

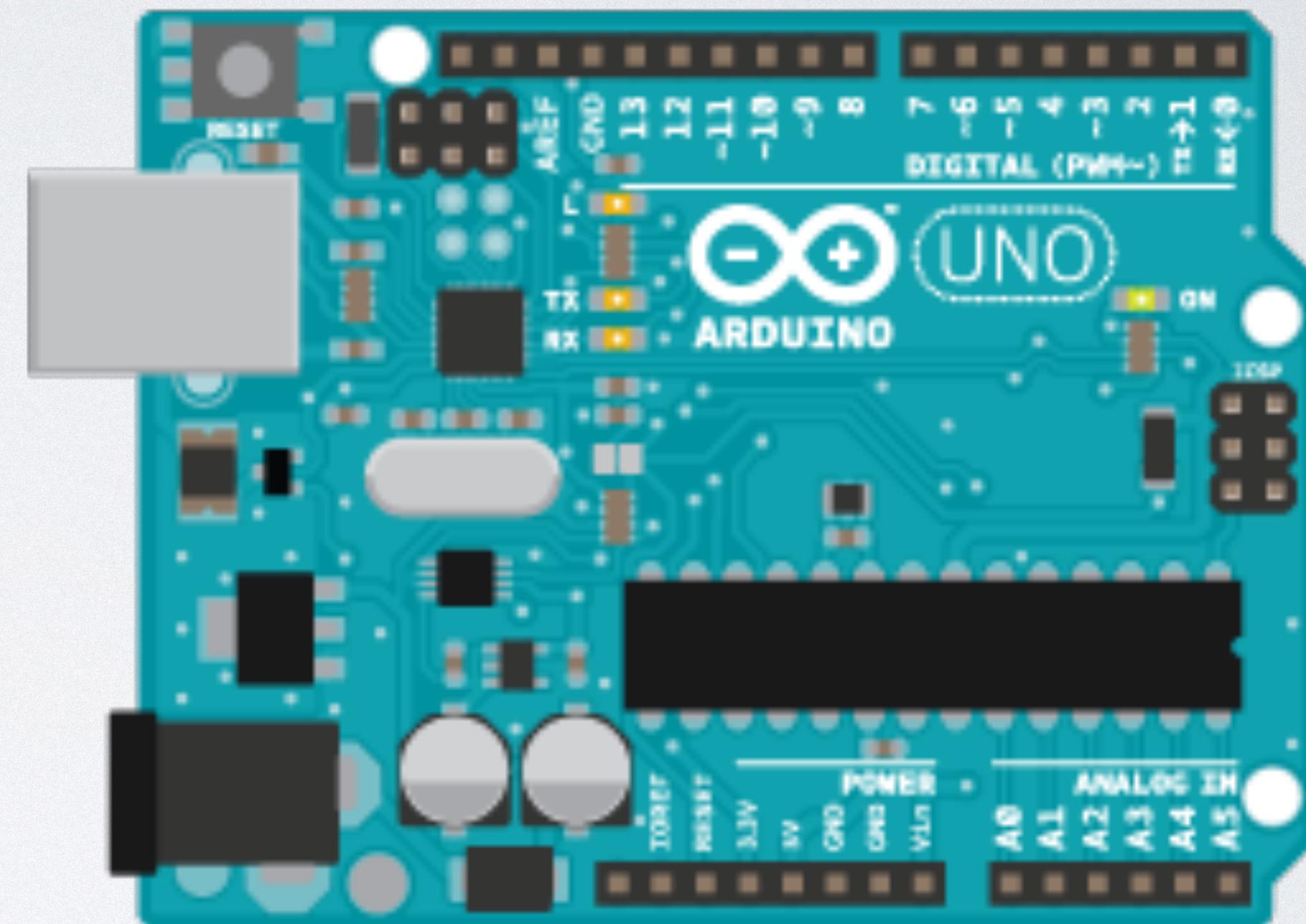
ARDUINO

Melchor Alejo Garau Madrigal

Estudiante de Ingeniería del Software en la Universidad de Málaga

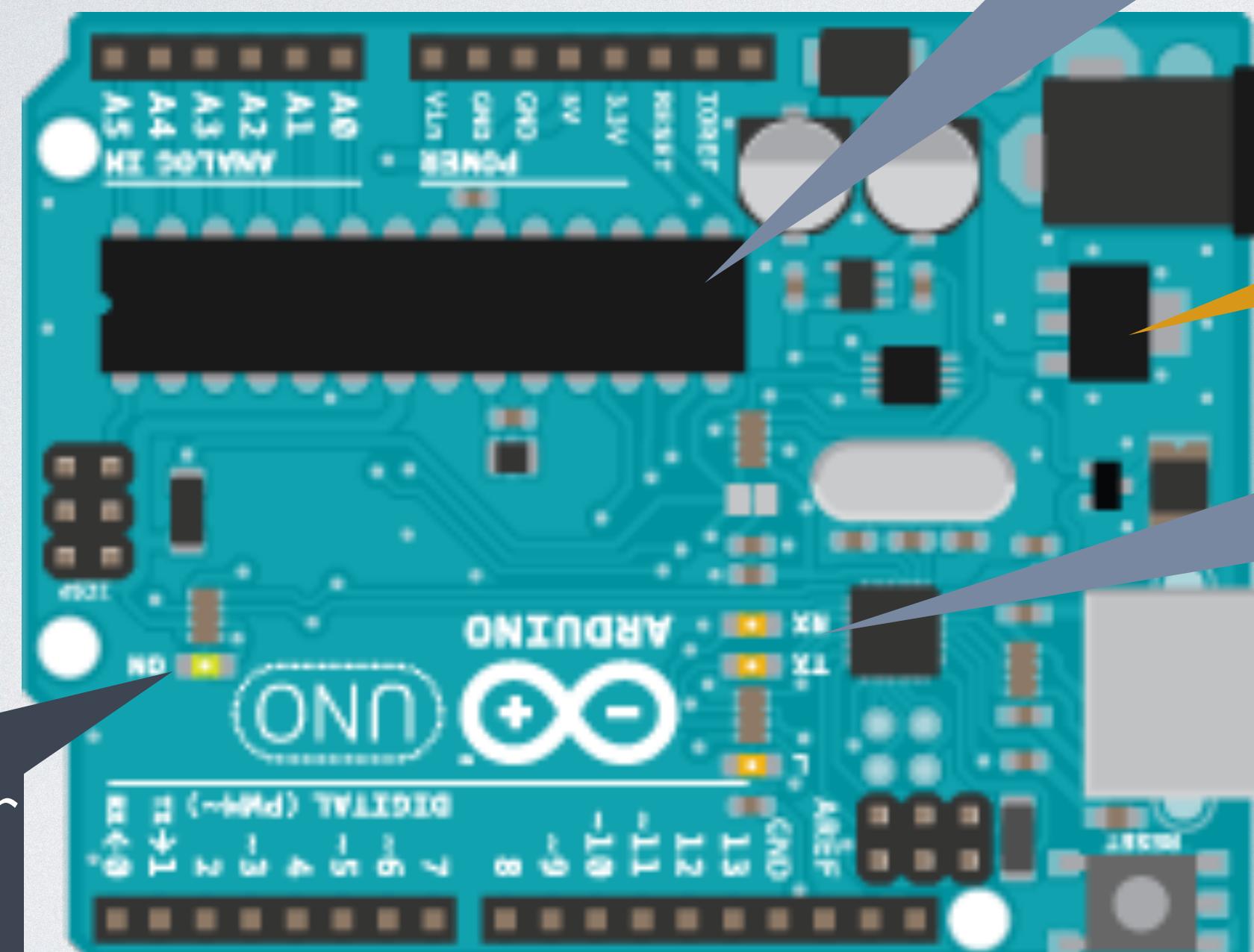
ARDUINO

Plataforma Open-Source que aproxima al usuario al mundo físico fácilmente usando un ordenador



GND, 5V & 3.3V

Analog Input



Circuito Integrado

Do Regulador de voltage

TX/RX LED

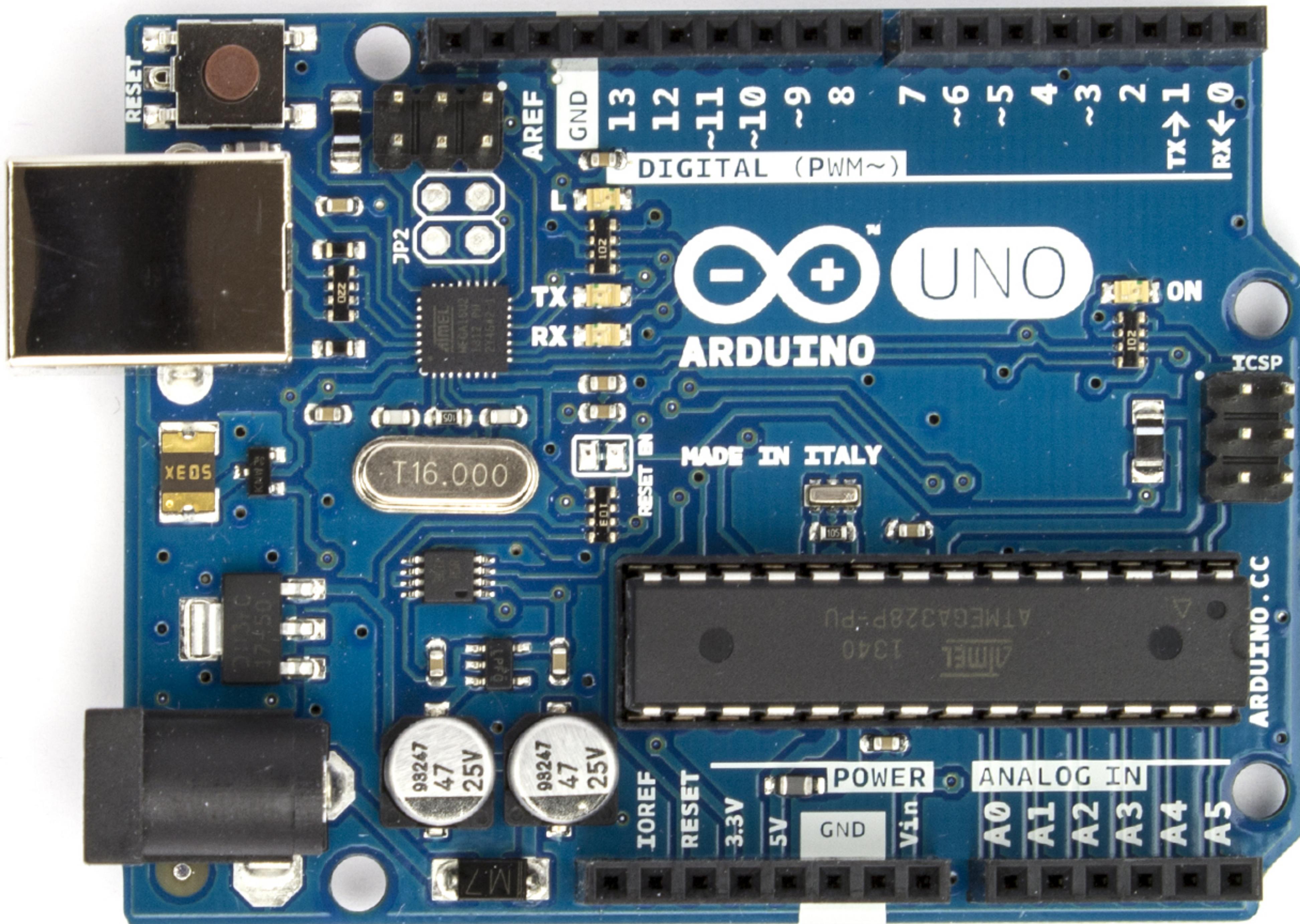
ARDUINO

Power
LED

Digital

Reset

USB





T16.000



MADE IN ITALY

RESET

ATMEGA328P-PU
ATMEG
1340

ARDUINO.cc

98247
47
25V

98247
47
25V

IOREF

RESET

3.3V

POWER

5V

GND

Vin

ANALOG IN

A0

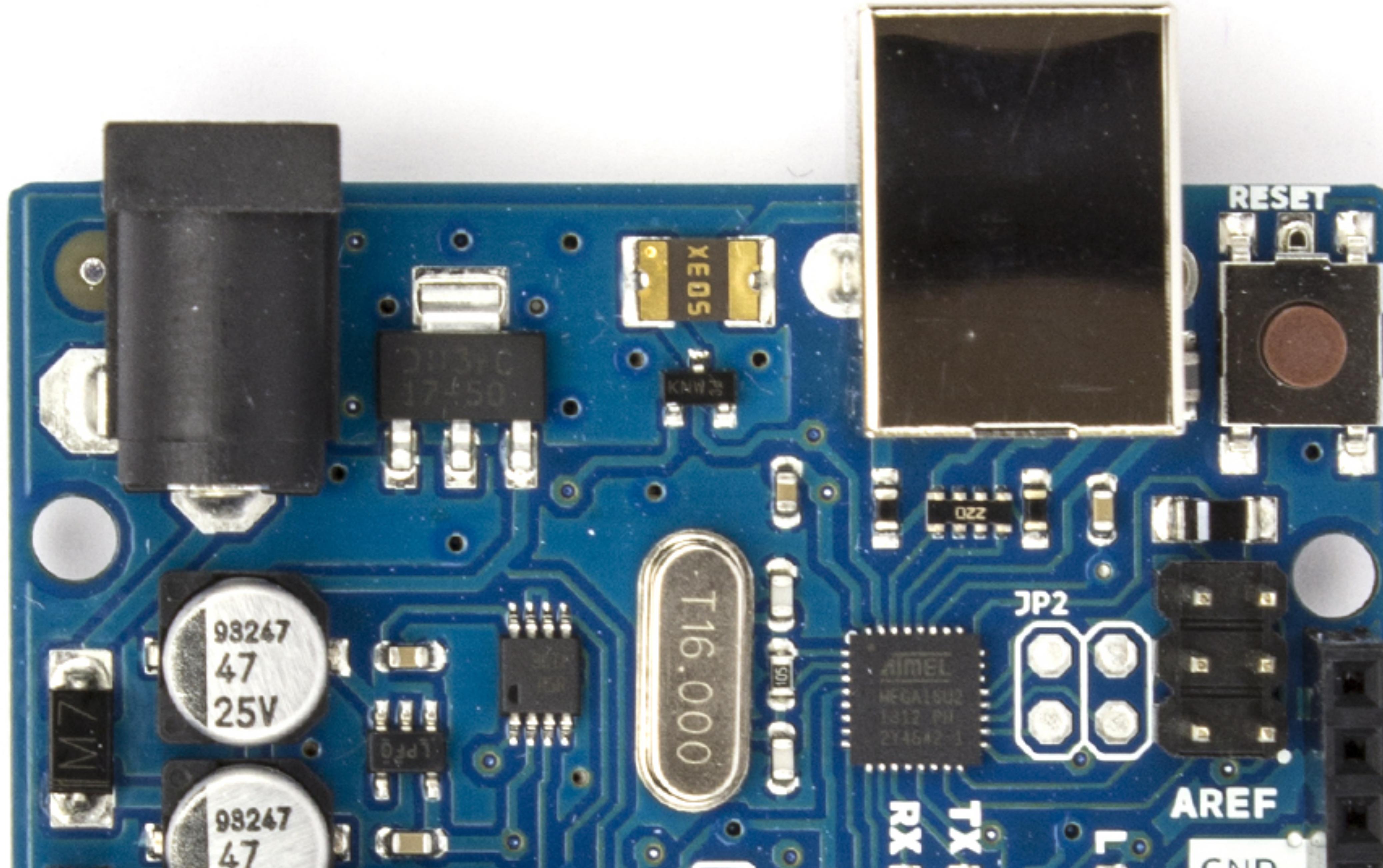
A1

A2

A3

A4

A5



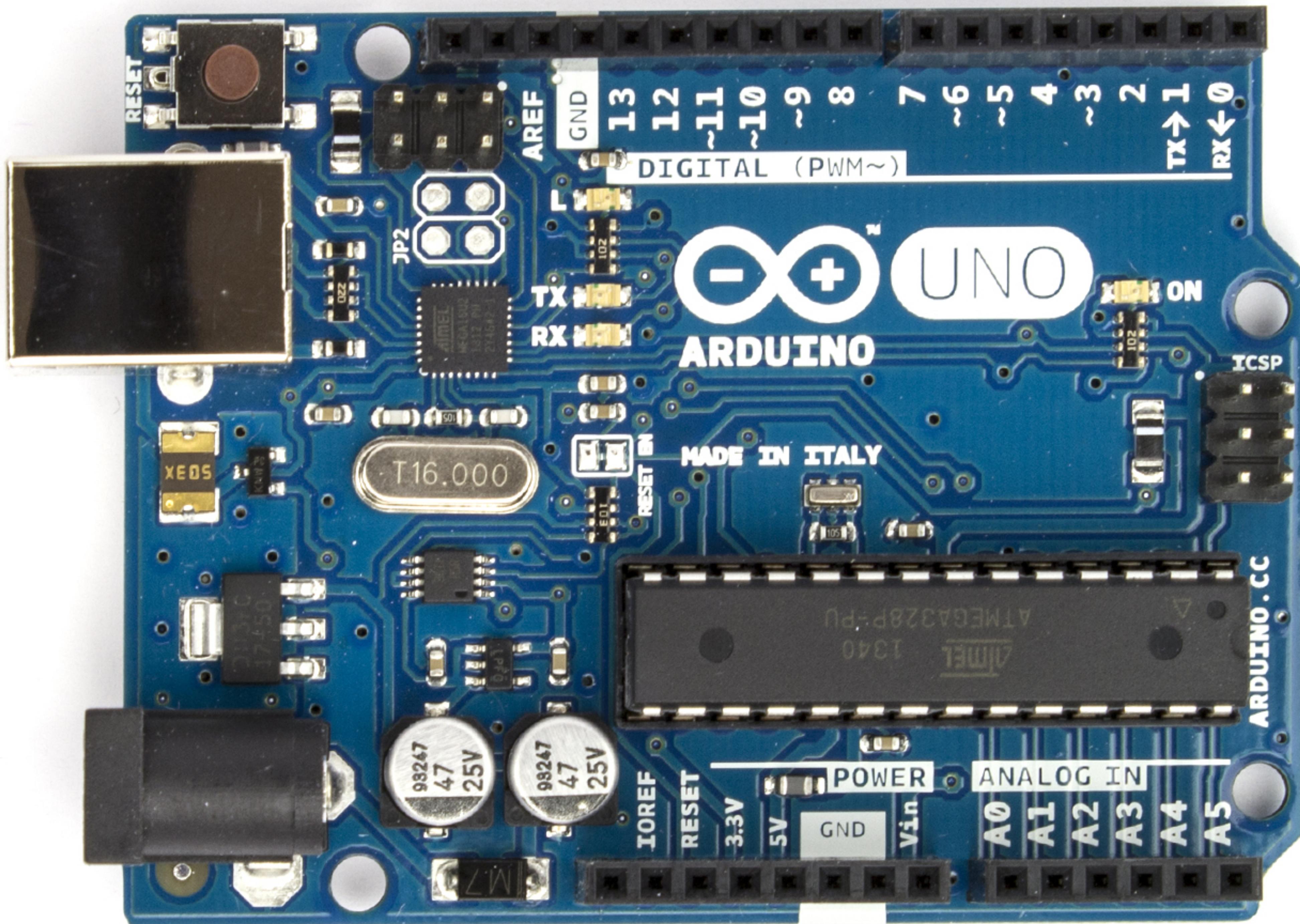
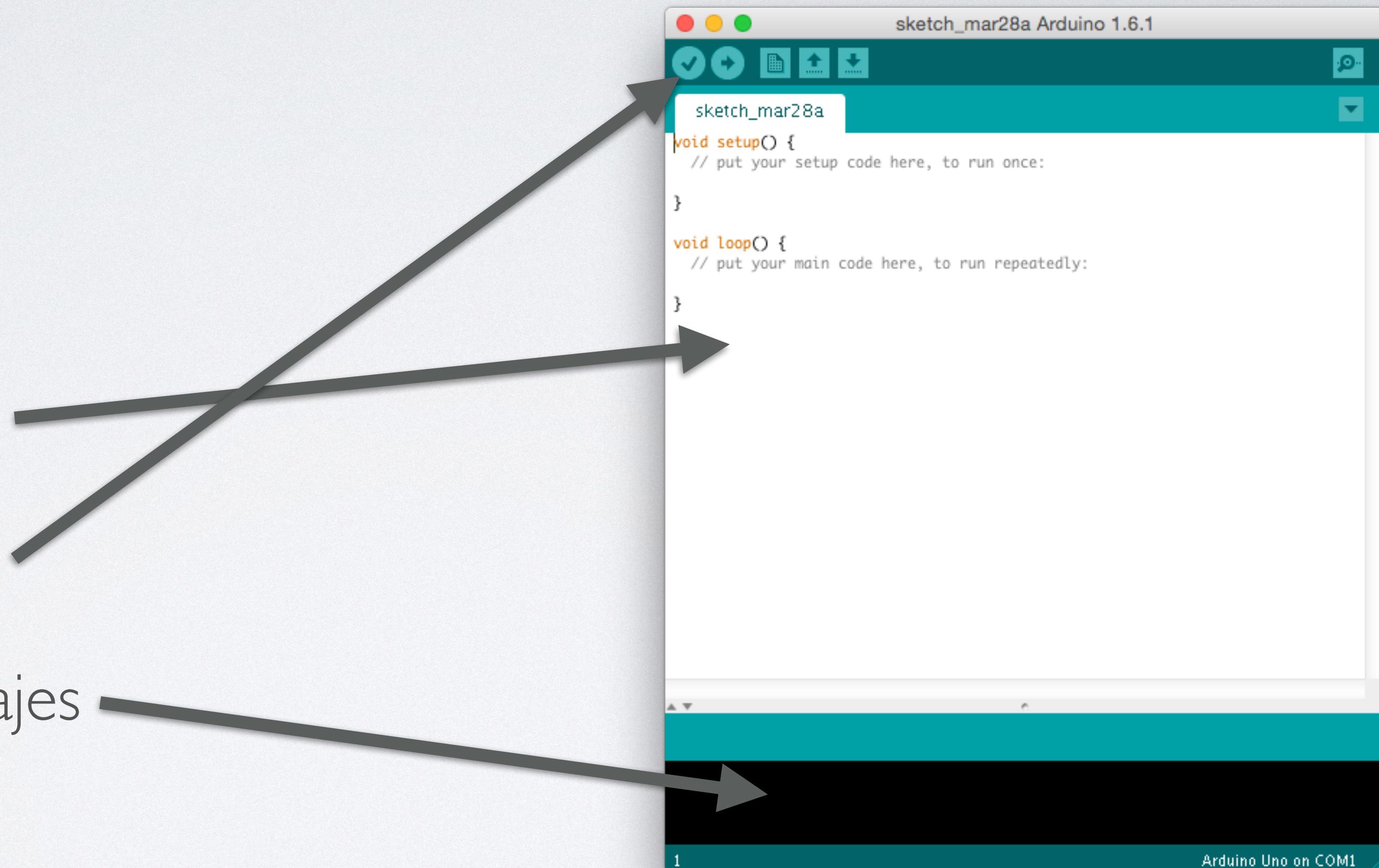




Foto de Aleks

IDE DE ARDUINO

- El IDE tiene:
 - Un editor de archivos
 - Un panel de acciones
 - Una consola de mensajes



LENGUAJE DE ARDUINO

- Arduino es un lenguaje de programación basado en C/C++
- Mantiene muchas de las estructuras de C

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);      // turn the LED on (HIGH is the voltage level)
  delay(1000);                // wait for a second
  digitalWrite(13, LOW);       // turn the LED off by making the voltage LOW
  delay(1000);                // wait for a second
}
}
delay(1000);                  // wait for a second
digitalWrite(13, HIGH);        // turn the LED on by making the voltage HIGH
```

SETUP()

Bloque de código que se ejecuta una sola vez

```
void setup() {  
    pinMode(13, OUTPUT);  
}
```

LOOP()

Bloque de código que se ejecuta todo el tiempo

```
void loop() {  
    digitalWrite(13, HIGH);  
    delay(1000);  
    digitalWrite(13, LOW);  
    delay(1000);  
}
```

PINMODE()

Función que configura el modo de funcionamiento de un pin.

PIN: el número del PIN digital

MODE: El tipo de conexión *INPUT* u *OUTPUT*

```
void setup() {  
    pinMode(13, OUTPUT);  
    pinMode(12, INPUT);  
    pinMode(PIN, MODE);  
}
```

DIGITALREAD() DIGITALWRITE()

Función que lee/escribe en un pin

PIN: El pin en donde actuar

VAL:Valor a leer/escribir (HIGH o LOW)

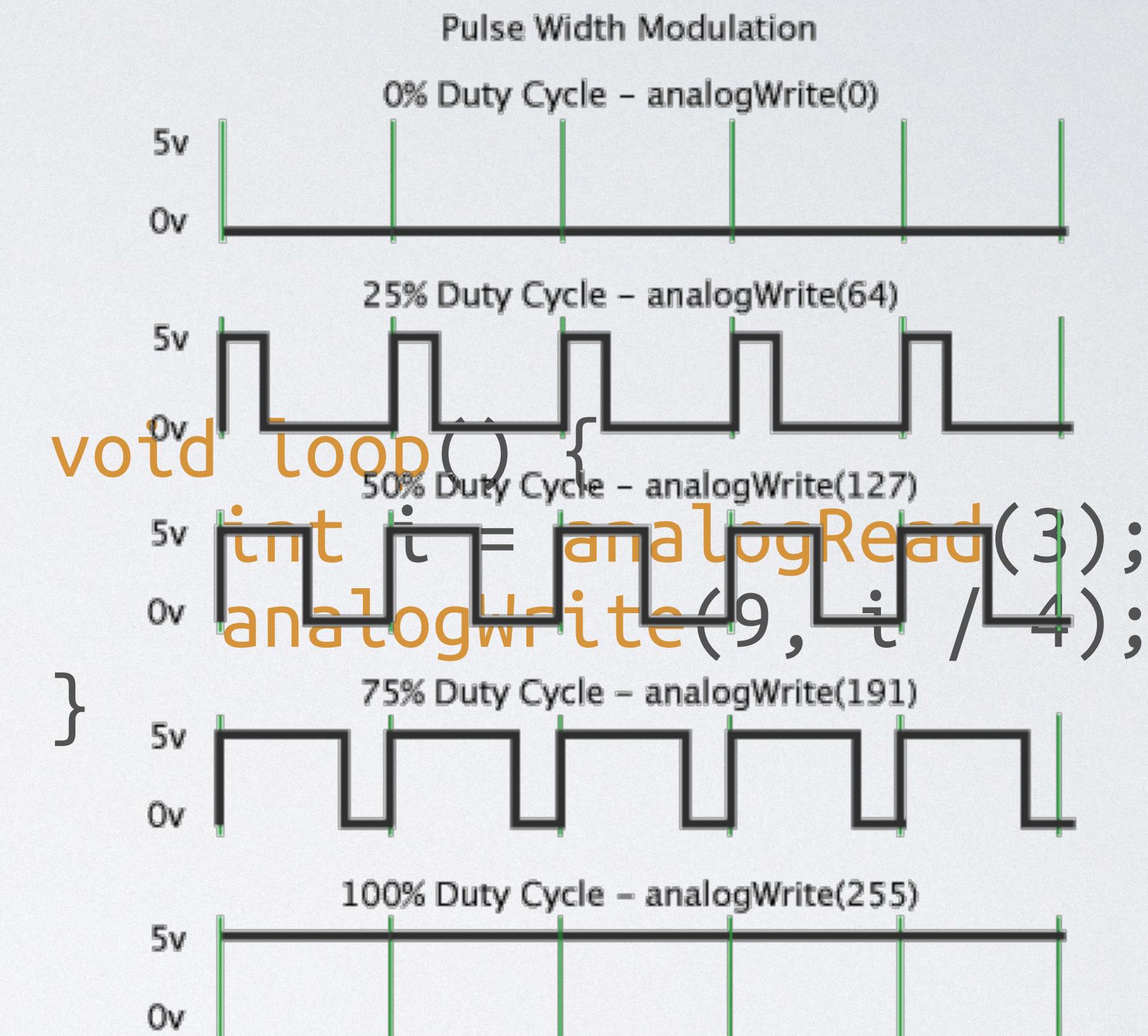
```
void loop() {  
    int i = digitalRead(13);  
    digitalWrite(12, i);  
}
```

ANALOGREAD() ANALOGWRITE()

Función que lee/escribe en un pin

PIN: El pin en donde actuar
(0-5 [read])

VAL: Valor a leer/escribir
(0 a 1023 [read] o a 255 [write])



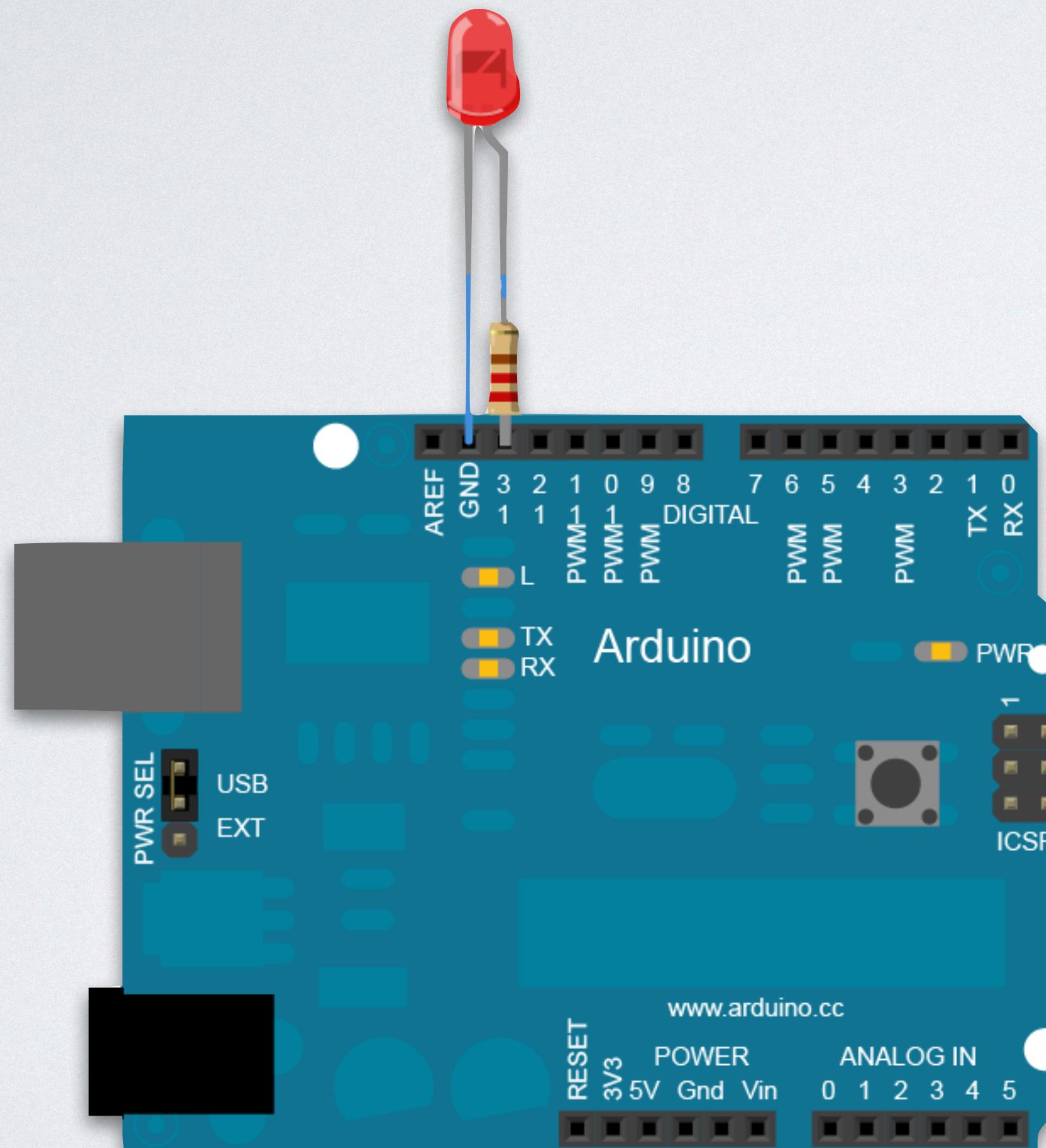
DELAY()

Función que hace parar el proceso de ejecución tantos milisegundos

TIME: tiempo a esperar en milisegundos

```
void loop() {  
    delay(1000);  
}
```

UN EJEMPLO: LED PARPADEANTE



```
sketch_mar28b Arduino 1.6.1
sketch_mar28b S
const int T_ESPERA = 1000;

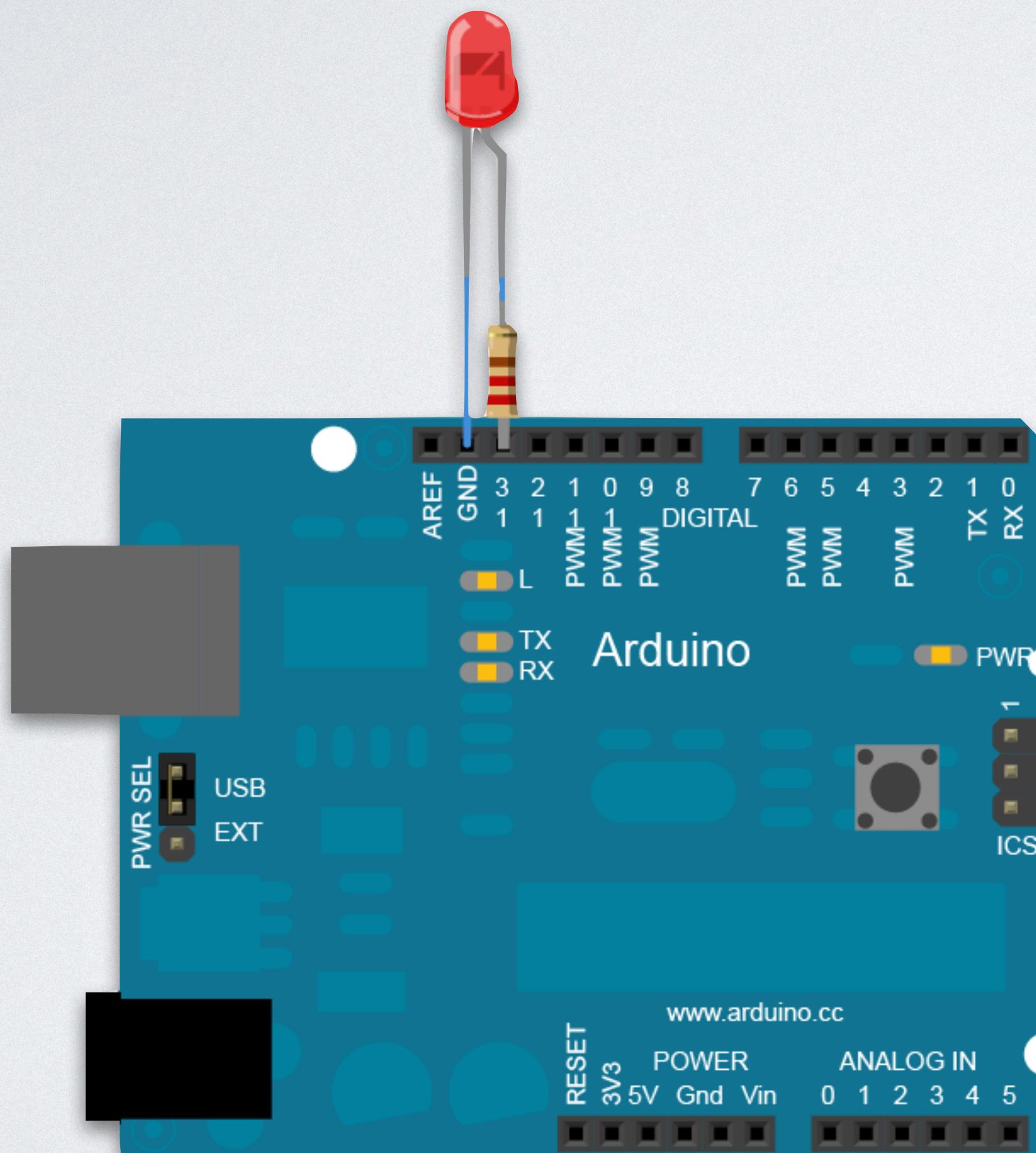
void setup() {
  pinMode(13, OUTPUT); //Preparamos para usar el pin 13
}

void loop() {
  digitalWrite(13, HIGH); //Encendemos el pin 13
  delay(T_ESPERA); //Esperamos T_ESPERA milisegundos
  digitalWrite(13, LOW); //Apagamos el pin 13
  delay(T_ESPERA); //Esperamos T_ESPERA milisegundos
}

Compilado
Sketch uses 1.030 bytes (3%) of program storage space. Maximum is 32.256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2.039 bytes for
local variables. Maximum is 2.048 bytes.

10
Arduino Uno on COM1
```

UN EJEMPLO: LED PARPADEANTE



sketch_mar28b Arduino 1.6.1

sketch_mar28b §

```
const int T_ESPERA = 1000;

void setup() {
  pinMode(13, OUTPUT); //Preparamos para usar el pin 13
}

void loop() {
  digitalWrite(13, HIGH); //Encendemos el pin 13
  delay(T_ESPERA); //Esperamos T_ESPERA milisegundos
  digitalWrite(13, LOW); //Apagamos el pin 13
  delay(T_ESPERA); //Esperamos T_ESPERA milisegundos
}
```

IF

Pregunta si una expresión se cumple y ejecuta un bloque de código si es así

```
void loop() {  
    if(input == HIGH) {  
        //Código  
    }  
}
```

IF/ELSE

Pregunta si una expresión se cumple y ejecuta un bloque de código si es así y si no ejecutará el otro

```
void loop() {  
    if(input == HIGH) {  
        //Código  
    } else if(expr) {  
        //Código  
    } else {  
        //Código  
    }  
}
```

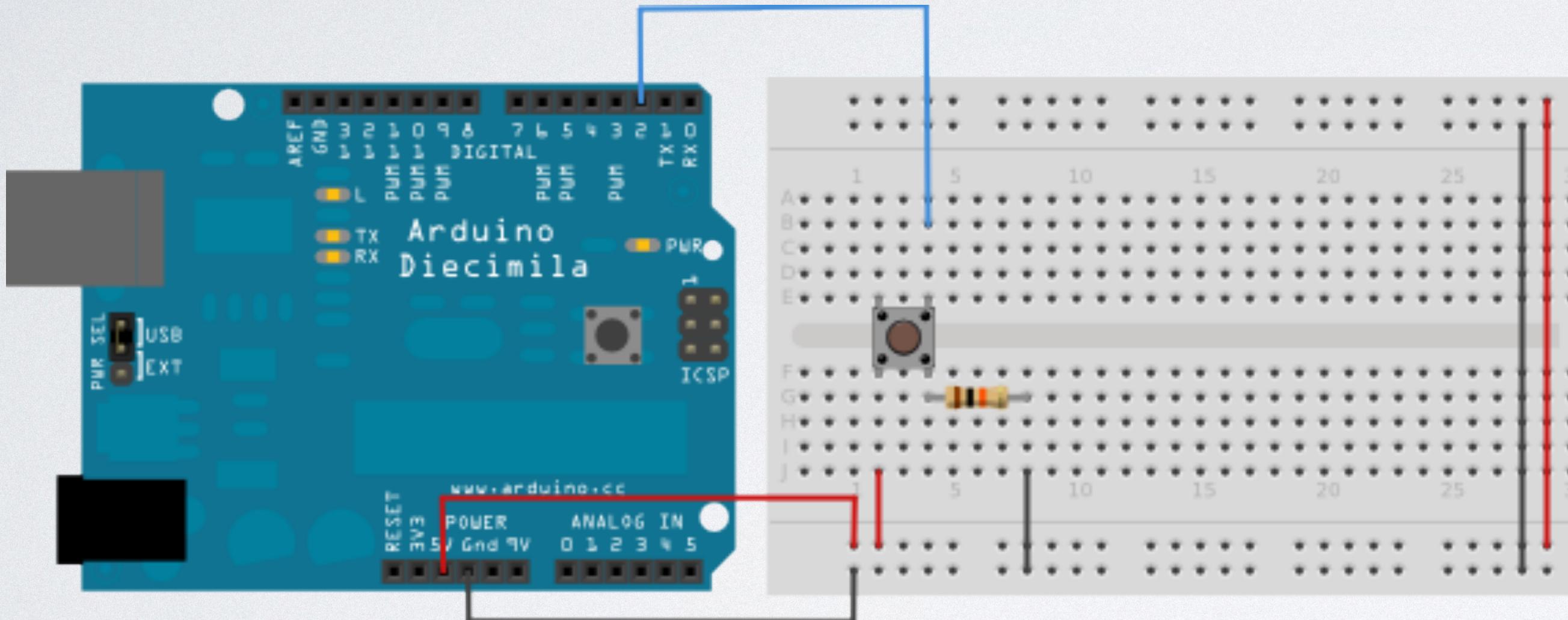
EXPRESIÓN BOLEADA

Expresión que contiene operadores que se resumen en Verdadero o Falso

$a == b$ ¿ a es igual a b ?
 $a != b$ ¿ a es distinto de b ?
 $a < b$ ¿ a es menor que b ?
 $a <= b$ ¿ a es menor o igual que b ?
 $a > b$ ¿ a es mayor que b ?
 $a >= b$ ¿ a es mayor o igual que b ?
 $!a$ no a

$expr1 \&& expr2$ a y b
 $expr1 \|\| expr2$ a o b

OTRO EJEMPLO: BOTÓN Y LED



sketch_mar28c Arduino 1.6.1

```
const int botonPin = 2, ledPin = 13;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(botonPin, INPUT);
}

void loop() {
  int estado = digitalRead(botonPin);

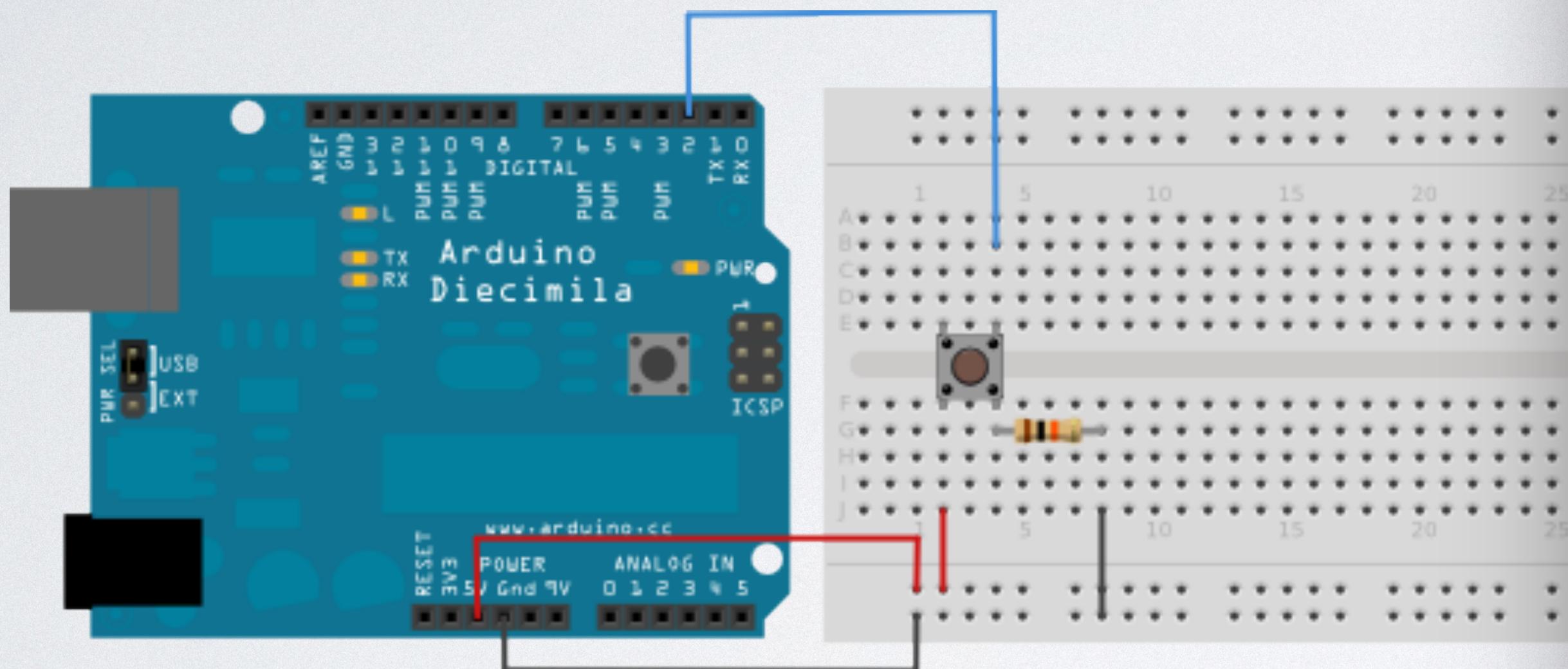
  //Si el estado es 'alto'...
  if(estado == HIGH) {
    //...encender led...
    digitalWrite(ledPin, HIGH);
  } //...si no...
  else {
    //...apagar led
    digitalWrite(ledPin, LOW);
  }
}
```

Compilado

Sketch uses 960 bytes (2%) of program storage space. Maximum is 32.256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2.039 bytes for local variables. Maximum is 2.048 bytes.

5 Arduino Uno on COM1

OTRO EJEMPLO: BOTÓN Y LED



sketch_mar28c Arduino 1.6.1

```
sketch_mar28c §

const int botonPin = 2, ledPin = 13;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(botonPin, INPUT);
}

void loop() {
  int estado = digitalRead(botonPin);

  //Si el estado es 'alto'...
  if(estado == HIGH) {
    //...encender led...
    digitalWrite(ledPin, HIGH);
  } //...si no...
  else {
    //...apagar led
    digitalWrite(ledPin, LOW);
  }
}
```

BOTÓN Y LED, QUE RECUERDA



The screenshot shows the Arduino IDE interface with a sketch titled "sketch_mar29a" running on version 1.6.1. The code implements a memory function using a button and an LED. It defines pins 2 and 13 for the button and LED respectively, and initializes variables for current and previous button states. The setup() function configures the pins and initializes the state variables. The loop() function reads the button state, toggles the LED if the state has changed from the previous reading, and updates the previous reading.

```
const int botonPin = 2, ledPin = 13;
bool ledOn, antiguo;

void setup() {
  pinMode(13, OUTPUT);
  pinMode(2, INPUT);

  ledOn = false;
  antiguo = false;
}

void loop() {
  bool actual = digitalRead(botonPin) == HIGH;

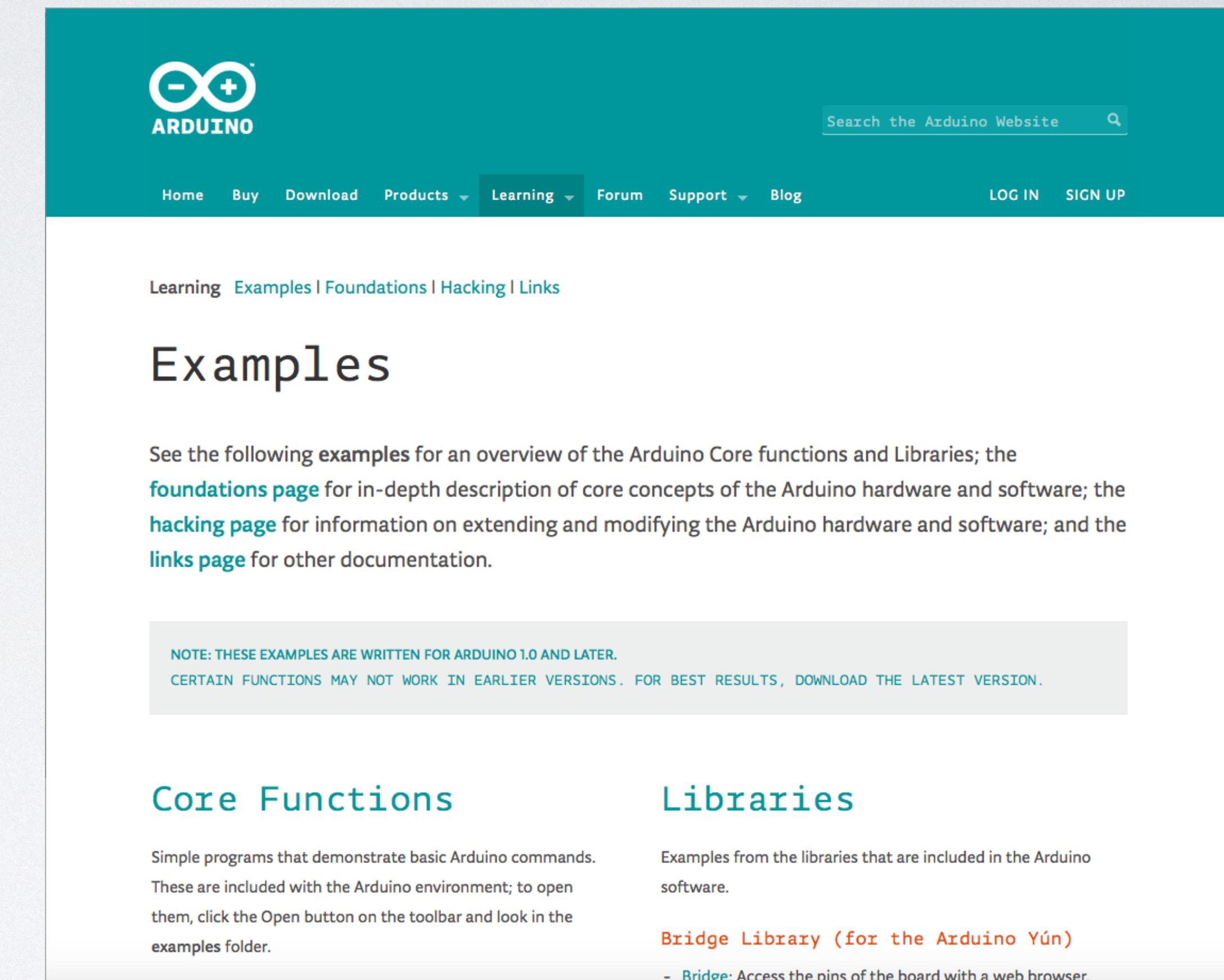
  if(actual && !antiguo) {
    if(ledOn) {
      digitalWrite(botonPin, LOW);
      ledOn = false;
    } else {
      digitalWrite(botonPin, HIGH);
      ledOn = true;
    }
  }

  antiguo = actual;
}
```

MÁS SOBRE ARDUINO: IDE Y LENGUAJE

<http://arduino.cc/en/Tutorial/HomePage>

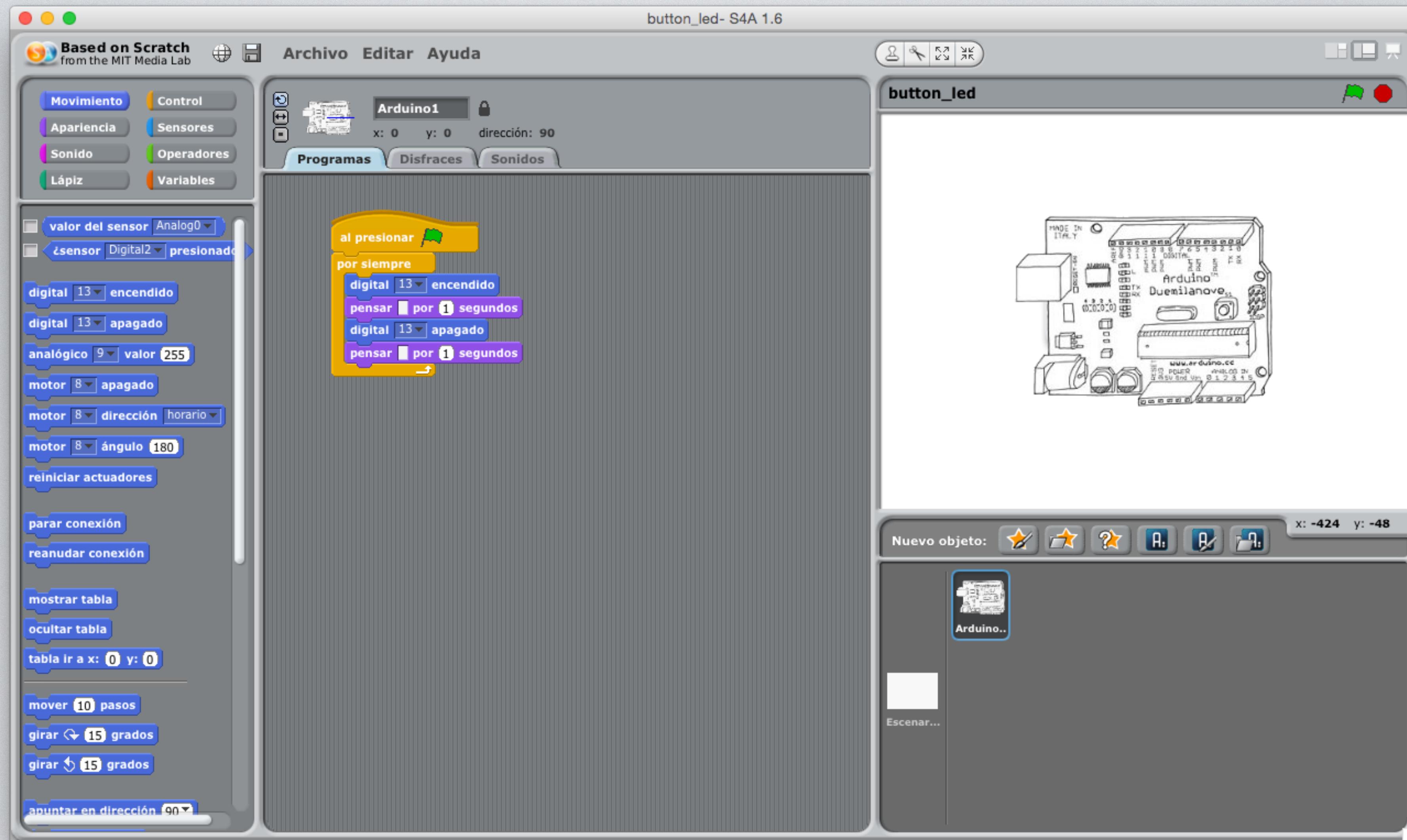
<http://arduino.cc/en/Reference/HomePage>



The screenshot shows the Arduino website's "Learning" section. The top navigation bar includes links for Home, Buy, Download, Products, Learning (which is currently selected), Forum, Support, and Blog. There is also a search bar and login/signup options. Below the navigation, there are links for Learning, Examples, Foundations, Hacking, and Links. The main content area is titled "Examples". It contains a note about the examples being for Arduino 1.0 and later, mentioning that certain functions may not work in earlier versions. Below this, there are two sections: "Core Functions" and "Libraries".

Core Functions
Simple programs that demonstrate basic Arduino commands. These are included with the Arduino environment; to open them, click the Open button on the toolbar and look in the examples folder.

Libraries
Examples from the libraries that are included in the Arduino software.
Bridge Library (for the Arduino Yún)
- [Bridge](#): Access the pins of the board with a web browser.



SCRATCH FOR ARDUINO

SCRATCH

- Scratch es un lenguaje de programación visual
- Orientada a objetos
- Multiplataforma y open-source

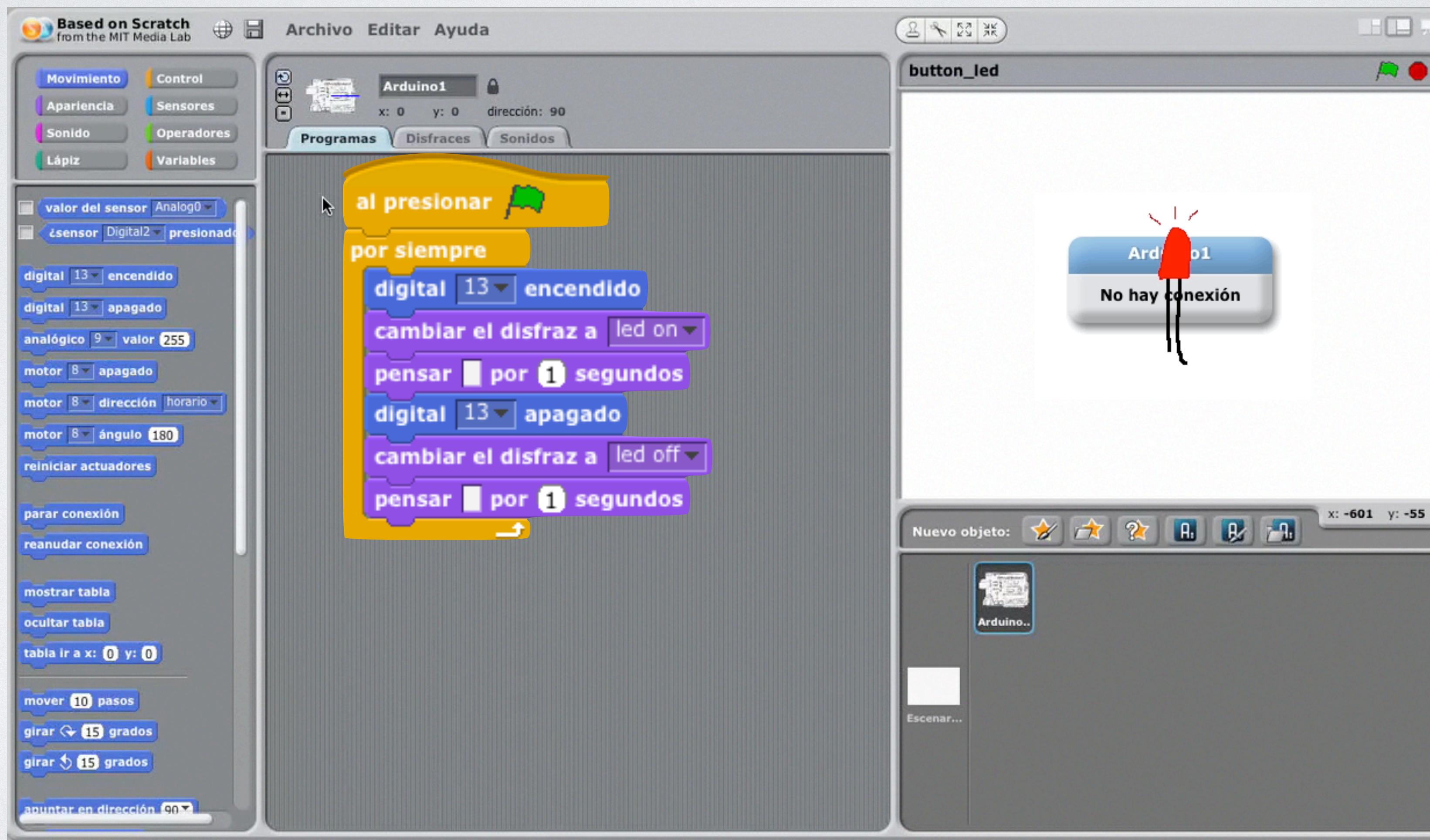
SCRATCH FOR ARDUINO

- Es una modificación de Scratch para Arduino
- Requiere una modificación en la placa (*firmware*)
- Fácil y rápido de programar
- Tiene ciertas limitaciones

SCRATCH FOR ARDUINO

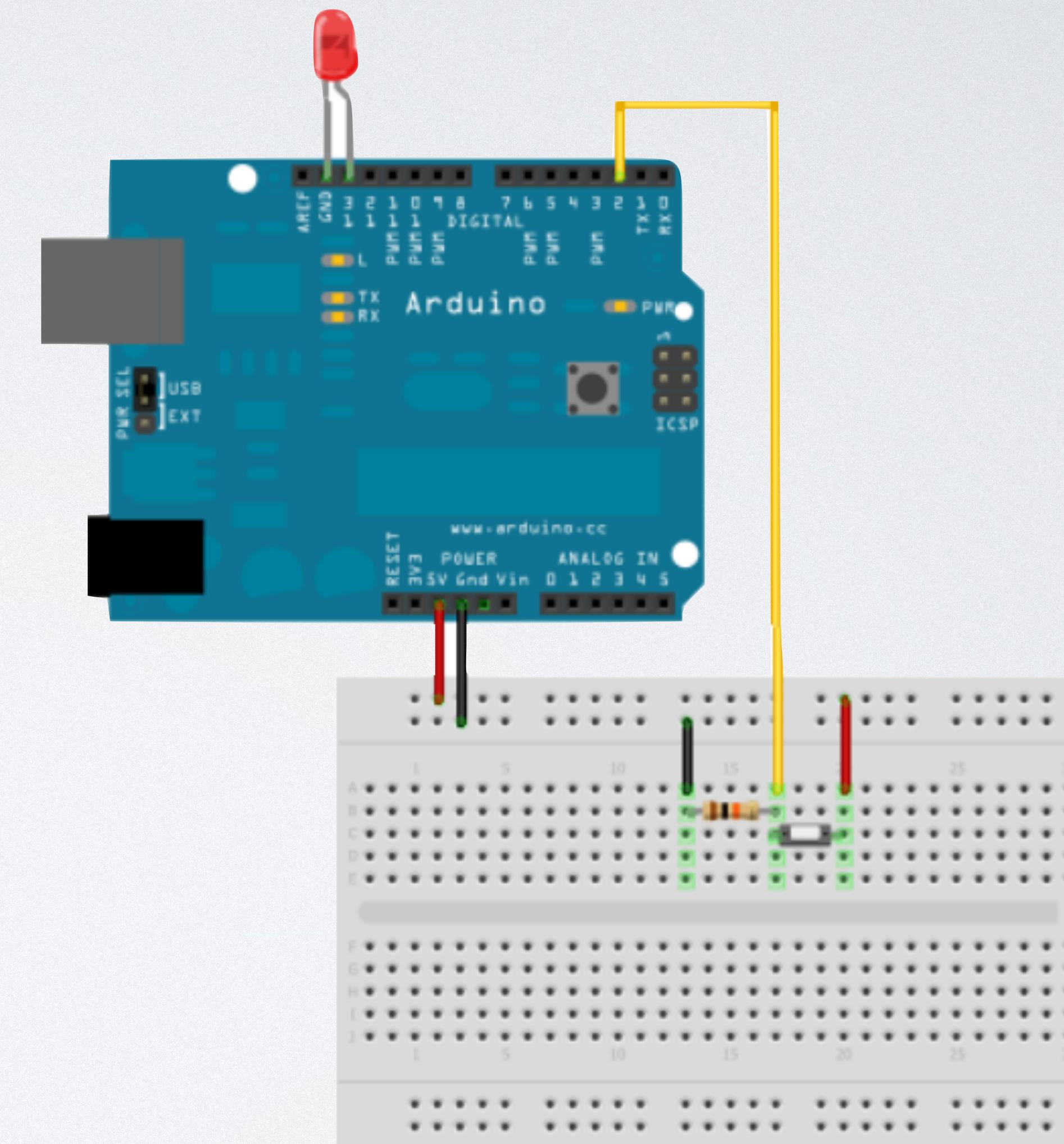
- Entradas y Salidas
 - Salidas digitales: 10, 11 y 13
 - Salidas analógicas: 5, 6, 9
 - Entradas analógicas: *todas las analógicas*
 - Entradas digitales: 2 y 3
 - Servomotores RC: 4, 7, 8, 12

EJEMPLO DEL LED PARPADEANTE

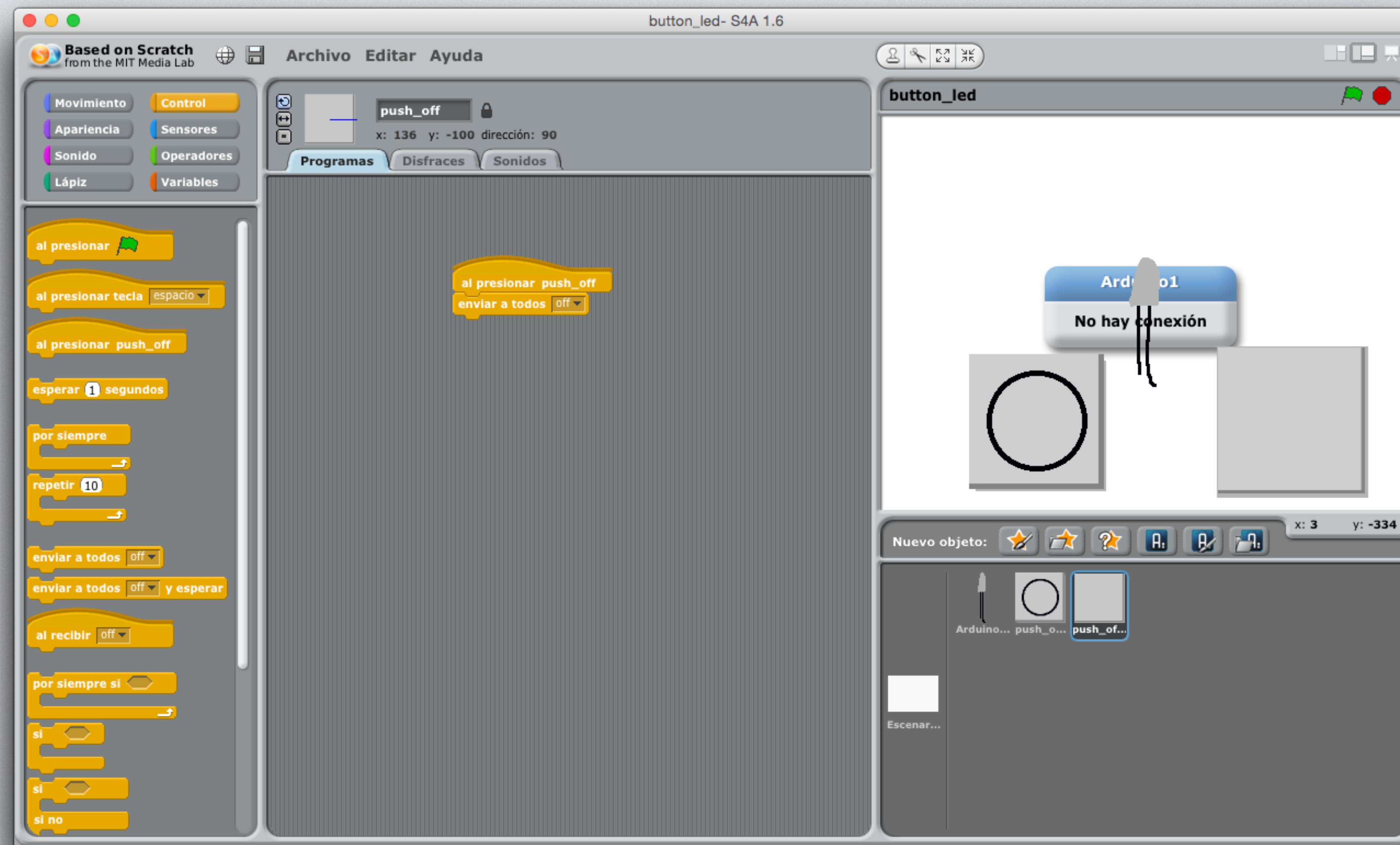


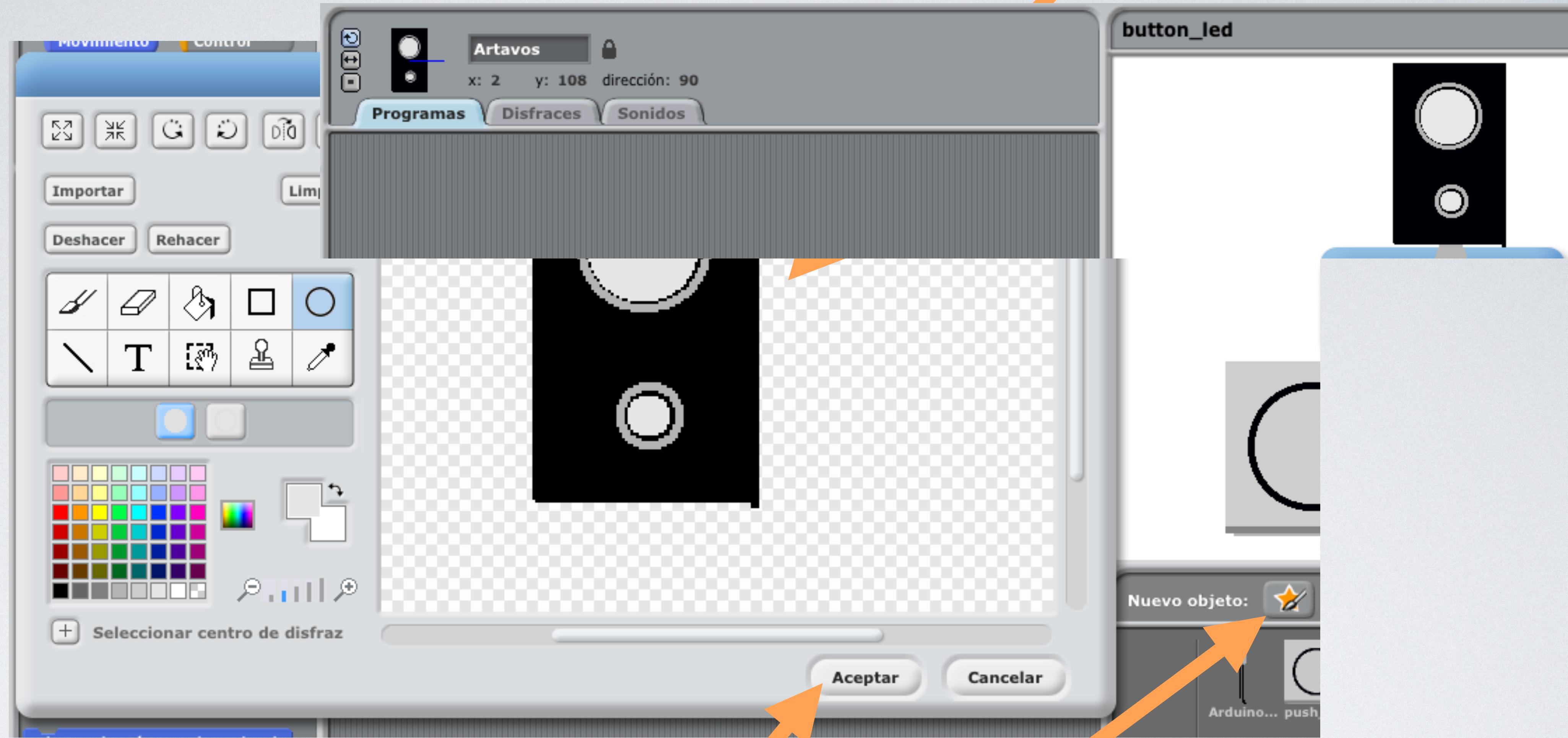
EJEMPLO DEL BOTÓN Y EL LED

```
al presionar bandera
por siempre
si
  ¿sensor Digital2 presionado? = true
    digital 13 encendido
    cambiar el disfraz a led on
  si no
    digital 13 apagado
    cambiar el disfraz a led off
```



¿BOTÓN VIRTUAL?





¿BOTÓN VIRTUAL?

BOTÓN Y LED, QUE RECUERDA



The script consists of the following blocks:

- Event: al presionar el botón
- Action: fijar ledOn a false
- Action: fijar oldValue a false
- Loop: por siempre
 - Action: fijar currentValue a *ísenor Digital1 pressed presionado?*
 - Decision: si *currentValue = true* y *oldValue = false*
 - Action: si *ledOn = false*
 - Action: digital 13 encendido
 - Action: fijar ledOn a true
 - Action: si no
 - Action: digital 13 apagado
 - Action: fijar ledOn a false
 - Action: fijar oldValue a *currentValue*

AHORA OS TOCA A VOSOTROS

¿QUE TAL HACER UN SEMÁFORO?