Elise Boudjema
Justine Lam
Team : Bad & Boudji

Internet of Things - Lab 1 & 2

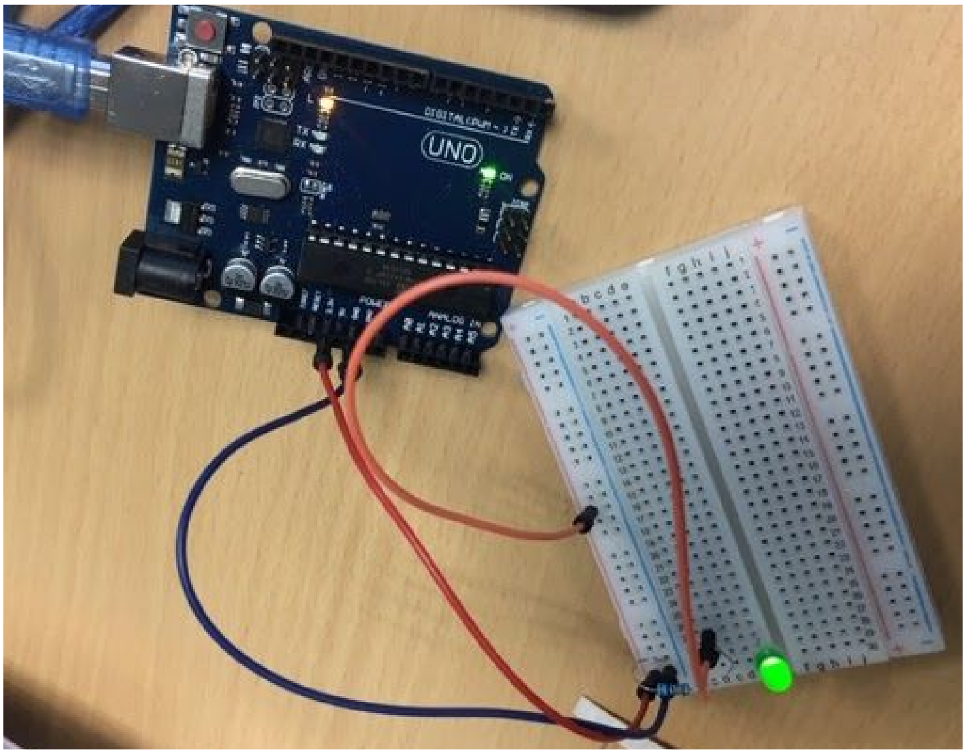
LAB 1 :

Step 1 :

For the first step, we didn’t have the battery so we could not verify if the led was working but we installed all the circuit the same way as in the picture.

Step 2 :

For this second step, we connected the circuit to an Arduino UNO board and when we tested the LED by connecting the orange wire to the red bus (VCC) the LED was working as you can see in the following picture.



Step 3 :

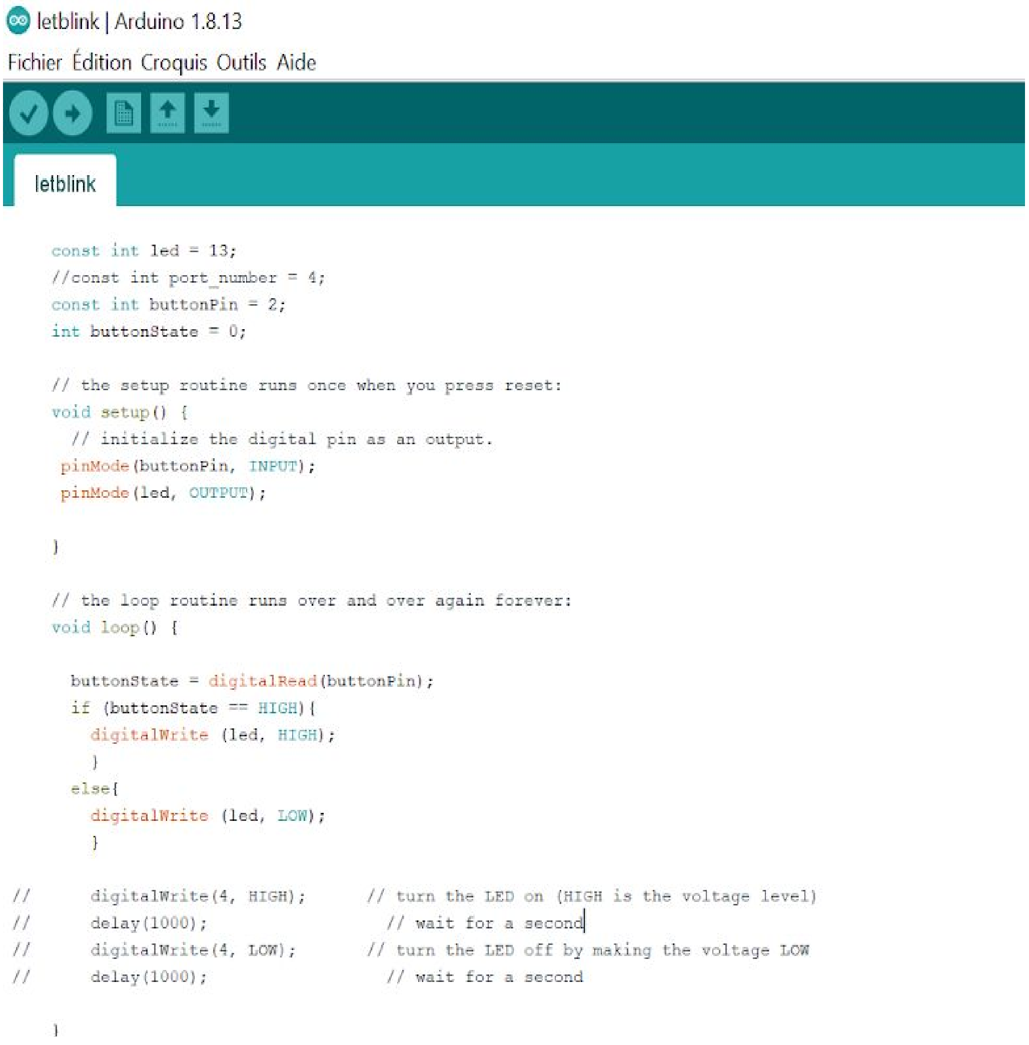
Then we downloaded Arduino IDE.

Step 4 :

Once the Arduino IDE opened, we started to modify the program in order to turn on and off the LED in port 4. The program contains two parts : a setup and a loop. The setup() function is called when a program starts. So we started by initializing the variables and pin modes. Then for the loop() function allows us to control the Arduino board.

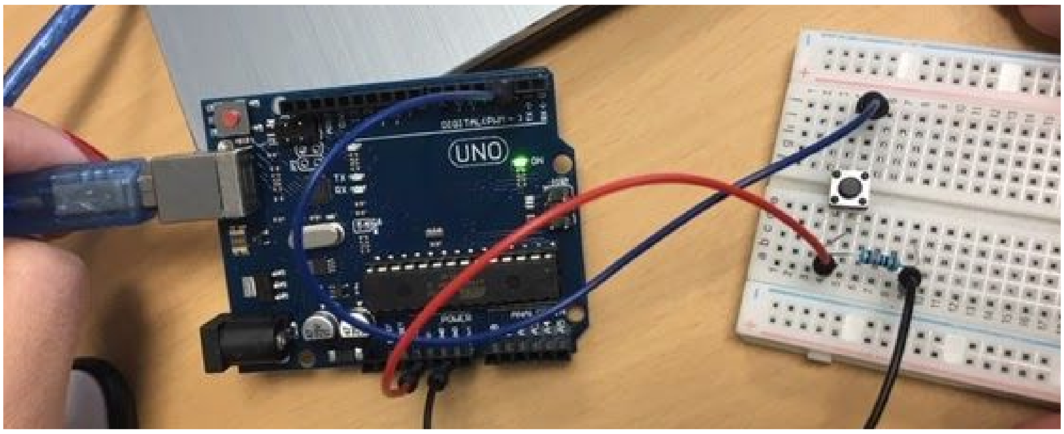
Step 5 :

For this step, we followed lesson 6. First, we installed a new circuit integrating a button so the LED will turn on when you press the button. Indeed, when the button is unpressed there is no connection between the two legs of the button while when the button is pressed the connection is made so that the pin is connected to 5V (we can read a HIGH).

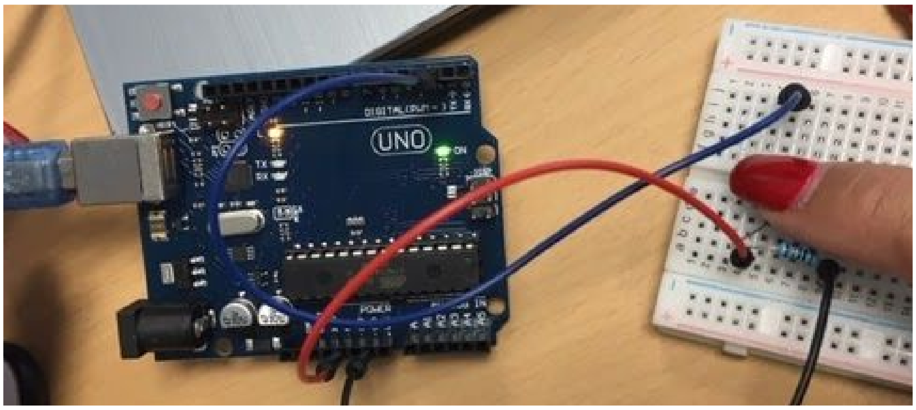

letblink | Arduino 1.8.13
Fichier Édition Croquis Outils Aide
letblink

```
    const int led = 13;
    //const int port_number = 4;
    const int buttonPin = 2;
    int buttonState = 0;

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

    }

    // the loop routine runs over and over again forever:
    void loop() {

      buttonState = digitalRead(buttonPin);
      if (buttonState == HIGH){
        digitalWrite (led, HIGH);
        }
      else{
        digitalWrite (led, LOW);
        }

//      digitalWrite(4, HIGH);      // turn the LED on (HIGH is the voltage level)
//      delay(1000);                  // wait for a second
//      digitalWrite(4, LOW);       // turn the LED off by making the voltage LOW
//      delay(1000);                  // wait for a second

    }
```

1) When we don't press the button :

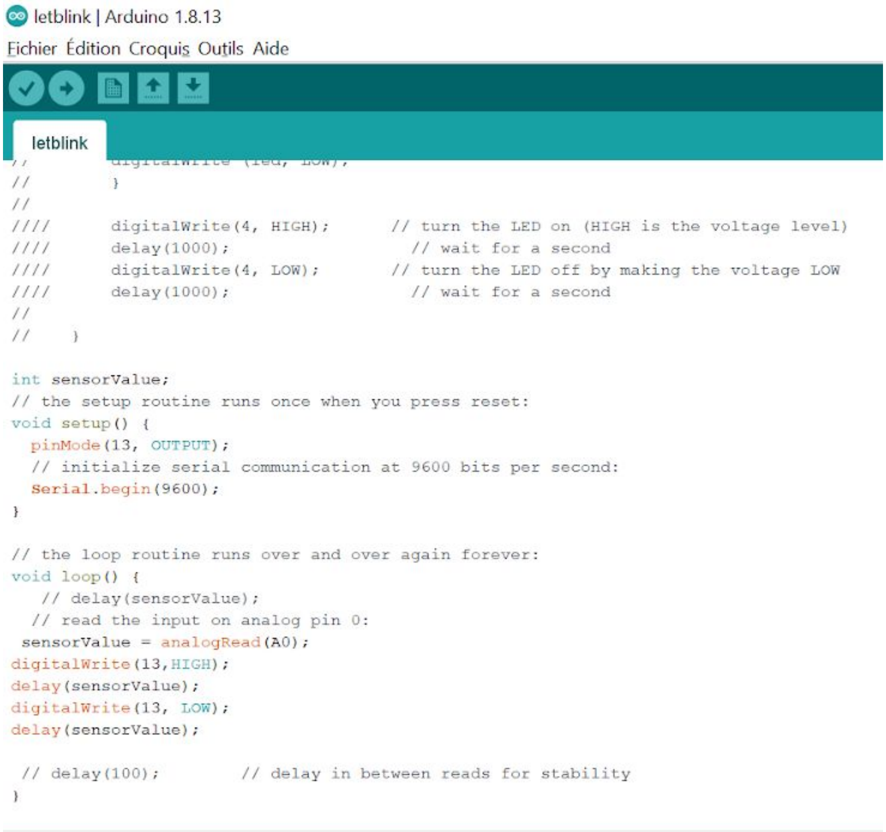


2) When we press the button :



Step 6 :

After that we followed lesson 4. For this step we used a variable resistor (a potentiometer) so we can read its value by using the analog input of the Arduino board and change the blink rate of the LED. In fact, the resistor's analog value is read as a voltage because this is how the analog inputs work.

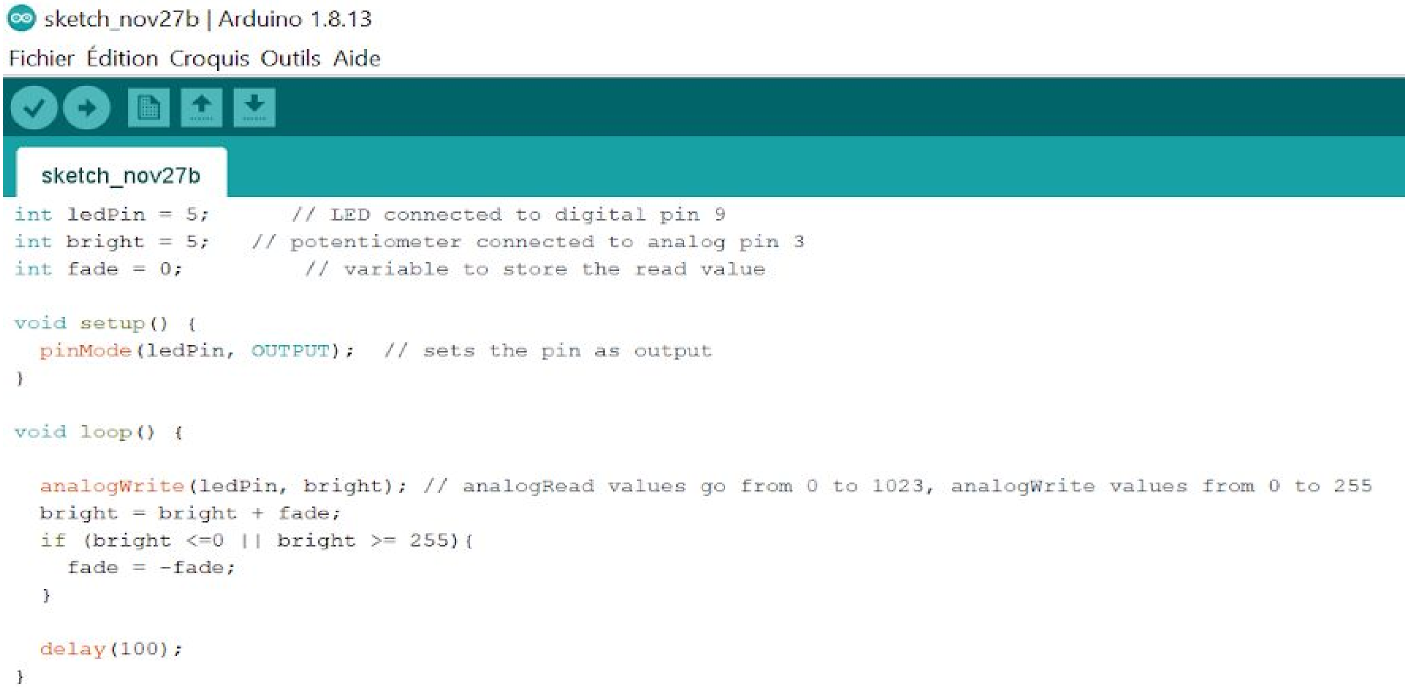


letblink | Arduino 1.8.13
Fichier Édition Croquis Outils Aide
letblink

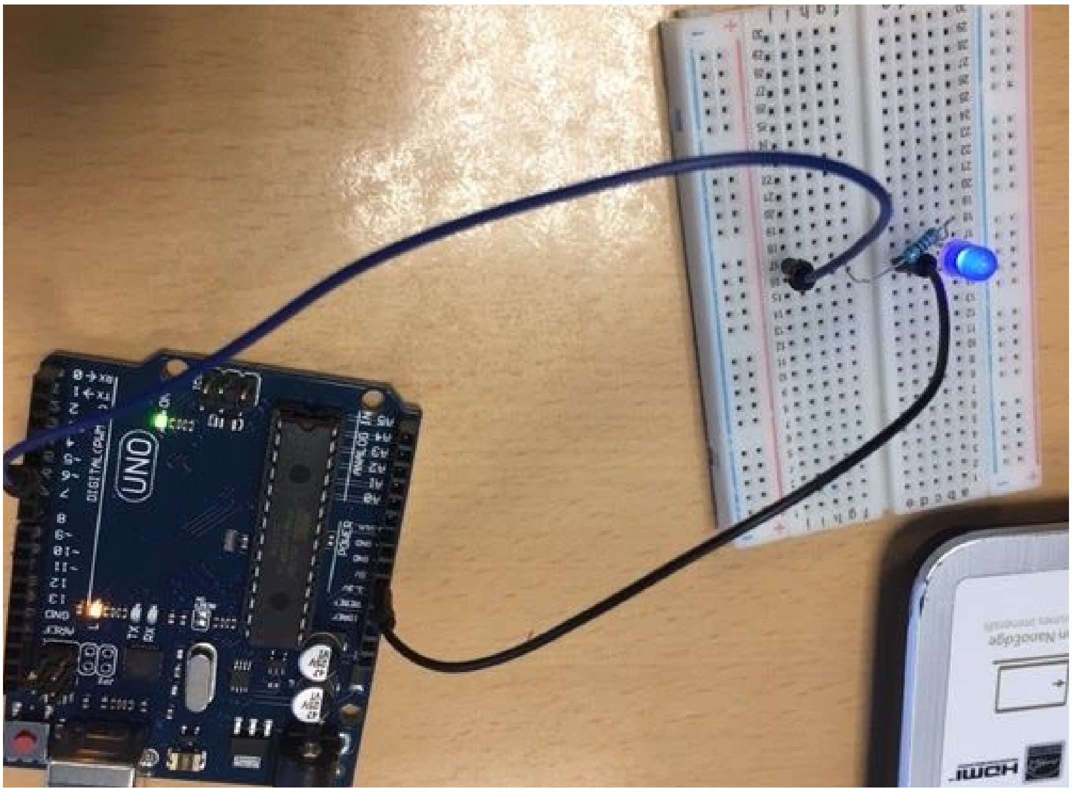
```
//          digitalWrite (led, LOW);
//          }
//
////        digitalWrite(4, HIGH);      // turn the LED on (HIGH is the voltage level)
////        delay(1000);                  // wait for a second
////        digitalWrite(4, LOW);       // turn the LED off by making the voltage LOW
////        delay(1000);                  // wait for a second
//
//      }

int sensorValue;
// the setup routine runs once when you press reset:
void setup() {
  pinMode(13, OUTPUT);
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
   // delay(sensorValue);
  // read the input on analog pin 0:
 sensorValue = analogRead(A0);
digitalWrite(13,HIGH);
delay(sensorValue);
digitalWrite(13, LOW);
delay(sensorValue);

 // delay(100);        // delay in between reads for stability
}
```

Step 7 :

Then it's lesson 5 : we used the function AnalogWrite.

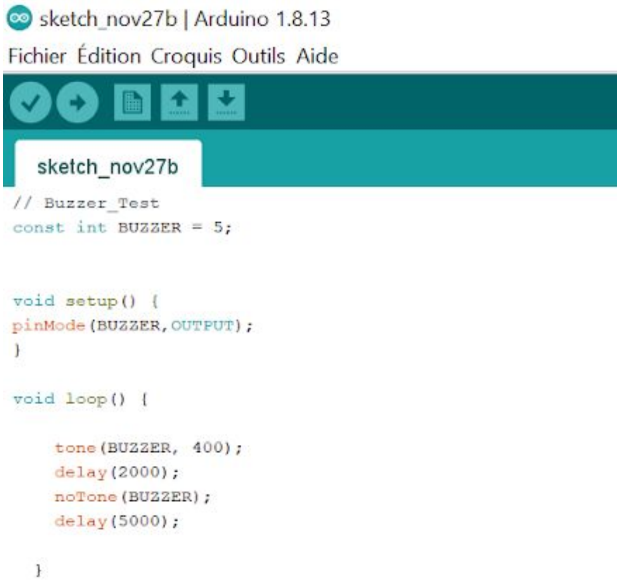

sketch_nov27b | Arduino 1.8.13
Fichier Édition Croquis Outils Aide
sketch_nov27b

```
int ledPin = 5;       // LED connected to digital pin 9
int bright = 5;    // potentiometer connected to analog pin 3
int fade = 0;         // variable to store the read value

void setup() {
  pinMode(ledPin, OUTPUT);   // sets the pin as output
}

void loop() {

  analogWrite(ledPin, bright); // analogRead values go from 0 to 1023, analogWrite values from 0 to 255
  bright = bright + fade;
  if (bright <=0 || bright >= 255){
    fade = -fade;
  }

  delay(100);
}
```



Step 8 : Buzzer

For the buzzer, we used the tone function : it selects the buzzer named BUZZER and the frequency that we choose to be =400Hz. We made it last 2 seconds (delay(2000)), made a 5 seconds (noTone and delay (5000)) pause then regenerated a 2 seconds tone as a loop.


sketch_nov27b | Arduino 1.8.13
Fichier Édition Croquis Outils Aide
sketch_nov27b

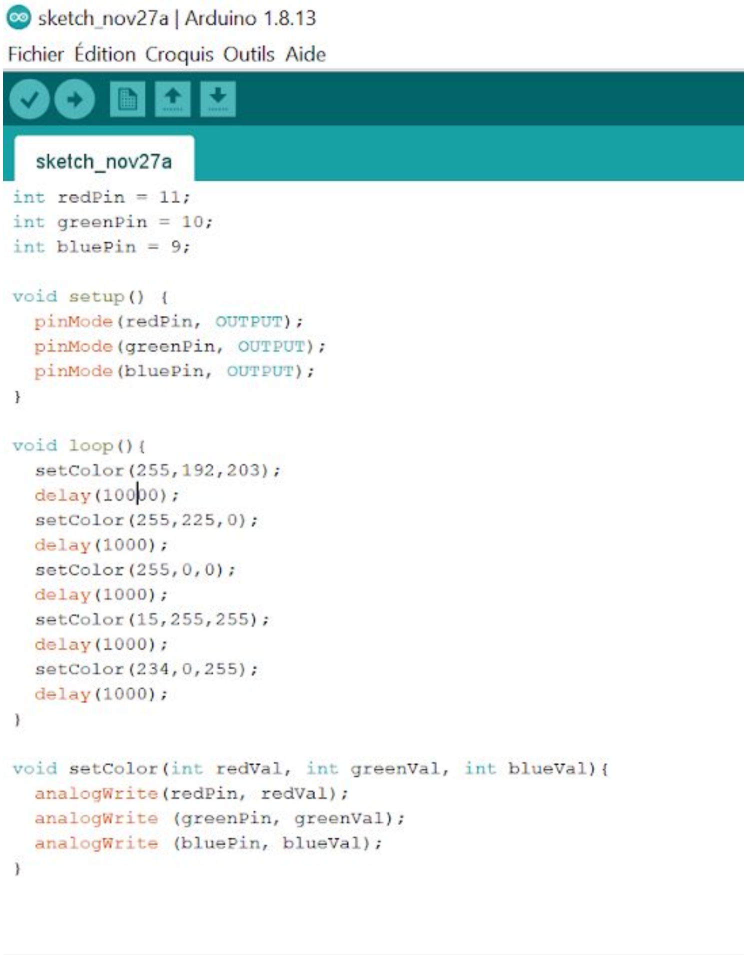
```
// Buzzer_Test
const int BUZZER = 5;


void setup() {
pinMode(BUZZER,OUTPUT);
}

void loop() {

    tone(BUZZER, 400);
    delay(2000);
    noTone(BUZZER);
    delay(5000);

  }
```

Step 9 : Fun RGB Led :

For this final step, we used the SetColor function to choose the RGB code of the color we wanted. We associated the led pins with one number from 0 to 255 and with that, we could have all the colors. After that, we called the setColor function.

LAB 2 :

Step 1 :


sketch_nov27a | Arduino 1.8.13
Fichier Édition Croquis Outils Aide
sketch_nov27a

```
int redPin = 11;
int greenPin = 10;
int bluePin = 9;

void setup() {
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
}

void loop(){
  setColor(255,192,203);
  delay(10000);
  setColor(255,225,0);
  delay(1000);
  setColor(255,0,0);
  delay(1000);
  setColor(15,255,255);
  delay(1000);
  setColor(234,0,255);
  delay(1000);
}

void setColor(int redVal, int greenVal, int blueVal){
  analogWrite(redPin, redVal);
  analogWrite (greenPin, greenVal);
  analogWrite (bluePin, blueVal);
}
```