



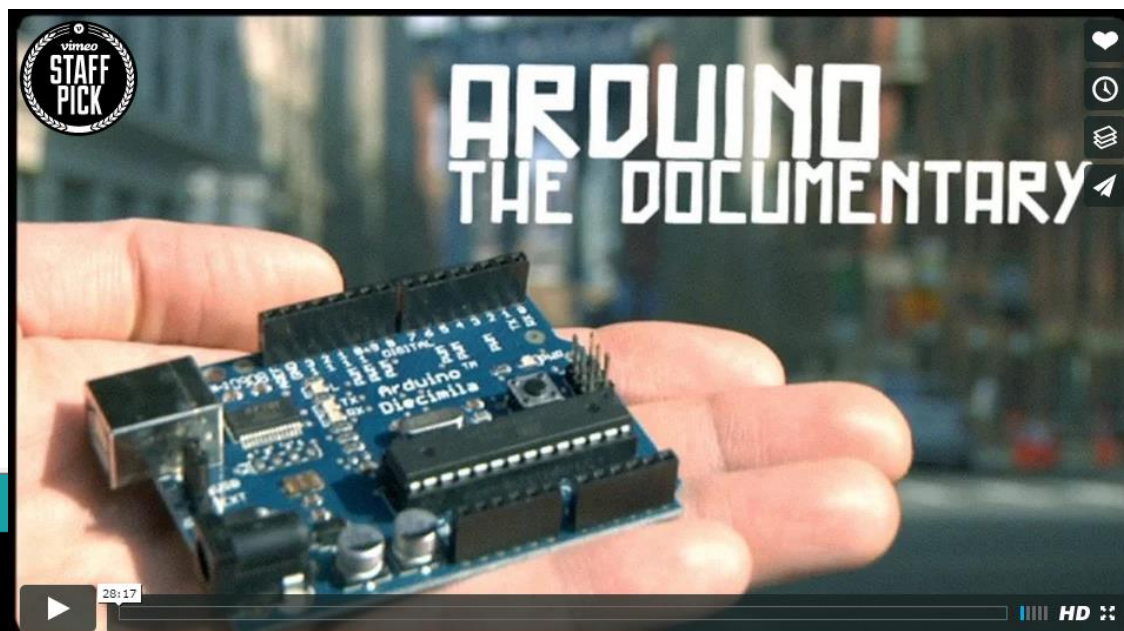
Une histoire d'Arduino ...

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://arduinhistory.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



Visible Language Workshop

<http://museum.mit.edu/150/115>

Date : 1975

Lieu : MIT

Muriel Cooper



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



IDE – Environnement de dév.

sketch_feb08a

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

```
void loop() {  
  // put your ma  
}
```

The screenshot shows the Arduino IDE environment with a web browser window open to arduino.cc/en/software. The browser window displays the Arduino Software page, which includes a navigation menu with links to Sketchbook, Examples, Libraries, and Serial Monitor. The main content area features a "CODE ONLINE" button and a "GETTING STARTED" link. Below this, there are promotional banners for "The Arduino Student Kit" and a "Downloads" section. The "Downloads" section highlights "Arduino IDE 1.8.13" and provides "DOWNLOAD OPTIONS" for Windows, Linux, and Mac OS X. A "Help" button is visible in the bottom right corner of the browser window.


up-to-date version of the IDE includes all libraries and also supports new Arduino boards.

[CODE ONLINE](#) [GETTING STARTED](#)

The Arduino Student Kit: bring the buzz home

The Arduino Student Kit: bring the buzz home

Downloads

 **Arduino IDE 1.8.13**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

Windows app Win 8.1 or 10 [Get](#)

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

Mac OS X 10.10 or newer

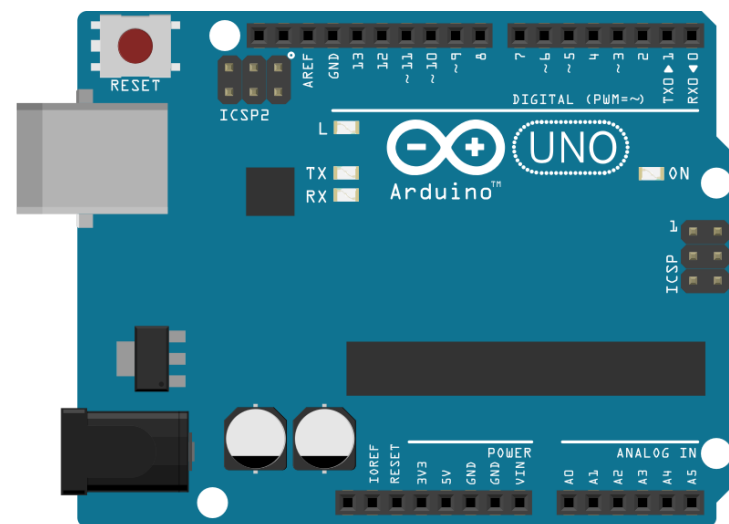
[?](#) Help

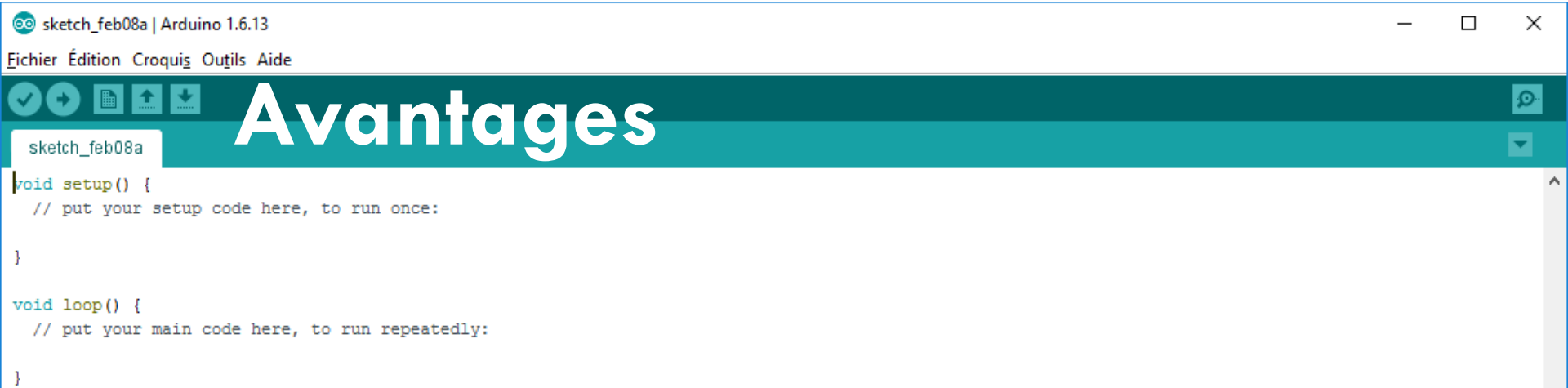
La carte générale ...

sketch_feb08a

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

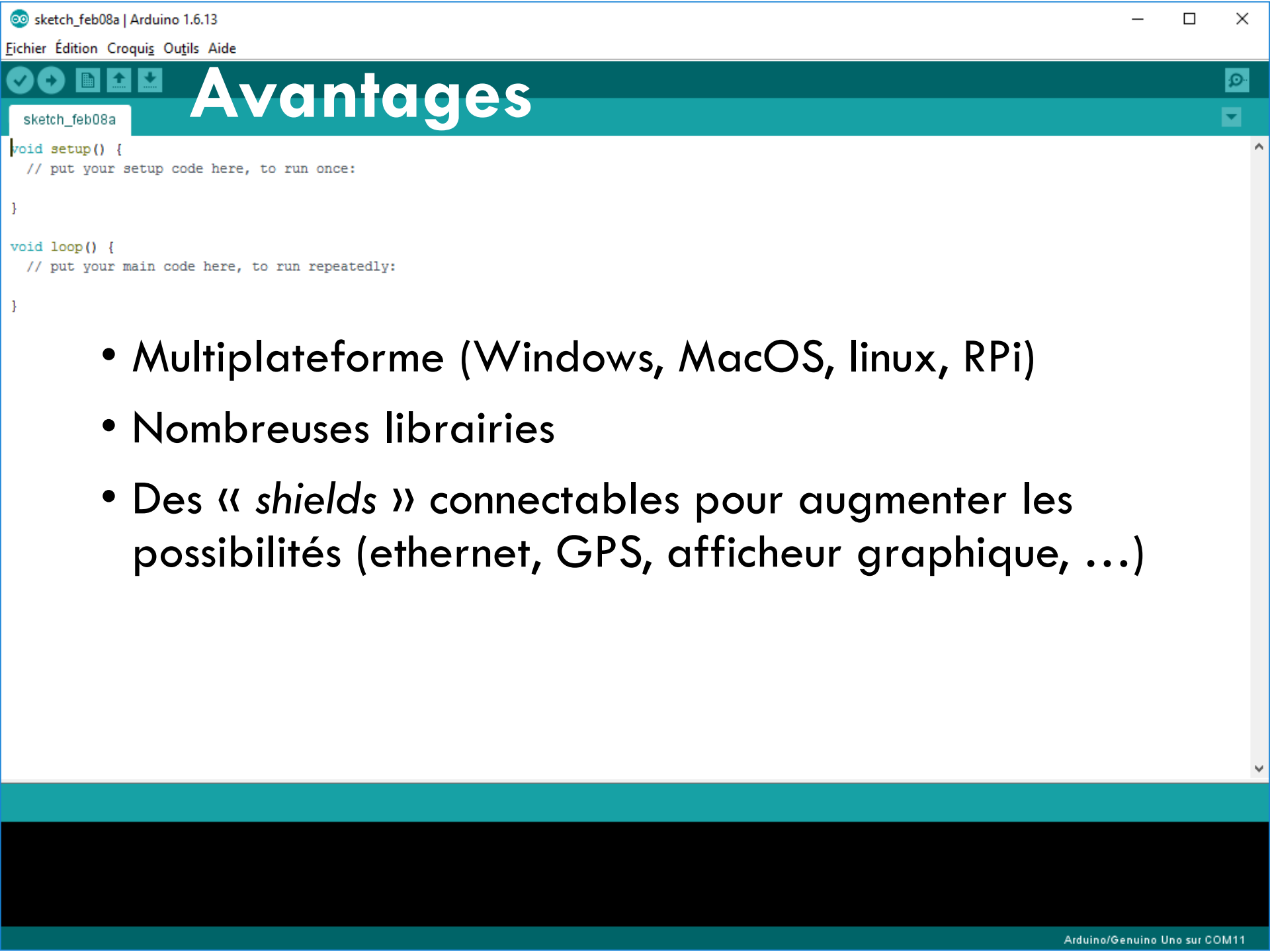
- Des entrées/sorties numériques
- Des entrées analogiques (A)
- ...





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

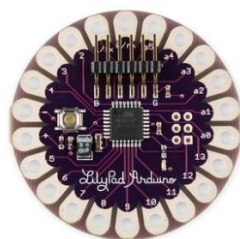
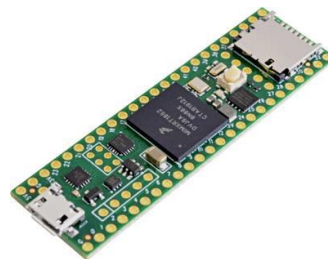
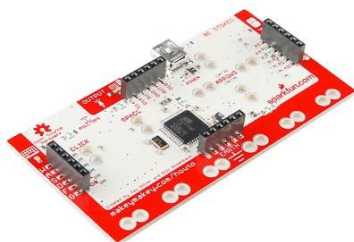
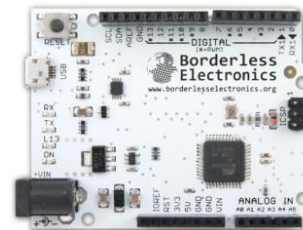
- Multiplateforme (Windows, MacOS, linux, RPi)
- Nombreuses librairies
- Des « *shields* » connectables pour augmenter les possibilités (ethernet, GPS, afficheur graphique, ...)

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.

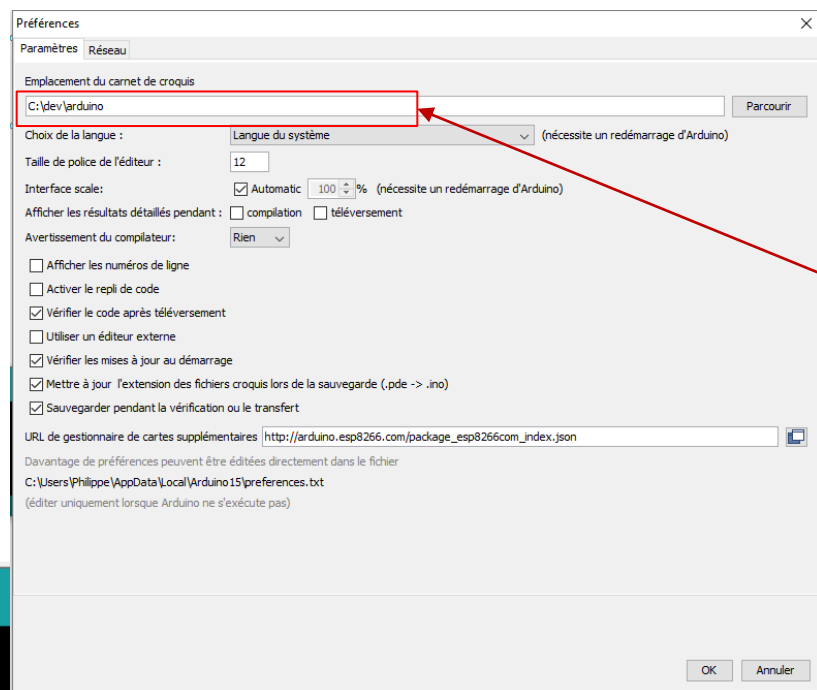
La base du programme Arduino est le « *sketch* »
(programme, prototype)
L'extension est le « **.ino** »

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

ELIPSE (C:) > dev > arduino > servo_HQ			Rechercher dans : s
Nom	Modifié le	Type	
servo_HQ.ino	24/02/2016 16:12	Fichier INO	

Deux fonctions basiques

sketch_feb08a

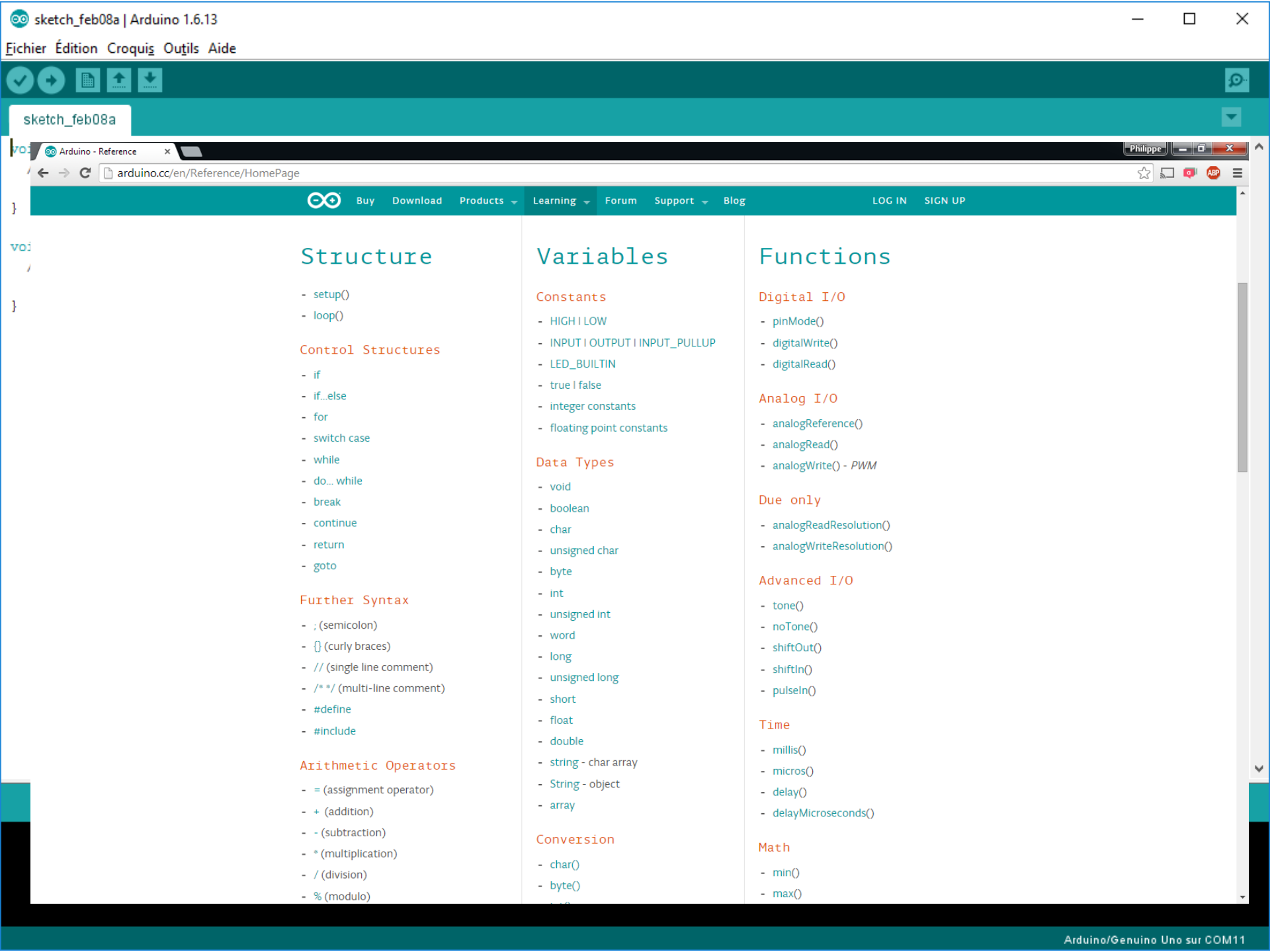
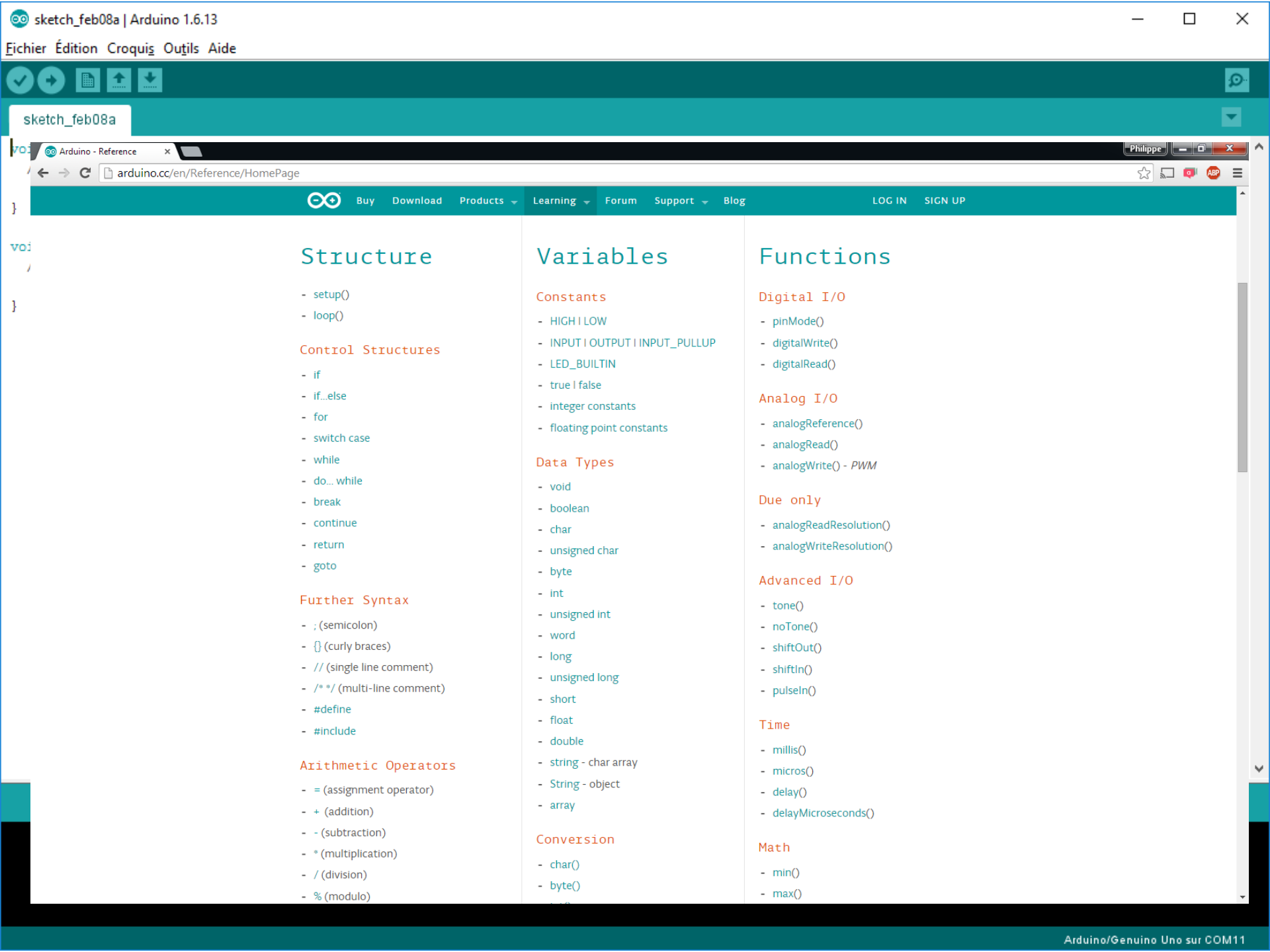
```
void setup() {  
  // put your setup code here, to run once:
```

```
}  
  
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 1.6.13

FichierÉditionCroquisOutilsAide

sketch_feb08a

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

IORef: 5V
Vin: 7-12V DC max.

IORef
Reset
+3.3V
+5V
Gnd
Gnd
Vin

ADC0 GPIO14
ADC1 GPIO15
ADC2 GPIO16
ADC3 GPIO17
SDA ADC4 GPIO18
SCL ADC5 GPIO19

Comm. ADC GPIO

ICSP:
Reset
Gnd

SCK
MISO
MOSI
+5V

GPIO18
GPIO19
Gnd
GPIO13
GPIO12
GPIO11
GPIO10
GPIO9
GPIO8
GPIO7
GPIO6
GPIO5
GPIO4
GPIO3
GPIO2
GPIO1
GPIO0

ADC4
ADC5
AREF
SCK
MISO
MOSI
CS
PWM11
PWM10
PWM9
PWM6
PWM5
PWM3
TX
RX

Serial: Serial is attached to pins 0 and 1, and to the USB-Serial microcontroller on board.

The Uno has a second microcontroller on board to handle USB-to-serial communications. This is the ICSP header for that microcontroller.

GPIO ADC Comm. PWM Interrupts

LED

INT1
INT0

Arduino/Genuino Uno sur COM11

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**

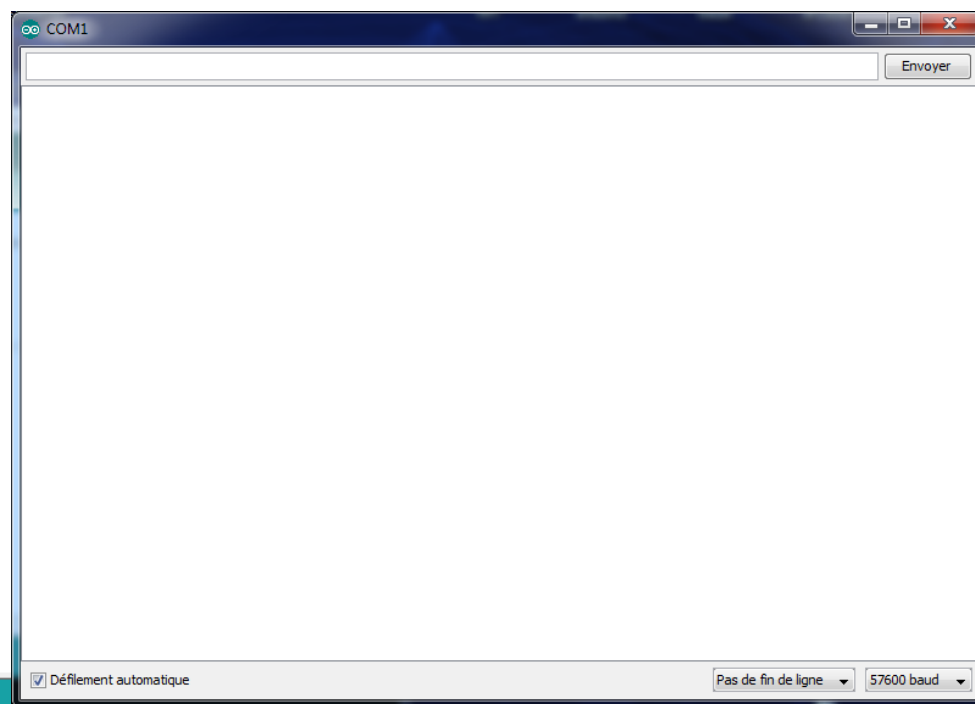


« Astuces »

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



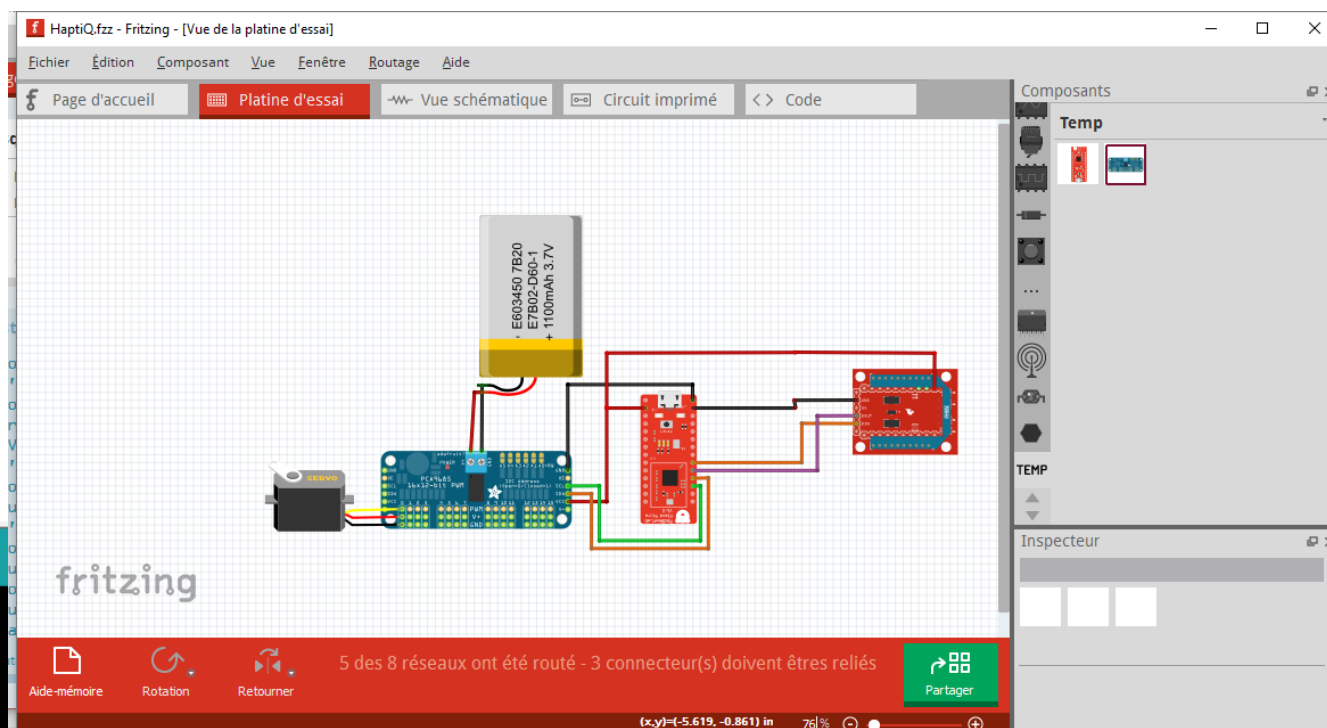
Un outil d'aide au montage

sketch_feb08a

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

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

- **Fritzing** - <http://fritzing.org> (payant depuis 2019)
<http://fritzing.org/download/0.9.3b/windows-64bit/fritzing.0.9.3b.64.pc.zip>



Un simulateur en ligne :

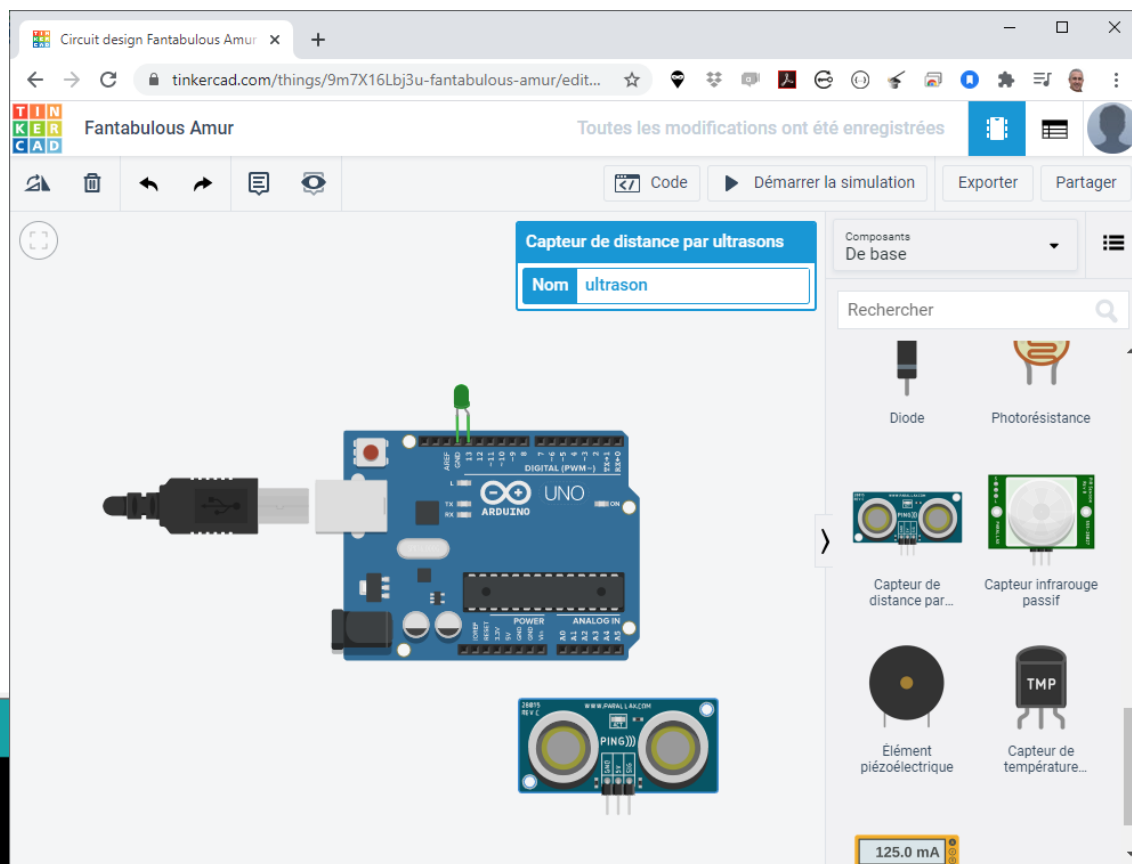
sketch_feb08a

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

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

- <https://www.tinkercad.com>

- Choisir **Circuits**



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 `1/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`

URL de gestionnaire de cartes supplémentaires

Entrez les URL supplémentaires, une par ligne

```
https://dl.espressif.com/dl/package_esp32_index.json  
http://arduino.esp8266.com/stable/package_esp8266com_index.json
```

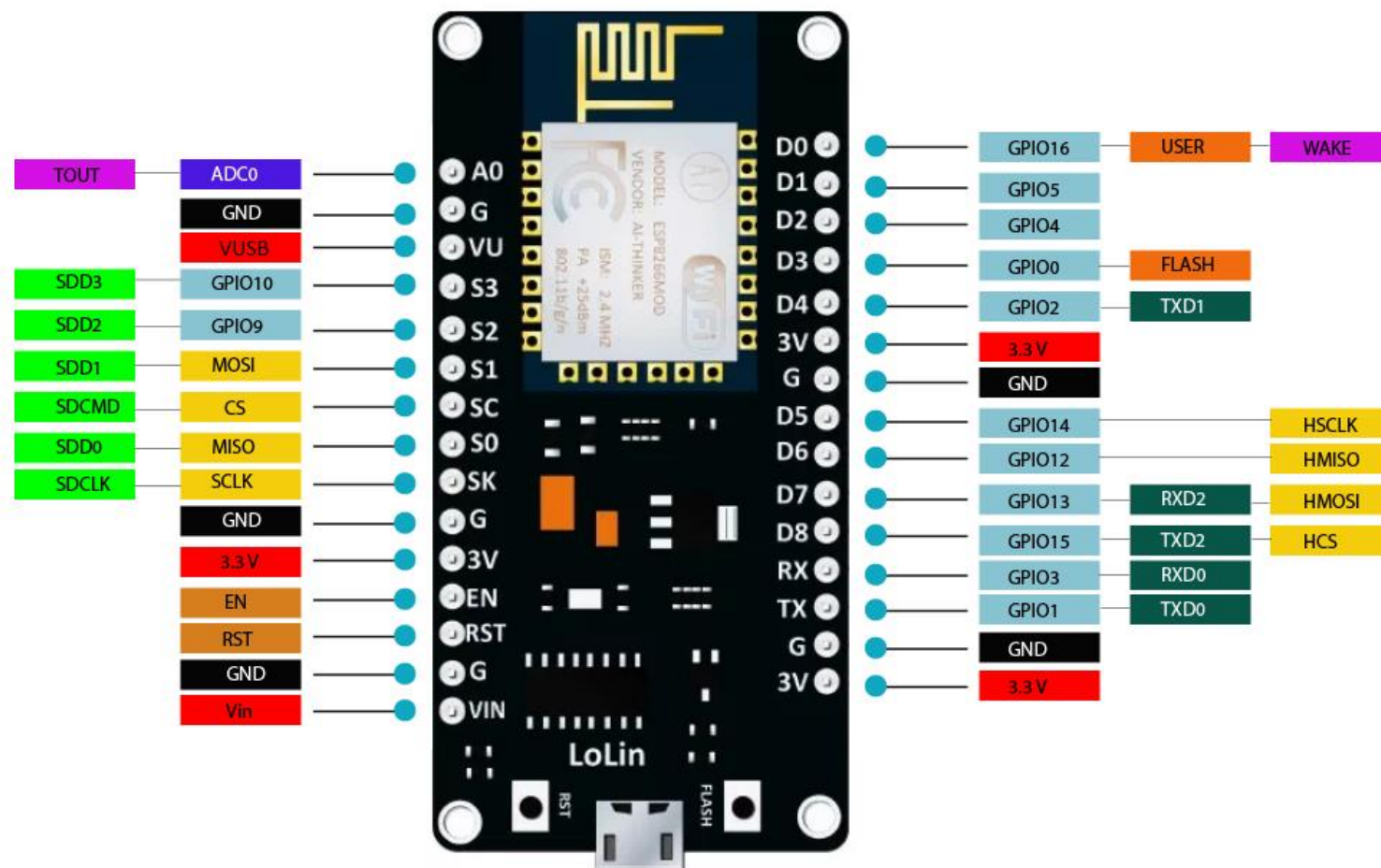
Cliquez pour la liste non-officielle des URL des cartes supportées

OK

Annuler

ESP8266

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

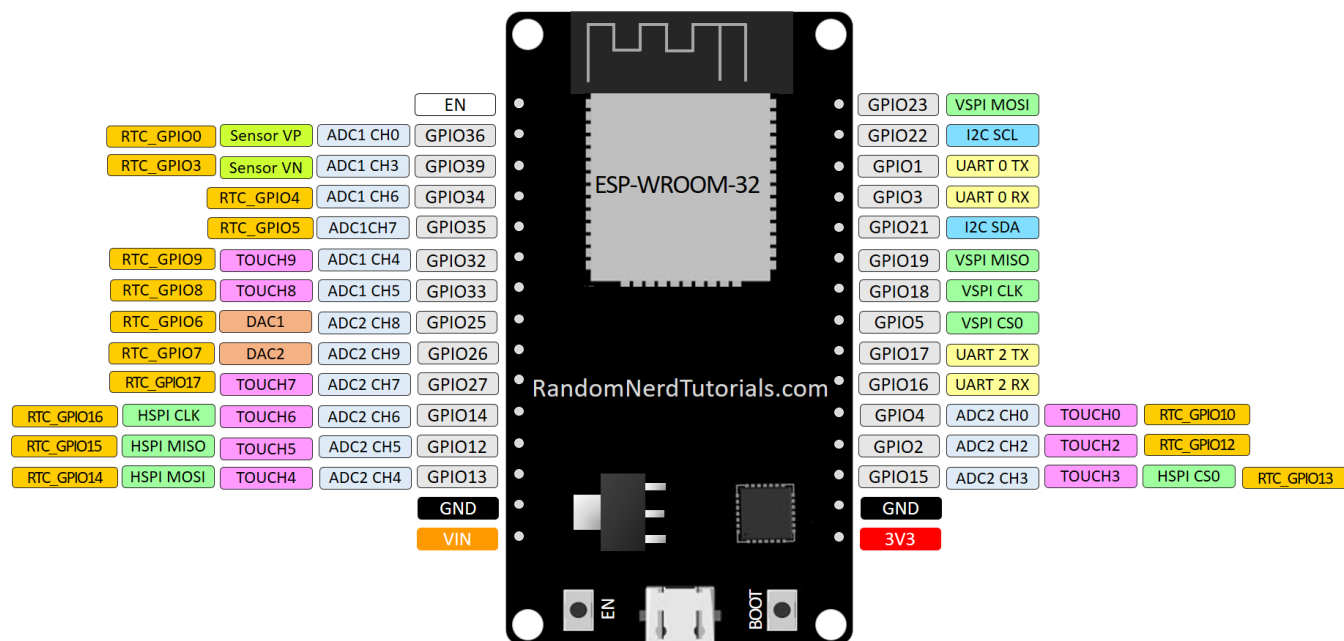
sketch_feb08a

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

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

ESP32 DEVKIT V1 – DOIT

version with 30 GPIOs



ESP32

sketch_feb08a

```

void setup
  // put y

}

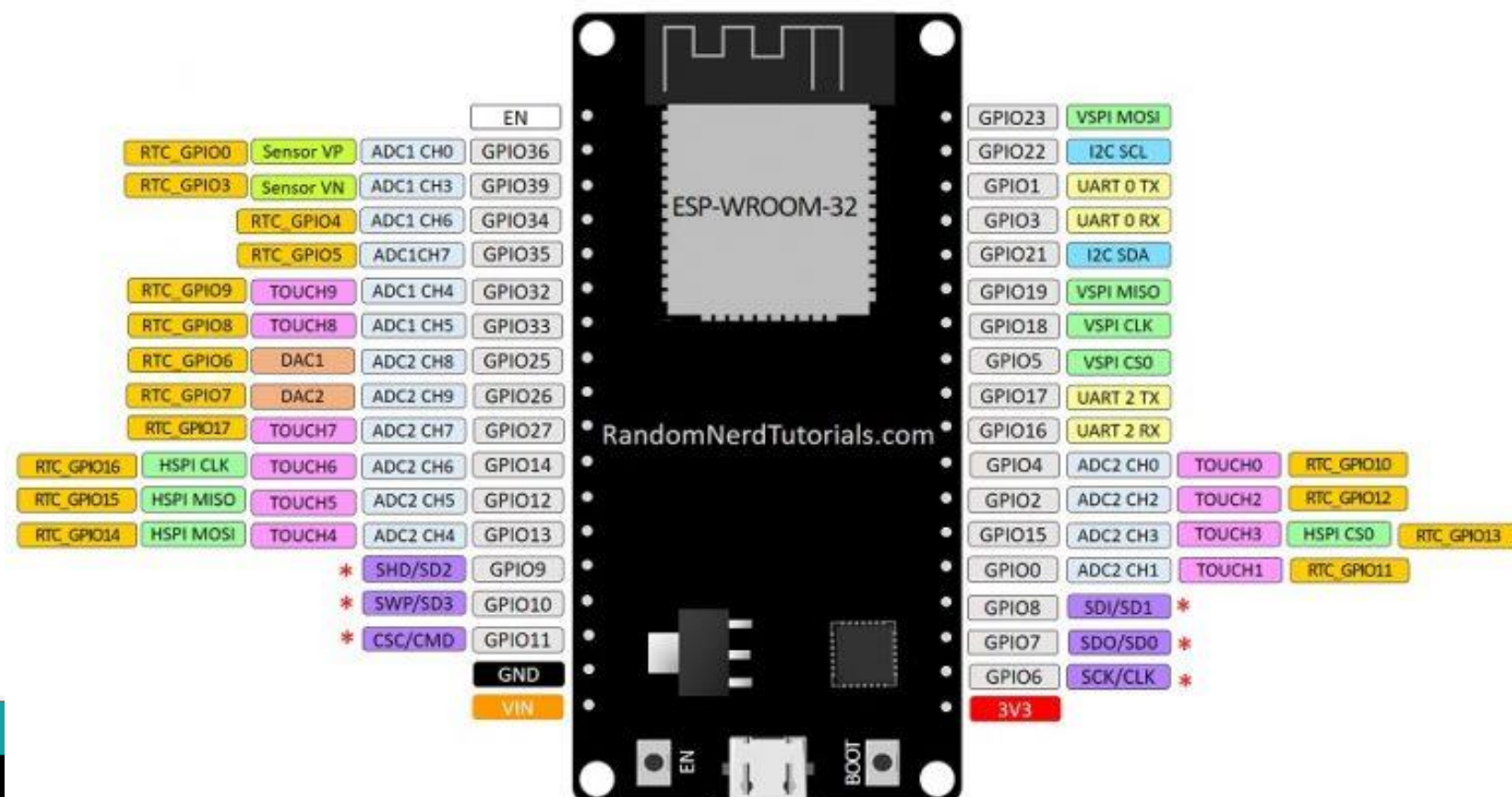
void loop(
  // put y

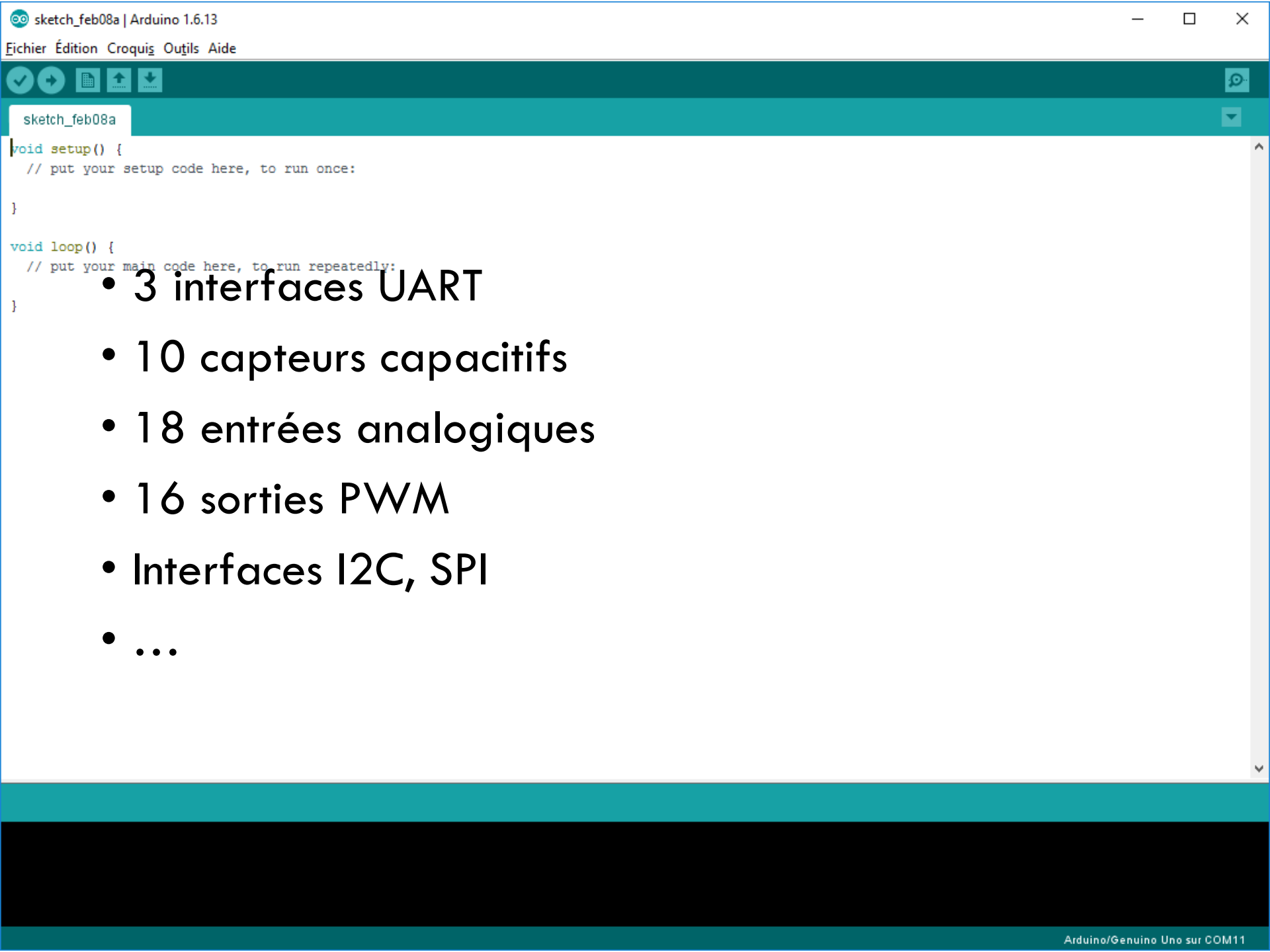
}

```

ESP32 DEVKIT V1 – DOIT

version with 36 GPIOs





sketch_feb08a

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

```
}
```

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

```
}
```

- 3 interfaces UART
- 10 capteurs capacitifs
- 18 entrées analogiques
- 16 sorties PWM
- Interfaces I2C, SPI
- ...



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 → GPIO 02 sur ESP32

→ 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

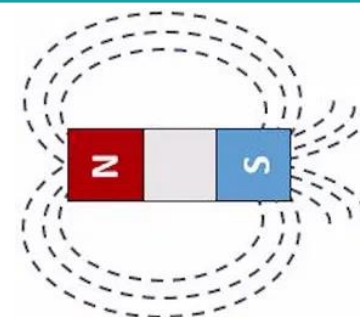
```
Serial.begin(rapidite_modulation)  
Serial.println()
```

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

Effet Hall

sketch_feb08a

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



- **hallRead()**
 - Un capteur à effet Hall

→ Écrire un programme qui allume la LED interne quand on approche un aimant (déterminer le seuil de déclenchement)

Capteur de toucher

sketch_feb08a

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

- **analogTouch()**
 - Des capteurs de toucher (T0 à T9)

→ Écrire un programme qui allume la LED interne quand on touche le capteur T0



Capteur T° et Humidité

sketch_feb08a

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



- **DHT** – capteur de T° et d'humidité

DHT sensor library

by Adafruit Version 1.4.1 **INSTALLED**

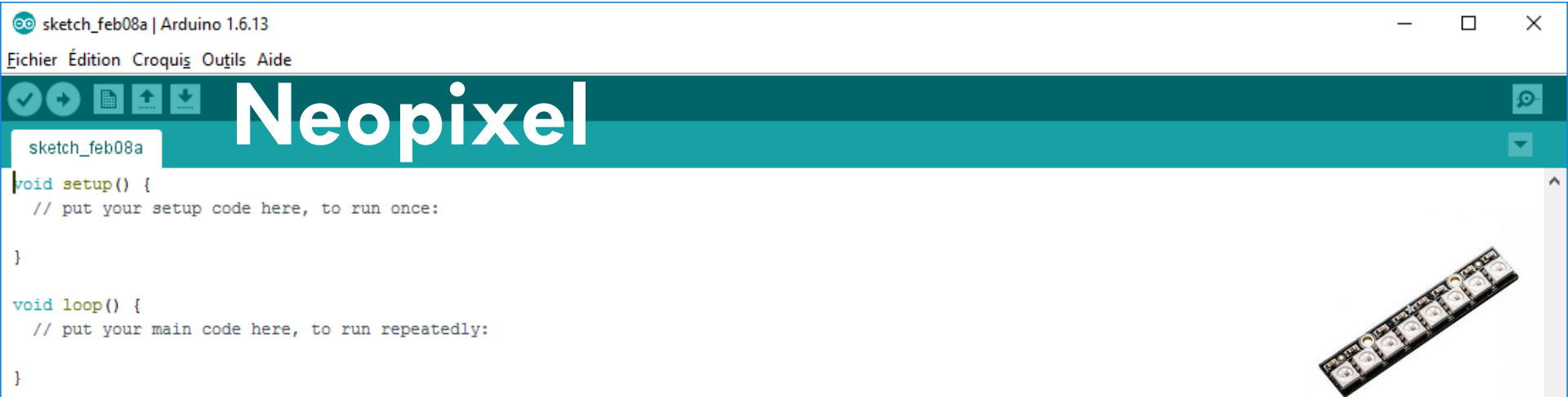
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors

[More info](#)

Sélectionner une version ▾

Installer

→ Ecrire un programme qui renvoie la température et l'humidité quand on touche le capteur T0



- **Adafruit Neopixel – des pixels RGB**

Adafruit NeoPixel

by **Adafruit** Version **1.7.0** **INSTALLED**

Arduino library for controlling single-wire-based LED pixels and strip. Arduino library for controlling single-wire-based LED pixels and strip.

[More info](#)

Sélectionner une version ▼

Installer

→ Modifier le code de telle manière que l'on puisse changer de couleur (aléatoire) quand on utilise un capteur de toucher

sketch_feb08a | Arduino 1.6.13

FichierÉditionCroquisOutilsAide

✓→📄⬆️⬇️

Lecteur NFC

sketch_feb08a

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



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 ▾

Installer

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

Arduino/Genuino Uno sur COM11



Borne wifi

sketch_feb08a

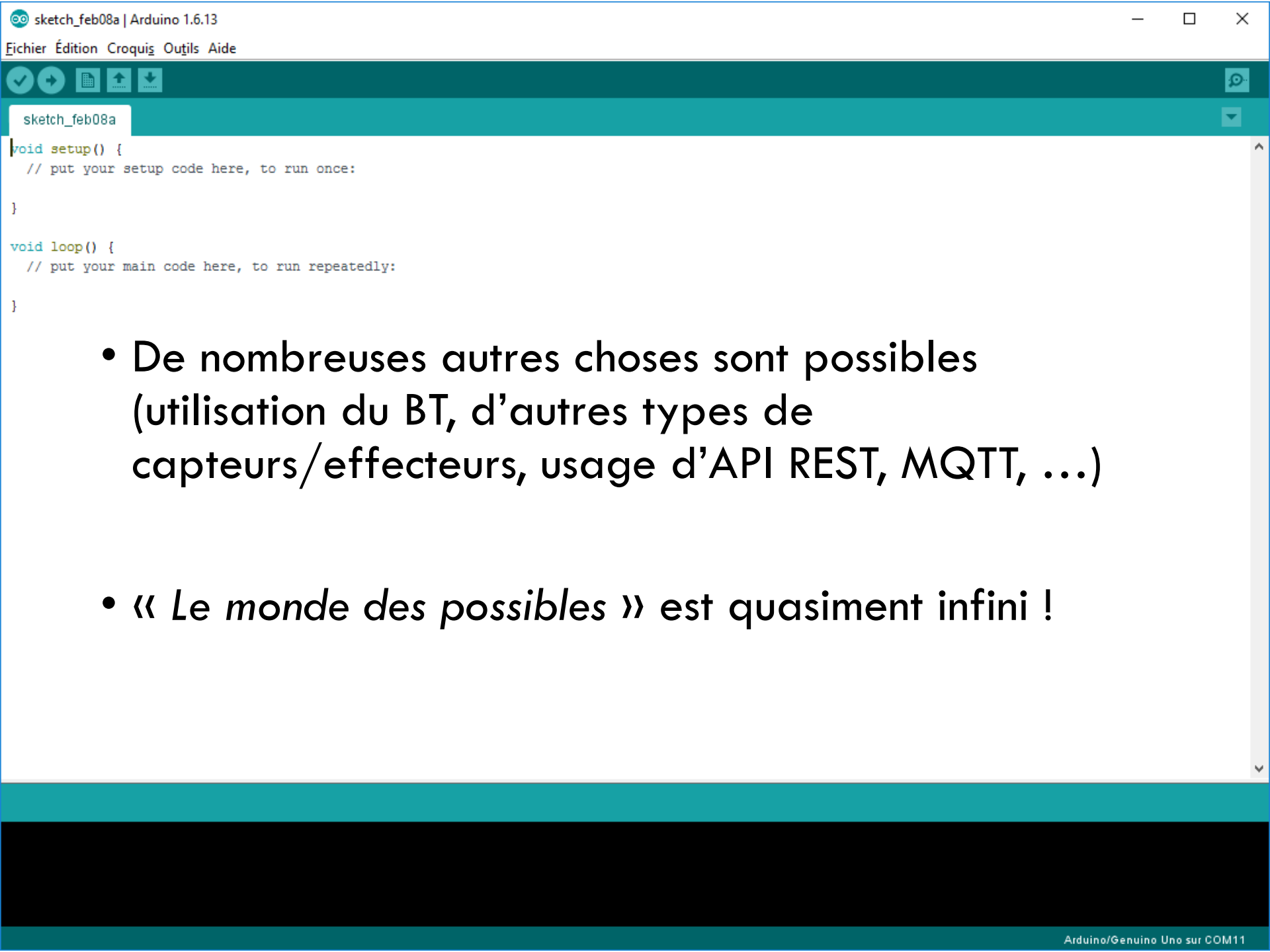
```
void setup() {  
  // put your setup code here, to run once:  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
}
```



- **Une borne et/ou un client wifi**

L'ESP32 peut servir d'AP (Access Point) et/ou être client d'un routeur Wifi

On peut y implémenter un (mini)-serveur web par exemple



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 BT, d'autres types de capteurs/effecteurs, usage d'API REST, MQTT, ...)
- « *Le monde des possibles* » est quasiment infini !

Mini-Projet

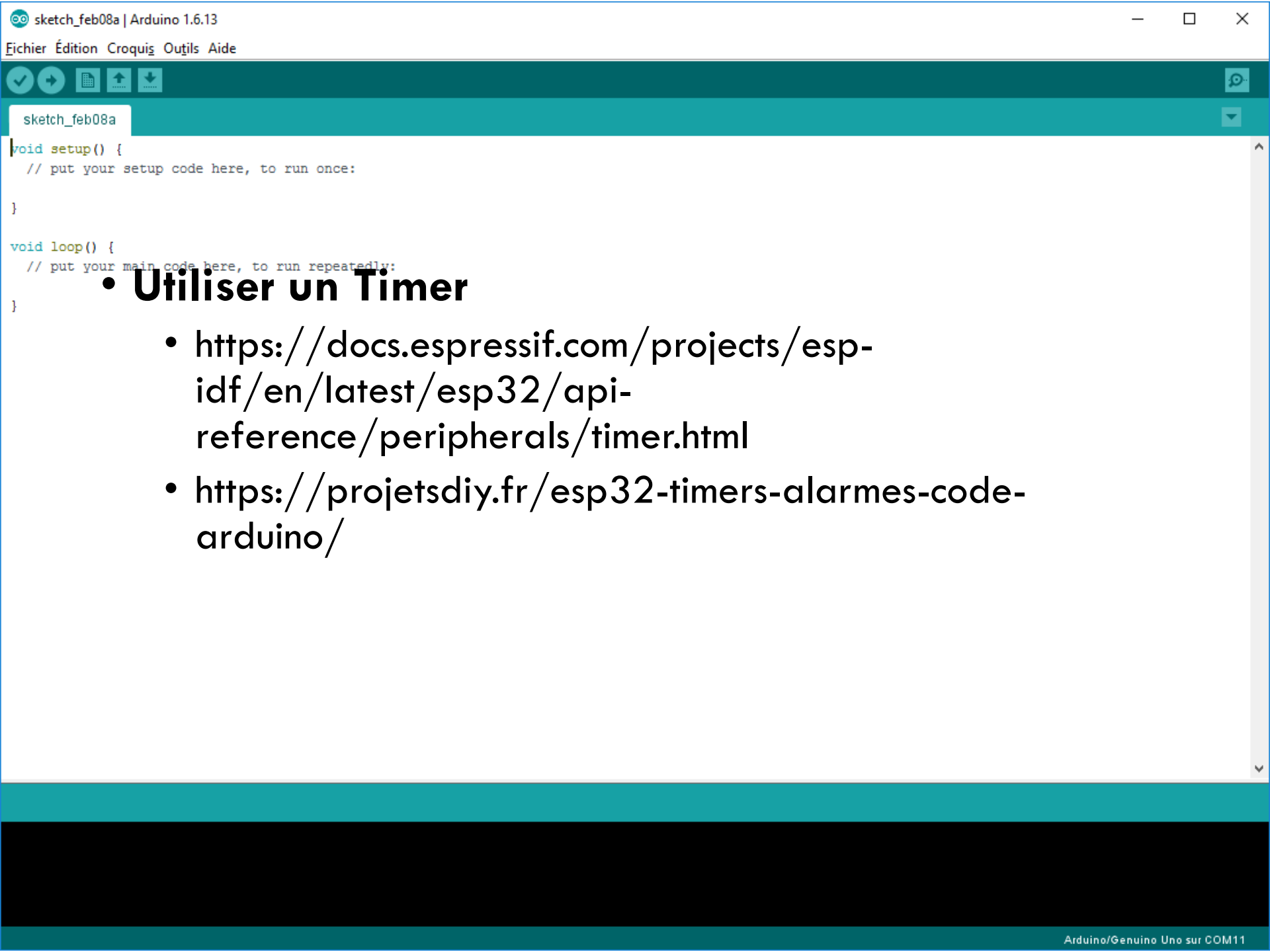
sketch_feb08a

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

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

Nous souhaitons gérer un feu tricolore (alternance vert 1 mn, orange 10 s, rouge 30 s).

- Un accès par wifi est possible qui permet de récupérer l'état courant du feu tricolore.
- Un appui sur un bouton d'appel permet de passer à l'orange puis au rouge
- Ajouter d'autres types de capteurs pour accéder à d'autres données
- Enfin Une commande secrète (par wifi) permet de changer aléatoirement l'état courant par une des trois couleurs



- **Utiliser un Timer**

- <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/timer.html>
- <https://projetsdiy.fr/esp32-timers-alarmes-code-arduino/>