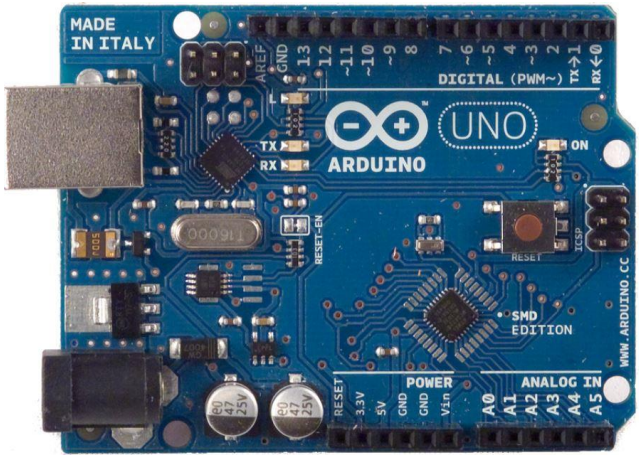
DEBUTER AVEC ARDUINO

Sommaire :

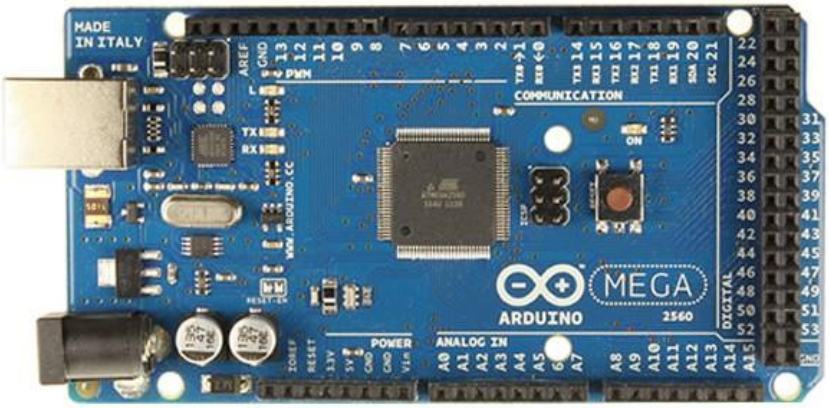
Les cartes Arduino

Les cartes communes et pas chères !!



Arduino Uno R3 (révision 3)

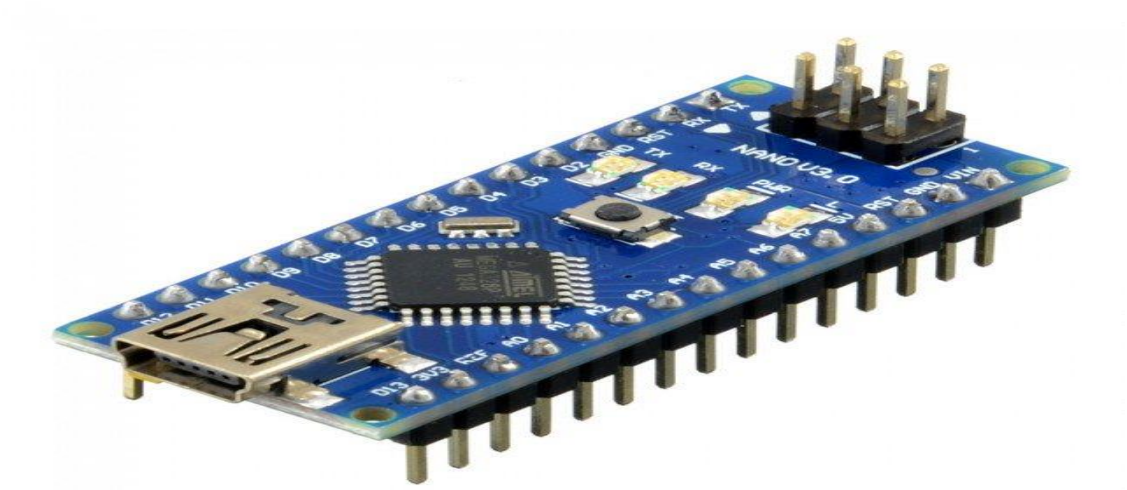
- **13 Entrées / sorties numérique**
- **6 Entrées analogique**
- **Tension délivrée 5V & 3.3V**
- **Tension admissible MAXI : 12v (avec adaptateur) Jack**
- **Prise USB classique**
- **Port série**



Arduino Méga 2560

- **54 Entrées / sorties numérique (dont 14 PWM)**
- **16 Entrées analogique**
- **Tension délivrée 5V & 3.3V**
- **Tension admissible MAXI : 12v (avec adaptateur) Jack**
- **Prise USB classique**
- **Port série**

La petite carte !

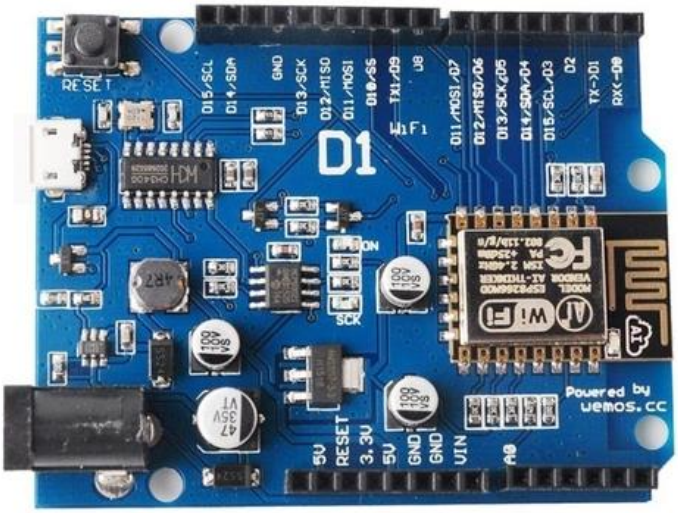


Arduino nano

- **22 Entrées / sorties numérique (dont 6 PWM)**
- **8 Entrées analogique**
- **Tension délivrée 5V & 3.3V**
- **Tension admissible 5V**
- **Prise USB nano**
- **Port série**

Les cartes compatibles

ESP8266 et exotiques



Wemos

Wemos D1 R1 ou Wemos D1 R2

Basée sur ESP8266

Avantage :

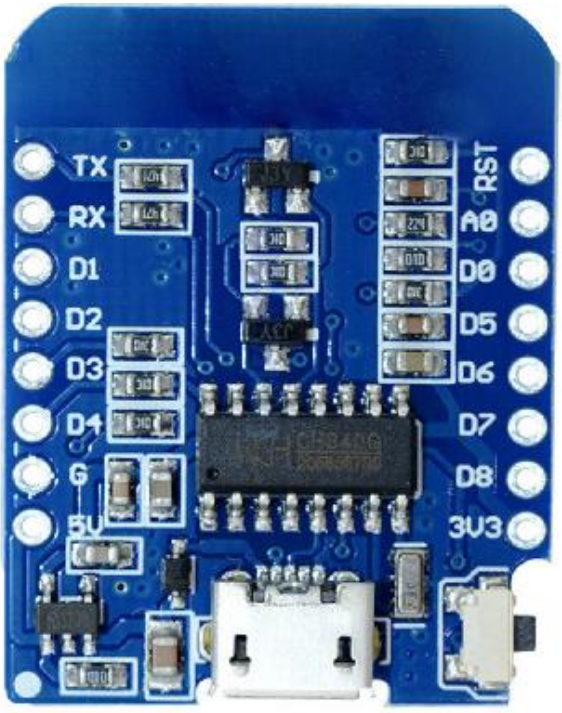
- **Vitesse 10 fois plus rapide que la uno**
- **Wifi**
- **Prise USB nano**
- **PWM sur toutes les broches digitales**

Inconvénient :

- **Tension délivrée : 3.3V (ici pas de 5V, ceci peut poser problème avec certains mofset)**
- **1 entrée analogique**

- **11 Entrées / sorties numérique (dont 11 PWM)**
- **1 Entrée analogique**
- **Tension délivrée 3.3V**
- **Tension admissible MAXI : 12v (avec adaptateur) Jack**
- **Prise USB nano**
- **Port série**

La version nano D1 mini



Avantage :

- **Vitesse 10 fois plus rapide que la nano**
- **Wifi**
- **Prise USB nano**
- **PWM sur toutes les broches digitales**

Inconvénient :

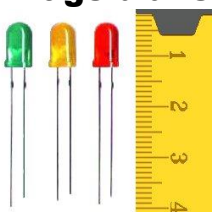
- **Tension délivrée : 3.3V (ici pas de 5V, ceci peut poser problème avec certains mofset)**
- **1 entrée analogique**

- **11 Entrées / sorties numérique (dont 11 PWM)**
- **1 Entrée analogique**
- **Tension délivrée 3.3V**
- **Tension admissible MAXI : ?**
- **Prise USB nano**
- **Port série**

Vous trouverez les infos sur les cartes wemos à cet URL : https://wiki.wemos.cc

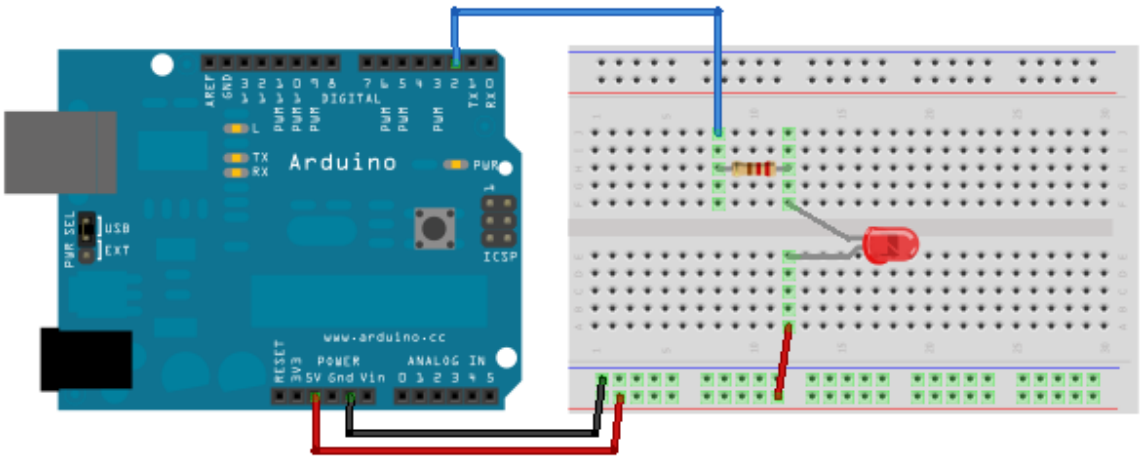
1 Les Sorties Numériques

Allumage d'une led



Voici le montage de base

Montage sur plaque d'essai



Dans notre exemple nous utiliserons le pin 13 ☺ car elle a une led intégrée sur la carte (ce qui nous permet de vérifier si ça marche).

Nous passons désormais au code dans l'IDE Arduino.

```
// Ici on place les déclarations
// Les librairies
// Les Variables
// Les constantes
```

```
void setup()
{
 // put your setup code here, to run once:
 // ICI on place le code à effectuer une seule fois
 // Initialisation de la broche 13 comme étant une sortie
    pinMode( 13, OUTPUT); // version arduino
 // pinMode( D13, OUTPUT); // version ESP8266 (wemos D1 et autres)

}
```

```
void loop()
 {
  // put your main code here, to run repeatedly:
  // ICI c'est le code qui se répète en boucle (à l'infini)
  // Écriture en sortie (broche 13) d'un état BAS
     digitalWrite( 13, LOW); // pour arduino
  // digitalWrite( D13, LOW); // pour ESP8266
}
```

Alors dans la première partie on place comme indiqué
Les librairies, et les variables (on appelle ça les déclarations).

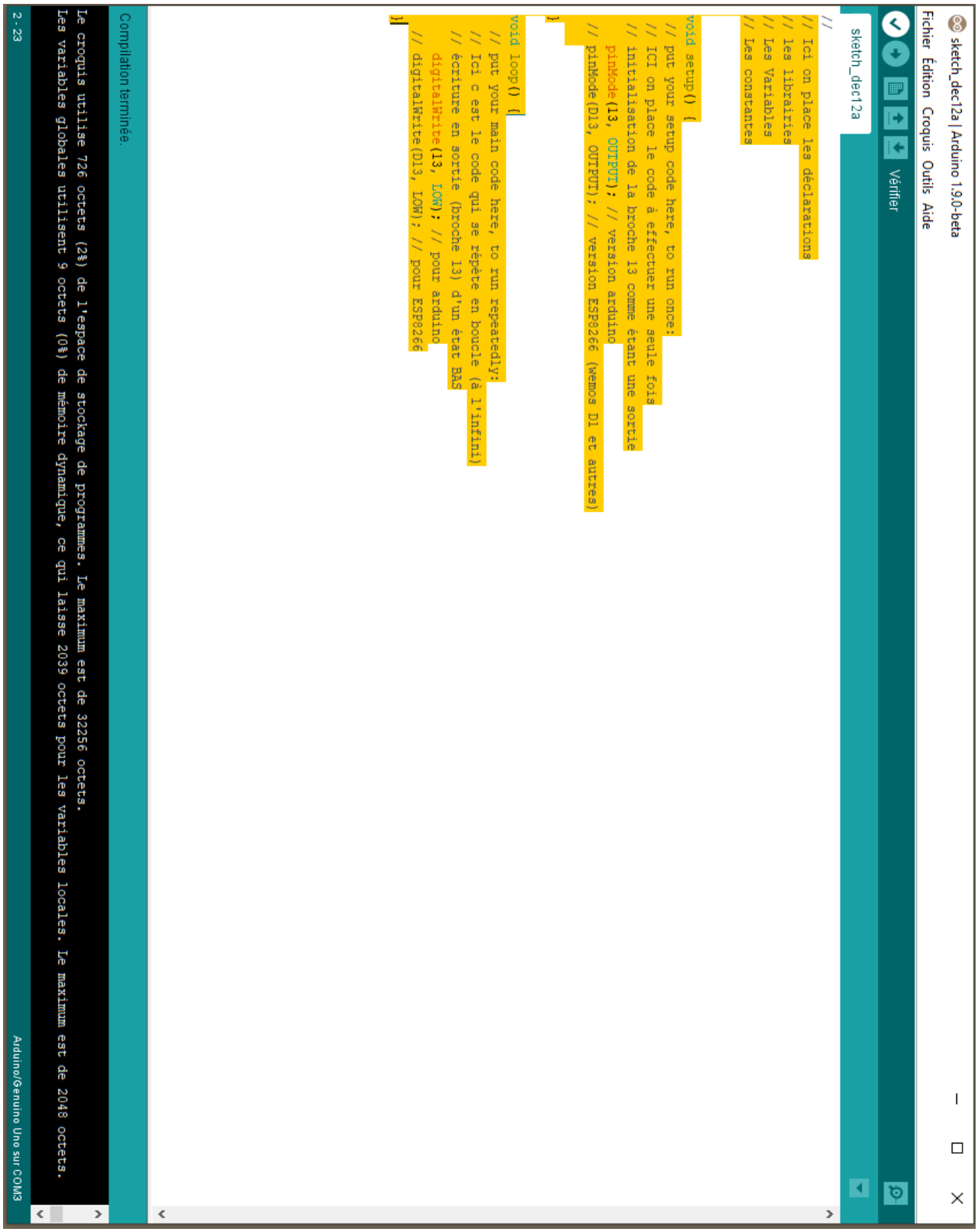
Dans la deuxième partie, on initialise les broches il faut le faire qu'une seule fois…
On peut aussi se servir de cette partie pour tester les éléments … (Led, récepteurs, et afficheurs).

Dans la troisième partie, on écrit le programme qui sera répété à l'infini !!!

Dans cette partie on peut tout faire ☺ ^^

Si notre led, n'est pas allumée, Remplacez LOW, par HIGH.

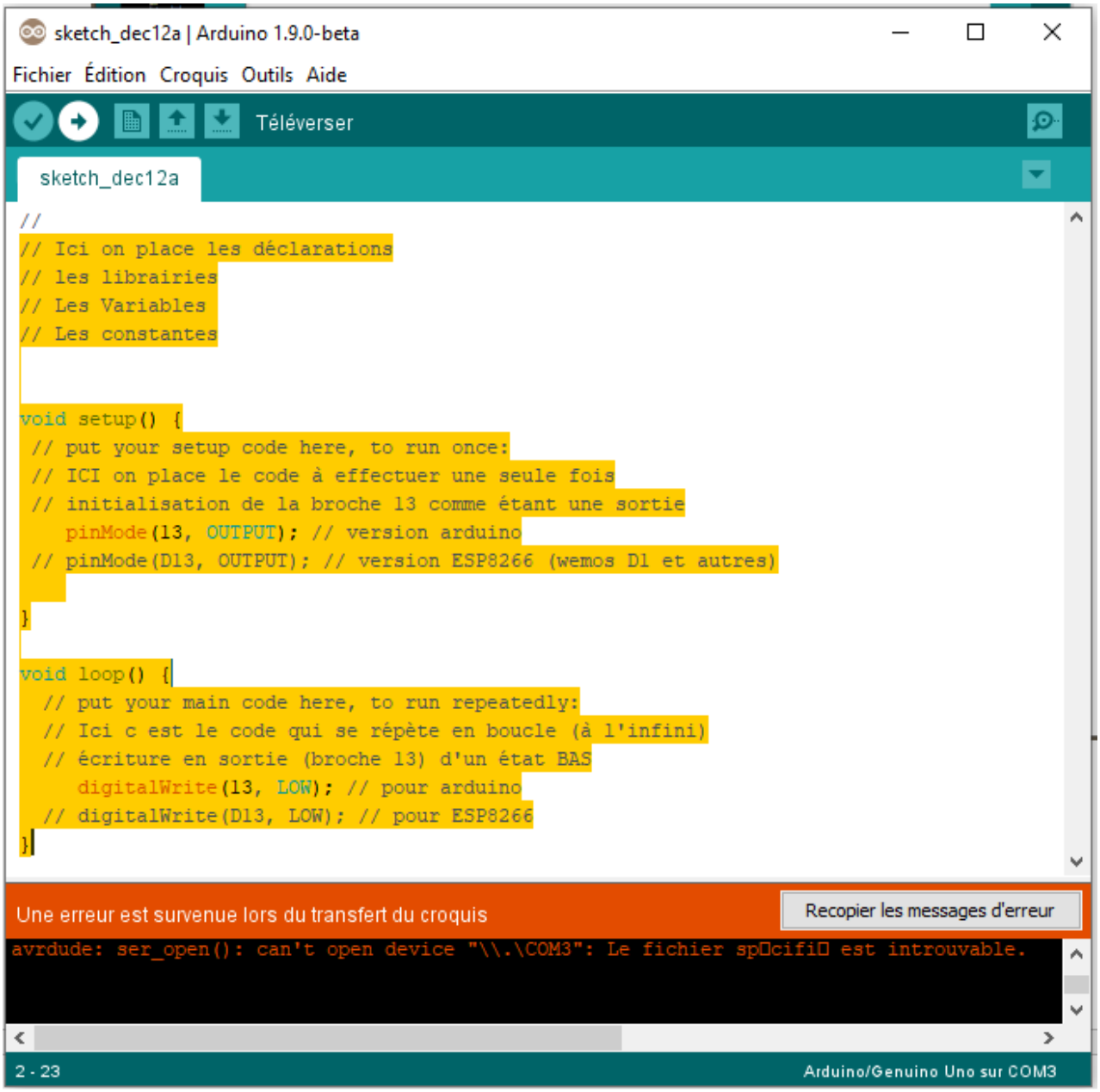
Attention, Pour éviter les erreurs, N'oubliez pas les ; en fin de ligne.



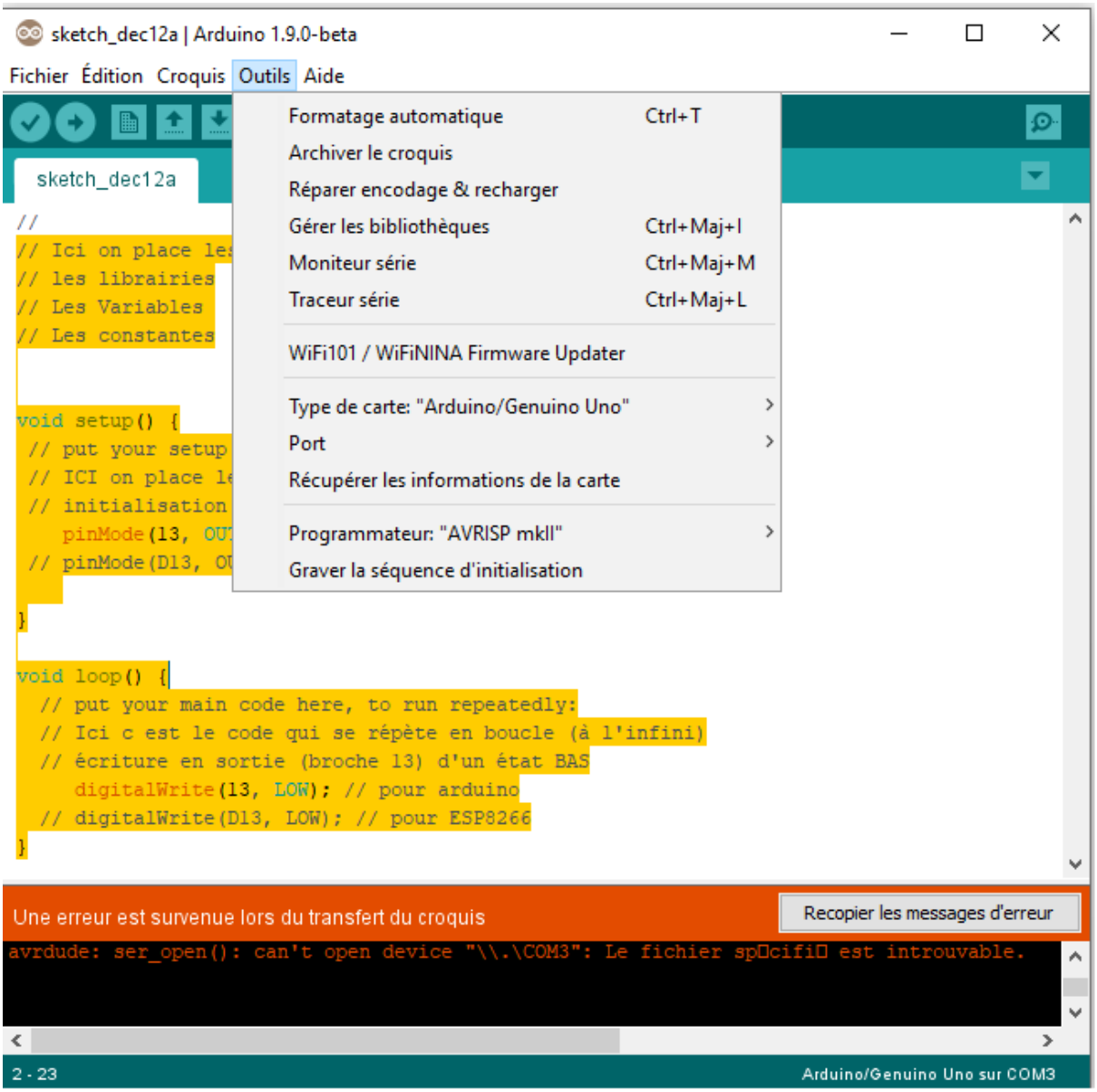
La coche surlignée en blanc permet de vérifier si y a des erreurs.

Une fois les erreurs éliminées, on peut TELEVERSER. TELEVERSER veut dire : Compiler le programme puis le transférer dans notre carte !!!

Si votre éditeur arduino ne peut pas communiquer avec la carte vous aurez une erreur comme ceci.



Dans ce cas, on se place ici :



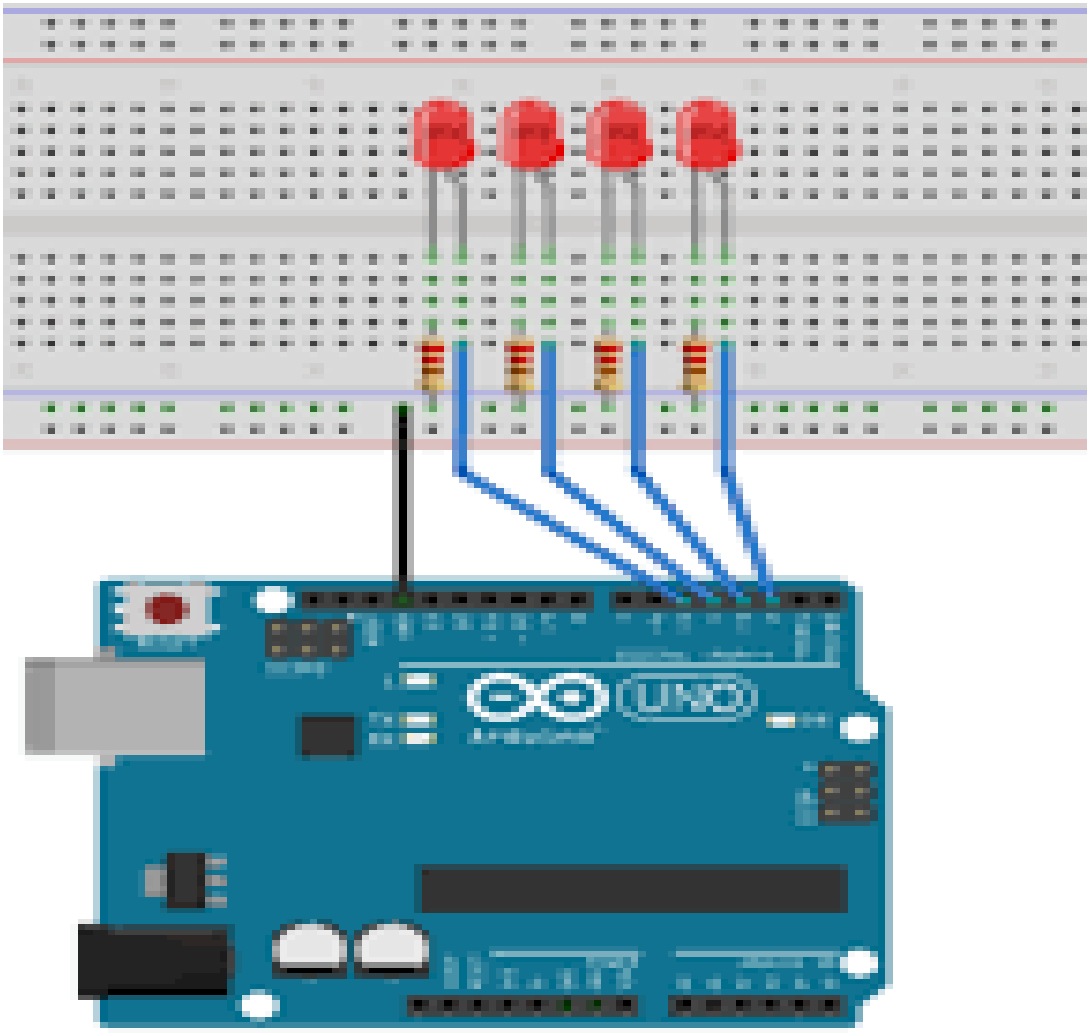
On choisit le bon ‘Type de carte’ : Arduino uno .

On choisit le ‘Port’ (si la carte est reconnue par l’ordinateur elle s’affiche ici).

On relance le TELEVERSEMENT et ça doit marcher, dans ce cas, le bandeau d’en bas reste dans la couleur verte.

Bon on a allumé une led, mais on peut faire quoi d’autre ?

Allumons désormais 4 Led



Notre nouveau programme ☺

```
// pour plus de facilités on va nommer nos led dans des constantes
const int led_rouge_1 = 2;
const int led_rouge_2 = 3;
const int led_rouge_3 = 4;
const int led_rouge_4 = 5; // attention à spécifier le D devant le numéro de broche si vous est sur ESP
```

Donc pour rendre notre programme plus facile à comprendre on défini nos leds dans les constantes

Ce qui nous permet d' < oubliez > que nous sommes sur arduino ou sur ESP et on se concentre sur le code.

Une constante est une variable qui ne bougera JAMAIS.

Ou pour imager une page de livre qu'on ne peut pas modifier !! on lui donne une valeur au début et elle n'est plus modifiable.

```
void setup() {
 // put your setup code here, to run once:
 // ICI on place le code à effectuer une seule fois
 // initialisation des broches avec le noms des broches et non un adressage direct,  comme étant des sorties
   pinMode(led_rouge_1, OUTPUT);
   pinMode(led_rouge_2, OUTPUT);
   pinMode(led_rouge_3, OUTPUT);
   pinMode(led_rouge_4, OUTPUT);
   // ici on teste les Leds on allumes les 4 en meme temps pour voir si ca marche
   digitalWrite(led_rouge_1, LOW);
   digitalWrite(led_rouge_2, LOW);
   digitalWrite(led_rouge_3, LOW);
   digitalWrite(led_rouge_4, LOW);
   // on attends une seconde et on eteint tout
   delay(1000);
   // la fonction delay ( en milliseconde)
   digitalWrite(led_rouge_1, HIGH);
   digitalWrite(led_rouge_2, HIGH);
   digitalWrite(led_rouge_3, HIGH);
   digitalWrite(led_rouge_4, HIGH);
    // on attends une seconde et on lance notre programme
   delay(1000);
}
```

Voilà ici nous initialisons nos broches, et comme on a des constantes avec les noms des led c'est plus facile à faire !!

Ensuite comme cette partie du programme ne s'exécute qu'une seul fois, on teste si toutes nos led fonctionnent bien, donc on les allume en même temps, on attend une seconde, on les éteint, et hop on passe à la partie < INTELLIGENTE > du programme.

J'aimerai attirer votre attention sur les points qui génèrent 90% des erreurs d'écriture de programme !!

Les { }
Et les
Point-virgule ;

Ils sont responsables
Des plus gros bugs informatiques
Alors concentrez-vous

Donc le reste du programme allume et éteint chaque led chacune à leur tour,
Avec un délai de 1 seconde entre chaque
Le programme allume les 4 led ensemble pendant 4 secondes, les éteint, on attend une seconde et on recommence.

```
void loop() {
  // put your main code here, to run repeatedly:
  // Ici c est le code qui se répète en boucle (à l'infini)
  // écriture en sortie (broche 13) d'un état BAS
  // on allume la led 1, pendant une seconde
     digitalWrite(led_rouge_1, LOW);
    delay(1000);
  // on eteind la led 1
   digitalWrite(led_rouge_1, HIGH);
   // on attends une seconde avant d'allumer la led 2
    delay(1000);
     // on allume la led 2, pendant une seconde
     digitalWrite(led_rouge_2, LOW);
    delay(1000);
  // on eteind la led 2
   digitalWrite(led_rouge_2, HIGH);
   // on attends une seconde avant d'allumer la led 3
    delay(1000);
     // on allume la led 3, pendant une seconde
     digitalWrite(led_rouge_3, LOW);
    delay(1000);
  // on eteind la led 3
   digitalWrite(led_rouge_3, HIGH);
   // on attends une seconde avant d'allumer la led 4
    delay(1000); // on allume la led 4, pendant une seconde
     digitalWrite(led_rouge_4, LOW);
    delay(1000);
  // on eteind la led 4
   digitalWrite(led_rouge_4, HIGH);
   // on attends une seconde avant d'allumer la led 4
    delay(1000);
    // et la on allume les 4 led ensemble pendant 4 sec
    digitalWrite(led_rouge_1, LOW);
    digitalWrite(led_rouge_2, LOW);
    digitalWrite(led_rouge_3, LOW);
    digitalWrite(led_rouge_4, LOW);
    delay(4000);
    digitalWrite(led_rouge_1, HIGH);
    digitalWrite(led_rouge_2, HIGH);
    digitalWrite(led_rouge_3, HIGH);
    digitalWrite(led_rouge_4, HIGH);
    // ici on attend une seconde et on recommence
    delay(1000);
}
```