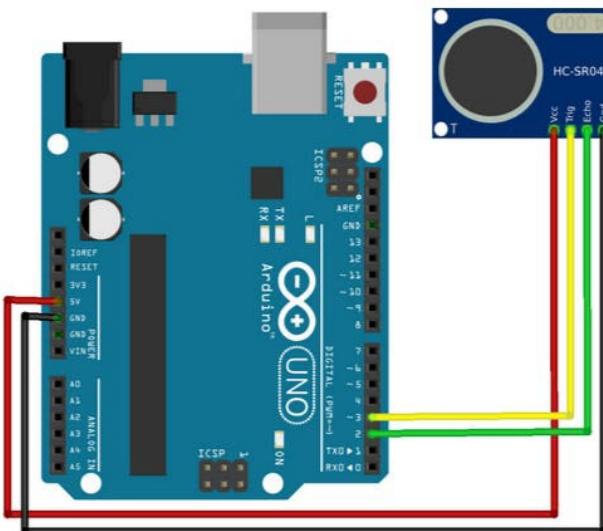


Exercices de démarrage

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

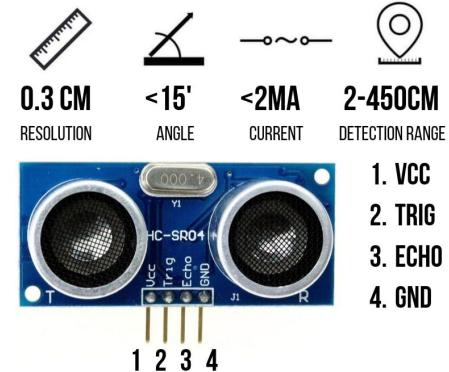
void loop() {
  // put your main code here, to run repeatedly:
}
```

- Un capteur de distance



ARDUINO >> HC-SR04

5V	—	VCC
D2	—	ECHO
D3	—	TRIG
GND	—	GND



Exercices de démarrage

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

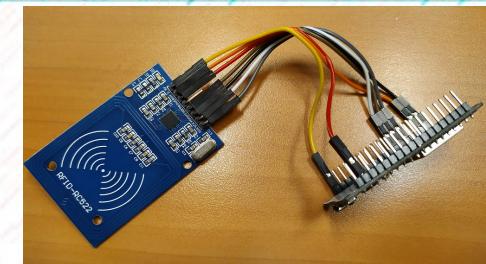


→ Écrire un programme qui envoie sur le port série la distance perçue par l'arduino avec le plus proche objet et allume la led **LED_BUILTIN** si la distance est inférieure à 20 cm

Exercices de démarrage

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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



- **RFID-522 – Un lecteur NFC**

MFRC522
by GithubCommunity Version 1.4.7 INSTALLED
Arduino RFID Library for MFRC522 (SPI) Read/Write a RFID Card or Tag using the ISO/IEC 14443A/MIFARE interface.
[More info](#)

Sélectionner une version

SDA	GPIO21
SCK	GPIO18
MOSI	GPIO23
MISO	GPIO19
IRQ	NOT USED
GND	GND
RST	GPIO22
3v3	3v3

→ Modifier le code fourni qui permet d'allumer/éteindre une LED quand on présente une carte NFC spécifique

sketch_feb08a

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

- De nombreuses autres choses sont possibles (utilisation du Bluetooth, d'autres types de capteurs/(effecteurs, usage d'API, ...)
- « *Le monde des possibles* » est quasiment infini !

Méthode de programmation

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

Pour programmer, nous aborderons la programmation sous la forme de Machine à Etats

- Un Etat → du code à exécuter
- Des événements (changement de valeur de capteurs, réception de données, ...) nous ferons changer d'état

La boucle **loop** ne contiendra qu'une instruction **switch/case**

Machine à états

```
sketch_feb08a

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

}

void loop() {
  // put your main code here, to run repeatedly
  typedef enum {ETEINT=LOW, ALLUME=HIGH} MAE;
  // l'énumération est définie sous le type MAE
  MAE mae;

  ...

  void loop() {
    switch(mae) {
      case ALLUME: ...
        break;
        ...
    }
  }
}
```

Machine à états

```
sketch_feb08a

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

void loop() {
  // put your main code here, to run repeatedly
}
  typedef enum {ETEINT=LOW, ALLUME=HIGH} MAE; // l'énumération est définie sous le type MAE
MAE mae;

...
void loop() {
  switch(mae) {
    case ALLUME: ...
      break;
      ...
  }
}
```

Les ESP

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

- Les ESP (8266 et 32) sont une série de micro-contrôleurs intégrant la gestion du wifi et du bluetooth (jusqu'à BLE)
- Ils sont peu onéreux et très appréciés dans le domaine de l'IoT !

Installer ESP8266 et ESP32

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

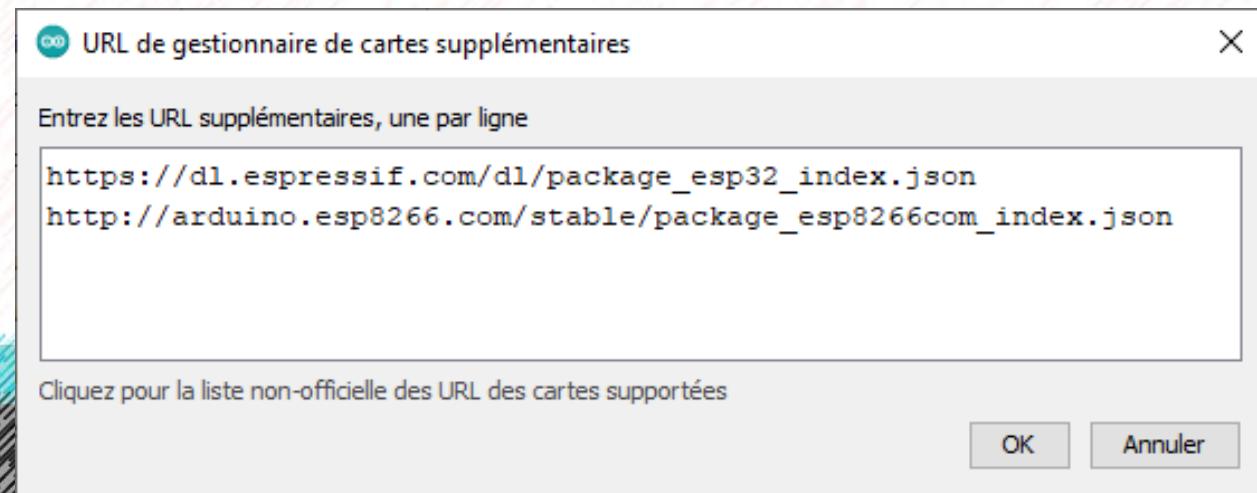
void loop() {
  // put your main code here, to run repeatedly:
}
```

- **Dans** Fichier | Préférences

URL de gestionnaire de cartes supplémentaires https://dl.espressif.com/dl/package_esp32_index.json, http://arduino.esp8266.com/stable/package_esp8266com_index.json 

https://dl.espressif.com/dl/package_esp32_index.json

http://arduino.esp8266.com/stable/package_esp8266com_index.json

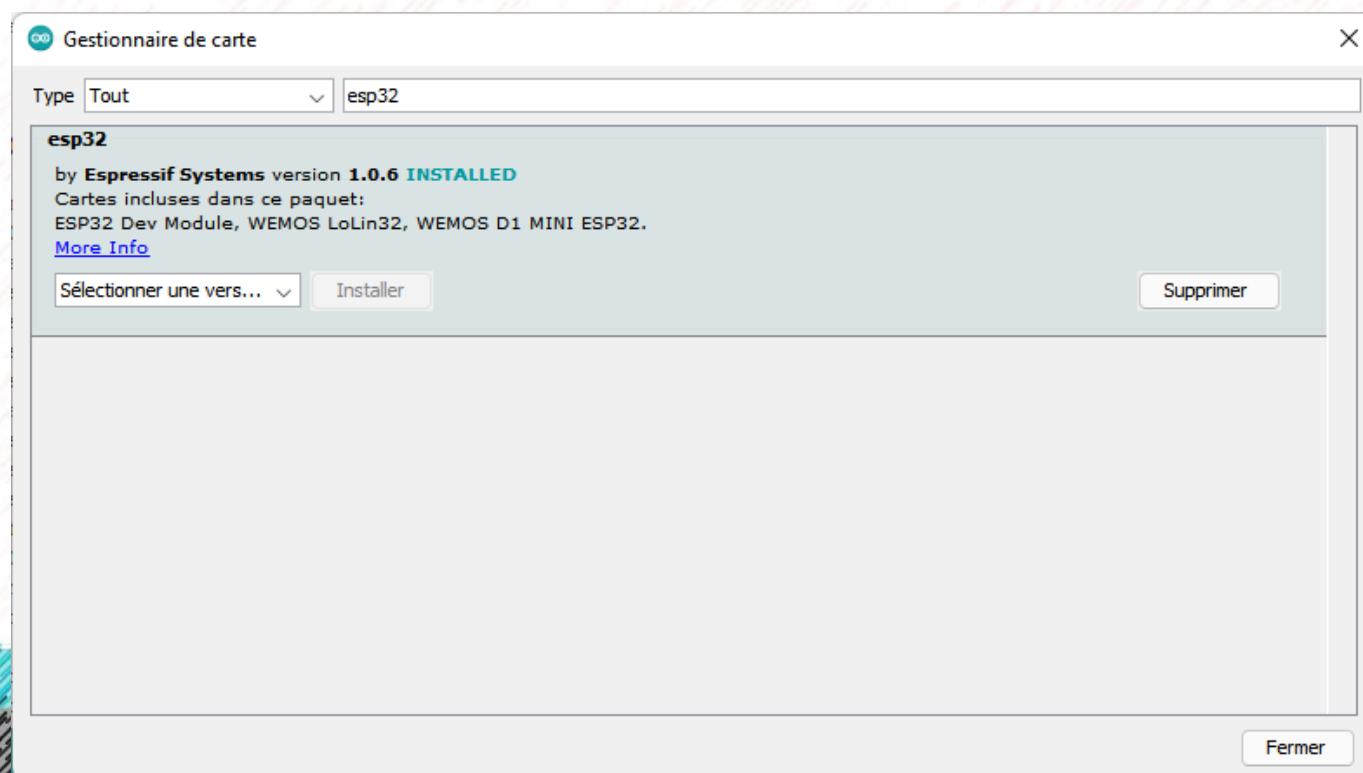


Installer ESP8266 et ESP32

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

- Dans Outils | Type de carte | Gestionnaire de carte

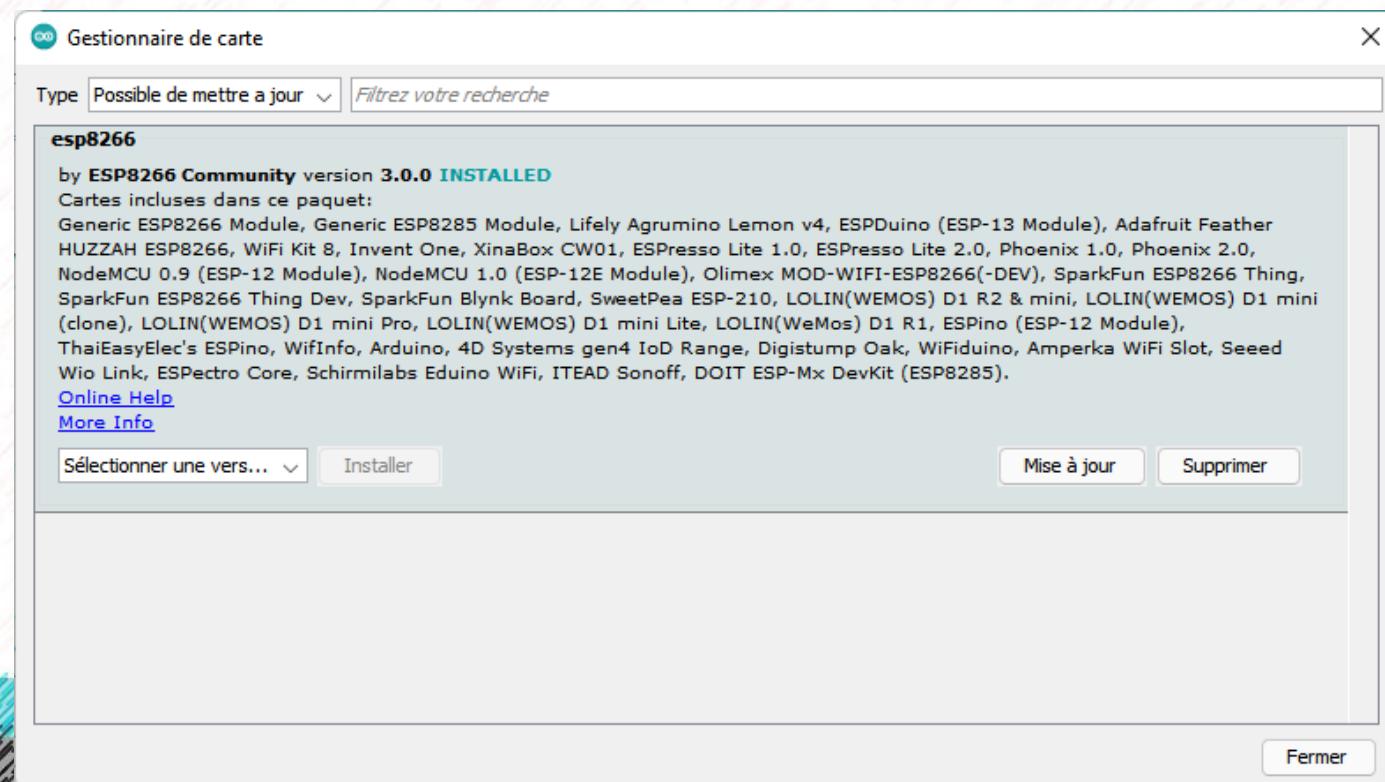


Installer ESP8266 et ESP32

```
sketch_feb08a
void setup() {
  // put your setup code here, to run once:
}

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

- Dans Outils | Type de carte | Gestionnaire de carte



Installer ESP8266 et ESP32

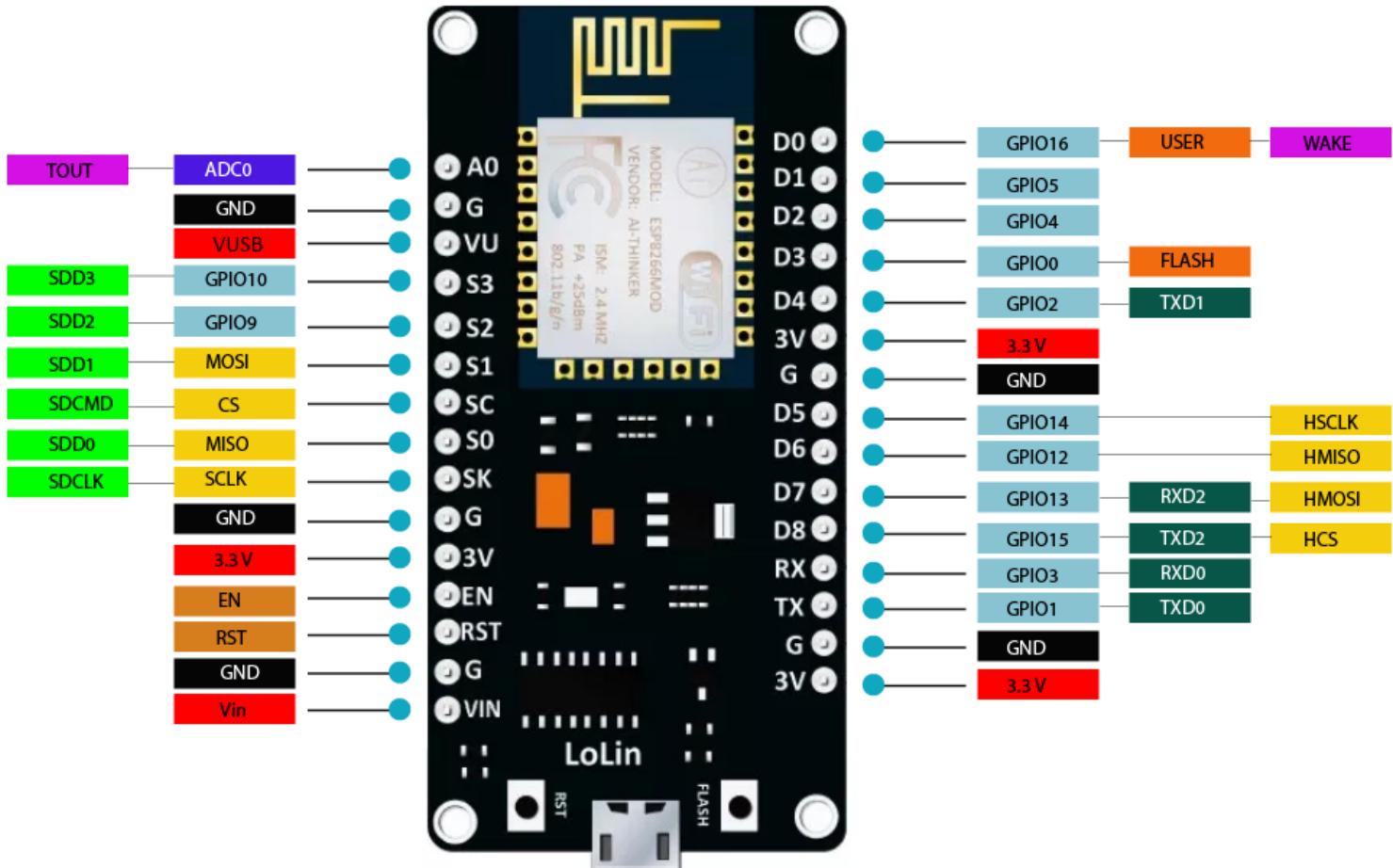
The screenshot shows the Arduino IDE interface. The top menu bar includes 'Fichier', 'Édition', 'Croquis', 'Outils', and 'Aide'. The 'Outils' menu is expanded, showing the 'Board' section. A sub-menu for 'WiFi101 / WiFiNINA Firmware Updater' is also visible. The main workspace displays a sketch named 'Blink' with the following code:

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

A tooltip for the 'Blink' sketch provides information about the LED pin and the correct LED model. The bottom-left corner shows the status 'Téléversement terminé' (Upload completed) and the message 'Leaving... Hard resetting via RTS pin...'. The bottom right shows the board selection dropdown with 'ESP32 Arduino' highlighted.

ESP8266 pinout

sketch_feb08a



[NodeMCU V3 Pinout](#)

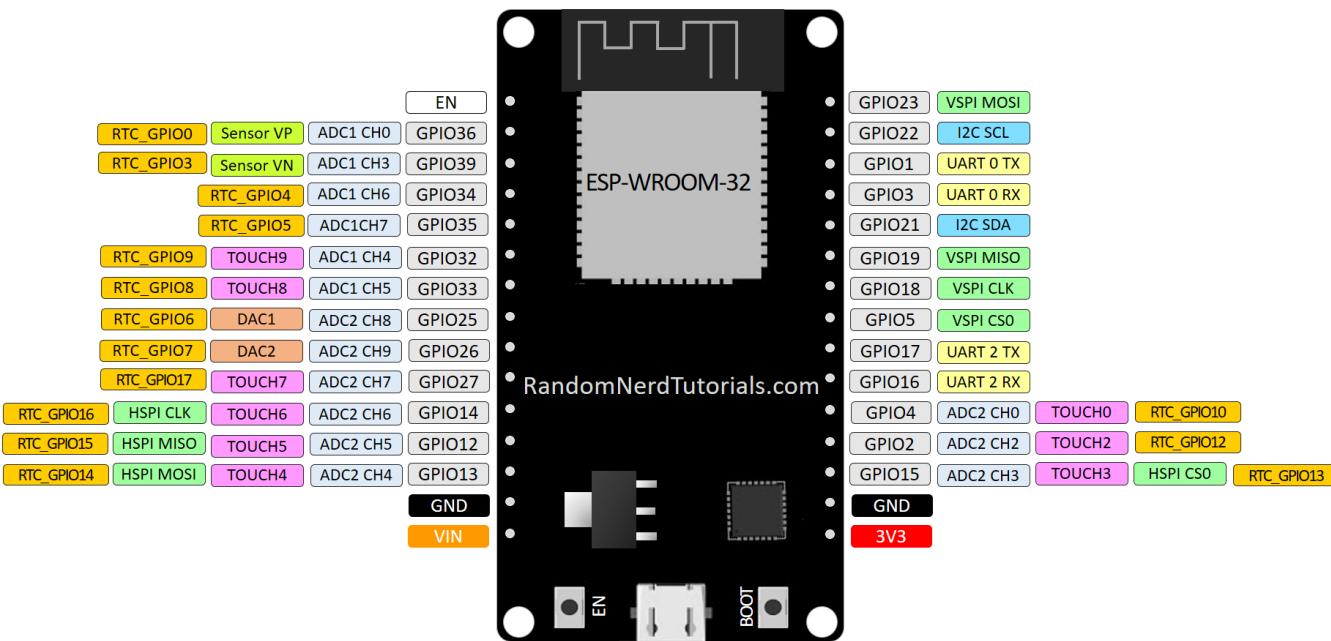
www.TheEngineeringProjects.com

D0	GPIO16
D1	GPIO05
D2	GPIO04
D3	GPIO00
D4	GPIO02
D5	GPIO14
D6	GPIO12
D7	GPIO13
D8	GPIO15
D9	GPIO03
D10	GPIO01

ESP32 pinout

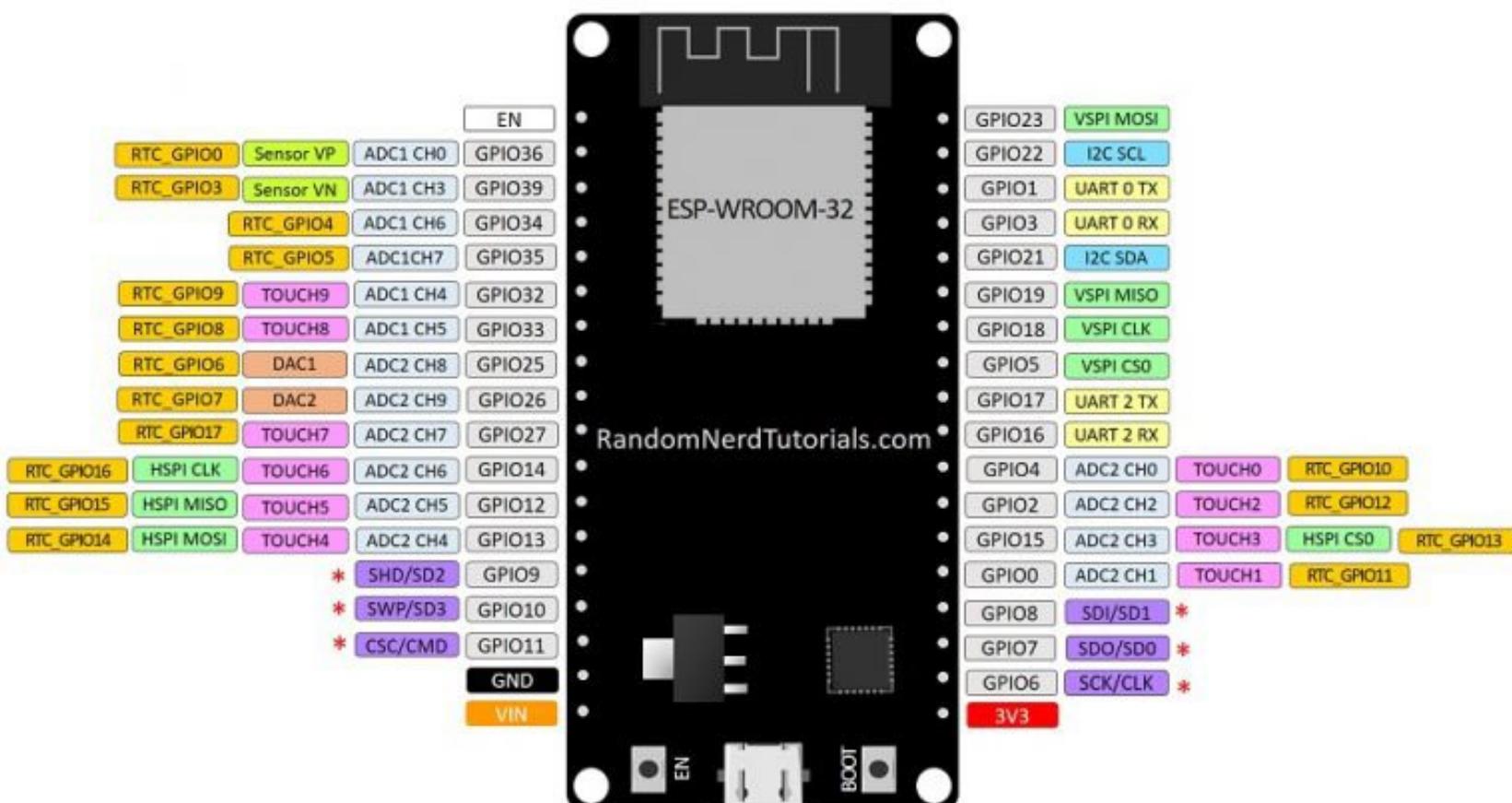
```
sketch_feb08a  
void setup() {  
  // put your setup  
}  
  
void loop() {  
  // put your main  
}
```

ESP32 DEVKIT V1 – DOIT version with 30 GPIOs



ESP32 pinout

ESP32 DEVKIT V1 - DOIT version with 36 GPIOs



* Pins SCK/CLK, SDO/SDO, SDI/SD1, SDO/SD0, SHD/SD2, SWP/SD3 and CSC/CMD, namely, GPIO6 to GPIO11 are connected to the integrated SPI flash integrated on ESP-WROOM-32 and are not recommended for other uses.