

Computación Ubicua

Sesión 1 – Introducción a Arduino

Ana Castillo Martínez

Javier Albert Segui

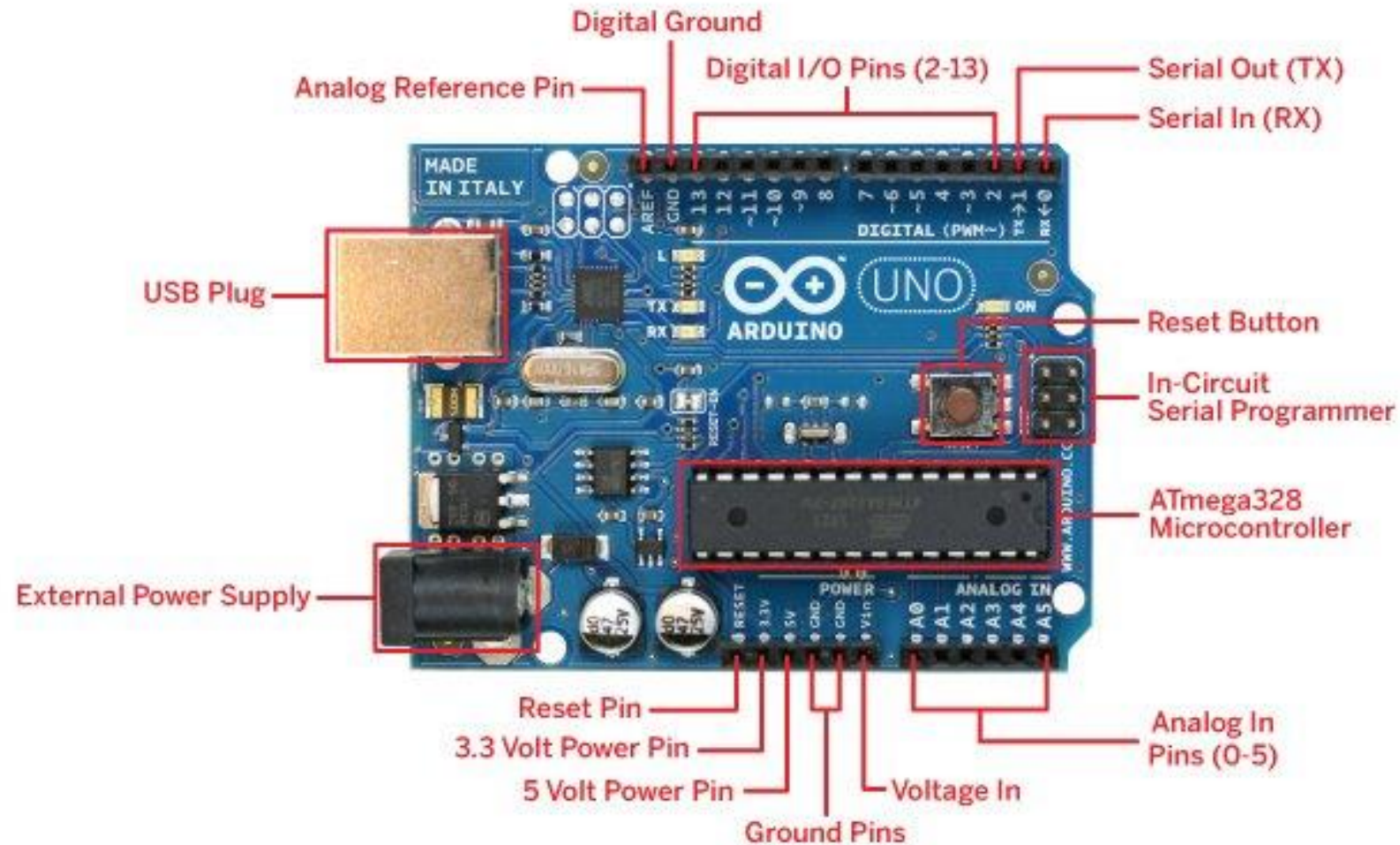
Arduino

- Plataforma de creación de electrónica de Código abierto
- Está basada en Hardware y Software libre, flexible y fácil de utilizar
- Ofrece un entorno de programación (IDE) con el que crear aplicaciones
- Cuenta con una gran comunidad que apoya el Desarrollo, comparte conocimiento, elabora librerías para facilitar su uso, publica proyectos que sirven de apoyo

Arduino = HW + SW + Comunidad



Placa Arduino



Modelos de Arduino



Entorno de Desarrollo

○ <https://www.arduino.cc/en/Main/Software>

Download the Arduino IDE



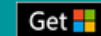
ARDUINO 1.8.9

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, for Windows XP and up
Windows ZIP file for non admin install

Windows app Requires Win 8.1 or 10



Mac OS X 10.8 Mountain Lion or newer

Linux 32 bits

Linux 64 bits

Linux ARM 32 bits

Linux ARM 64 bits

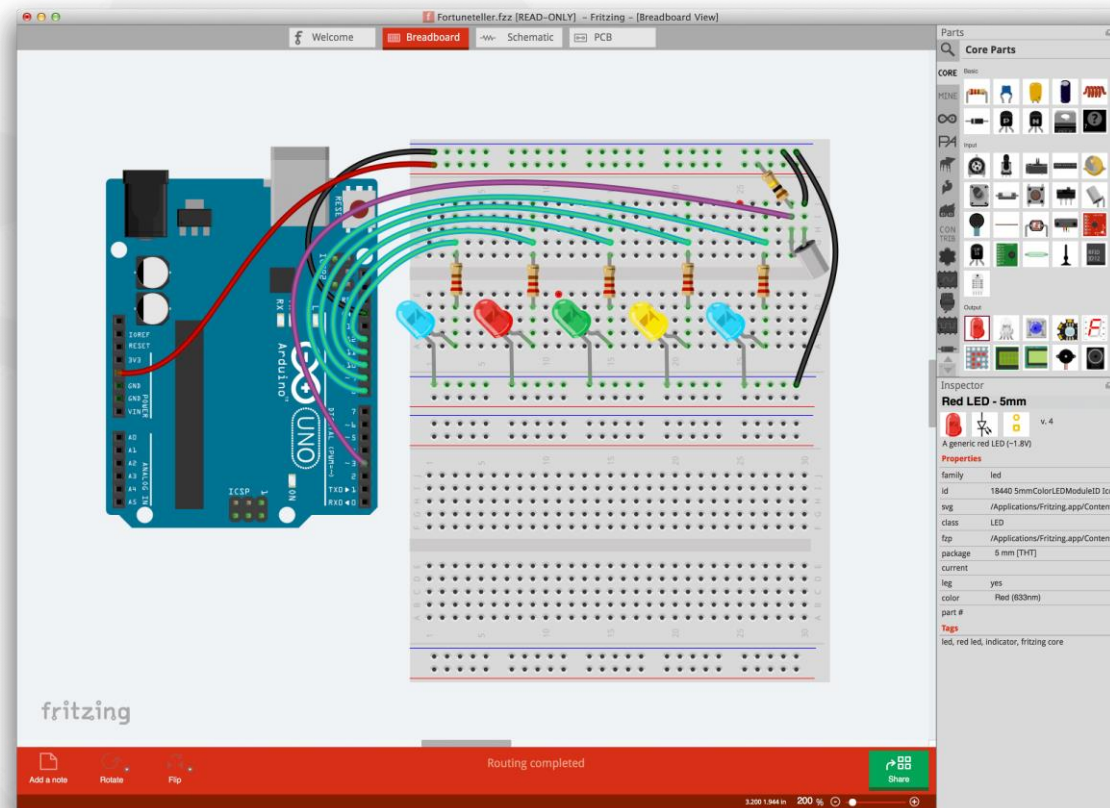
[Release Notes](#)

[Source Code](#)

[Checksums \(sha512\)](#)

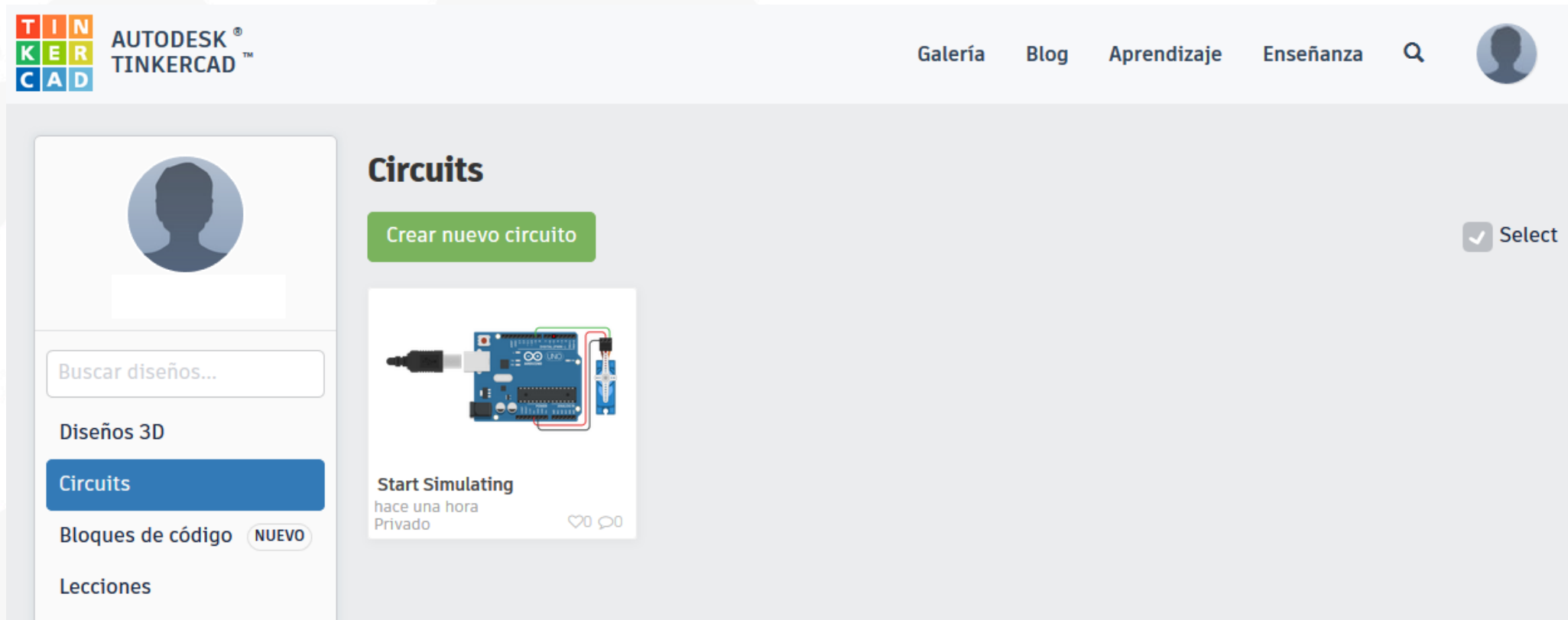
Documentación Proyectos

○ <https://fritzing.org/home/>



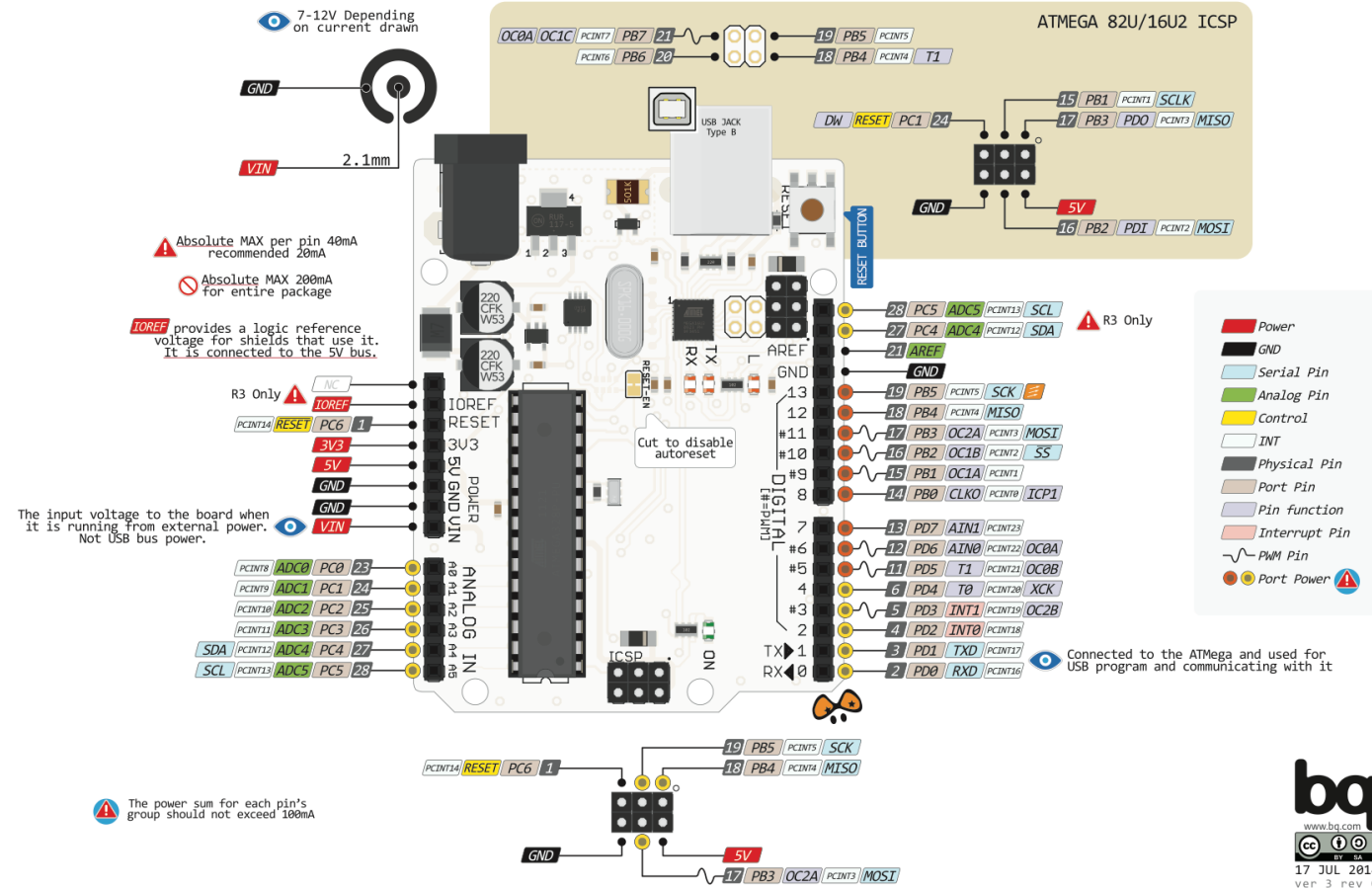
Simulador de Arduino

○ <https://www.tinkercad.com>

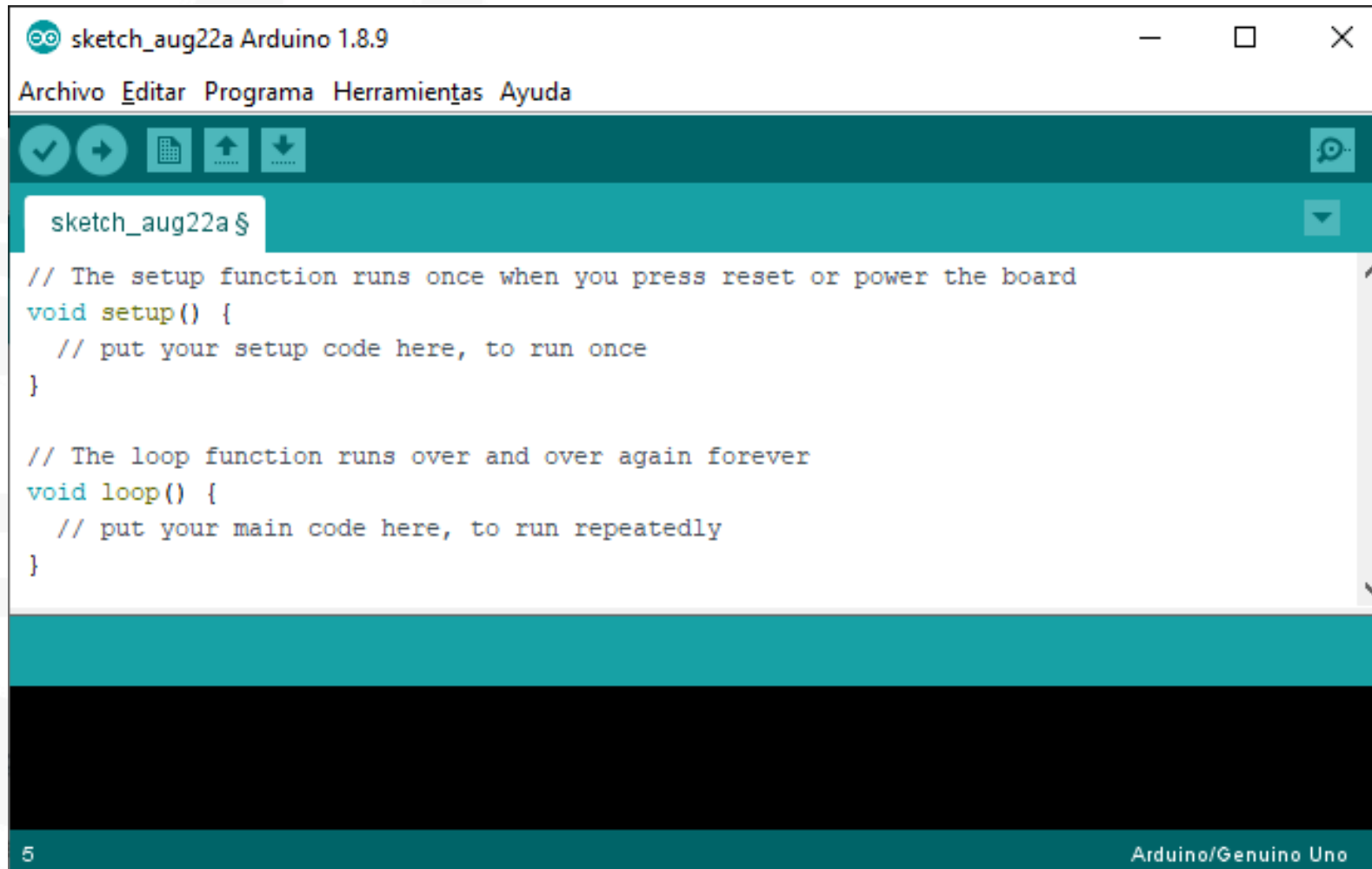


Esquema Arduino Pinout

UNO PINOUT



Sintaxis Arduino



The screenshot shows the Arduino IDE window titled "sketch_aug22a Arduino 1.8.9". The menu bar includes "Archivo", "Editar", "Programa", "Herramientas", and "Ayuda". The toolbar contains icons for opening, saving, and running. The code editor displays the following C++ code:

```
sketch_aug22a $  
  
// The setup function runs once when you press reset or power the board  
void setup() {  
    // put your setup code here, to run once  
}  
  
// The loop function runs over and over again forever  
void loop() {  
    // put your main code here, to run repeatedly  
}
```

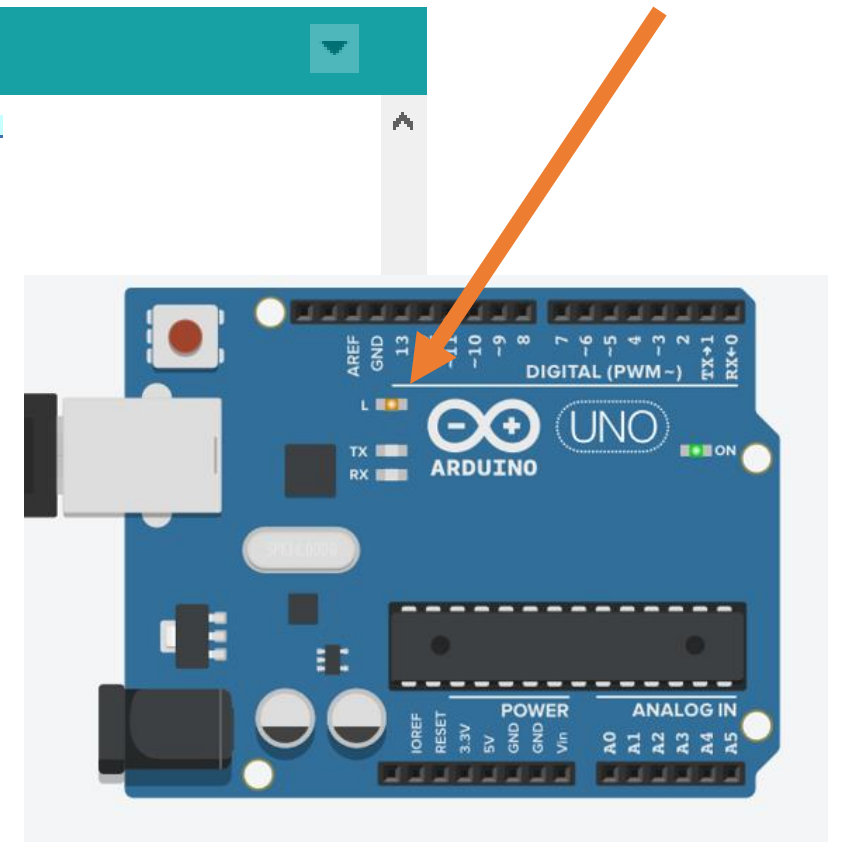
The status bar at the bottom indicates "5" on the left and "Arduino/Genuino Uno" on the right.

Primer programa Arduino

sketch_aug22a \$

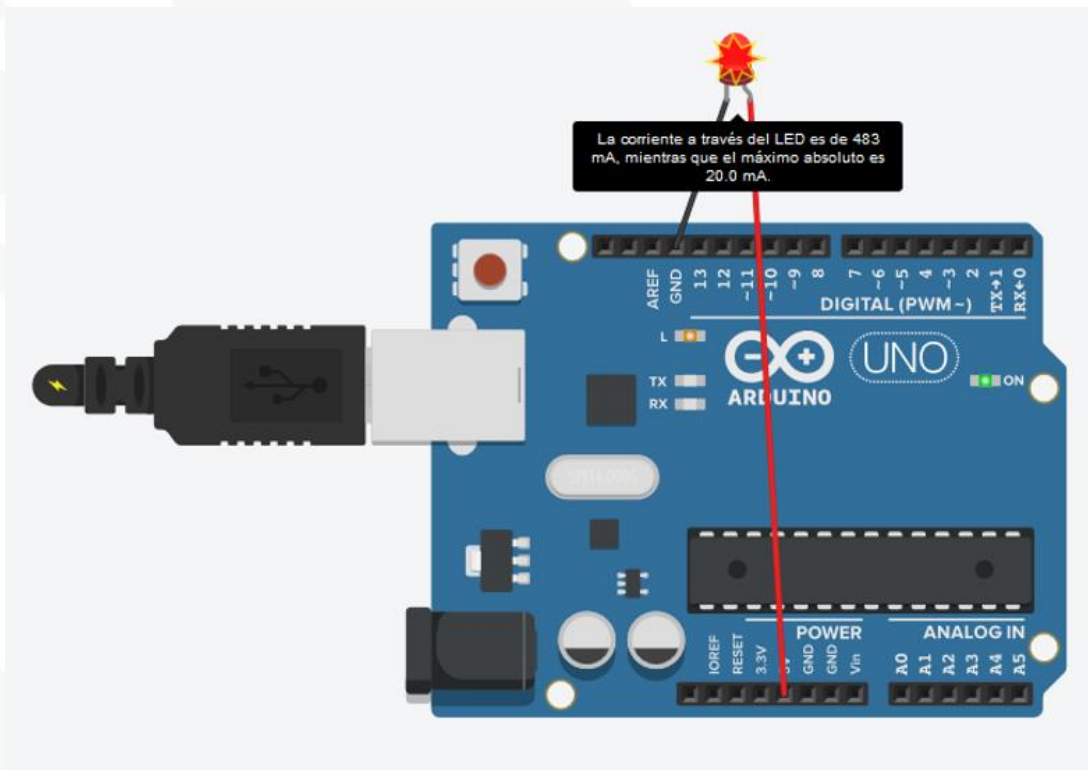
```
// The setup function runs once when you press reset or power the board
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

// The loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); //Turn the led on
  delay(1000);                     //Wait 1 second
  digitalWrite(LED_BUILTIN, LOW);  //Turn the led off
  delay(1000);                     //Wait 1 second
}
```

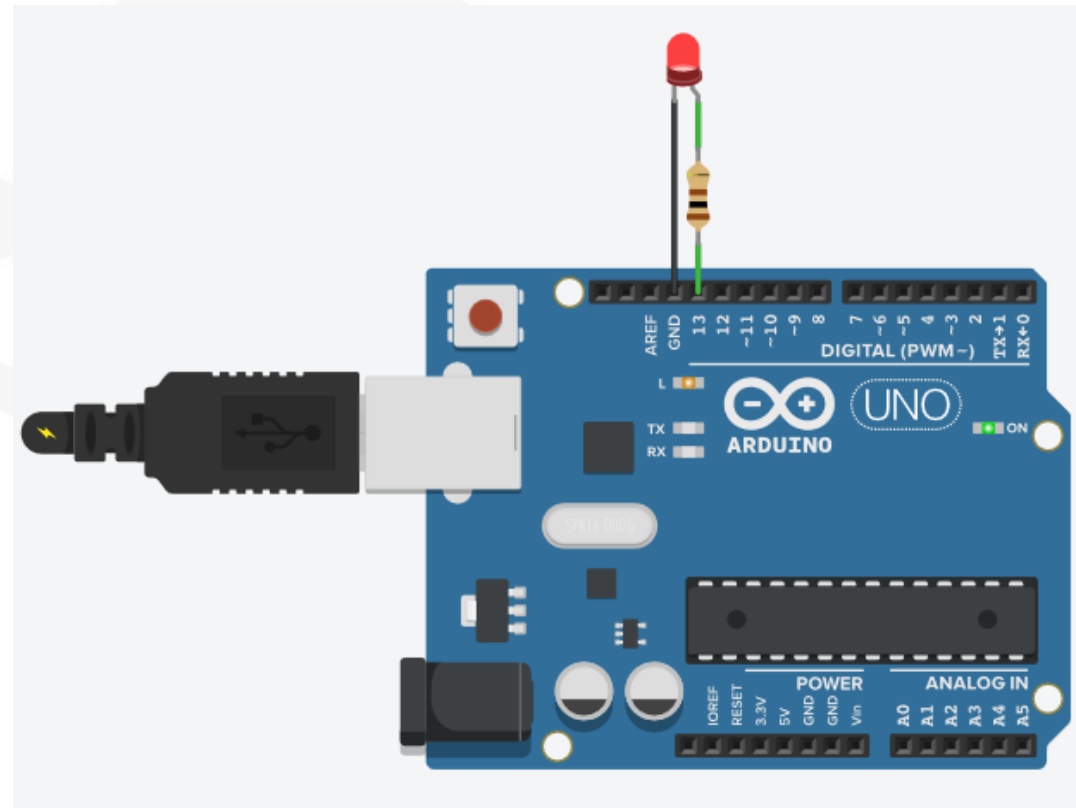


Problemas de voltaje

- Algunos componentes pueden estropearse en caso de usar un voltaje superior al recomendado.



Problemas de voltaje



Electrónica Analógica

- Resistencias

- Limita el flujo de electricidad en un circuito, reduciendo el voltaje y la corriente

- Condensadores

- Almacena y libera energía eléctrica en un circuito

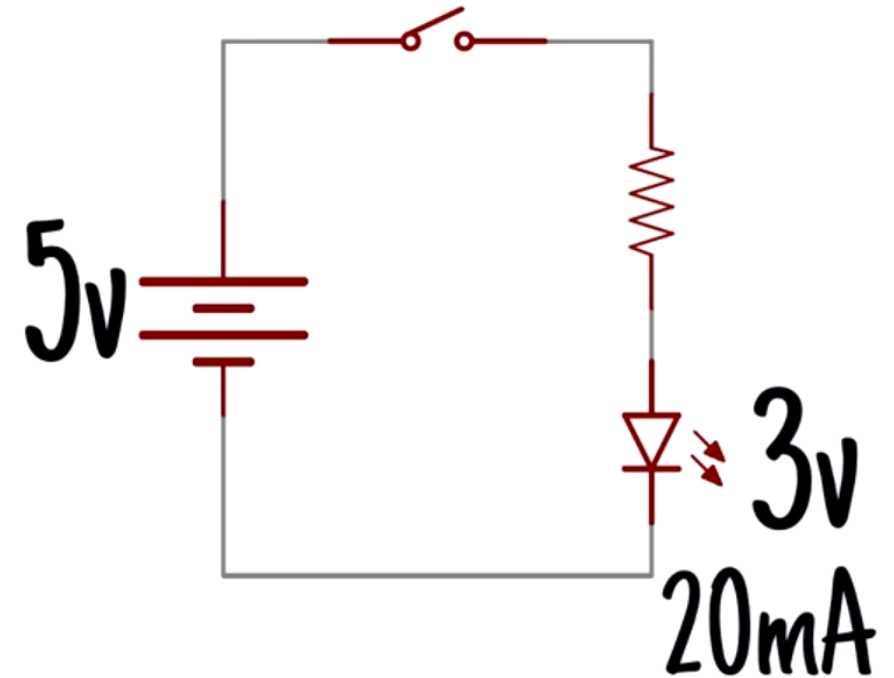
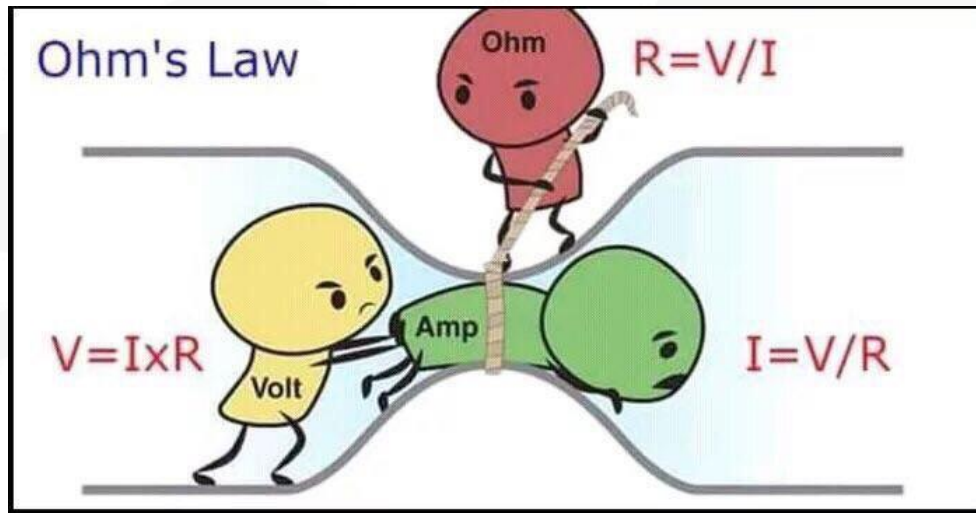
- Diodos

- Permite que la electricidad fluya en una única dirección

- Transistores

- Permite entregar una señal de salida en función de una señal de entrada

Ley de Ohm

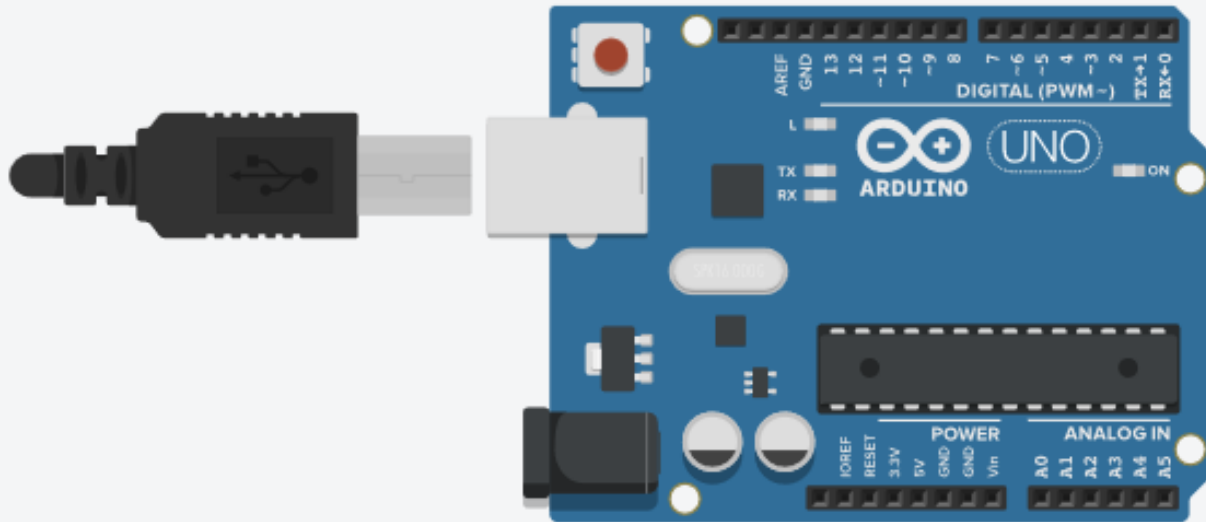


$$R = 100\Omega$$

Comunicación Serie

- Permite la comunicación entre la placa Arduino y el ordenador
- Funciones básicas:
 - `Serial.begin(speed)`
 - Establece la velocidad de bits por segundo para la transmisión en serie
 - `Serial.read()`
 - Lee los datos entrantes del puerto serie
 - `Serial.print(val,[format])`
 - Imprime los datos al puerto serie como texto ASCII
 - `Serial.println(val,[format])`
 - Imprime los datos al puerto serie como texto ASCII seguido de un salto de línea
 - `Serial.available()`
 - Devuelve el número de bytes disponibles para ser leídos por el puerto serie

Comunicación Serie

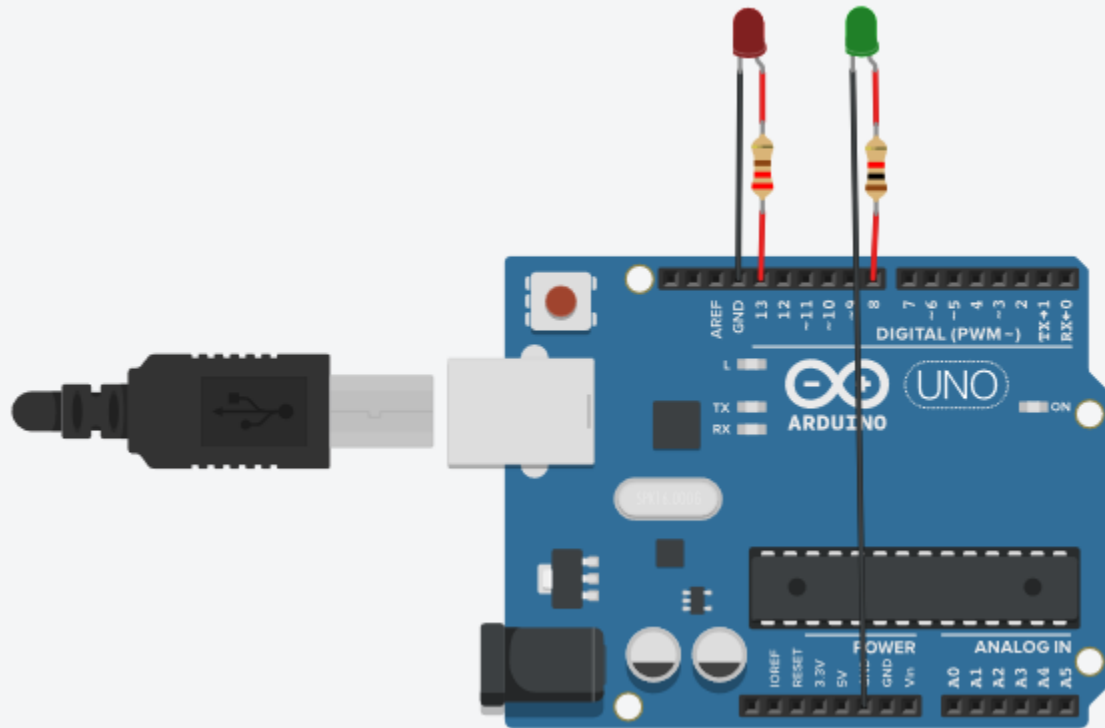


```
1 void setup()
2 {
3   Serial.begin(9600);
4   pinMode(13, OUTPUT);
5 }
6
7 void loop()
8 {
9   digitalWrite(13, HIGH);
10  Serial.println("Led ON");
11  delay(1000); // Wait for 1000 millisecond(s)
12  digitalWrite(13, LOW);
13  Serial.println("Led OFF");
14  delay(1000); // Wait for 1000 millisecond(s)
15 }
```

Monitor en serie

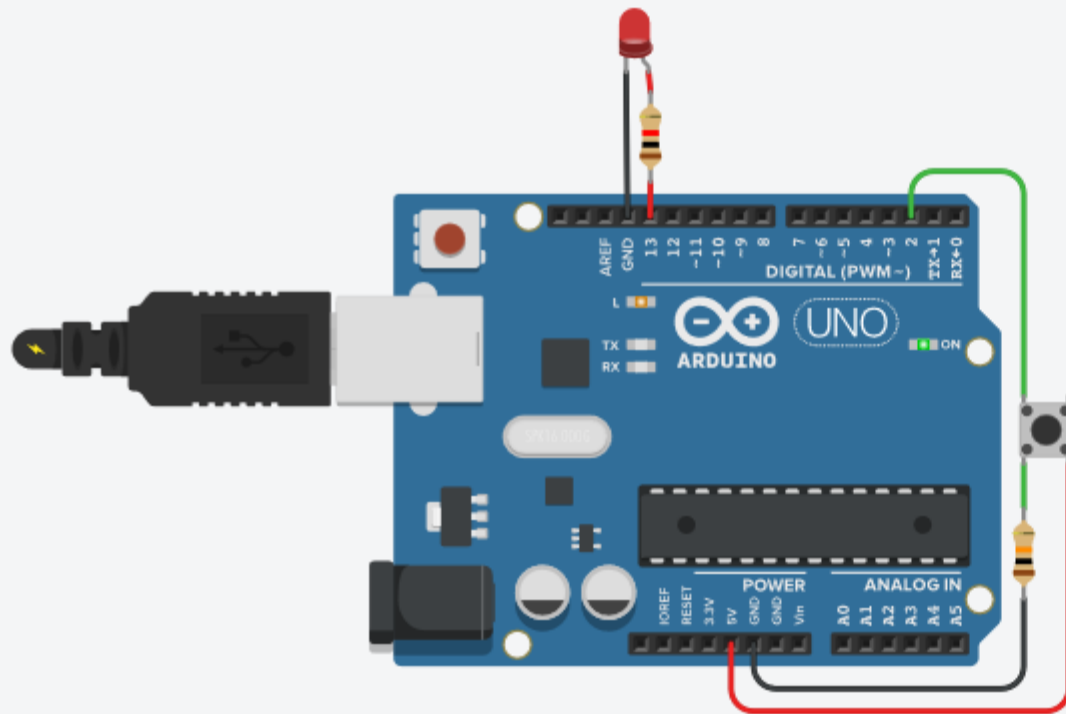
Led ON
Led OFF
Led ON
Led OFF
Led ON
Led OFF

Comunicación Serie



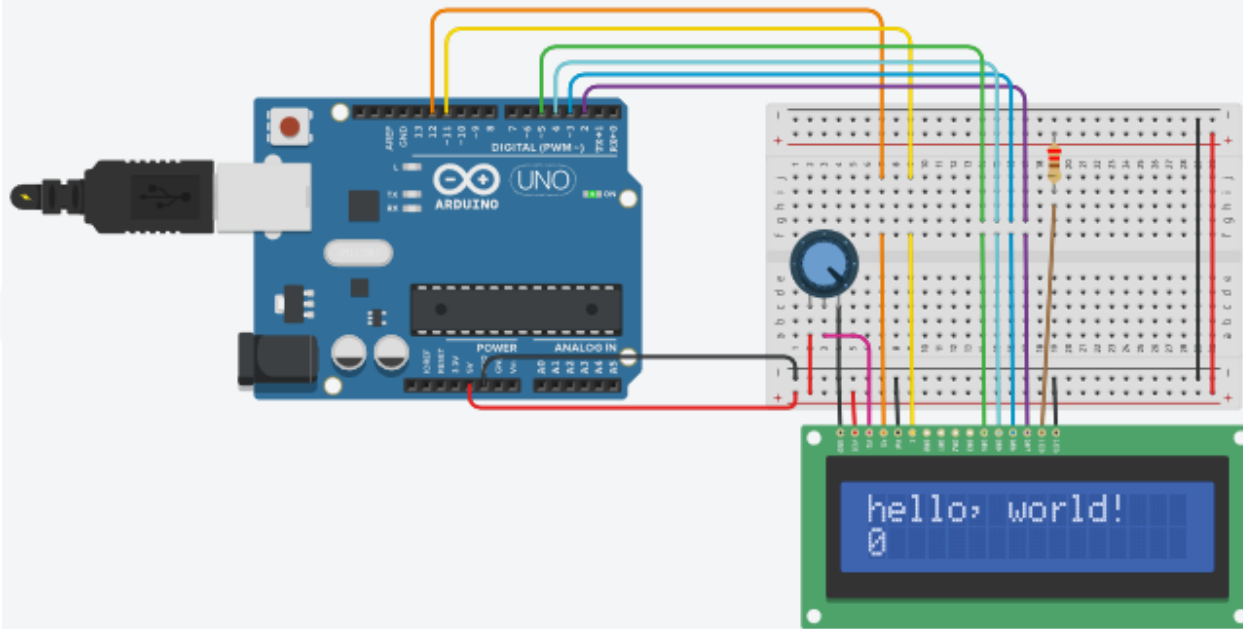
```
1
2 int pin=13;
3 String a;
4 void setup()
5 {
6     Serial.begin(9600);
7     pinMode(13, OUTPUT);
8     pinMode(8, OUTPUT);
9 }
10
11 void loop()
12 {
13     if (Serial.available() > 0) {
14         a= Serial.readString();// read the incoming data as string
15         Serial.println(a);
16         pin =8;
17     }else
18     {
19         pin=13;
20     }
21     digitalWrite(pin, HIGH);
22     delay(500); // Wait for 1000 millisecond(s)
23     digitalWrite(pin, LOW);
24     delay(500); // Wait for 1000 millisecond(s)
25 }
```

Interacción con botones



```
1  int buttonState = 0;
2
3  void setup()
4  {
5      pinMode(2, INPUT);
6      pinMode(13, OUTPUT);
7  }
8
9  void loop()
10 {
11     // read the state of the pushbutton value
12     buttonState = digitalRead(2);
13     // check if pushbutton is pressed
14     if (buttonState == HIGH) {
15         // turn LED on
16         digitalWrite(13, HIGH);
17     } else {
18         // turn LED off
19         digitalWrite(13, LOW);
20     }
21     delay(10);
22 }
```


Uso de librerías



```
1 // include the library code:
2 #include <LiquidCrystal.h>
3
4 // initialize the library with the numbers of the interface pins
5 LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
6
7 void setup() {
8     // set up the LCD's number of columns and rows:
9     lcd.begin(16, 2);
10    // Print a message to the LCD.
11    lcd.print("hello, world!");
12 }
13
14 void loop() {
15     // set the cursor to column 0, line 1
16     // (note: line 1 is the second row, since counting begins with 0):
17     lcd.setCursor(0, 1);
18     // print the number of seconds since reset:
19     lcd.print(millis() / 1000);
20 }
21
```