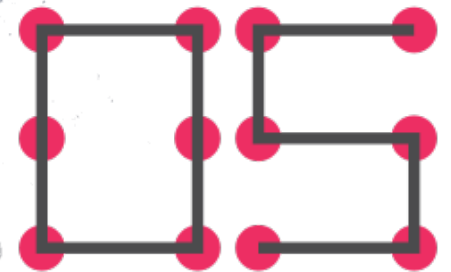


Taller básico de Arduino

Y su aplicación en el ámbito de las Neurociencias



MARCOS COLETTI
coletti.marcos@gmail.com



TALLERES OPEN SOURCE

Agenda

Presentación

Qué es Arduino? Conceptos básicos de Hardware y Software para su uso mediante ejemplos prácticos

Manos a la obra

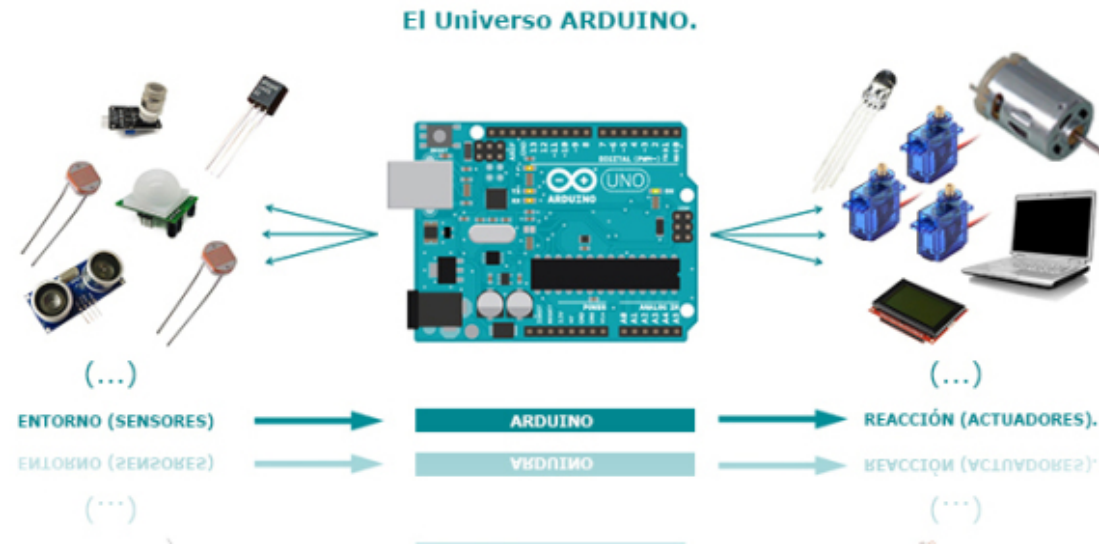
Actividad en grupos en la plataforma Tinkercad

Revisión conjunta, ejemplos y cierre

Comentarios sobre la actividad. Consultas.

Duración Total: 90min

Qué es Arduino?



Ejemplos de aplicación:

Centrifugas

Laberinto para estudio del ritmo circadiano

Impresora
3D

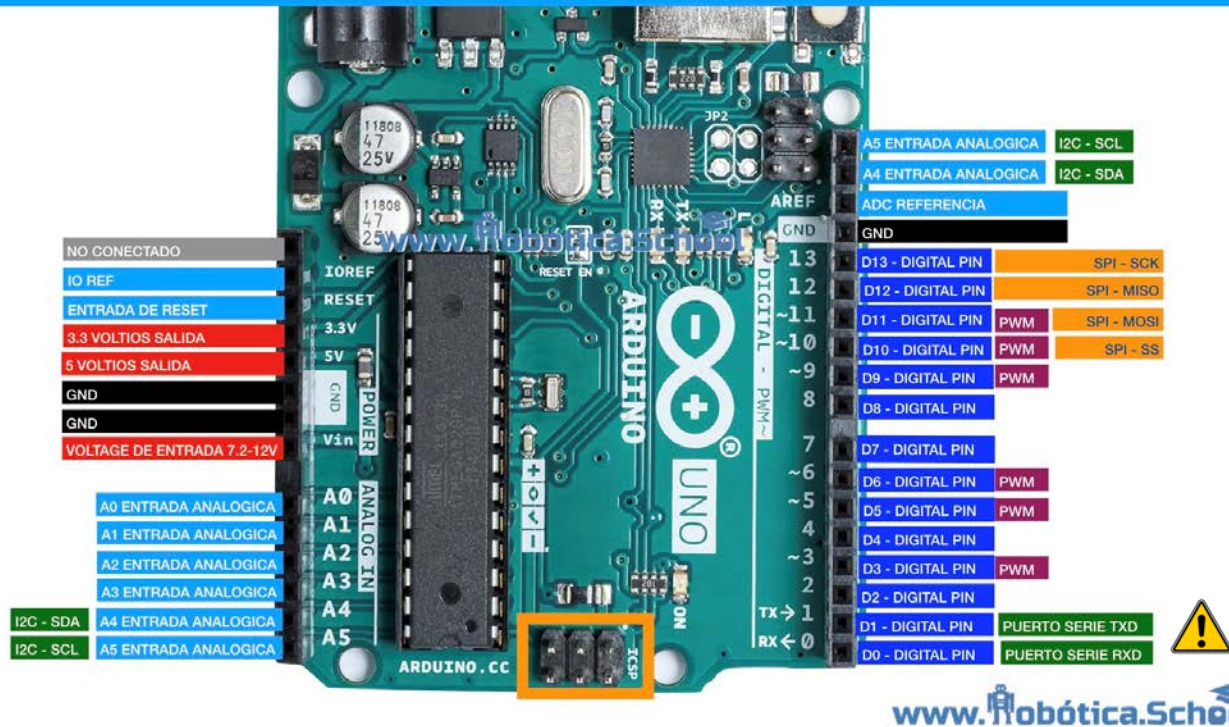
Datalogger para
jaulas

Caja operante usando un
ipod-touch

Cámara de exposición a
vapor de nicotina

Hardware del Arduino

ARDUINO UNO PIN OUT



Microcontroller & USB-to-serial converter	ATmega328P & Atmega16U2
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

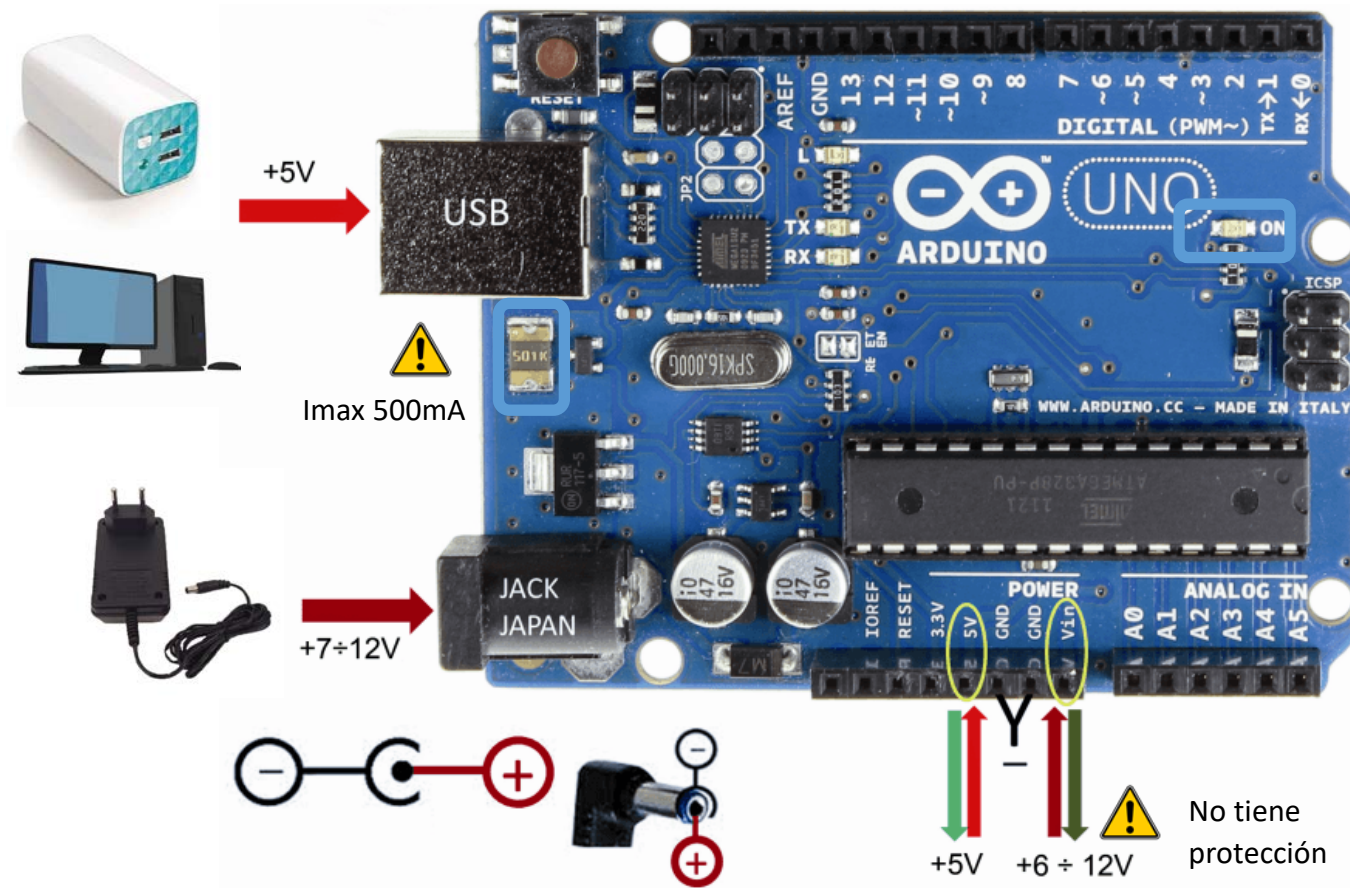
- 14 pines digitales que pueden ser configurados como entradas o salidas.
- 6 pines PWN (serigrafiados con ~) entre los pines digitales.
- 6 pines analógicos serigrafiados desde A0 hasta A5 para las entradas analógicas.
- 3 pines GND para conectar a tierra nuestros circuitos.
- 2 pines de alimentación de 5V y 3.3V respectivamente.

- I2C (pines 4 y 5)
- SPI (pines 10,11,12,13)
- Comunicación serie (pines 0 y 1) + Puerto USB
- Boton de reset



Hardware del Arduino

Alimentación de la placa



Límites de voltaje de entrada

- 7~12 V recomendado
- 6~20 V limite absoluto
- Pines Entrada/Salida (E/S): -0.5V a +5.5V (el máximo real es $V_{cc} + 0.5V$ para un arduino de 5V)

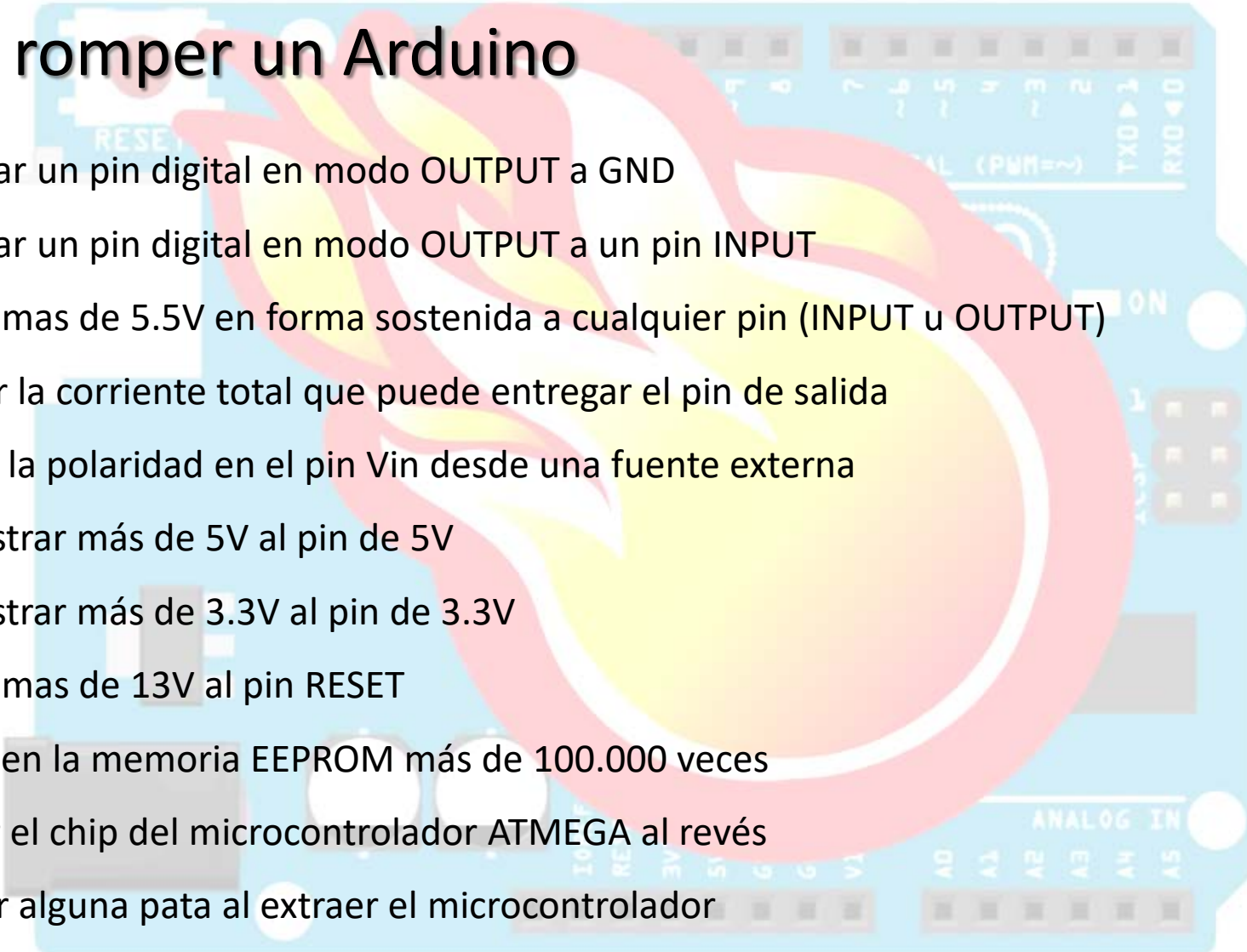
Límites de corriente de salida:

- Si es alimentado por USB: un total de 500 mA
- Si es alimentado por fuente externa o batería: un total de 500 mA~1 A
- Máximo individual por pin de E/S: 40 mA
- Suma de todas las Entradas/Salidas combinadas (SIN incluir el pin de "5V"): 200 mA

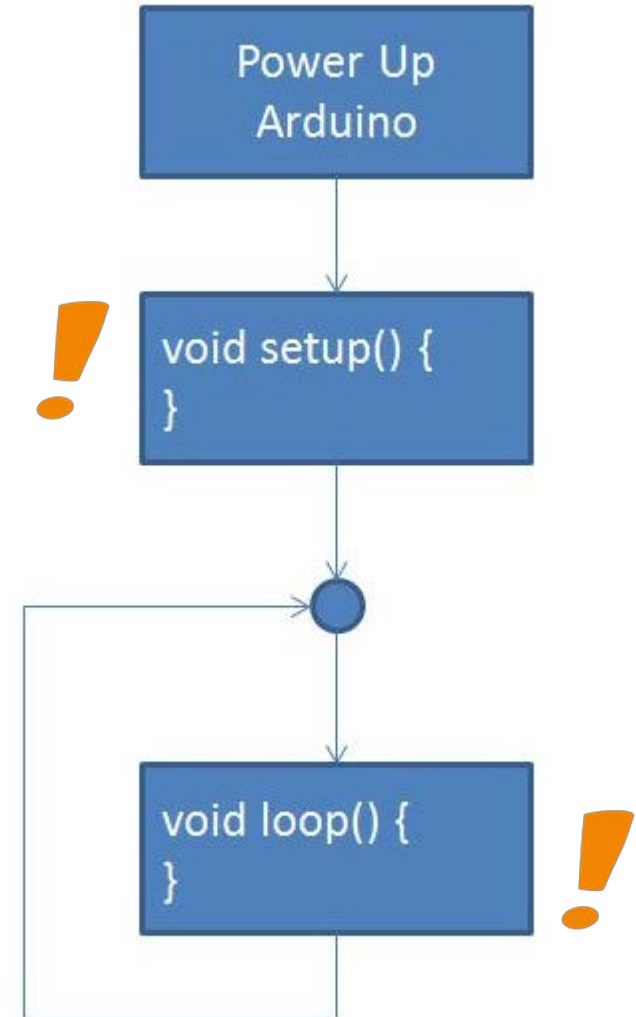
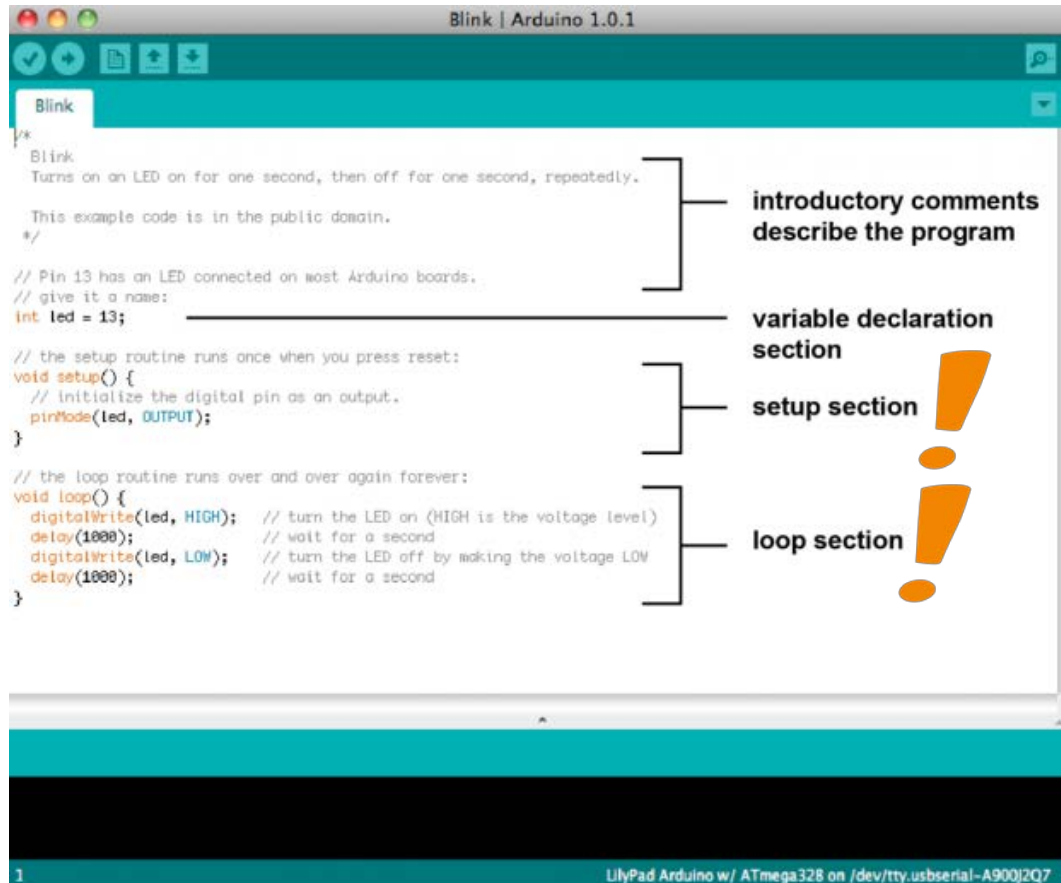
Hardware del Arduino

Como romper un Arduino

1. Conectar un pin digital en modo OUTPUT a GND
2. Conectar un pin digital en modo OUTPUT a un pin INPUT
3. Aplicar mas de 5.5V en forma sostenida a cualquier pin (INPUT u OUTPUT)
4. Exceder la corriente total que puede entregar el pin de salida
5. Invertir la polaridad en el pin Vin desde una fuente externa
6. Suministrar más de 5V al pin de 5V
7. Suministrar más de 3.3V al pin de 3.3V
8. Aplicar mas de 13V al pin RESET
9. Grabar en la memoria EEPROM más de 100.000 veces
10. Instalar el chip del microcontrolador ATMEGA al revés
11. Romper alguna pata al extraer el microcontrolador



Los programas (Sketch)



Entradas y Salidas (E/S) digitales

pinMode

SINTAXIS: pinMode(**#pin**, **modo**);

#pin: Es el numero del pin que se quiere configurar. Está escrito en la tarjeta Arduino

modo: INPUT, INPUT_PULLUP, OUTPUT

digitalWrite

SINTAXIS: digitalWrite(**#pin**, **estado**);

#pin: Es el numero del pin que se quiere cambiar el estado

estado: HIGH, LOW

digitalRead

SINTAXIS: digitalRead(**#pin**);

#pin: Es el numero del pin que se quiere saber el estado lógico de entrada

TIP. Las E/S analógicas se pueden usar también como E/S digitales llamándolas por su nombre A0,A1,etc



RECORDAR. No superar los 40mA de salida de cada pin de salida y 200mA máx en conjunto

Simulación en TinkerCad




Recursos en: [GitHub del evento](#)



Simulador: www.tinkercad.com



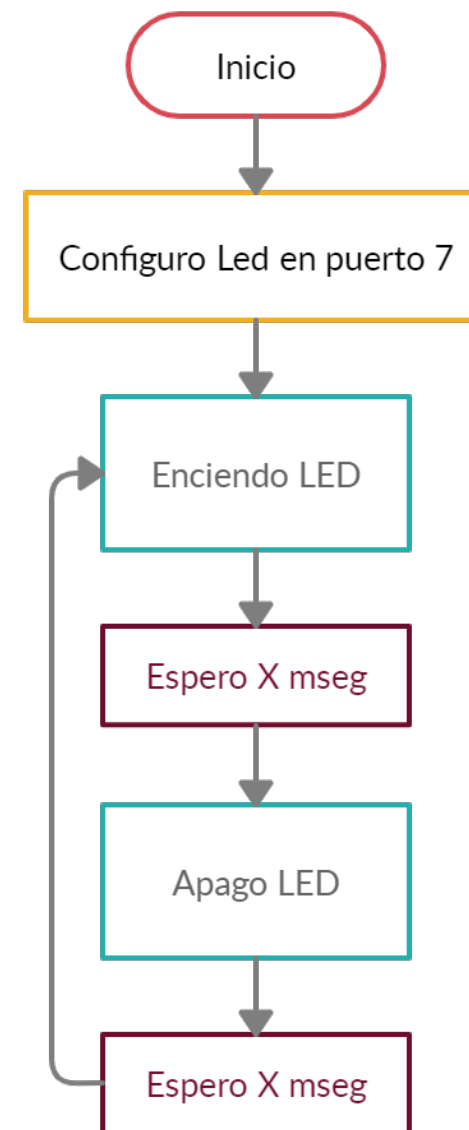
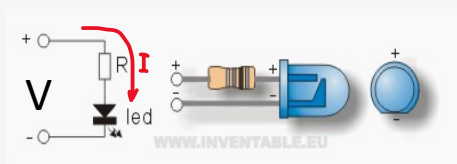
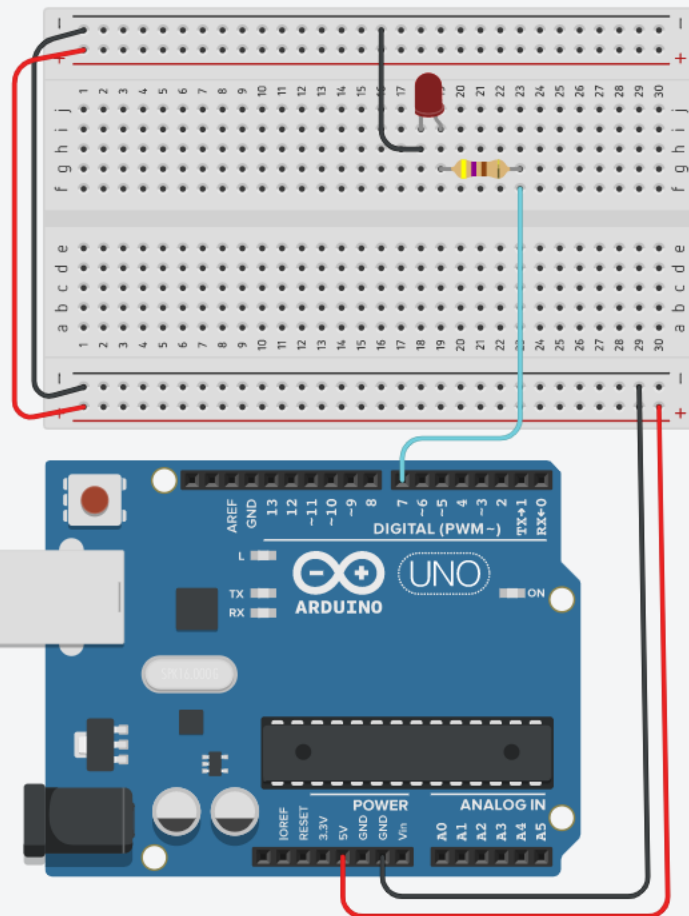
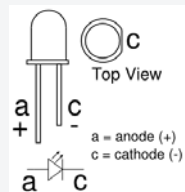
Preguntar siempre!


 Código Iniciar simulación

Exportar

Compartir

Blinking LED

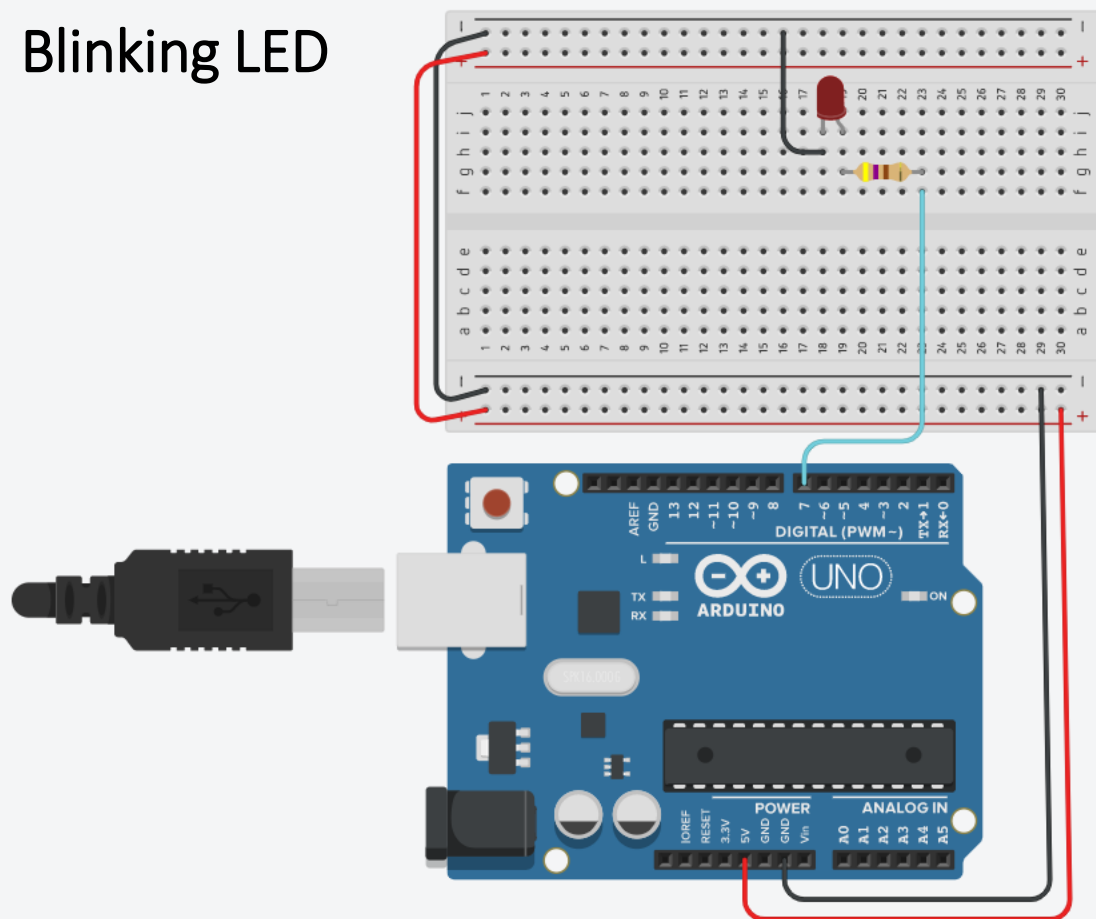


 Código Iniciar simulación

Exportar

Compartir

Blinking LED

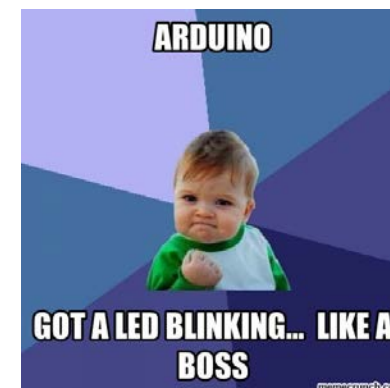


Texto



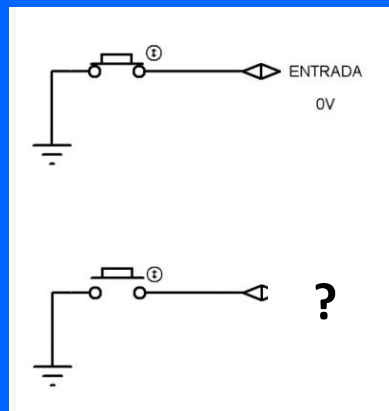
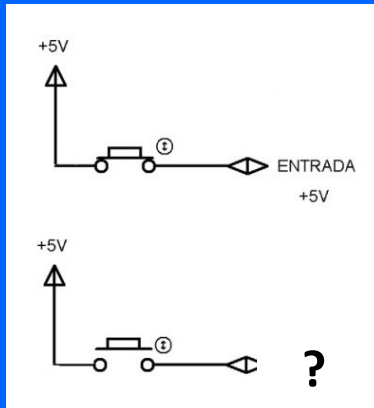
1 (Arduino Uno R3)

```
1 //Salidas
2 const int led = 7;
3
4 void setup()
5 {
6   pinMode(led,OUTPUT);    //defino pin de led como salida
7   digitalWrite(led, LOW); //arranco con estado LOW la salida LED
8 }
9
10 void loop()
11 {
12   digitalWrite(led, HIGH); //enciendo el LED
13   delay(500); //espero 500mseg
14   digitalWrite(led,LOW); //apago el LED
15   delay(200); //espero 200mseg
16 }
```

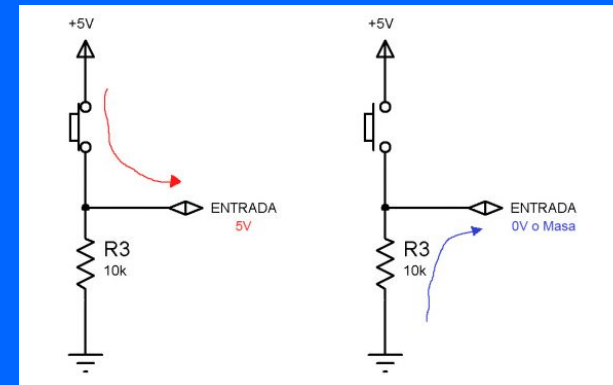


Monitor en serie

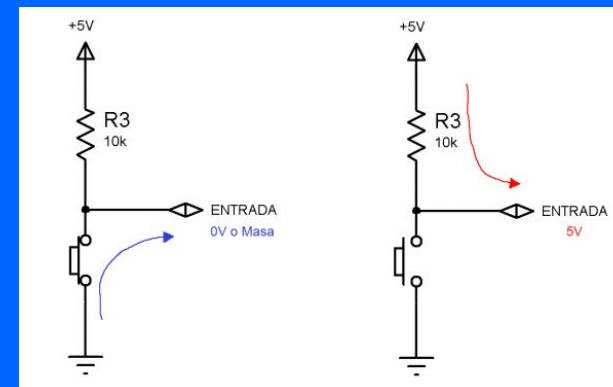
Agreguemos un pulsador




Resistencia Pull-Down



Resistencia Pull-Up

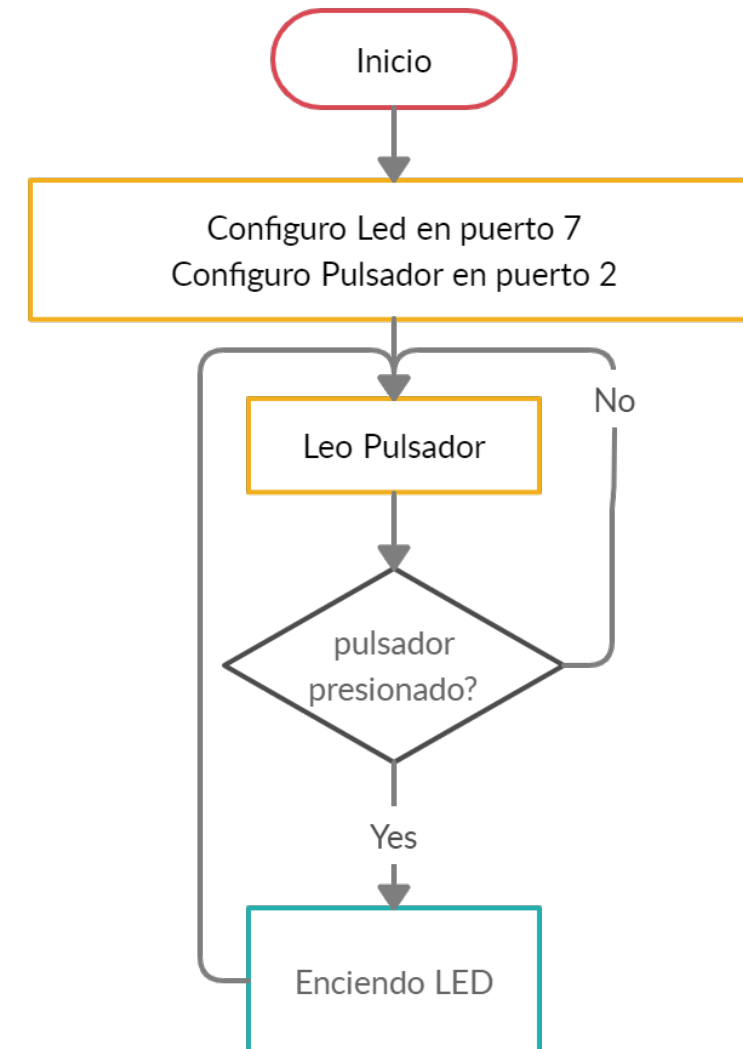
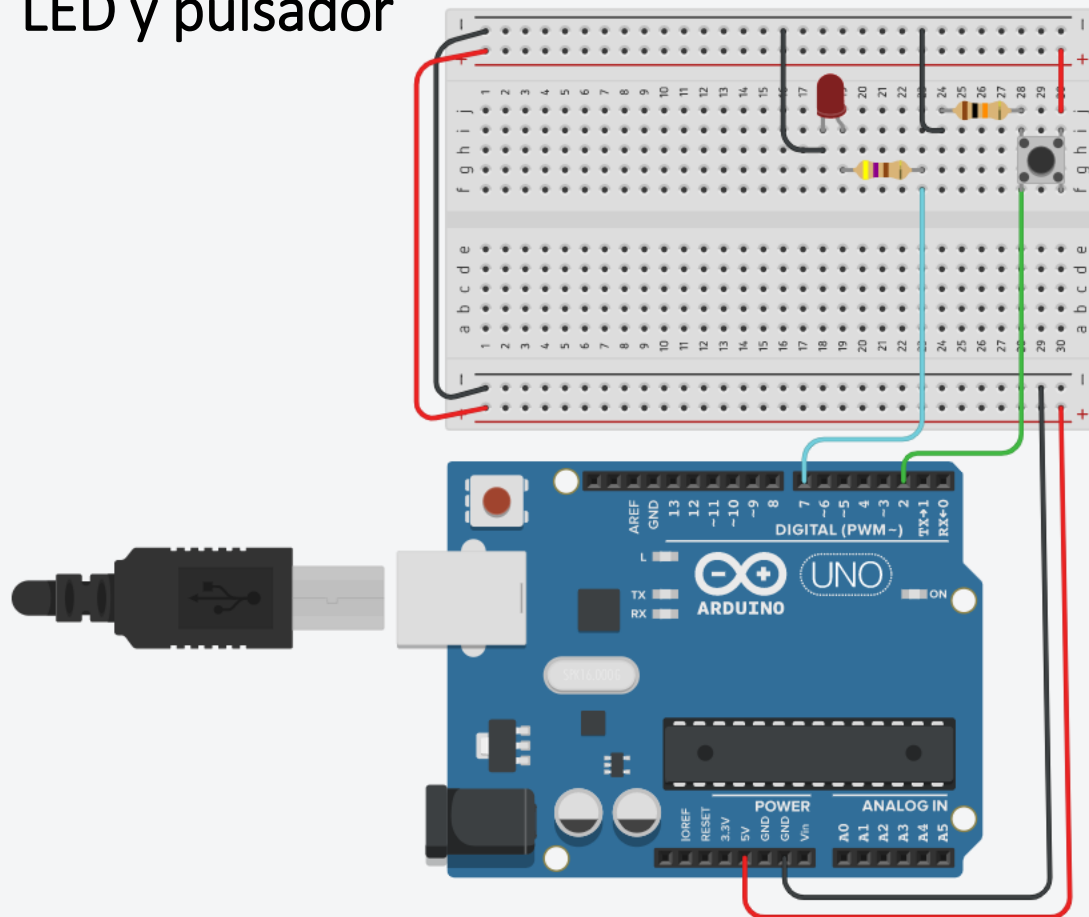


 Código Iniciar simulación

Exportar

Compartir

LED y pulsador





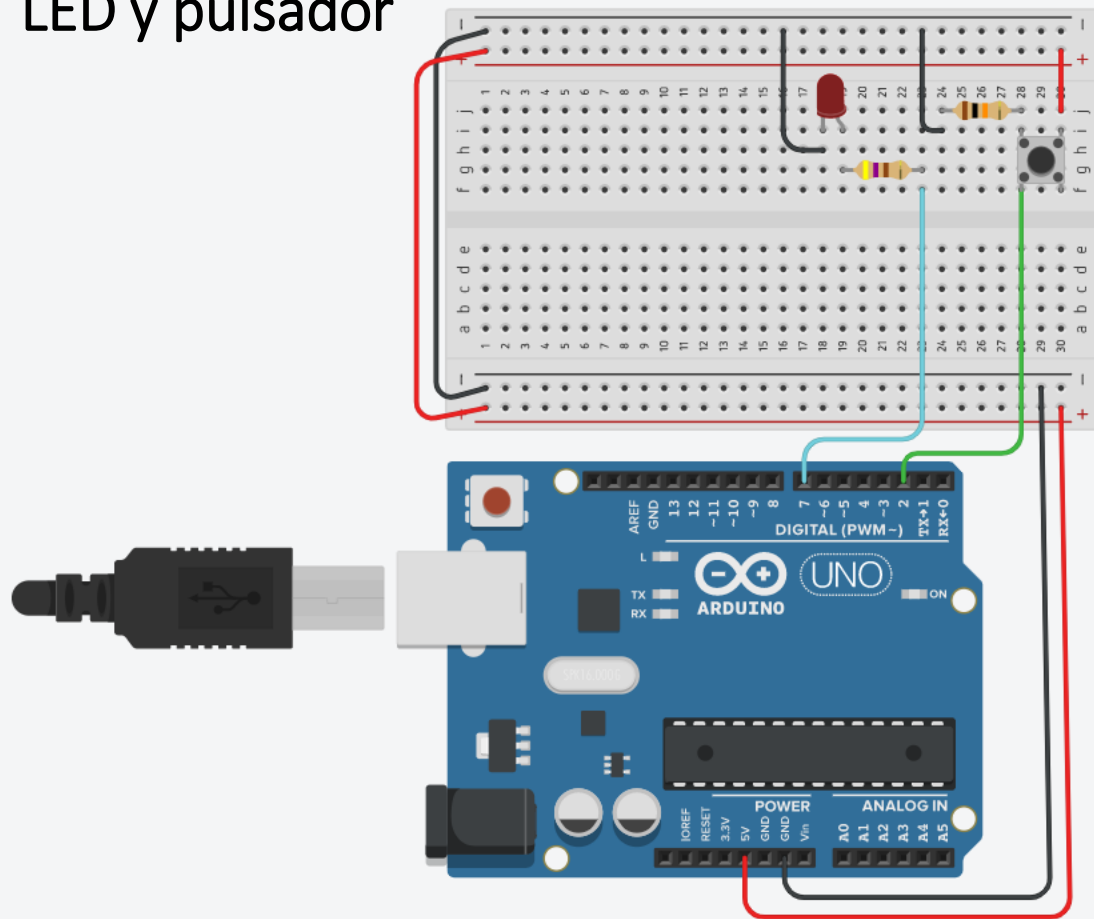
Código

▶ Iniciar simulación

Exportar

Compartir

LED y pulsador



Texto



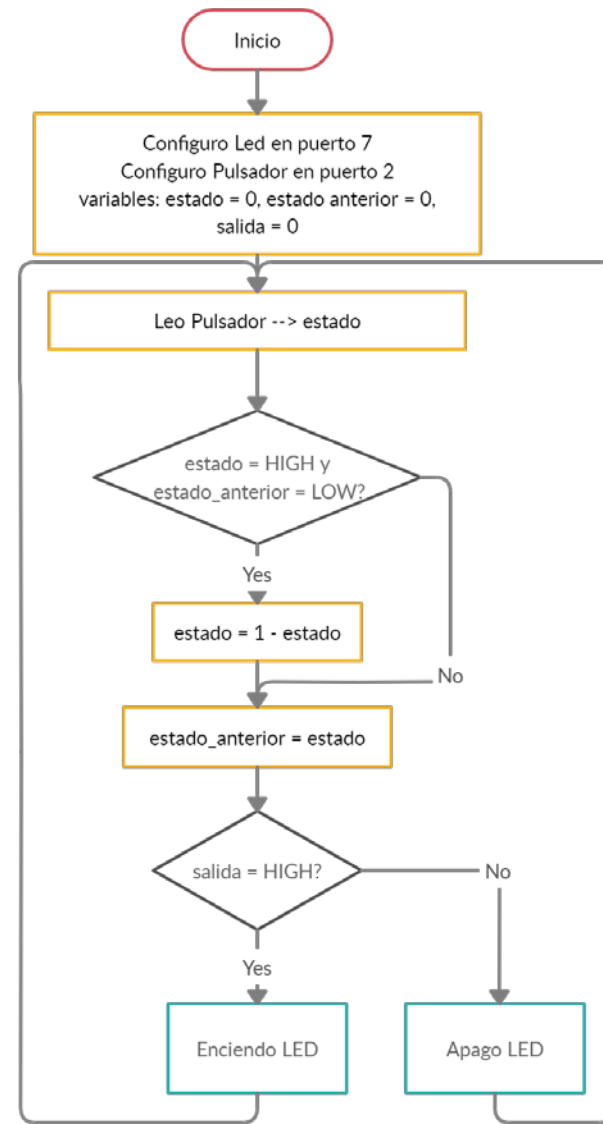
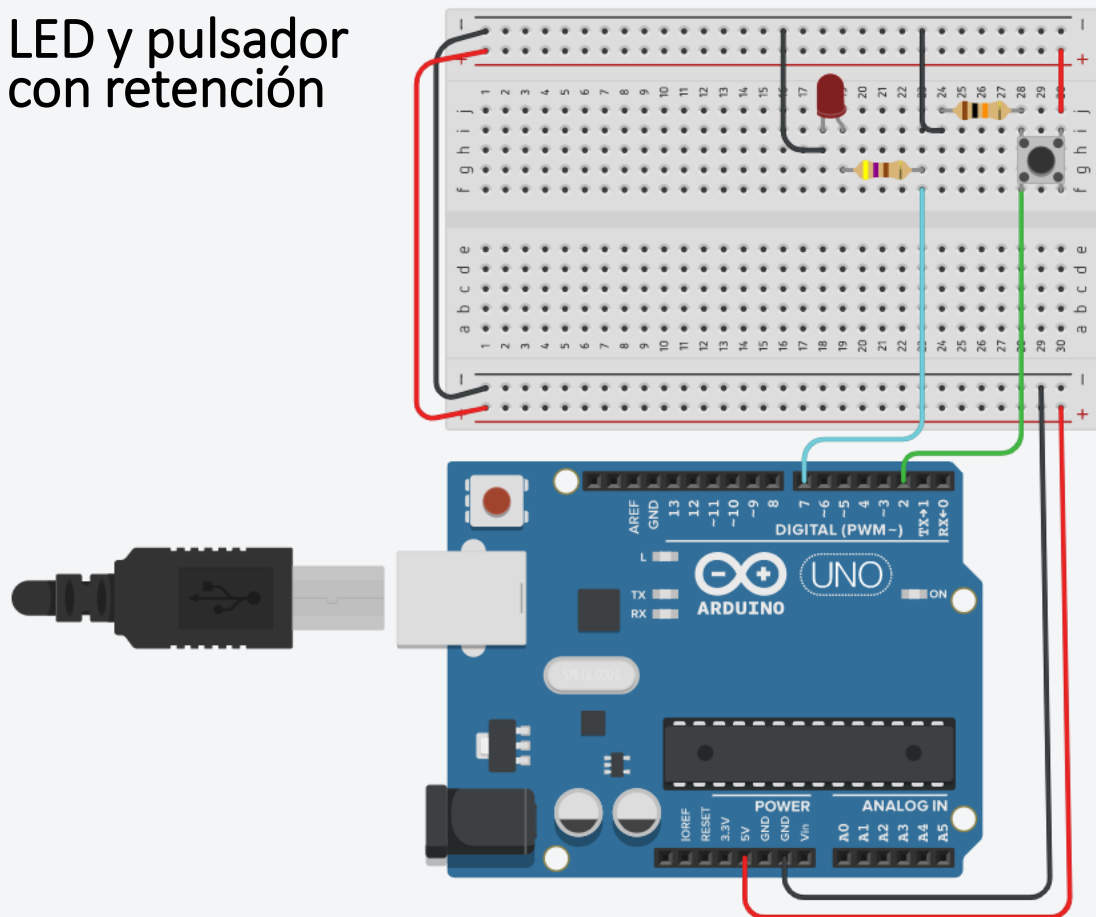
1 (Arduino Uno R3)

```
1 //Salidas
2 const int led = 7;
3
4 //Entradas
5 const int pulsador = 2;
6
7 //Variables auxiliares
8 int estado = 0; //estado del pulsador
9
10 void setup()
11 {
12   pinMode(led,OUTPUT); //defino pin de led como salida
13   pinMode(pulsador,INPUT); //defino pin de pulsador como entrada
14   digitalWrite(led, LOW); //arranco con estado LOW la salida LED
15 }
16
17 void loop()
18 {
19   estado = digitalRead(pulsador); //leo estado del puls.
20   if(estado == 1) //pregunto por el estado del pulsador
21   {
22     digitalWrite(led, HIGH); //si esta pulsado, prendo el LED
23   }
24   else
25   {
26     digitalWrite(led,LOW); //caso contrario, mantengo apagado LED
27   }
28 }
```



Monitor en serie

LED y pulsador con retención





Código

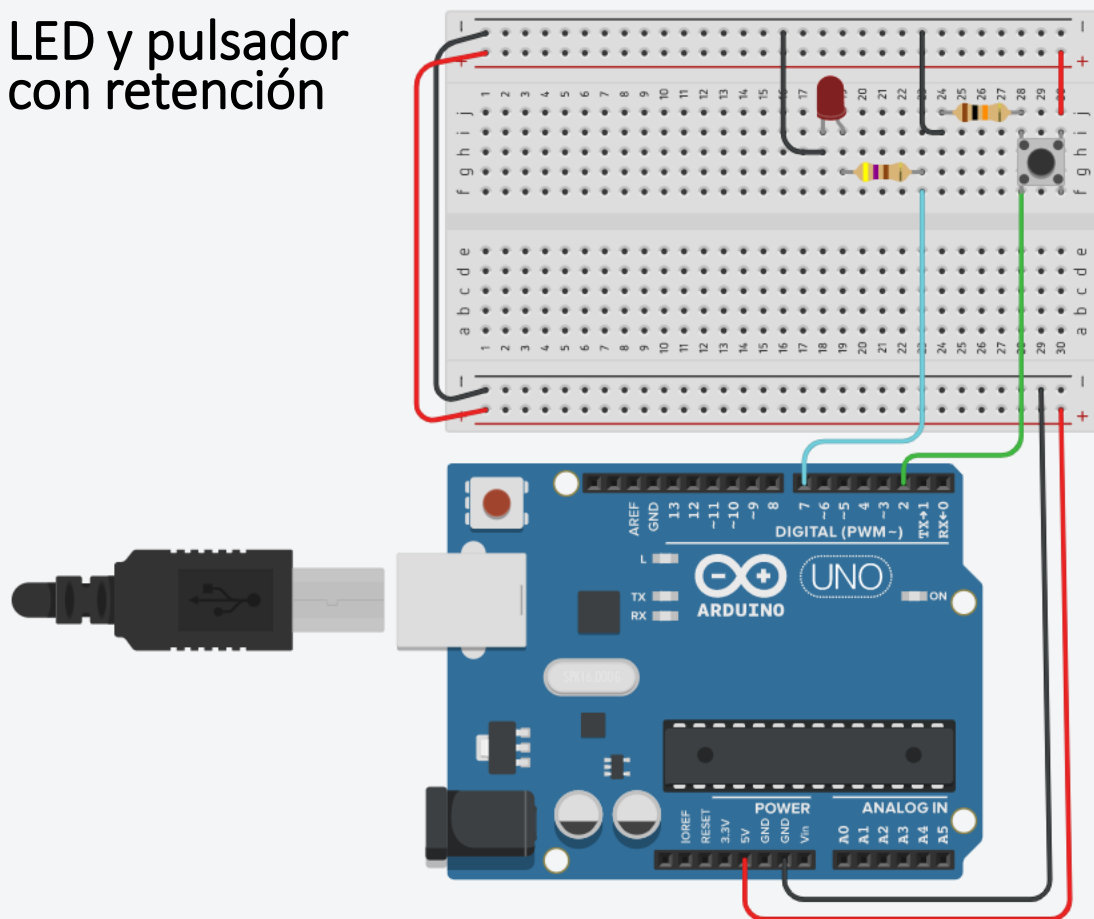


Iniciar simulación

Exportar

Compartir

LED y pulsador con retención



Texto



1 (Arduino Uno R3)

```
1  const int led = 7;
2  const int pulsador = 2;
3
4  int salida = 0;           //estado del led
5  int estadoanterior = 0;  //estado anterior del pulsador
6  int estado = 0;          //estado del pulsador
7
8  void setup()
9  {
10     pinMode(led,OUTPUT);    //defino pin de led como salida
11     pinMode(pulsador,INPUT); //defino pin de pulsador como entrada
12     digitalWrite(led, LOW); //arranco con estado LOW la salida LED
13 }
14
15 void loop()
16 {
17     estado = digitalRead(pulsador); //leo estado actual del puls.
18     if(estado == HIGH && estadoanterior == LOW)
19     {
20         salida = 1 - salida; //cambia 1<-->0
21         delay(20); //DEBOUNCE
22     }
23     estadoanterior = estado;
24     if(salida == 1)
25     {
26         digitalWrite(led, HIGH);
27     }
28     else
29     {
30         digitalWrite(led,LOW);
31     }
32 }
```



Monitor en serie

Manos a la obra...



Manos a la obra

Grupos aleatorios de 5 personas

Uno entra a TinkerCad y comparte pantalla. Pero trabajan todos juntos.

Valerse de los recursos en GitHub

Probar conectando de diferentes formas los componentes. Discutir la forma de conectarlos

Modificar el código

Comentar entre todos modificaciones a hacer al código. Simularlas.

Solicitar asistencia

Pueden solicitar ayuda en cualquier momento clickeando en el icono 

Duración de la Actividad: 30min


Actividad propuesta

Comenzar de cero con una placa Arduino y un protoboard.

- Conectar un LED al puerto D7
- Conectar un pulsador en el puerto D2.
- Escribir el código para setear cada puerto como entrada o salida según corresponda.
Hacer que el LED se encienda al presionar el pulsador y se apague al soltarlo.

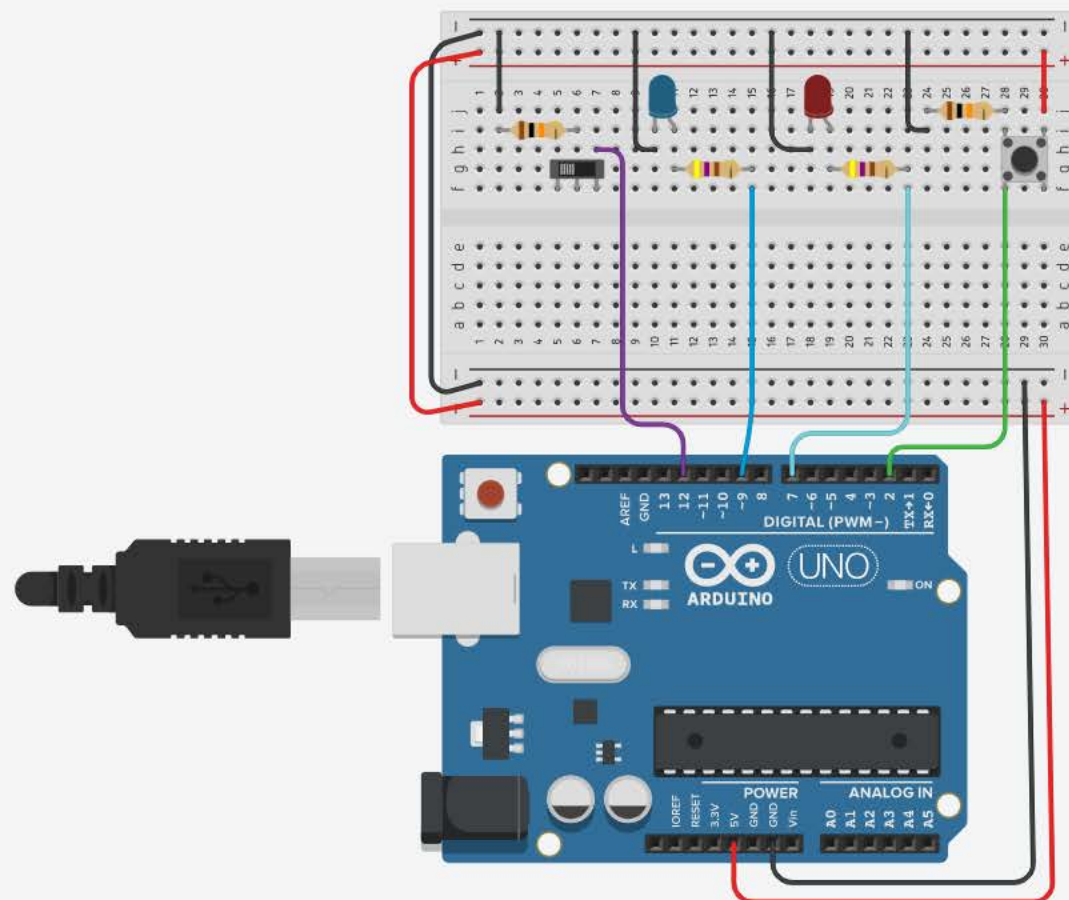
EXTRA

- Agregar otro LED (de un color diferente) en el puerto D9.
- Modifica el código para que un led se encienda al presionar el pulsador y el otro led se encienda de forma intermitente durante todo el tiempo.

 Código Iniciar simulación

Exportar

Compartir

Componentes
Básico

Buscar



Resistencia



LED



Pulsador



Potenciómetro



Condensador

Interruptor
deslizante

Recursos para seguir

Arduino and Electronics Reference sheet

[En el GitHub del evento](#)

Biblioteca de Libros Varios de Arduino

[Carpeta de Drive](#)

Sitio oficial de la comunidad Arduino

www.arduino.cc

Si puedes imaginarlo,
puedes crearlo...

Walt Disney (Tom Fitzgerald)

Muchas Gracias!