

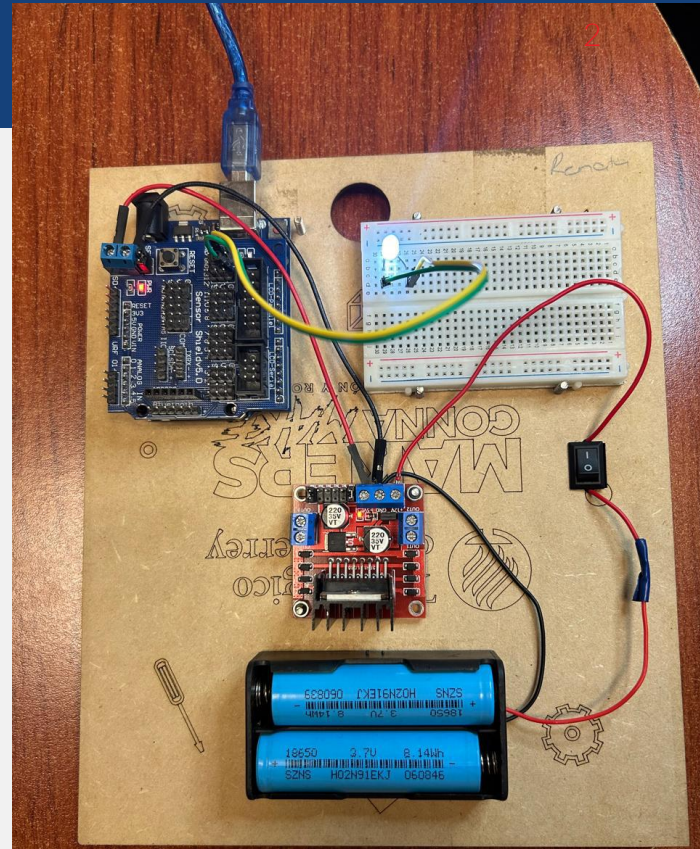
Introducción a los Robots



¡La clase pasada!

Antes de nada revisa que tu circuito encienda. Revisa si necesitas poner a cargar tus pilas o si hay que arreglar algún cable.

Listo!



Bienvenido!

Anatomía



Proceso de construcción y elementos de un robot

Aunque todos los robots varían en su complejidad, todos se van armando tomando la siguiente secuencia:

- A) S3 - Diseño Chasis y componentes principales
- B) S4- Instalación y cableado Sistema eléctrico
- C) S5- Programación del Sistema de control
- D) S6- Cableado, instalación de Actuadores y variadores
- E) S7- Cableado y programación de Sensores
- F) S8- Configuración y programación del sistema de Comunicación
- G) S9 -Pruebas

Breadboard



¿Qué es un breadboard / protoboard?

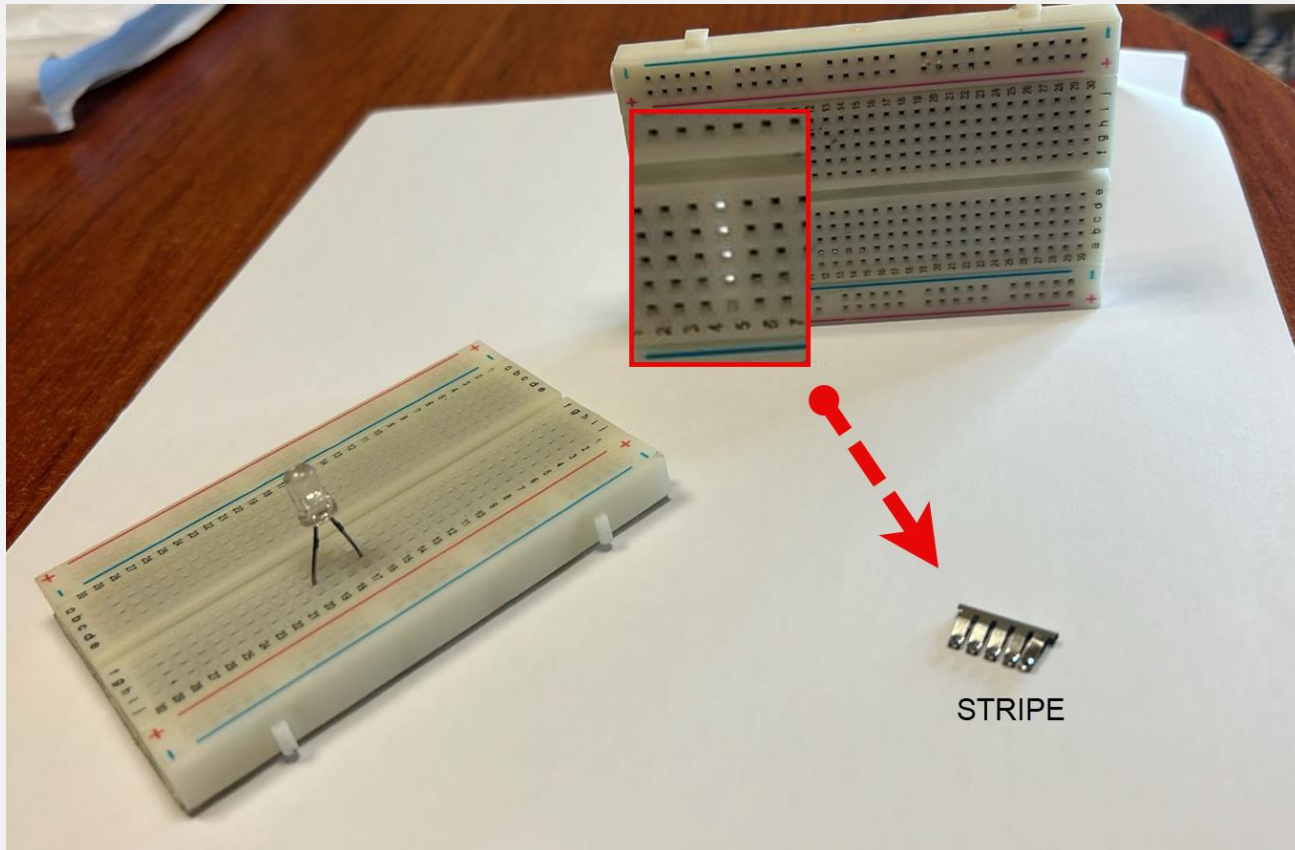
6

- El "protoboard" o "breadboard" es un tablero con orificios conectados eléctricamente entre sí, habitualmente siguiendo patrones de líneas, en el cual se pueden insertar componentes electrónicos y cables para el armado y prototipado de circuitos electrónicos y sistemas similares.



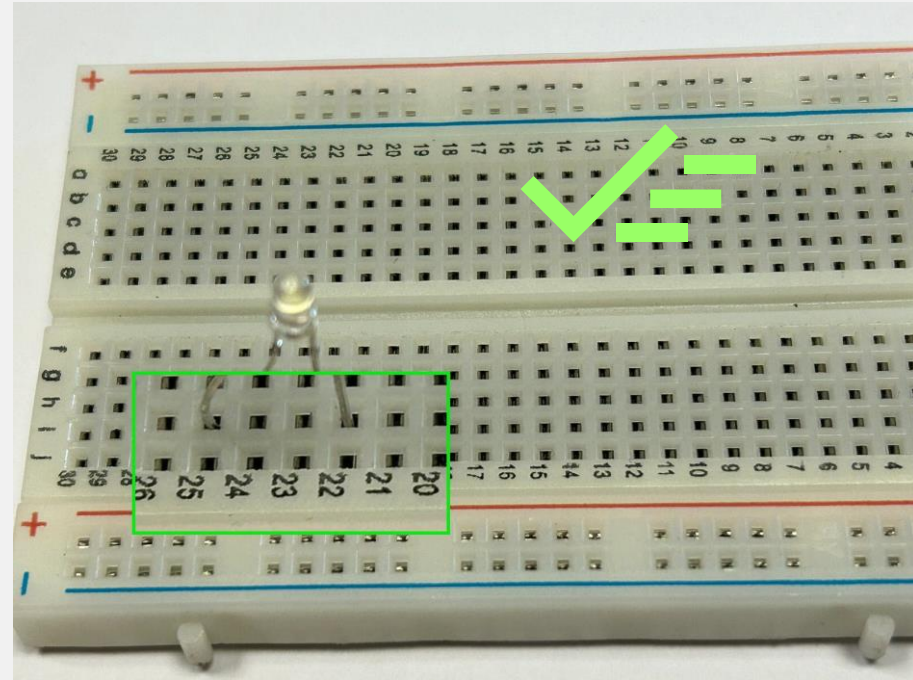
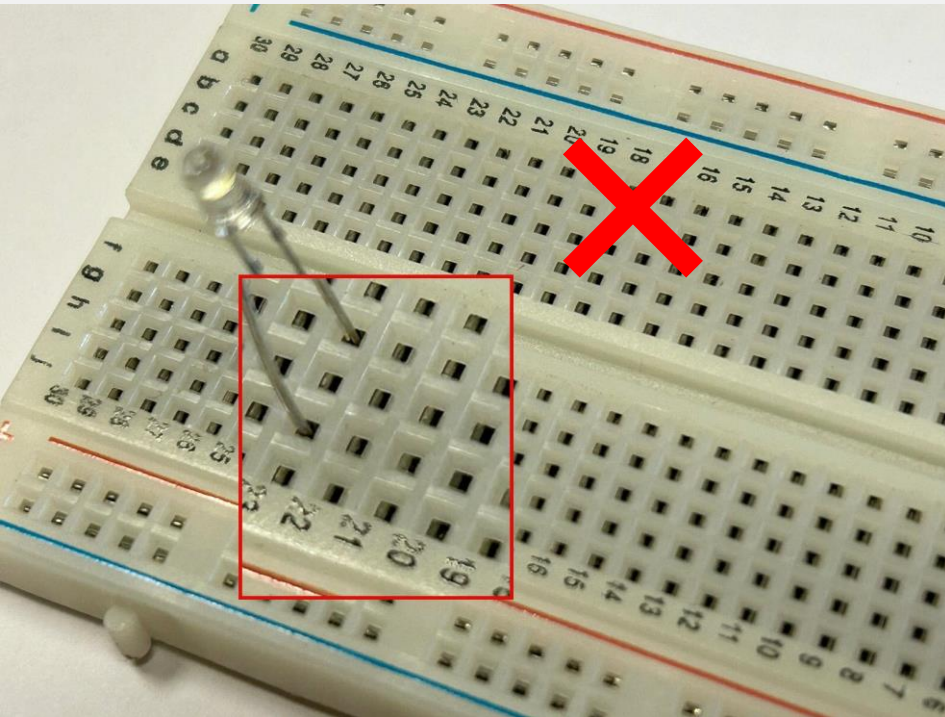
Rails & Stripes

7



Correcto e Incorrecto

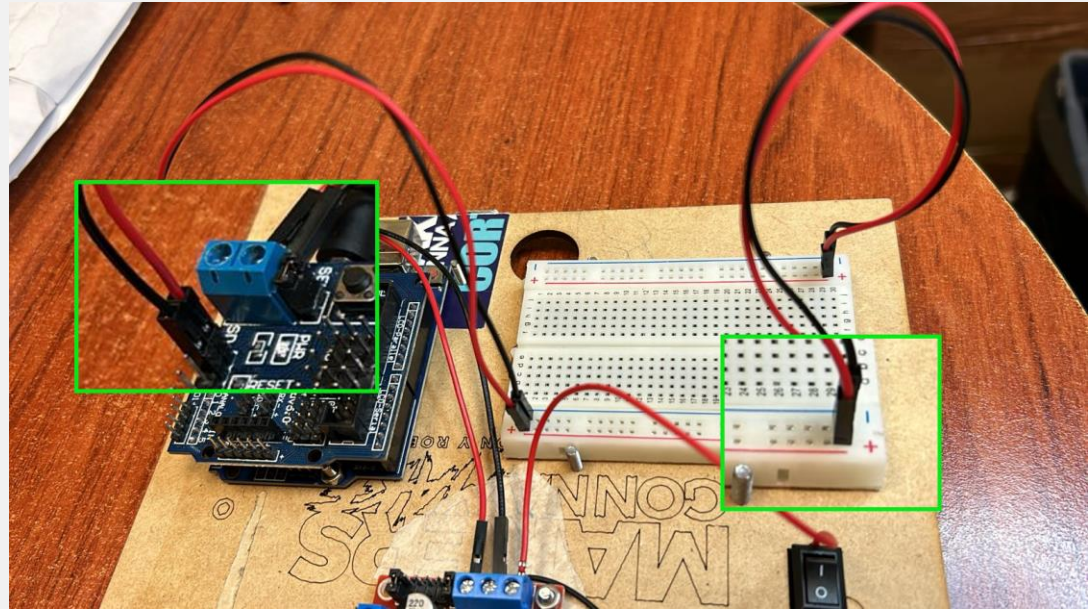
8



¡Tiene que ver con la estructura interna de la placa!, vamos a ver

Puentes

Primero energizaremos nuestra placa de pruebas, usando puentes entre nuestro Arduino y nuestra breadboard, como se muestra.

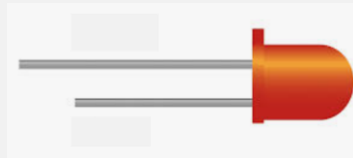


Leds



Polaridad

Antes de continuar es importante que sepas que hay componentes que **DEBES** conectar correctamente a **+** y **-**. Esto significa que son componentes con polaridad. Un ejemplo son los LEDs.



Hay componentes que no importan como se conecten como son las resistencias, o los botones los cuales no tienen polaridad.



LED

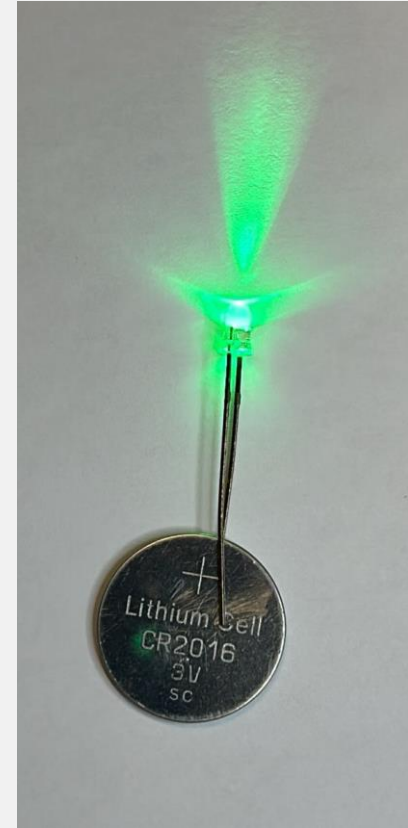
12

Un LED (acrónimo del concepto inglés light-emitting diode) es un diodo emisor de luz. En su interior hay un semiconductor que, al ser atravesado por una corriente emite luz.



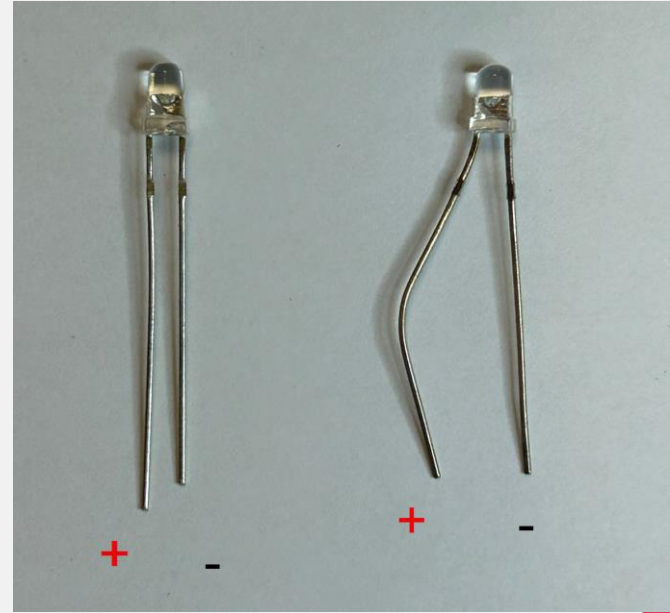
Probando un LED

- La forma más sencilla de probar un LED es usando una pila de reloj o de botón de 3V. La más común es la CR2016



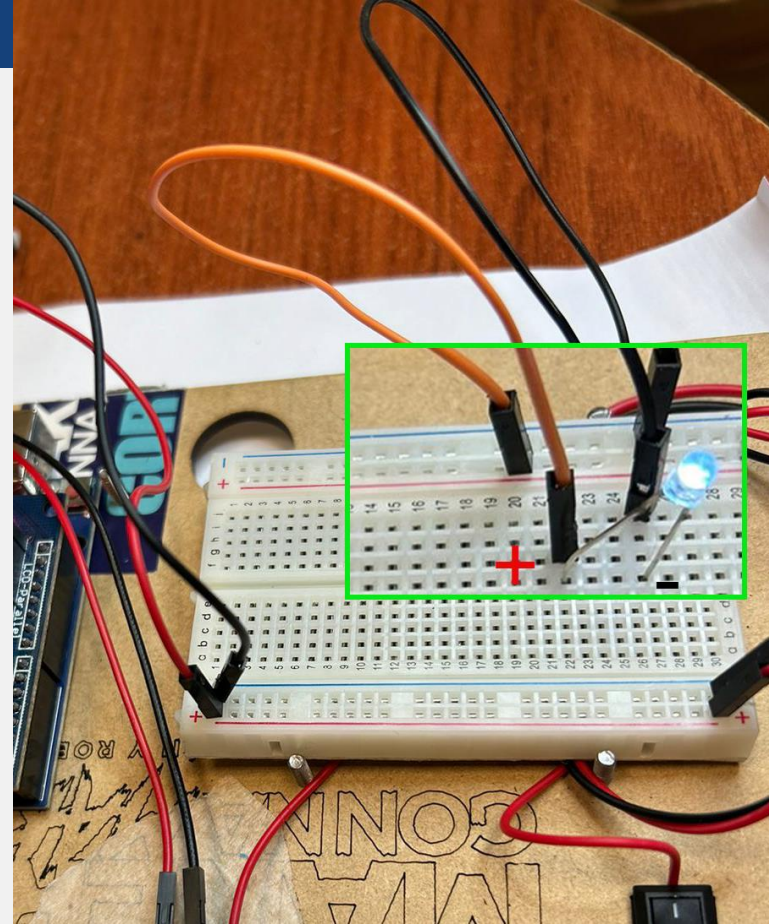
Identificando la polaridad

- Un truco para identificar la polaridad de un LED es doblar su polo más largo como se muestra a continuación, además de que es más sencillo conectarlo



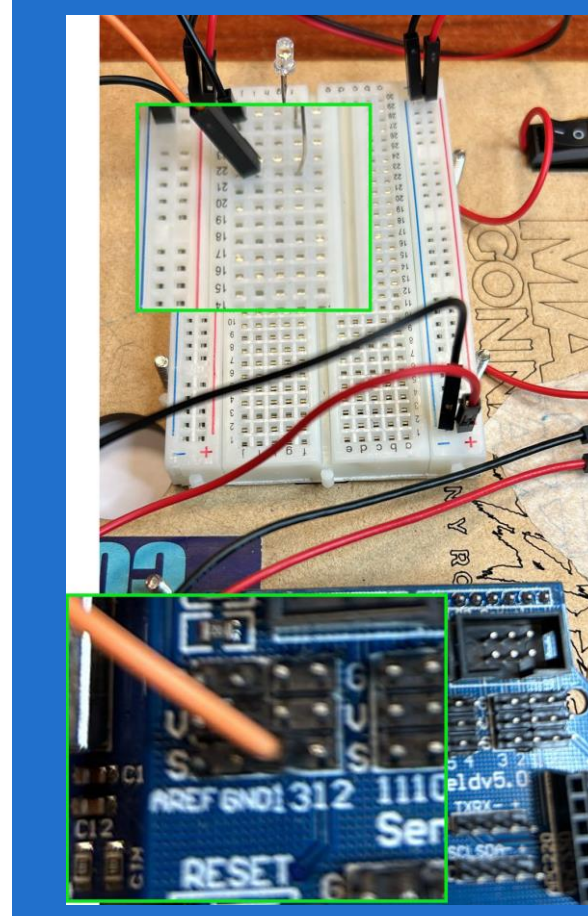
Probando nuestros puentes

- Para probar si nuestra placa de pruebas tiene energía, podemos colocar un LED y probamos que tenga energía en ambos lados.

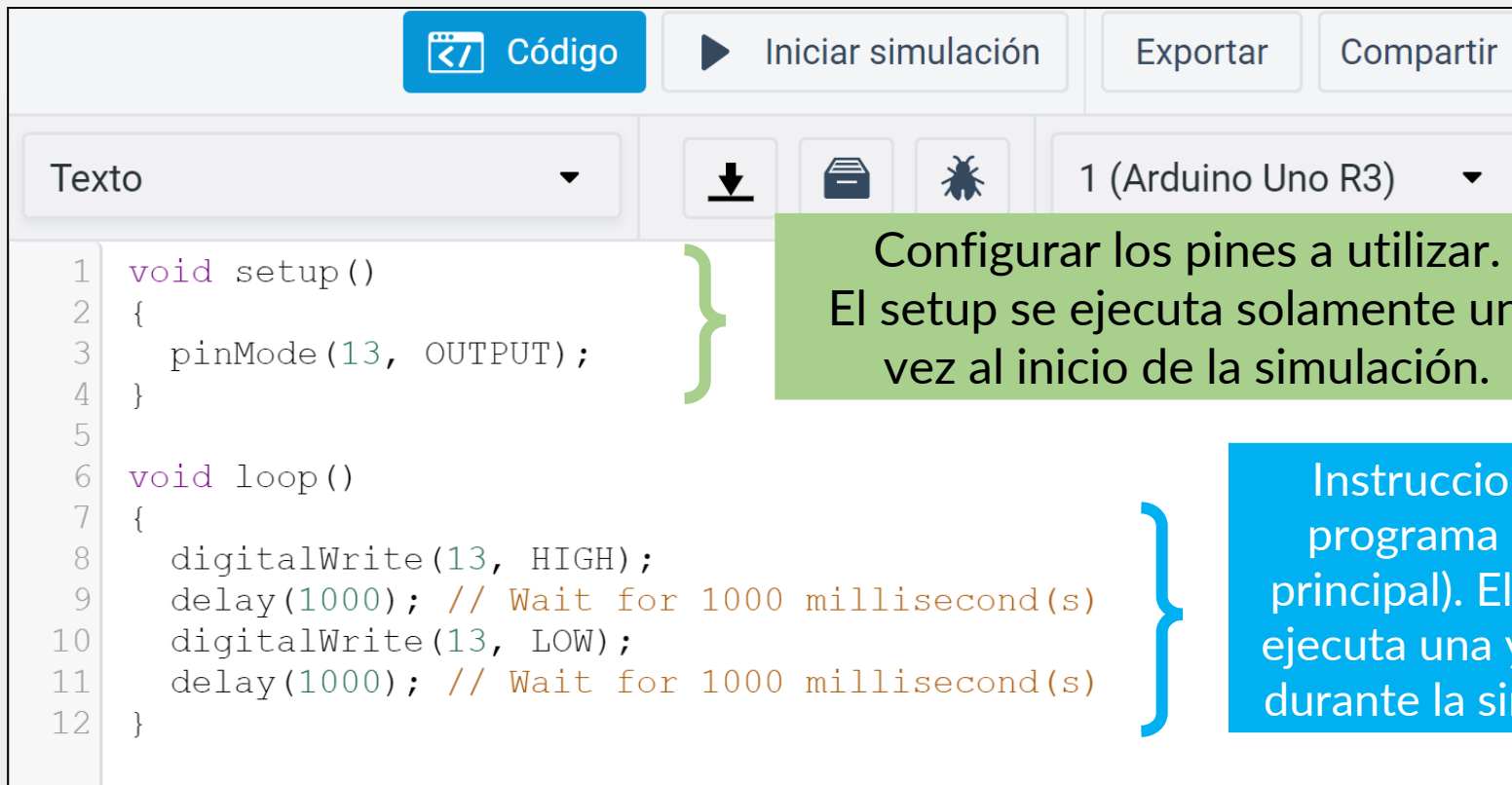


Ahora vamos a programar nuestro LED

Vamos a conectar el polo positivo de nuestro LED al puerto 13 de nuestra placa.



Estructura del código



The screenshot shows the Arduino IDE interface. At the top, there are buttons for 'Código' (Code), 'Iniciar simulación' (Start simulation), 'Exportar' (Export), and 'Compartir' (Share). Below these, there is a dropdown menu for 'Texto' (Text) and a toolbar with icons for download, save, and debug. To the right of the toolbar, it says '1 (Arduino Uno R3)'. The main area is the code editor, which contains the following code:

```
1 void setup()
2 {
3   pinMode(13, OUTPUT);
4 }
5
6 void loop()
7 {
8   digitalWrite(13, HIGH);
9   delay(1000); // Wait for 1000 millisecond(s)
10  digitalWrite(13, LOW);
11  delay(1000); // Wait for 1000 millisecond(s)
12 }
```

Two callouts are present:

- A green callout with a bracket pointing to the `setup()` function, stating: "Configurar los pines a utilizar. El setup se ejecuta solamente una vez al inicio de la simulación." (Configure the pins to use. The setup is executed only once at the start of the simulation.)
- A blue callout with a bracket pointing to the `loop()` function, stating: "Instrucciones del programa (código principal). El loop() se ejecuta una y otra vez durante la simulación." (Program instructions (main code). The loop() is executed once and over again during the simulation.)

Comandos básicos de Arduino

pinMode()

- Configura el pin especificado para que se comporte como INPUT o OUTPUT.

digitalWrite()

- Escribe un valor HIGH o LOW en un pin digital.
- HIGH = 1 u ON
- LOW = 0 u OFF

delay()

- Pausa el programa por la cantidad de tiempo (en milisegundos) especificado como parámetro. (Hay 1000 milisegundos en un segundo).



Botón




Boton

Es un interruptor que cuando se presiona deja pasar la energía. Como ya dijimos no tiene polaridad.

Ahora vamos a colocar un botón que cuando se presione, se encienda el LED.



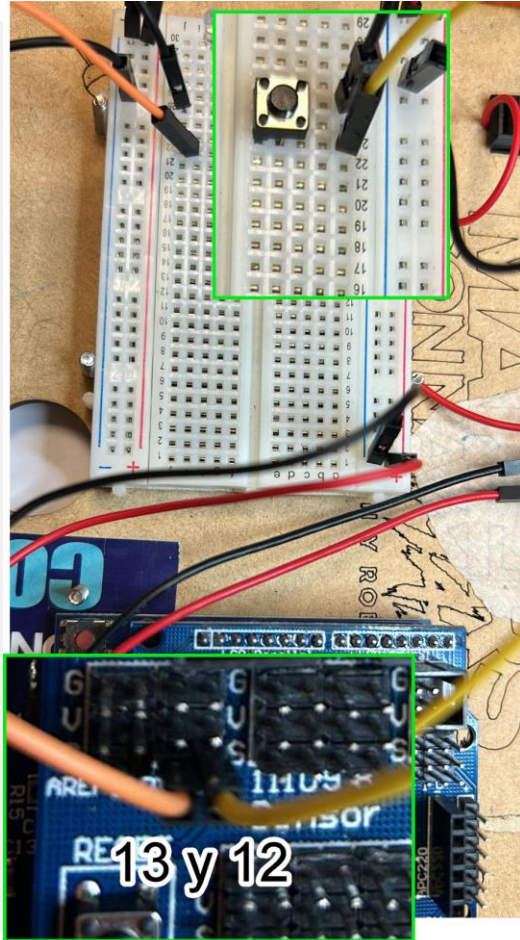
Programando botones

- Los botones son componentes tipo INPUT, los botones específicamente tipo **INPUT_PULLUP**  son
- Los botones también se configuran en el setup() con la función **pinMode()**; `pinMode(pinNumber, INPUT_PULLUP);`
- **digitalRead(pinNumber)**; será la función que nos permita leer las acciones que suceden en el botón
- En el ejemplo de hoy usaremos una estructura condicional **if-else y variables**
- El funcionamiento del ejemplo será: cuando presiono el botón, el LED se enciende; de lo contrario, estará apagado.



Observa:

- Ya sabes como conectar un LED, solamente recuerda que tener presente su número de pin es importante, en este ejemplos el pin **13 (naranja)**.
- El **botón** requiere una conexión similar a la del led, una terminal a tierra/ground y la otra a un pin digital, en este caso, pin **12 (amarillo)**.

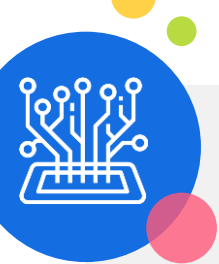




• Ahora, escribe y prueba este código:

23

```
void setup() {  
  pinMode(12, INPUT_PULLUP);  
  pinMode(13, OUTPUT);  
}  
  
// the loop function runs over and over again forever  
void loop() {  
  int bot = digitalRead(12);  
  if (bot == LOW) // Si se presiona el botón  
    digitalWrite(13, HIGH);    // prende el LED  
  else  
    digitalWrite(13, LOW);    // apaga el LED  
}
```



Trabaja en parejas para resolver el ejercicio:

Agrega un buzzer, conéctalo al puerto 11 y recuerda que es un componente polarizado.

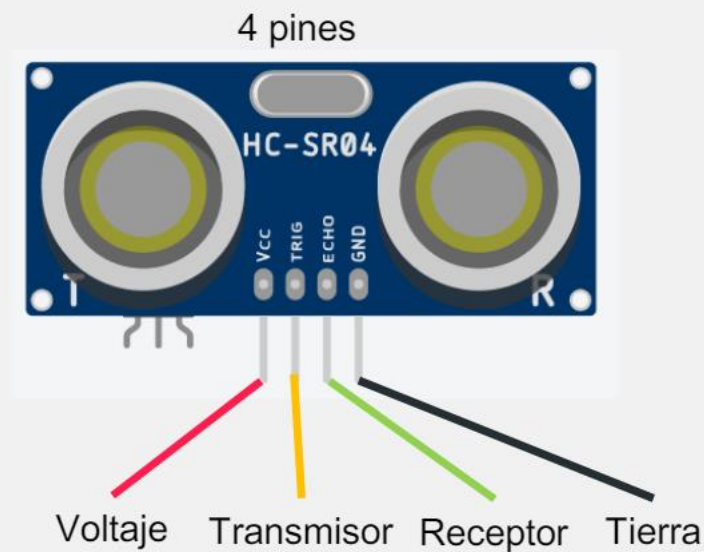


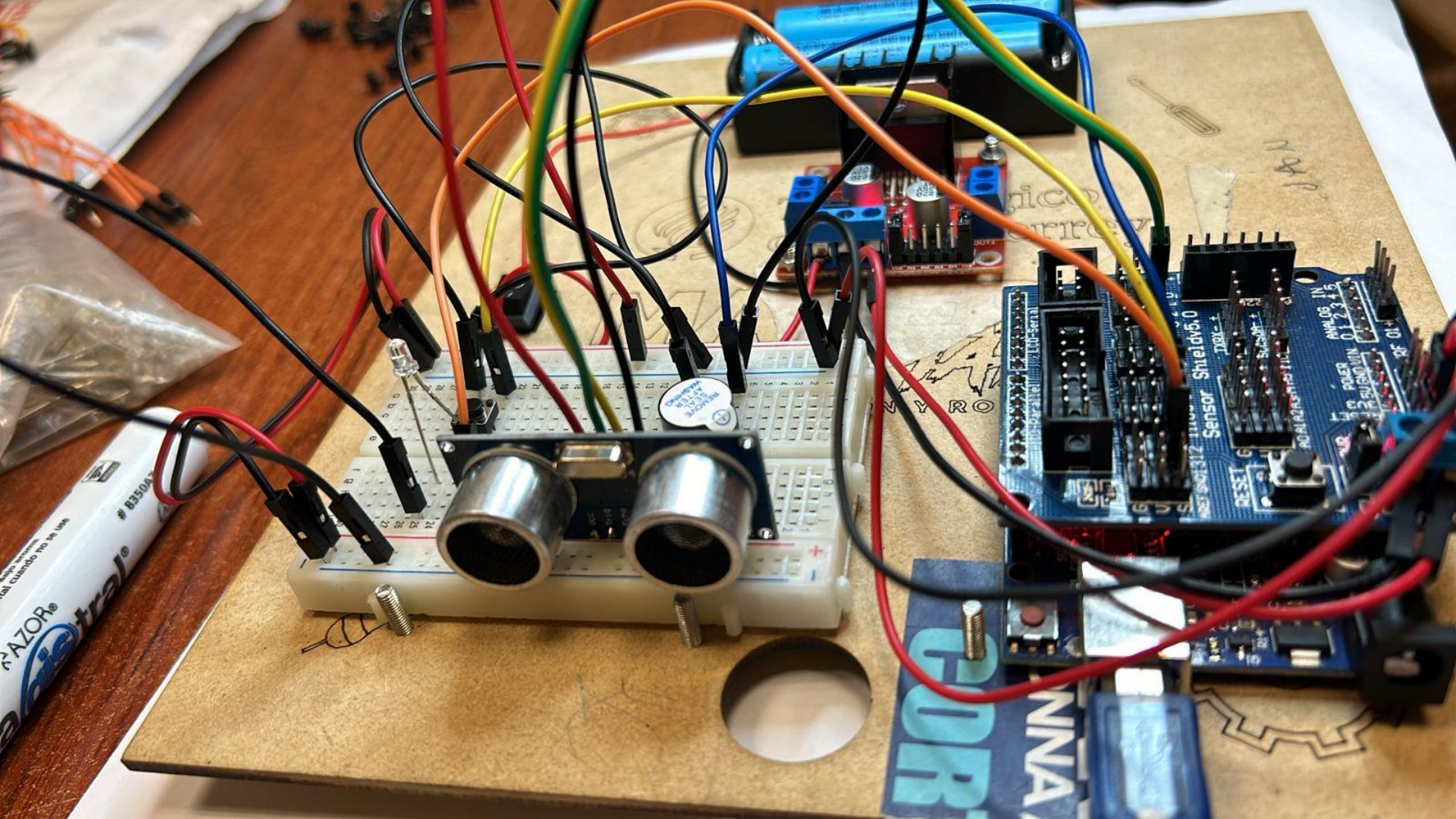
Ultrasonico



Sensor Ultrasónico

- El sensor ultrasónico consta de dos elementos. Un Transmisor (Trigger) y un receptor (Echo).





Función - ultrasonic

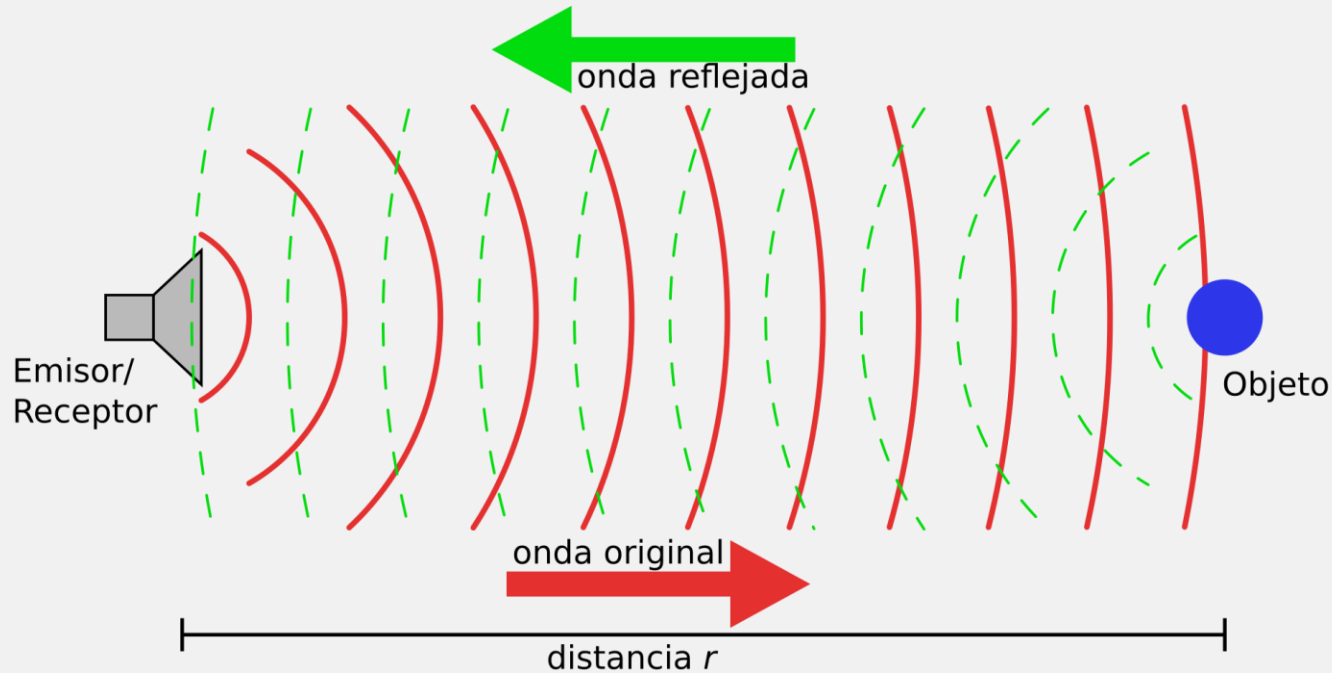
<https://github.com/Keybot-5716/ingJR/blob/main/ultra.ino>

```
int ultra()
{
    float tiempo_de_espera,distancia;
    digitalWrite (trigger,LOW); // ponemos en bajo el pin 8 durante 2 microsegundos
    delayMicroseconds(2);
    digitalWrite (trigger, HIGH); // ahora ponemos en alto pin 8 durante 10 microsegundos;
    delayMicroseconds (10); // pues este el momento en que emite el sonido durante 10 segundos
    digitalWrite (trigger, LOW); // ahora ponemos en bajo pin 8
    tiempo_de_espera = pulseIn (echo,HIGH); // pulseIn, recoge la señal del sonido que emite el trigger
    distancia =(tiempo_de_espera/2)/29.15; // formula para hallar la distancia
    Serial.print (distancia); // imprimimos la distancia en cm
    Serial.println ("cm");
    delay (1000);
    return distancia;
}
```



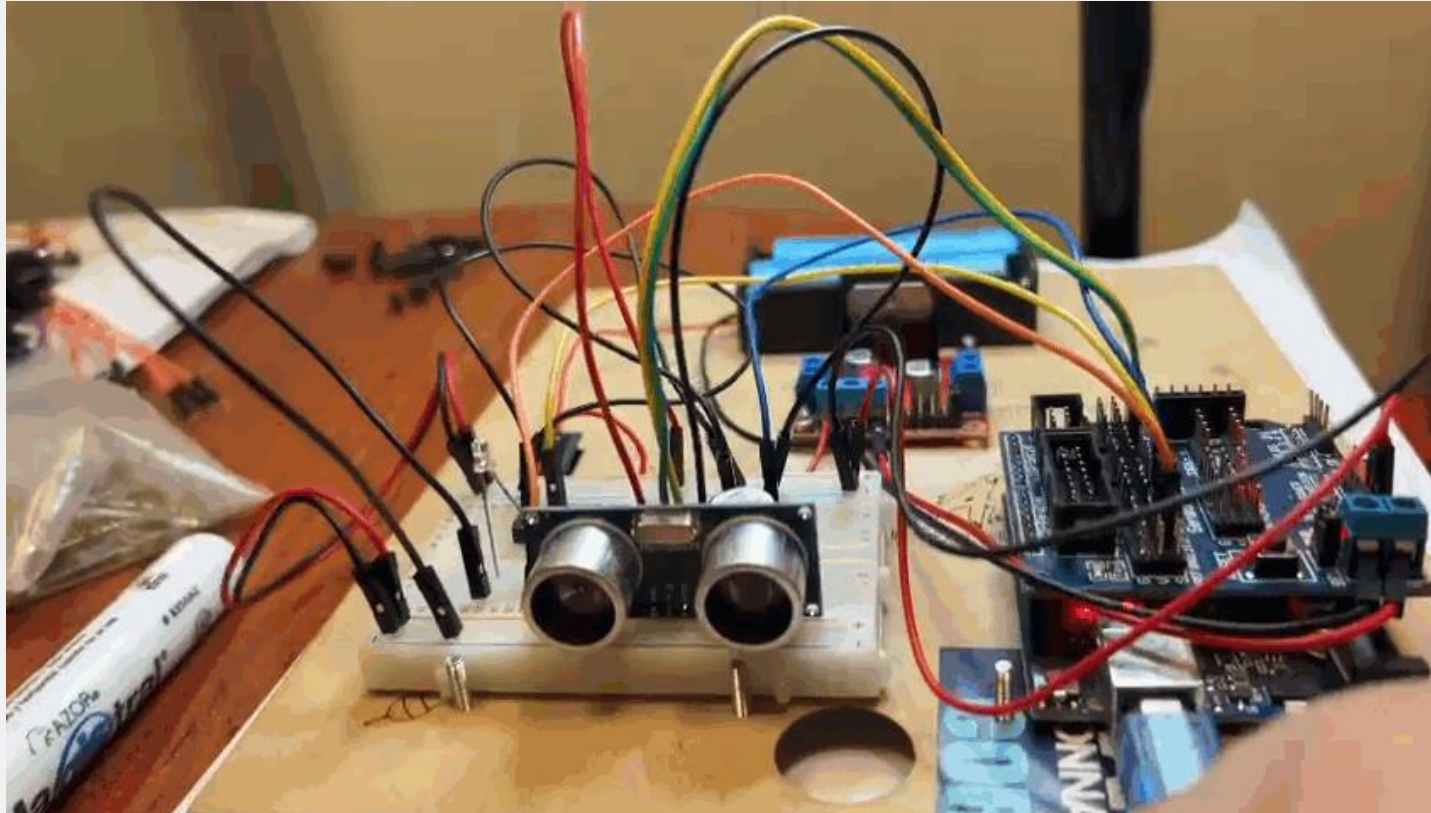
¿Cómo funciona?

29



Demo

30



Creado por:

Adriana Guadalupe Pastrana De la O

apastran@tec.mx

Estado de México

Ramiro Casas Gómez

rcasas@tec.mx

Eugenio Garza Lagüera

Alejandro Ehécatl Correa Cerón

alejandro.correacr@tec.mx

Valle Alto

Modificado por:

Luis Enrique S. García

luis.garcia@tec.mx

Hidalgo