



KIT DE INICIO OTRONIC ARDUINO UNO

Kit de inicio Arduino UNO: tutorial y 16 proyectos

Paso a paso: qué es un Arduino, qué hay en tu kit, cómo funciona Codey y 16 proyectos para construir tú mismo.

codey.online/es/starter-kits/uno/tutorial



Contenido

¡Bienvenido! ¡Tú vas a crearlo! 🚀	3
¿Qué es un Arduino UNO?	4
¿Qué hay en tu kit de inicio?	5
Breadboard	6
Cables jumper	7
LED (lucecita)	8
Resistencia 220 Ω	9
Pulsador	10
Zumbador activo	11
Sensor de luz (LDR)	12
Sensor de temperatura DHT11	13
Sensor de distancia HC-SR04	14
Sensor de movimiento (PIR)	15
Pantallita OLED 0,91"	16
Servomotor SG90	17
Anillo LED RGB (8 LEDs)	18
Relé 5V	19
Clip para batería de 9V	20
¿Qué es vibe coding?	21
Crea tu cuenta gratis	22
Así funciona Codey	23
Conectar y subir	24
¡Manos a la obra! 🎉	25
Tu primer circuito en breadboard	26
Luz intermitente	27
Semáforo	28
Botón de luz	29
Timbre	30
Anillo arcoíris	31
Mini estación meteorológica	32
Sensor de aparcamiento	33
Lamparita de noche	34
Alarma de movimiento	35
Barrera	36
Juego de reacción	37
Dado electrónico	38
Sensor de aparcamiento deluxe	39
Medidor de distancia con barra de luz	40
Ventilador inteligente	41
Barrera automática	42

KIT DE INICIO OTRONIC ARDUINO UNO

¡Bienvenido! ¡Tú vas a crearlo! 🚀

Con este kit de inicio construyes tus propios aparatos inteligentes. Una estación meteorológica, un sensor de aparcamiento, una alarma... Y lo mejor: no necesitas saber programar. Codey te ayuda.

Mira este tutorial corto un momento. Así sabrás qué hay en tu kit y qué puedes hacer con él. Después empezarás enseguida con tu primer proyecto.

Así funciona:

- 2 ¿Qué es un Arduino y qué puedes hacer con él?
- 3 Todas las piezas de tu kit, una por una
- 4 ¿Qué es vibe coding? Programar con AI
- 5 Crear tu cuenta gratis
- 6 Así funciona Codey
- 7 Conectar tu Arduino y subir tu código
- 8 Empezar con 16 proyectos

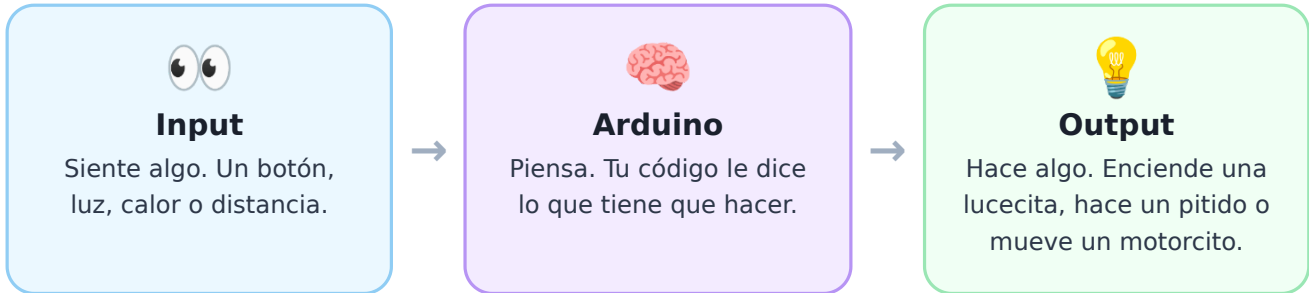


🕒 Esto dura unos 15 minutos. Siempre puedes saltarlo y volver más tarde.

¿Qué es un Arduino UNO?

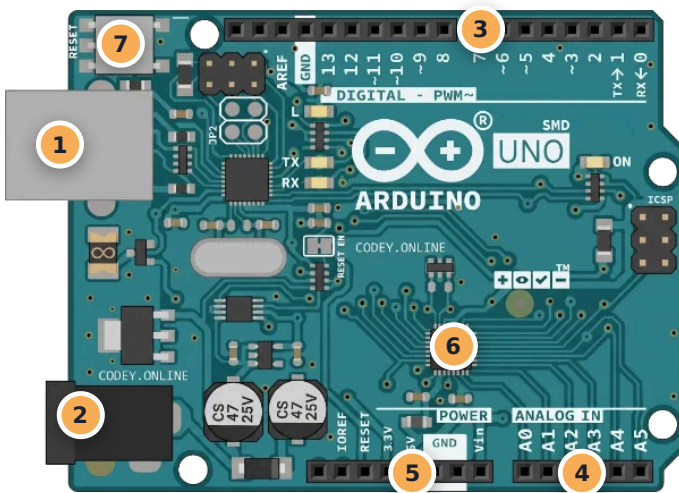
El Arduino es un ordenador pequeño. Tú le dices lo que tiene que hacer, y él lo hace. Una y otra vez, sin cansarse.

Así piensa un Arduino



Ejemplo: el sensor de distancia ve que te acercas (input). El Arduino piensa: "¡demasiado cerca!" Y el zumbador empieza a pitar (output). Así funciona un sensor de aparcamiento en un coche.

¿Qué hay en la placa?



- 1 Puerto USB: aquí va el cable a tu ordenador. Por ahí recibe energía y código.
- 2 Conector de alimentación: para una batería o adaptador (7-12V), si no usas un ordenador.
- 3 Pines digitales D0 a D13: encendido o apagado. Para lucecitas, botones y zumbadores.
- 4 Pines analógicos A0 a A5: miden cuánto. Por ejemplo, cuánta luz hay.
- 5 Pines de alimentación: 5V (más) y GND (menos) dan energía a tus componentes.
- 6 El chip: el cerebro. Aquí se guarda tu código.
- 7 Botón de reset: inicia tu código otra vez.

Dos palabras que vas a ver a menudo:

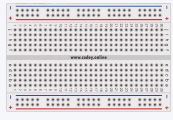
- **5V** es el más. De aquí viene la corriente (5 voltios, como un cargador de teléfono).
- **GND** (ground) es el menos. Por aquí vuelve la corriente. Cada componente necesita un cable a GND.

Los **pines** son los agujeritos de los lados. Ahí metes los cables. Un pin puede ser una **entrada** (escucha) o una **salida** (da corriente). Eso lo decides en el código.

¿Qué hay en tu kit de inicio?

Tu kit está lleno de componentes. Algunos sienten algo (sensores). Otros hacen algo (lucecitas, zumbador, motorcito). Haz clic en Siguiente y los conocerás uno por uno.

Base: con esto construyes todo



Breadboard

La placa de pruebas: construir sin soldar

En tu kit: 1x



Cables jumper

Cables Dupont: los cablecitos que conectan todo

En tu kit: 40x



Resistencia 220 Ω

Frena la corriente

En tu kit: 10x

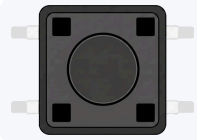


Clip para batería de 9V

Haz que tu proyecto funcione sin computadora

En tu kit: 1x

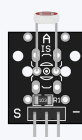
Entrada: estas piezas sienten o miden algo



Pulsador

Un botoncito para pulsar

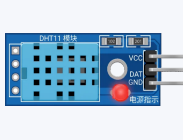
En tu kit: 5x



Sensor de luz (LDR)

Mide cuánta luz u oscuridad hay

En tu kit: 1x



Sensor de temperatura DHT11

Mide la temperatura y la humedad del aire

En tu kit: 1x



Sensor de distancia HC-SR04

Mide distancia con sonido, como un murciélago

En tu kit: 1x



Sensor de movimiento (PIR)

Ve si alguien se mueve

En tu kit: 1x

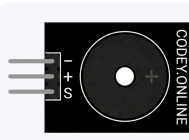
Salida: estas piezas hacen algo



LED (lucecita)

Una lucecita en 5 colores

En tu kit: 10x



Zumbador activo

Hace pitidos

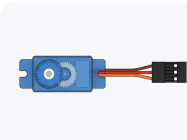
En tu kit: 1x



Pantallita OLED 0,91"

Una pantalla pequeña para texto y números

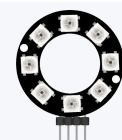
En tu kit: 1x



Servomotor SG90

Un motorcito que gira a una posición

En tu kit: 1x



Anillo LED RGB (8 LEDs)

8 lucecitas inteligentes de cualquier color

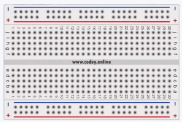
En tu kit: 1x



Relé 5V

Un interruptor que tu Arduino puede mover

En tu kit: 1x



Breadboard

La placa de pruebas: construir sin soldar

En tu kit: 1x

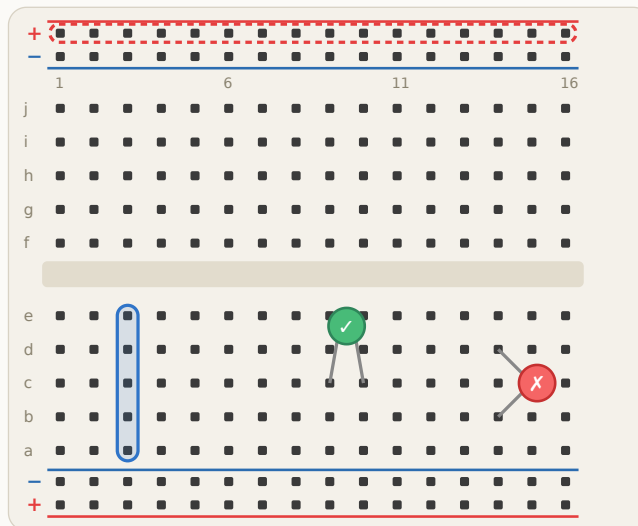
¿Qué es?

Un breadboard es una plaquita llena de agujeros. Metes piezas y cables en ella. Debajo de los agujeros hay tiras de metal que conectan algunos agujeros entre sí. Así haces un circuito sin soldar. ¿Te equivocaste? Solo sácalo y prueba otra vez.

Así lo conectas

Así funciona:

- **Filas de 5:** en la parte del medio siempre hay 5 agujeros juntos conectados (a-e y f-j). Todo lo que pongas en la misma fila queda conectado.
- **La ranura del medio** separa los dos lados. Izquierda (a-e) y derecha (f-j) *no* están conectadas.
- **Tiras de corriente (+ y -)** en los lados largos están conectadas en toda su longitud. Conecta el + a 5V y el - a GND de tu Arduino. Ahora tienes corriente por todas partes.



Tira de corriente: toda la longitud está conectada

5 agujeros en una fila están conectados

✗ Mal: las dos patitas están en la misma fila

Ranura: izquierda y derecha no están conectadas

✓ Bien: las patitas están en filas diferentes

¿Qué puedes hacer con esto?

- Probar circuitos sin soldar
- Dar corriente a varias piezas a la vez con las tiras + y -
- Poner un LED con una resistencia bien en serie

Consejo

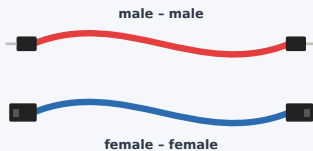
Nunca pongas las dos patitas de una pieza en la misma fila de 5. Entonces quedan conectadas entre sí y la corriente pasa por ahí. Mejor pon una pieza cruzando la ranura.

¡Pruébalo ahora mismo! En el ejercicio "Tu primer circuito en breadboard" construyes un LED con una resistencia en la breadboard y haces que se encienda y parpadee. →

Proyectos con esta pieza

★ Tu primer circuito en breadboard

Parte 2 de 15



Cables jumper

Cables Dupont: los cablecitos que conectan todo

En tu kit: 40x

🔍 ¿Qué es?

Con cables jumper conectas tu Arduino con las piezas. También se llaman **cables Dupont**, por los conectores negros de las puntas. Hay dos tipos en tu kit:

- **Male-male** (pin en los dos lados): para meter en el Arduino y en el breadboard.
- **Female-female** (agujero en los dos lados): para poner sobre los pines de un módulo.

💡 Así lo conectas

Mete el cable con firmeza en el agujero o sobre el pin. Si juntas un cable male y uno female, tienes un cable male-female: útil para conectar un módulo directamente a tu Arduino.

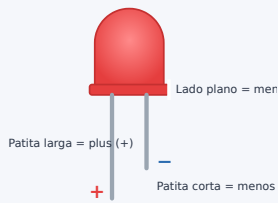
✨ ¿Qué puedes hacer con esto?

- Conectar tu Arduino con el breadboard
- Conectar módulos sensores
- Llevar corriente (5V) y GND a tus piezas

💡 Consejo

Usa colores fijos: rojo para 5V y negro para GND. Así ves enseguida si la corriente está bien.

Parte 3 de 15



LED (lucecita)

Una lucecita en 5 colores

En tu kit: 10x

¿Qué es?

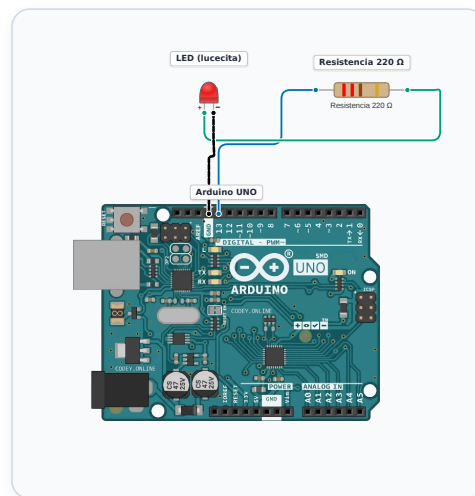
Un LED es una lucecita que usa muy poca corriente. Tu kit tiene 10 LEDs en 5 colores. Un LED funciona solo en una dirección, así que tienes que ponerlo bien.

Así lo conectas

- La **patita larga** es el plus (+). Va a un pin de tu Arduino pasando por una resistencia de 220 Ω .
- La **patita corta** es el menos (-). Va a GND. En ese lado el borde de la lucecita también es plano.

¡Usa siempre una resistencia! Sin resistencia, el LED recibe demasiada corriente y se rompe.

Esquema de conexión



- Arduino UNO · D13 → Resistencia 220 Ω · 1
- Resistencia 220 Ω · 2 → LED (lucecita) · ANODE
- LED (lucecita) · CATHODE → Arduino UNO · GND

¿Qué puedes hacer con esto?

- Una luz parpadeante
- Un semáforo
- Una lucecita que muestra si algo está encendido o apagado

Consejo

¿Tu LED no hace nada? Dale la vuelta. Solo funciona en una dirección.

Proyectos con esta pieza

★ Tu primer circuito en breadboard

★ Luz intermitente

★ Semáforo

★ Botón de luz

★★ Alarma de movimiento

★★ Juego de reacción

Parte 4 de 15

Rojo, rojo, marrón = 220 Ω

Resistencia 220 Ω

Frena la corriente

En tu kit: 10x

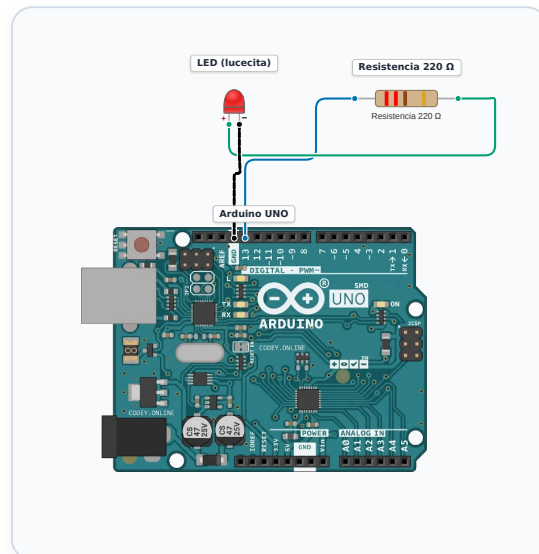
¿Qué es?

Una resistencia frena un poco la corriente. Como un grifo que cierras a medias. Así un LED recibe justo la corriente que necesita y no se rompe. Las resistencias de tu kit son de 220 ohm (Ω). Lo ves por los anillitos de colores: rojo, rojo, marrón.

Así lo conectas

Una resistencia no tiene plus ni menos: puedes ponerla en cualquier dirección. Ponla entre el pin del Arduino y la patita larga del LED. Un lado va al pin, el otro lado al LED.

Esquema de conexión



- Arduino UNO · D13 → Resistencia 220 Ω · 1
- Resistencia 220 Ω · 2 → LED (lucecita) · ANODE
- LED (lucecita) · CATHODE → Arduino UNO · GND

¿Qué puedes hacer con esto?

- Proteger cada LED
- Limitar la corriente hacia una pieza

Consejo

Una resistencia por cada LED. ¿Usas tres LEDs? Entonces también necesitas tres resistencias.

Proyectos con esta pieza

★ Tu primer circuito en breadboard

★ Luz intermitente

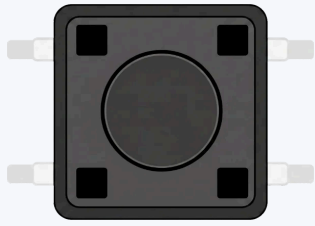
★ Semáforo

★ Botón de luz

★★ Alarma de movimiento

★★ Juego de reacción

Parte 5 de 15



Pulsador

Un botoncito para pulsar

En tu kit: 5x

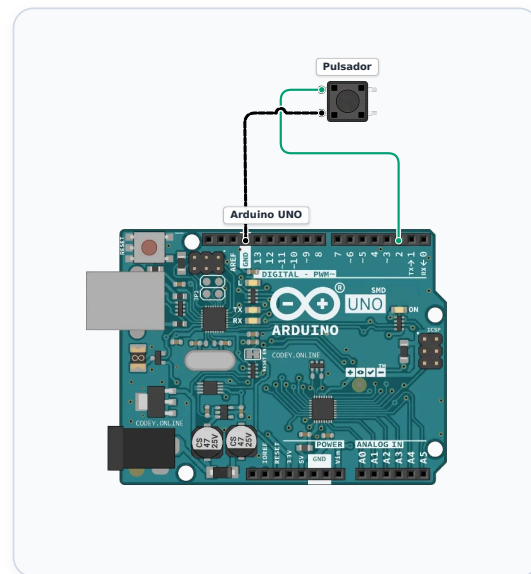
¿Qué es?

Un pulsador hace contacto mientras lo pulsas. Si lo sueltas, el contacto desaparece. Tu Arduino puede notar si el botón está pulsado. Tu kit tiene 5 botones.

Así lo conectas

El botón tiene 4 patitas. Solo necesitas 2: una a un pin (por ejemplo D2) y una a GND. En el código pones el pin en `INPUT_PULLUP`. Entonces el pin está HIGH si no pulsas y LOW si sí pulsas.

Esquema de conexión



● Arduino UNO · D2 → Pulsador · 1

● Arduino UNO · GND → Pulsador · 2

✨ ¿Qué puedes hacer con esto?

- Encender y apagar una lucecita
- Un timbre
- Un jueguito: ¿quién pulsa más rápido?

💡 Consejo

¿No sabes seguro qué patitas usar? Toma dos patitas que estén en diagonal, una frente a la otra. Así siempre estará bien. Pon el botón sobre la ranura de tu breadboard.

Proyectos con esta pieza

★ Botón de luz

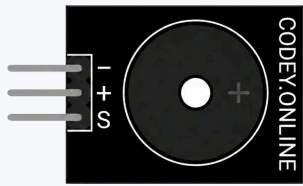
★ Timbre

★★ Barrera

★★ Juego de reacción

★★ Dado electrónico

Parte 6 de 15



Zumbador activo

Hace pitidos

En tu kit: 1x

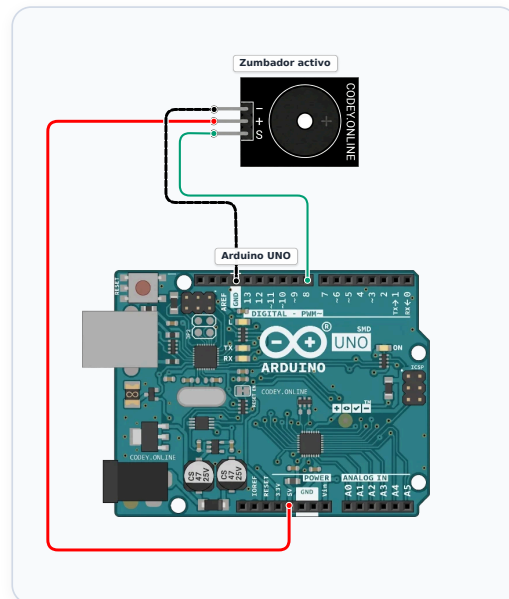
¿Qué es?

El zumbador hace un sonido de pitido. Es un zumbador **activo**: pita solo en cuanto recibe corriente. Solo tienes que encenderlo y apagarlo, igual que una lucecita.

Así lo conectas

- S (señal) a un pin, por ejemplo D8
- + a 5V
- - a GND

Esquema de conexión



● Arduino UNO · GND → Zumbador activo · -

● Arduino UNO · 5V → Zumbador activo · +

● Arduino UNO · D8 → Zumbador activo · S

¿Qué puedes hacer con esto?

- Una alarma
- Un sensor de aparcamiento que pita más rápido cuando te acercas
- Un timbre o un temporizador

Consejo

¿Quieres un pitido corto? Enciende el zumbador, espera 100 milisegundos y apágalo otra vez.

Proyectos con esta pieza

★ Timbre

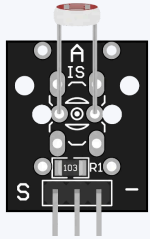
★★ Sensor de aparcamiento

★★ Alarma de movimiento

★★★★ Sensor de aparcamiento deluxe

★★★★ Barrera automática

Parte 7 de 15



Sensor de luz (LDR)

Mide cuánta luz u oscuridad hay

En tu kit: 1x

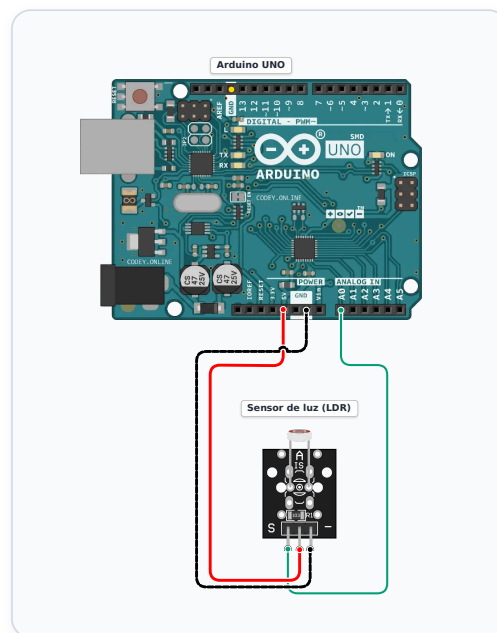
¿Qué es?

Este sensor mide cuánta luz hay. Tu Arduino lee un número entre 0 y 1023. Con mucha luz obtienes un número distinto que en la oscuridad. Así tu Arduino sabe si es de día o de noche.

Así lo conectas

- S a un pin analógico, por ejemplo A0
- + a 5V
- - a GND

Esquema de conexión



● Arduino UNO · A0 → Sensor de luz (LDR) · S

● Arduino UNO · 5V → Sensor de luz (LDR) · +

● Arduino UNO · GND → Sensor de luz (LDR) · -

¿Qué puedes hacer con esto?

- Una luz de noche que se enciende sola
- Un medidor de luz
- Una alarma que suena si alguien abre el cajón

Consejo

Mira en el monitor serie qué número ves. Luego pon tu mano sobre el sensor. Así ves qué número corresponde a "oscuro".

Proyectos con esta pieza

★★ **Lamparita de noche**

Parte 8 de 15



Sensor de temperatura DHT11

Mide la temperatura y la humedad del aire

En tu kit: 1x

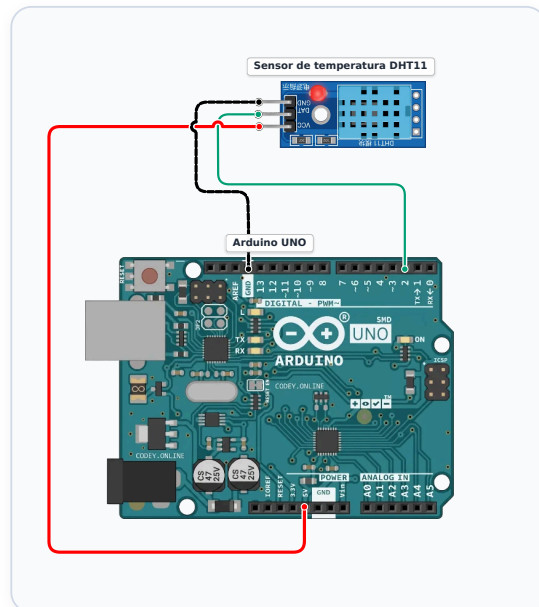
¿Qué es?

El DHT11 mide cuánto calor hace (en grados Celsius) y cuánta humedad hay en el aire (en porcentaje). Da una nueva medición cada 2 segundos.

Así lo conectas

- **DAT** (datos) a un pin, por ejemplo D2
- **VCC** a 5V
- **GND** a GND

Esquema de conexión



- Arduino UNO · 5V → Sensor de temperatura DHT11 · VCC
- Arduino UNO · GND → Sensor de temperatura DHT11 · GND
- Arduino UNO · D2 → Sensor de temperatura DHT11 · DAT

¿Qué puedes hacer con esto?

- Una mini estación meteorológica
- Un ventilador que se enciende si hace demasiado calor
- Un aviso si hay demasiada humedad en el baño

Consejo

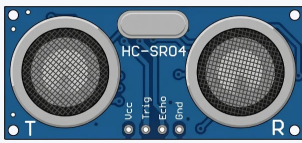
Sopla sobre el sensor. ¡Entonces verás que la humedad sube enseguida!

Proyectos con esta pieza

★★★ Mini estación meteorológica

★★★★ Ventilador inteligente

Parte 9 de 15



Sensor de distancia HC-SR04

Mide distancia con sonido, como un murciélago

En tu kit: 1x

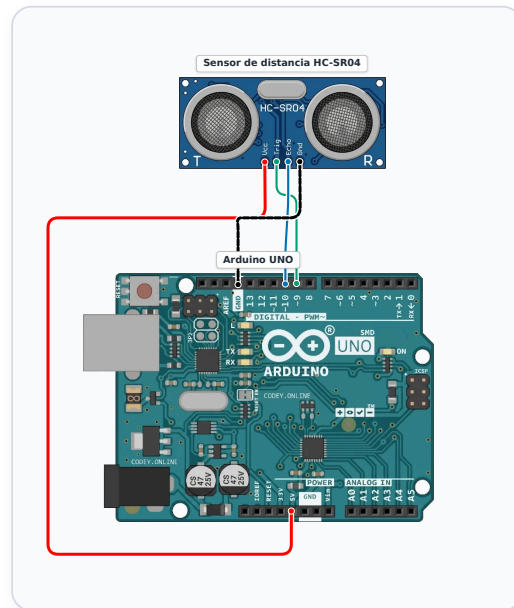
¿Qué es?

Este sensor envía un sonido que tú no puedes oír. El sonido choca contra algo y vuelve. El sensor mide cuánto tarda. Así Arduino sabe a qué distancia está algo: desde 2 cm hasta unos 4 metros.

Así lo conectas

- **Trig** (enviar) a D9
- **Echo** (recibir) a D10
- **Vcc** a 5V
- **Gnd** a GND

Esquema de conexión



- Arduino UNO · 5V → Sensor de distancia HC-SR04 · Vcc
- Arduino UNO · GND → Sensor de distancia HC-SR04 · Gnd
- Arduino UNO · D9 → Sensor de distancia HC-SR04 · Trig
- Arduino UNO · D10 → Sensor de distancia HC-SR04 · Echo

🌟 ¿Qué puedes hacer con esto?

- Un sensor de aparcamiento
- Un medidor de distancia
- Una barrera que se abre sola

💡 Consejo

Los dos "ojos" son el altavoz y el micrófono. Apúntalos recto hacia lo que quieres medir.

Proyectos con esta pieza

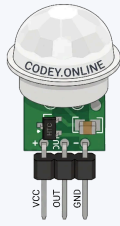
★★★ Sensor de aparcamiento

★★★★ Sensor de aparcamiento deluxe

★★★★ Medidor de distancia con barra de luz

★★★★ Barrera automática

Parte 10 de 15



Sensor de movimiento (PIR)

Ve si alguien se mueve

En tu kit: 1x

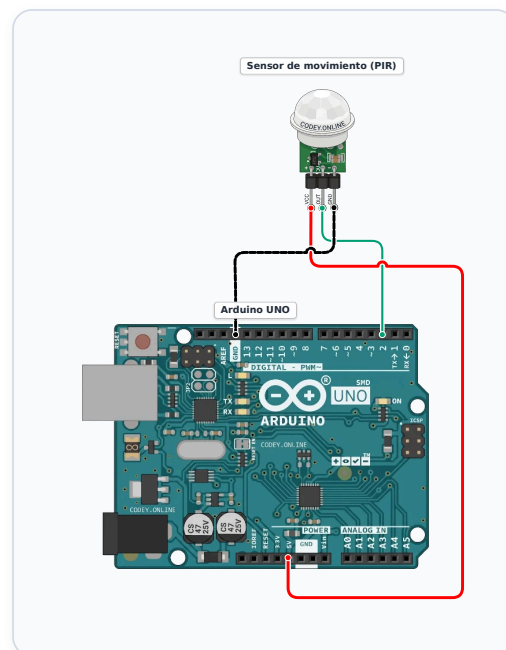
¿Qué es?

Un sensor PIR siente el calor de personas y animales. ¿Se mueve algo caliente delante del sensor? Entonces su salida se pone HIGH. Así funcionan también las lámparas que se encienden solas cuando pasas.

Así lo conectas

- **OUT** a un pin, por ejemplo D2
- **VCC** a 5V
- **GND** a GND

Esquema de conexión



- Arduino UNO · 5V → Sensor de movimiento (PIR) · VCC
- Arduino UNO · GND → Sensor de movimiento (PIR) · GND
- Arduino UNO · D2 → Sensor de movimiento (PIR) · OUT

✨ ¿Qué puedes hacer con esto?

- Una alarma antirrobo
- Una lámpara que se enciende cuando entras
- Contar cuántas veces pasa alguien

💡 Consejo

Después de encenderlo, el sensor necesita unos segundos para acostumbrarse a la habitación. Quédate quieto un momento.

Proyectos con esta pieza

★★ Alarma de movimiento

Parte 11 de 15



Pantallita OLED 0,91"

Una pantalla pequeña para texto y números

En tu kit: 1x

¿Qué es?

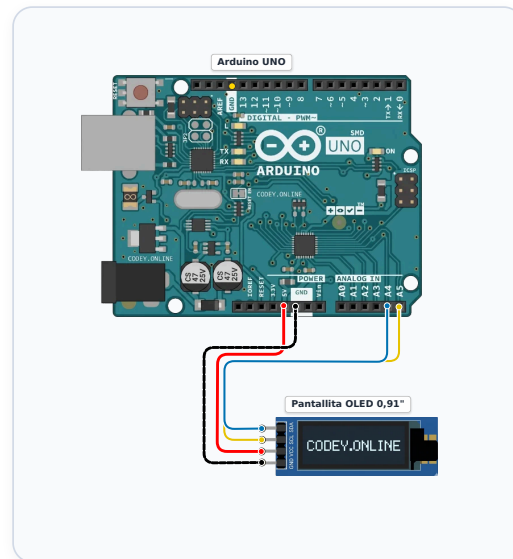
Una pantallita pequeña y nítida de 128 por 32 pixels. Puedes poner texto, números y dibujos sencillos. Por ejemplo la temperatura o una distancia.

Así lo conectas

La pantalla habla por **I2C**: solo dos cables de datos.

- **SDA** a A4
- **SCL** a A5
- **VCC** a 5V
- **GND** a GND

Esquema de conexión



- Arduino UNO · A4 → Pantallita OLED 0,91" · SDA
- Arduino UNO · A5 → Pantallita OLED 0,91" · SCL
- Arduino UNO · 5V → Pantallita OLED 0,91" · VCC
- Arduino UNO · GND → Pantallita OLED 0,91" · GND

✨ ¿Qué puedes hacer con esto?

- Mostrar la temperatura
- Mostrar la distancia en centímetros
- Mostrar una puntuación o el tiempo de reacción

💡 Consejo

¿La pantalla sigue negra? Comprueba que SDA y SCL no estén al revés.

Proyectos con esta pieza

★★★ Mini estación meteorológica

★★★ Juego de reacción

★★★★ Sensor de aparcamiento deluxe

★★★★ Ventilador inteligente

Parte 12 de 15



Servomotor SG90

Un motorcito que gira a una posición

En tu kit: 1x

¿Qué es?

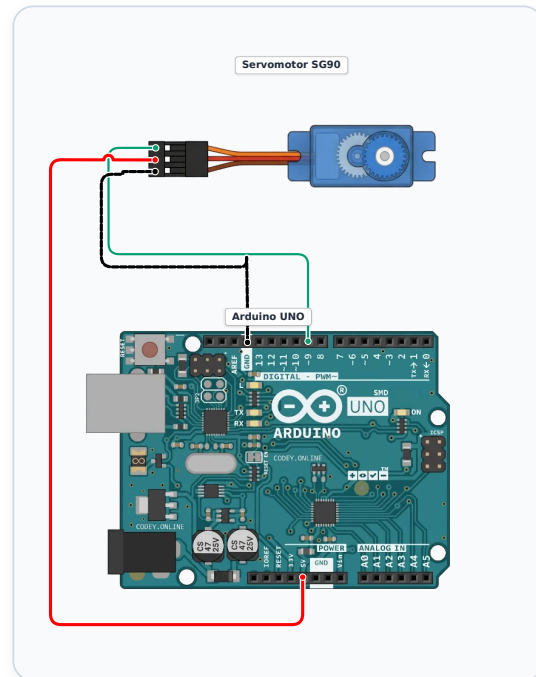
Un servo es un motorcito que puedes girar a una posición exacta, de 0 a 180 grados. Por ejemplo, le dices "ve a 90 grados" y va justo allí. En el eje pones un bracito.

Así lo conectas

El servo tiene tres cablecitos:

- **Naranja o amarillo** (señal) a un pin, por ejemplo D9
- **Rojo** a 5V
- **Marrón** a GND

Esquema de conexión



● Arduino UNO · GND → Servomotor SG90 · GND

● Arduino UNO · 5V → Servomotor SG90 · VCC

● Arduino UNO · D9 → Servomotor SG90 · PWM

✨ ¿Qué puedes hacer con esto?

- Una barrera
- Una aguja que señala la temperatura
- Un bracito de robot que saluda

💡 Consejo

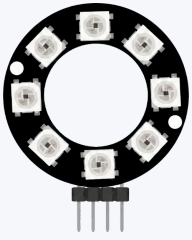
No gires el bracito con la mano. Entonces se pueden romper los engranajes de dentro.

Proyectos con esta pieza

★★★ Barrera

★★★★ Barrera automática

Parte 13 de 15



Anillo LED RGB (8 LEDs)

8 lucecitas inteligentes de cualquier color

En tu kit: 1x

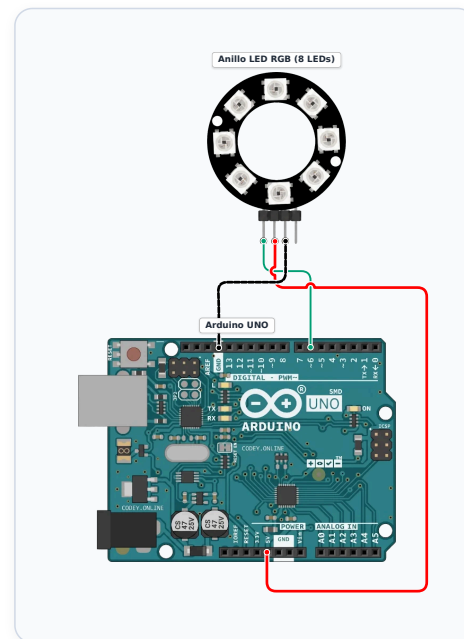
¿Qué es?

Un anillo con 8 lucecitas inteligentes. Cada lucecita puede ponerse de cualquier color, y las controlas una por una. ¡Con solo un cable de datos! Es súper divertido para efectos de luz.

Así lo conectas

- **DI** (entrada de datos) a un pin, por ejemplo D6
- **5V** a 5V
- **GND** a GND
- **DO** lo dejas suelto (ahí puedes poner otro anillo detrás)

Esquema de conexión



● Arduino UNO · 5V → Anillo LED RGB (8 LEDs) · 5V

● Arduino UNO · GND → Anillo LED RGB (8 LEDs) · GND

● Arduino UNO · D6 → Anillo LED RGB (8 LEDs) · DI

✨ ¿Qué puedes hacer con esto?

- Una lámpara arcoíris
- Una barra de luz que muestra lo cerca que está algo
- Un dado de luz

💡 Consejo

No pongas el brillo muy alto en el código. Es mejor para tus ojos y para tu Arduino.

Proyectos con esta pieza

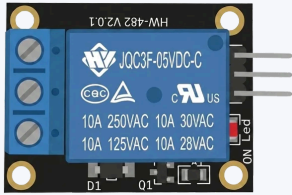
★ Anillo arcoíris

★★ Lamparita de noche

★★ Dado electrónico

★★★★ Medidor de distancia con barra de luz

Parte 14 de 15



Relé 5V

Un interruptor que tu Arduino puede mover

En tu kit: 1x

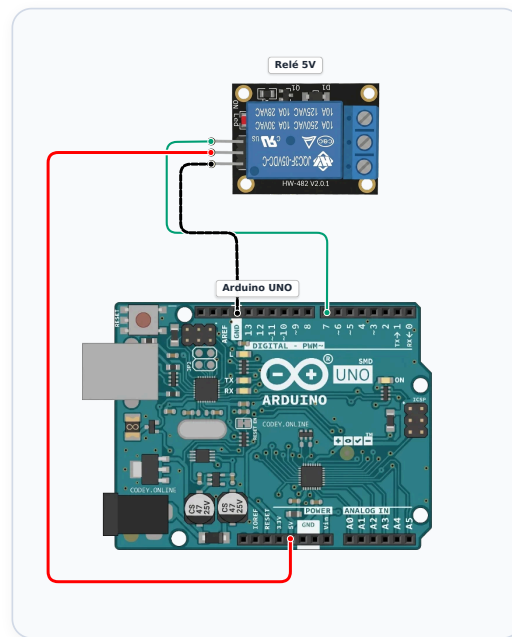
¿Qué es?

Un relé es un interruptor que tu Arduino puede encender y apagar. Lo oyes hacer "clic". Con un relé, tu Arduino puede encender y apagar otro aparato, por ejemplo un ventilador pequeño o una lámpara con batería.

Así lo conectas

- S a un pin, por ejemplo D7
- + a 5V
- - a GND
- El aparato que quieres encender y apagar lo conectas a **COM** y **NO**. NO significa: normalmente apagado, encendido cuando el relé hace clic.

Esquema de conexión



● Arduino UNO · GND → Relé 5V · -

● Arduino UNO · 5V → Relé 5V · +

● Arduino UNO · D7 → Relé 5V · S

¿Qué puedes hacer con esto?

- Un ventilador que se enciende cuando hace calor
- Una lámpara con temporizador
- Una bombita de agua para las plantas

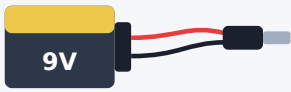
Consejo

La seguridad es lo primero: usa el relé solo con baterías o baja tensión (hasta 12V). ¡Nunca con el enchufe (230V)!

Proyectos con esta pieza

★★★★ Ventilador inteligente

Parte 15 de 15



Clip para batería de 9V

Haz que tu proyecto funcione sin computadora

En tu kit: 1x

🔍 ¿Qué es?

Con este clip pones una batería de 9V en tu Arduino. Así tu proyecto también funciona sin computadora. Muy práctico si quieres enseñarlo o ponerlo en algún lugar.

🔌 Así lo conectas

Conecta la batería al clip y mete el conector redondo en la toma negra de corriente de tu Arduino. Tu código sigue en el Arduino, así que empieza enseguida.

✨ ¿Qué puedes hacer con esto?

- Colgar tu sensor de aparcamiento en el garaje
- Poner tu estación meteorológica en el alféizar de la ventana
- Enseñar tu proyecto a tus amigos

💡 Consejo

Quita la batería cuando termines. Así durará mucho más.

¿Qué es vibe coding?

Antes tenías que aprender a programar durante meses. Ahora solo cuentas con tus propias palabras lo que quieres hacer. La IA escribe el código por ti.

Eso se llama **vibe coding**: tú tienes la idea, la IA hace el trabajo de escribir. Por ejemplo, dices: *"Haz una luz de noche que se encienda cuando se haga oscuro."* En unos pocos segundos el código está listo, con un esquema de conexiones incluido.

La IA de ahora programa muy bien. Muchas veces incluso mejor y más rápido que una persona. Conoce todos los componentes, hace menos errores al escribir y explica todo con calma.

Construir un proyecto: antes y ahora

Programar tú mismo

Investigar, escribir, buscar errores, volver a intentarlo...

± 10 uur

Con Codey

Contar lo que quieres, subirlo, listo.

± 5 minuten ↗

Así funciona



Tú cuentas lo que quieres hacer.



Codey escribe el código y dibuja cómo conectar todo.



Tú lo construyes igual y subes el código.



¿Quieres otra cosa? Dilo, y Codey lo cambia.

¡También aprendes con esto! Codey explica qué hace el código. Así entiendes cada vez mejor cómo funciona.

Crea tu cuenta gratis

Para trabajar con Codey necesitas una cuenta. Es gratis y solo tarda un minutito. Con tu kit de inicio, después puedes usar Codey sin límite.

1 Regístrate

Haz clic en "Crear cuenta" y escribe tu correo electrónico y una contraseña. ¿Tienes una cuenta de Google? Entonces haz clic en "Continuar con Google" y ya estará listo.

2 Confirma tu correo electrónico

Recibirás un correo de Codey Online. Haz clic en el enlace de ese correo. ¿No ves ningún correo? Mira también en tu correo no deseado.

3 Inicia sesión

Inicia sesión con tu correo electrónico y tu contraseña. Entonces volverás automáticamente a este tutorial, en el siguiente paso.

Ya tengo una cuenta: iniciar sesión →

👉 Ve a codey.online/uno para crear tu cuenta gratis.

👤 ¿Aún eres pequeño? Entonces crea la cuenta junto con tu padre, tu madre u otro adulto.

Así funciona Codey

Codey es tu compañero para programar. Todo está en una sola pantalla. Estos son los botones más importantes.

The screenshot shows the Codey Online web interface. On the left is a sidebar with a chat window, project archives, and navigation buttons. The main area displays an Arduino IDE with a sketch named 'sketch.ino'. The code is for a mini weather station using an Adafruit SSD1306 display and a DHT11 sensor. Annotations are placed over the interface: 1 points to the chat window, 2 points to the code editor, 3 points to the 'Compile and upload to board' button, 4 points to a connection diagram, 5 points to the 'Monitor serial' button, and 6 points to the 'My kit' button.

1 El chat

Aquí hablas con Codey. Escribe lo que quieres hacer o cambiar, con palabras normales.

2 El código

Aquí está el código que Codey escribe para ti. Tú también puedes escribir en él.

3 Compilar y subir a la placa

Un botón para todo: Codey revisa tu código, lo prepara (compilar) y lo envía por el cable USB a tu Arduino (subir). ¿Hay un error? Entonces Codey muchas veces lo arregla solo.

4 Esquema de conexión

Muestra qué cable va en cada sitio. Lo ves en el chat; haz clic en él para verlo grande.

5 Monitor serie

Aquí ves los mensajes que envía tu Arduino, por ejemplo la temperatura o una distancia.

6 Mi kit

Todas las piezas de tu starterkit. Codey solo usa cosas que tú tienes.

Bueno saberlo: con tu starterkit puedes usar Codey sin límite en tus chats del kit. Puedes preguntarle cualquier cosa a Codey, también "¿qué hace esta línea?" o "¿por qué no funciona?".

Conectar y subir

¿Tu código está listo? Entonces ahora va a tu Arduino. Lo haces en tres pasos.

Importante: usa **Google Chrome** o **Microsoft Edge** en tu computadora. Solo esos navegadores pueden hablar con tu Arduino por USB. Safari, Firefox y la mayoría de los teléfonos (todavía) no pueden hacerlo.

1 Conecta tu Arduino

Pon el cable USB en tu Arduino y en tu computadora. Se encenderá una lucecita verde (ON).

2 Haz clic en Compilar y subir a la placa

Codey revisa tu código y lo prepara (compilar). La primera vez a veces tarda un minutito.

3 Elige tu Arduino en la ventana de puertos

Aparece una ventana con puertos. Elige tu Arduino y haz clic en Conectar. Las lucecitas RX y TX parpadean un momento: tu código está entrando (subiendo). ¡Listo!

¿Cómo se llama mi Arduino en la ventana de puertos?

Eso depende de tu computadora. ¿Ves varios? Saca el cable un momento y mira cuál desaparece. Ese es tu Arduino.

Windows

Un **puerto COM** con un número, por ejemplo `Arduino Uno (COM4)` o `USB-SERIAL CH340 (COM5)`. COM1 casi nunca es tu Arduino.

Mac

Algo con `usbmodem` o `usbserial`, por ejemplo `/dev/cu.usbmodem14101`, `cu.usbserial-1420` o `cu.wchusbserial1420`. Elige siempre la versión `cu.`.

Linux

`/dev/ttyACM0` o `/dev/ttyUSB0`. ¿No tienes acceso? Añádate al grupo `dialout` e inicia sesión de nuevo.

Chromebook

Chrome muestra el nombre del dispositivo, por ejemplo `Arduino Uno`, `USB2.0-Serial` o `USB Serial`.

Monitor serie: algunos proyectos envían mensajes a tu computadora, por ejemplo la temperatura. Los ves si haces clic en Monitor serie en Codey. Así ves lo que tu Arduino "piensa".

¿No funciona? ¡No te preocupes!

▼ No veo mi Arduino en la ventana de puertos.

Prueba con otro cable USB. Algunos cables solo sirven para cargar y no pueden enviar datos. Prueba también otro puerto USB de tu computadora.

▼ Veo CH340 o USB2.0-Serial, pero no puedo subir el código (Windows/Mac).

Entonces tu Arduino usa un chip CH340. Instala una sola vez el driver gratuito CH340 (busca "CH340 driver") y reinicia tu computadora.

▼ La subida falla a la mitad.

Cierra otros programas que usen el Arduino, como Arduino IDE o una segunda pestaña de Codey. Saca el cable un momento, vuelve a conectarlo y prueba otra vez.

▼ Funciona, pero mi circuito no hace nada.

Revisa los cables con el esquema de conexión. ¿5V y GND están bien puestos? Después pregúntale a Codey: "Mi LED no se enciende, ¿qué puede ser?"

¡Manos a la obra!

Elige un proyecto y haz clic en HAZ ESTO. Codey prepara el código y el esquema de conexión para ti. Empieza con un proyecto de una estrella ★ si nunca has hecho nada con Arduino.



★ ⌚ 10 minutos

Tu primer circuito en breadboard

Practica con la breadboard: enciende un LED y luego hazlo parpadear.



★ ⌚ 10 minutos

Luz intermitente

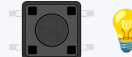
Tu primer proyecto: hacer parpadear un LED.



★ ⌚ 15 minutos

Semáforo

Tres LEDs que funcionan como un semáforo de verdad.



★ ⌚ 15 minutos

Botón de luz

Pulsa el botón y la lámpara se enciende. Otra vez: se apaga.



★ ⌚ 10 minutos

Timbre

Pulsa el botón para un alegre ding-dong.



★ ⌚ 10 minutos

Anillo arcoíris

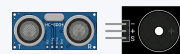
El anillo LED gira con todos los colores del arcoíris.



★★ ⌚ 20 minutos

Mini estación meteorológica

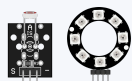
Temperatura y humedad en tu propia pantallita.



★★ ⌚ 20 minutos

Sensor de aparcamiento

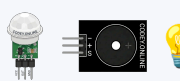
Cuanto más cerca, más rápido pita. Como en un coche.



★★ ⌚ 15 minutos

Lamparita de noche

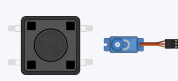
¿Se oscurece? Entonces la lámpara se enciende sola.



★★ ⌚ 20 minutos

Alarma de movimiento

¿Pasa alguien? ¡Entonces suena la alarma!



★★ ⌚ 20 minutos

Barrera

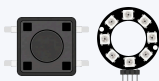
Pulsa el botón y la barrera se abre.



★★ ⌚ 25 minutos

Juego de reacción

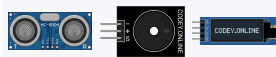
¿Qué tan rápido eres? Mide tu tiempo de reacción.



★★ ⌚ 15 minutos

Dado electrónico

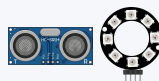
Pulsa el botón y tira un dado de luz.



★★★ ⌚ 30 minutos

Sensor de aparcamiento deluxe

Pitidos y la distancia en centímetros en una pantallita.



★★★ ⌚ 25 minutos

Medidor de distancia con barra de luz

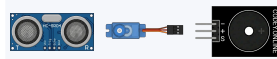
Cuanto más cerca, más lucecitas: verde, naranja, rojo.



★★★ ⌚ 30 minutos

Ventilador inteligente

¿Demasiado calor? Entonces el relé enciende el ventilador.



★★★ ⌚ 30 minutos

Barrera automática

¿Viene un coche? Entonces la barrera se abre sola.

¿Tienes una idea propia? ¡Mucho mejor! Abre Codey y cuéntale qué quieres hacer. Por ejemplo: "Haz una lucecita que parpadee más rápido cuando haga más calor."

★ Principiante ⌚ 10 minutos

Tu primer circuito en breadboard

Un ejercicio para descubrir cómo funciona una breadboard. Montas un LED con una resistencia en la breadboard. Primero lo enciendes con la tira de 5V, sin código. Después lo haces parpadear con pin 13.

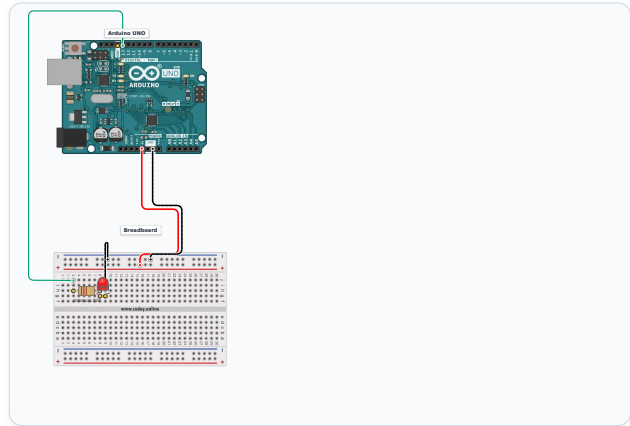
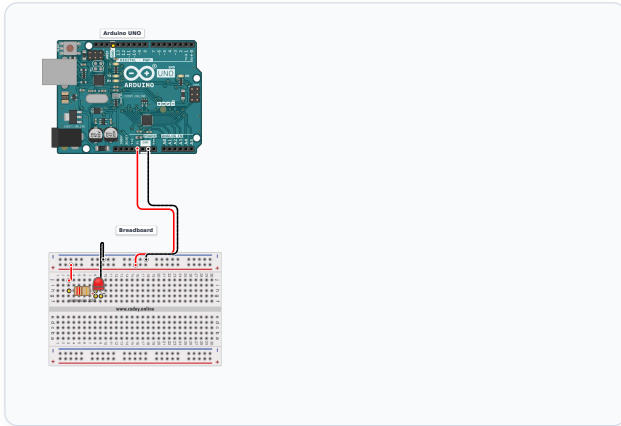
Esquema de conexión

🔧 Paso A: enciende el LED (sin código)

El cable rojo va de la tira + a la fila 3. La corriente va así: 5V → tira + → resistencia → LED → tira - → GND. ¿Se enciende el LED? ¡Entonces has usado bien la breadboard!

💡 Paso B: haz parpadear el LED con pin 13

Quita el cable rojo entre la tira + y la fila 3. Pon en su lugar un cable de pin 13 a la fila 3. Ahora el Arduino decide cuándo recibe corriente el LED.



🛒 ¿Qué necesitas?



1x Arduino UNO



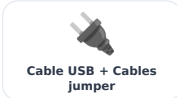
1x Breadboard



1x LED (lucecita)



1x Resistencia 220 Ω



Cable USB + Cables jumper

🔧 Así lo construyes

1. Conecta 5V del Arduino con la tira + de arriba de la breadboard (cable rojo).
2. Conecta GND del Arduino con la tira - de al lado (cable negro). Ahora tienes dos tiras de corriente.
3. Pon la resistencia de 220 Ω desde el agujero H3 hasta el agujero H8. Así queda en dos filas diferentes.
4. Pon la patita larga del LED en G8 (la misma fila que la resistencia) y la patita corta en G9.
5. Conecta el agujero J9 con la tira - (cable negro). Así la corriente puede volver a GND.
6. Paso A: conecta la tira + con el agujero J3 (cable rojo). ¡El LED se enciende!
7. Paso B: quita ese cable rojo y conecta pin 13 con el agujero J3 (cable verde).
8. Deja que Codey haga el código: haz clic en MAAK DIT, o escribe tú en el chat: "Haz que el LED en pin 13 parpadee cada segundo." Después haz clic en Compilar y subir a la placa.

💻 El código

```
// Tu primer circuito en breadboard
// El LED en la breadboard se enciende y se apaga. Recibe corriente por pin 13.

const int LED_PIN = 13; // El cable verde va de pin 13 a la breadboard

void setup() {
  pinMode(LED_PIN, OUTPUT); // Pin 13 es una salida: da corriente
}

void loop() {
  digitalWrite(LED_PIN, HIGH); // LED encendido
  delay(1000); // esperar 1 segundo
  digitalWrite(LED_PIN, LOW); // LED apagado
  delay(1000); // esperar 1 segundo
}
```

🧠 ¿Qué hace el código?

- Pin 13 es una salida. Con **HIGH** da 5V, con **LOW** no da nada.
- En `loop()` encendemos el LED, esperamos 1 segundo, lo apagamos y volvemos a esperar.
- La corriente pasa por la breadboard: pin 13 → fila 3 → resistencia → fila 8 → LED → fila 9 → tira - → GND.

🔗 Prueba esto...

- Haz que el LED parpadee el doble de rápido
- Añade un segundo LED en la breadboard que parpadee por turnos
- Haz que el LED parpadee tres veces rápido y luego se quede apagado un momento

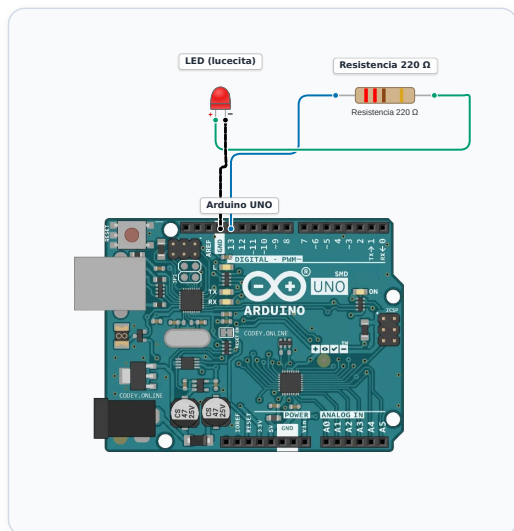
🔧 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/breadboard-led y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ Principiante ⌚ 10 minutos

Luz intermitente

¡Tu primer proyecto! Un LED se enciende y se apaga, como una luz intermitente. Así aprendes cómo Arduino enciende y apaga una salida.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Pon el LED en el breadboard, con las patitas en dos filas diferentes.
2. Pon una resistencia de 220 Ω desde la fila de la patita larga hasta otra fila.
3. Conecta esa otra fila con un cable a pin 13 de tu Arduino.
4. Conecta la patita corta con un cable a GND.

El código

```
// Luz intermitente
// Un LED se enciende y se apaga, como una luz intermitente.

const int LED_PIN = 13; // El LED está en pin 13 (con la resistencia de 220 ohm)
const int TIEMPO_ESPERA = 500; // Cuánto tiempo queda encendido o apagado el LED (en milisegundos)

void setup() {
  // setup() se ejecuta una vez cuando Arduino arranca
  pinMode(LED_PIN, OUTPUT); // Pin 13 es una salida: da corriente
}

void loop() {
  // loop() se repite una y otra vez, para siempre
  digitalWrite(LED_PIN, HIGH); // LED encendido
  delay(TIEMPO_ESPERA); // Esperar un poco
  digitalWrite(LED_PIN, LOW); // LED apagado
  delay(TIEMPO_ESPERA); // Esperar un poco
}
```

¿Qué hace el código?

- En `setup()` decimos que pin 13 es una salida.
- En `loop()` encendemos el LED, esperamos un poco, lo apagamos y volvemos a esperar.
- Después `loop()` empieza de nuevo. Así el LED sigue parpadeando.

Prueba esto...

- Haz que el LED parpadee el doble de rápido
- Haz una señal SOS: corto-corto-corto, largo-largo-largo, corto-corto-corto
- Haz que el LED se encienda y se apague suave, como si respirara

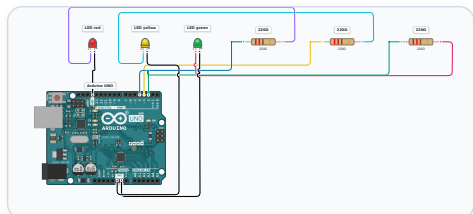
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/blink y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ Principiante ⌚ 15 minutos

Semáforo

Construyes un semáforo de verdad con un LED rojo, uno amarillo y uno verde. Se encienden por turnos, igual que en la calle.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Pon los tres LEDs uno al lado del otro en el breadboard.
2. Dale a cada LED su propia resistencia de 220 Ω hacia su pin: rojo a 4, amarillo a 3, verde a 2.
3. Conecta todas las patitas cortas con la tira negativa (-) del breadboard.
4. Conecta la tira negativa con GND de tu Arduino.

El código

```
// Semáforo
// Tres LEDs hacen justo lo que hace un semáforo de verdad: verde, amarillo, rojo.

const int ROJO = 4;    // LED rojo en pin 4
const int AMARILLO = 3; // LED amarillo/naranja en pin 3
const int VERDE = 2;   // LED verde en pin 2

void setup() {
  pinMode(ROJO, OUTPUT);
  pinMode(AMARILLO, OUTPUT);
  pinMode(VERDE, OUTPUT);
}

// Enciende una lucécita y apaga las otras dos
void ponerLuz(int encendida) {
  digitalWrite(ROJO, encendida == ROJO ? HIGH : LOW);
  digitalWrite(AMARILLO, encendida == AMARILLO ? HIGH : LOW);
  digitalWrite(VERDE, encendida == VERDE ? HIGH : LOW);
}

void loop() {
  ponerLuz(VERDE); // Verde: puedes pasar
  delay(5000);
  ponerLuz(AMARILLO); // Amarillo: cuidado, se pondrá rojo
  delay(2000);
  ponerLuz(ROJO); // Rojo: ¡para!
  delay(5000);
}
```

¿Qué hace el código?

- Cada LED tiene su propio pin: rojo en 4, amarillo en 3 y verde en 2.
- La función `ponerLuz()` enciende una lucécita y apaga las otras dos.
- En `loop()` vamos de verde a amarillo y luego a rojo, con un tiempo de espera entre ellos.

Prueba esto...

- Haz que la lucécita amarilla parpadee tres veces antes de ponerse en rojo
- Añade un botón: si lo pulsas, la luz cambia más rápido a rojo (un botón de peatón)
- Haz un modo noche en el que solo parpadee la lucécita amarilla

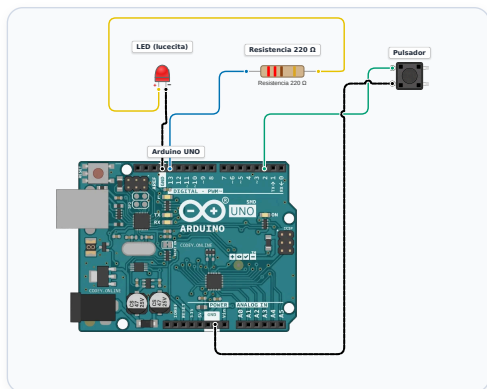
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/traffic-light y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ Principiante ⌚ 15 minutos

Botón de luz

Vas a hacer un botón de luz. Pulsas una vez: el LED se enciende. Pulsas otra vez: el LED se apaga. Así aprendes cómo Arduino "siente" un botón.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Pon el pulsador sobre la ranura de tu breadboard.
2. Conecta una patita con pin 2 y una patita del otro lado con GND.
3. Pon el LED con una resistencia de 220 Ω en pin 13, la patita corta hacia GND.

El código

```
// Botón de luz
// Pulsa una vez el botón: el LED se enciende. Otra vez: el LED se apaga.

const int BOTON_PIN = 2; // El pulsador está en pin 2 (el otro lado a GND)
const int LED_PIN = 13; // El LED está en pin 13 (con la resistencia)

bool ledEncendido = false; // Recuerda si el LED ahora está encendido o apagado
bool botonAnterior = HIGH; // Cómo estaba el botón la vez anterior

void setup() {
  pinMode(BOTON_PIN, INPUT_PULLUP); // Entrada con resistencia interna: suelto = HIGH, pulsado = LOW
  pinMode(LED_PIN, OUTPUT);
}

void loop() {
  bool boton = digitalRead(BOTON_PIN);

  // Solo reaccionar en el momento en que pulsas el botón
  if (botonAnterior == HIGH && boton == LOW) {
    ledEncendido = !ledEncendido; // Cambiar: encendido pasa a apagado, apagado pasa a encendido
    digitalWrite(LED_PIN, ledEncendido ? HIGH : LOW);
    delay(50); // Espera corta contra el "rebote" del botón
  }

  botonAnterior = boton;
}
```

¿Qué hace el código?

- El botón está en pin 2 con `INPUT_PULLUP`. Suelto, el pin está HIGH. Pulsado, está LOW.
- Recordamos si el LED está encendido o apagado en `ledEncendido`.
- Solo en el momento en que pulsas el botón, cambiamos `ledEncendido`.

Prueba esto...

- Haz que el LED parpadee en vez de quedarse encendido
- Haz que el LED se apague solo después de 10 segundos
- Cuenta cuántas veces se ha pulsado el botón y muéstralo en el monitor serie

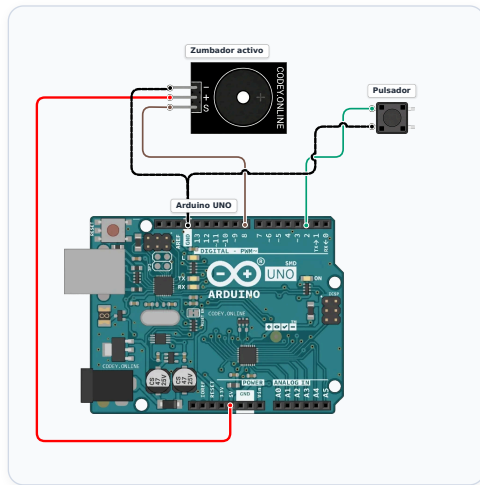
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/light-switch y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ Principiante ⌚ 10 minutos

Timbre

Vas a hacer tu propio timbre. ¡Pulsa el botón y el zumbador dice "ding-dong"!

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Pon el pulsador sobre la ranura del breadboard: una patita a pin 2, y una patita del otro lado a GND.
2. Conecta el zumbador: S a pin 8, + a 5V y – a GND.

El código

```
// Timbre
// Pulsa el botón y el zumbador hace un alegre "ding-dong".

const int BOTON_PIN = 2; // Pulsador en pin 2 (el otro lado a GND)
const int ZUMBADOR_PIN = 8; // Zumbador activo: S en pin 8

void setup() {
  pinMode(BOTON_PIN, INPUT_PULLUP); // Suelto = HIGH, pulsado = LOW
  pinMode(ZUMBADOR_PIN, OUTPUT);
}

// Hace pitar el zumbador un ratito
void pitido(int cuantoTiempo) {
  digitalWrite(ZUMBADOR_PIN, HIGH);
  delay(cuantoTiempo);
  digitalWrite(ZUMBADOR_PIN, LOW);
}

void loop() {
  if (digitalRead(BOTON_PIN) == LOW) { // ¡Alguien pulsa el timbre!
    pitido(300); // "Ding"
    delay(150);
    pitido(600); // "Dong"
    delay(1000); // Espera un poco, para que el timbre no siga sonando
  }
}
```

¿Qué hace el código?

- El botón está en pin 2 y el zumbador en pin 8.
- La función `pitido()` enciende el zumbador un momento y luego lo apaga otra vez.
- Si el botón está pulsado, hacemos sonar un pitido corto y uno largo.

Prueba esto...

- Toca una melodía alegre de cinco pitidos
- Haz que también parpadee un LED cuando alguien toque el timbre
- Cuenta cuántas veces han tocado el timbre y muéstralo en el monitor serie

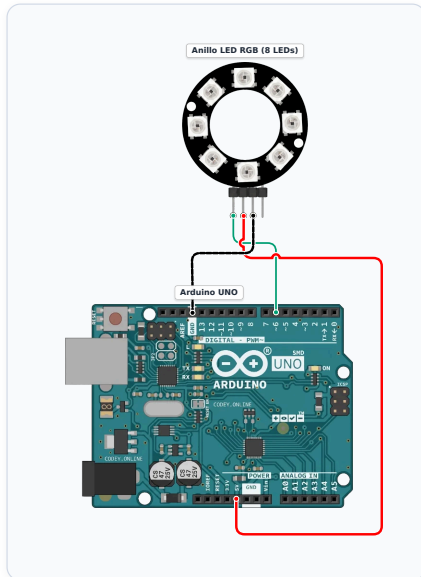
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/doorbell y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ Principiante ⌚ 10 minutos

Anillo arcoíris

Con el anillo LED haces un arcoíris que gira. Cada lucecita recibe otro color, y todos los colores se mueven en círculo.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Conecta el anillo LED: DI al pin 6, 5V a 5V y GND a GND.
2. ¡Eso es todo! Deja DO suelto.

El código

```
// Anillo arcoíris
// El anillo LED hace girar todos los colores del arcoíris.

#include <Adafruit_NeoPixel.h>

const int PIN_ANILLO = 6; // DI del anillo está en el pin 6
const int NUM_LEDS = 8; // El anillo tiene 8 lucecitas
const int VELOCIDAD = 20; // Cuanto menor, más rápido gira

Adafruit_NeoPixel anillo(NUM_LEDS, PIN_ANILLO, NEO_GRB + NEO_KHZ800);

void setup() {
  anillo.begin();
  anillo.setBrightness(40); // No muy fuerte: 0 = apagado, 255 = superbrillante
  anillo.show();
}

void loop() {
  // Mueve el arcoíris un poquito cada vez
  for (long color = 0; color < 65536; color += 512) {
    for (int i = 0; i < NUM_LEDS; i++) {
      // Cada lucecita recibe un color un poco distinto
      long colorLed = color + (i * 65536 / NUM_LEDS);
      anillo.setPixelColor(i, anillo.gamma32(anillo.ColorHSV(colorLed)));
    }
    anillo.show();
    delay(VELOCIDAD);
  }
}
```

¿Qué hace el código?

- Usamos la biblioteca `Adafruit_NeoPixel` para controlar el anillo.
- `ColorHSV()` crea un color del arcoíris. Cada lucecita va un poquito más adelante en el arcoíris.
- Si movemos el color un poquito cada vez, parece que el arcoíris gira.

Prueba esto...

- Haz que el arcoíris gire el doble de rápido
- Haz una luz de policía que parpadee en rojo y azul
- Haz que una sola lucecita dé vueltas, como una lucecita que corre

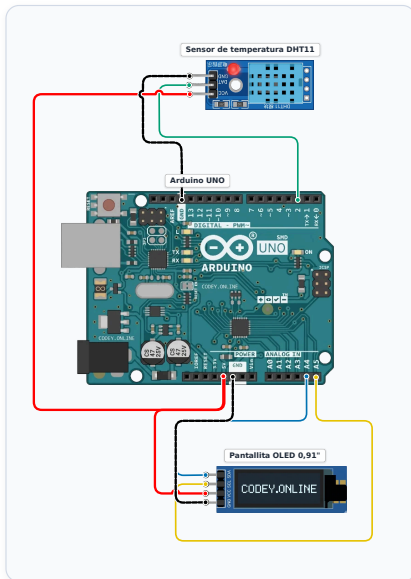
🔧 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/rainbow-ring y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ ★ Principiante avanzado ⌚ 20 minutos

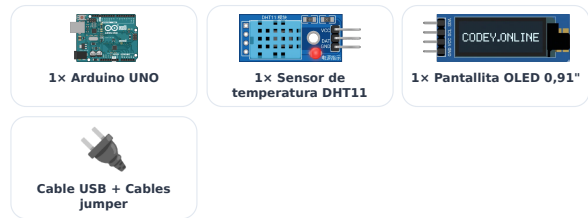
Mini estación meteorológica

Vas a hacer una mini estación meteorológica. El DHT11 mide la temperatura y la humedad, y la pantallita OLED las muestra.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Conecta el DHT11: DAT al pin 2, VCC a 5V y GND a GND.
2. Conecta la pantalla OLED: SDA a A4, SCL a A5, VCC a 5V y GND a GND.
3. Consejo: dos piezas necesitan 5V y GND. Usa las tiras + y - de tu breadboard.

El código

```
// Mini estación meteorológica
// El DHT11 mide la temperatura y la humedad. La pantallita OLED lo muestra.

#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <DHT.h>

const int DHT_PIN = 2; // DAT del DHT11 está en pin 2
DHT dht(DHT_PIN, DHT11);

Adafruit_SSD1306 pantalla(128, 32, &Wire, -1); // OLED de 128 x 32 píxeles (SDA = A4, SCL = A5)

void setup() {
  Serial.begin(9600);
  dht.begin();
  if (!pantalla.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // 0x3C es la dirección de la pantalla
    Serial.println(F("Pantalla no encontrada. Revisa los cables de SDA y SCL."));
    while (true); // Parar: sin pantalla no tiene sentido
  }
  pantalla.setTextColor(SSD1306_WHITE);
}

void loop() {
  float temperatura = dht.readTemperature(); // En grados Celsius
  float humedad = dht.readHumidity(); // En porcentaje

  pantalla.clearDisplay();
  pantalla.setTextSize(1);
  pantalla.setCursor(0, 0);

  if (isnan(temperatura) || isnan(humedad)) { // ¿No se puede medir?
    pantalla.setTextSize(1);
    pantalla.print(F("No hay DHT11"));
  } else {
    pantalla.print(temperatura, 1); // El signo de grados
    pantalla.print((char)247);
    pantalla.println(F("C"));
    pantalla.print(humedad, 0);
    pantalla.print(F("% hum. "));

    Serial.print(F("Temperatura: "));
    Serial.print(temperatura);
    Serial.print(F(" C Humedad: "));
    Serial.print(humedad);
    Serial.println(F(" %"));
  }

  pantalla.display(); // Solo ahora aparece todo en la pantalla
  delay(2000); // El DHT11 solo puede medir una vez cada 2 segundos
}
```

¿Qué hace el código?

- El DHT11 está en el pin 2. Con `readTemperature()` y `readHumidity()` leemos los valores.
- La pantalla OLED habla por SDA (A4) y SCL (A5). Con `print()` ponemos texto en la pantalla.
- Cada 2 segundos llega una nueva medición. ¿No se puede medir? Entonces ves un mensaje.

Prueba esto...

- Muestra también la temperatura en Fahrenheit
- Haz que un LED se encienda si hace más de 25 grados
- Recuerda la temperatura más alta de hoy y muéstrala también

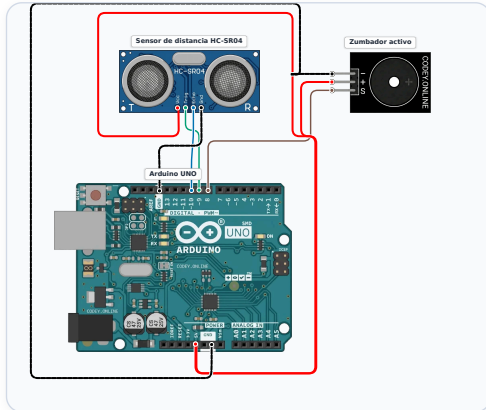
🔧 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/weather-station y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

☆☆ Principiante avanzado ⌚ 20 minutos

Sensor de aparcamiento

Vas a construir un sensor de aparcamiento, como en un coche. Cuanto más te acercas al sensor, más rápido pita el zumbador. ¿Muy cerca? Entonces suena un tono largo.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Conecta el HC-SR04: Trig al pin 9, Echo al pin 10, Vcc a 5V y Gnd a GND.
2. Conecta el zumbador: S al pin 8, + a 5V y - a GND.
3. Usa las tiras + y - de tu breadboard para 5V y GND.

El código

```
// Sensor de aparcamiento
// Cuanto más te acercas al sensor, más rápido pita el zumbador.
// Como en un coche!

const int TRIG_PIN = 9;    // Trig del HC-SR04 en pin 9
const int ECHO_PIN = 10;   // Echo del HC-SR04 en pin 10
const int ZUMBADOR_PIN = 8; // S del zumbador en pin 8

void setup() {
  Serial.begin(9600);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(ZUMBADOR_PIN, OUTPUT);
}

// Mide la distancia en centímetros
long medirDistancia() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);    // Envía un sonido corto (que
  // tú no oyes)
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  long tiempo = pulseIn(ECHO_PIN, HIGH, 30000); // ¿Cuánto tarda
  // en volver el eco?

  if (tiempo == 0) return 999;           // Sin eco: nada
  // cerca
  return tiempo / 58;                    // Convertir el
  // tiempo a centímetros
}

void loop() {
  long distancia = medirDistancia();
  Serial.print(F("Distancia: "));
  Serial.print(distancia);
  Serial.println(F(" cm"));

  if (distancia < 5) {
    digitalWrite(ZUMBADOR_PIN, HIGH); // Muy cerca: un tono largo
    delay(100);
  } else if (distancia < 50) {
    digitalWrite(ZUMBADOR_PIN, HIGH); // Pitido corto
    delay(50);
    digitalWrite(ZUMBADOR_PIN, LOW);
    delay(distancia * 10);             // Más cerca = pausa más
    // corta = pita más rápido
  } else {
    digitalWrite(ZUMBADOR_PIN, LOW); // Lejos: silencio
    delay(100);
  }
}
```

¿Qué hace el código?

- `medirDistancia()` envía un sonido y mide cuánto tarda en volver el eco. Eso lo convertimos a centímetros.
- A más de 50 cm: el zumbador está en silencio.
- Más cerca: la pausa entre los pitidos se hace más corta. Por debajo de 5 cm pita sin parar.

Prueba esto...

- Haz que el sensor de aparcamiento empiece a pitar ya desde 1 metro
- Añade un LED rojo que se encienda si está demasiado cerca
- Muestra la distancia en el monitor serie, con una barra de asteriscos

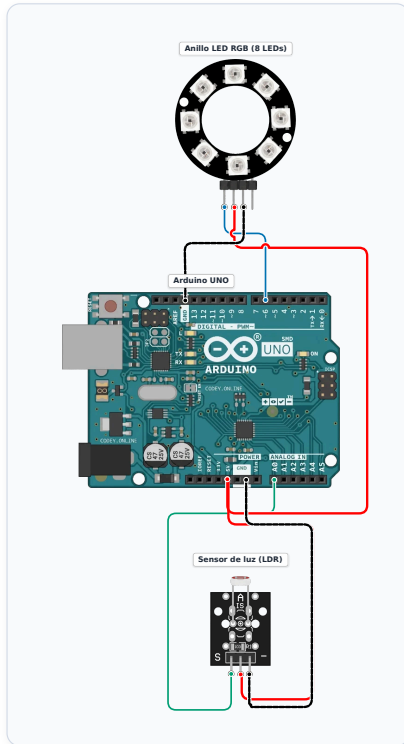
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/parking-sensor y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ ★ Principiante avanzado ⌚ 15 minutos

Lamparita de noche

Vas a hacer una lamparita de noche inteligente. Si se oscurece, el anillo LED se enciende suave. Si hay luz, se vuelve a apagar.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Conecta el sensor de luz: S a A0, + a 5V y – a GND.
2. Conecta el anillo LED: DI al pin 6, 5V a 5V y GND a GND.
3. Abre el monitor serie y mira qué número ves con luz y en oscuro. Ajusta `LIMITE_OSCURO` con eso.

El código

```
// Lamparita de noche
// ¿Se oscurece? Entonces el anillo LED se enciende suave. ¿Hay luz? Entonces se
// apaga.

#include <Adafruit_NeoPixel.h>

const int LUZ_PIN = A0; // S del sensor de luz en A0
const int ANILLO_PIN = 6; // DI del anillo LED en pin 6
const int NUM_LEDS = 8;

// Con este valor decimos que está "oscuro". Mira en el monitor serie qué número
// ves
// con luz y en oscuro, y pon aquí un número intermedio.
const int LIMITE_OSCURO = 500;

Adafruit_NeoPixel anillo(NUM_LEDS, ANILLO_PIN, NEO_GRB + NEO_KHZ800);

void setup() {
  Serial.begin(9600);
  anillo.begin();
  anillo.setBrightness(60);
  anillo.show();
}

void loop() {
  int luz = analogRead(LUZ_PIN); // Un número de 0 a 1023
  Serial.print(F("Sensor luz: "));
  Serial.println(luz);

  // Con este sensor, el número sube cuando se oscurece.
  // ¿En tu caso va al revés? Cambia > por <.
  bool oscuro = luz > LIMITE_OSCURO;

  for (int i = 0; i < NUM_LEDS; i++) {
    if (oscuro) {
      anillo.setPixelColor(i, anillo.Color(255, 140, 40)); // Luz amarilla
    } else {
      anillo.setPixelColor(i, 0); // Apagado
    }
  }
  anillo.show();
  delay(200);
}
```

¿Qué hace el código?

- El sensor de luz está en A0. `analogRead()` da un número de 0 a 1023.
- ¿El número es mayor que `LIMITE_OSCURO`? Entonces decimos que está oscuro.
- En la oscuridad damos a todas las lucecitas un color amarillo cálido. Si no, se apagan.

Prueba esto...

- Haz que la lamparita brille poco a poco más fuerte cuando se oscurezca
- Haz que el color cambie despacito, como una lámpara tranquila
- Haz que la lamparita se apague sola después de 1 minuto

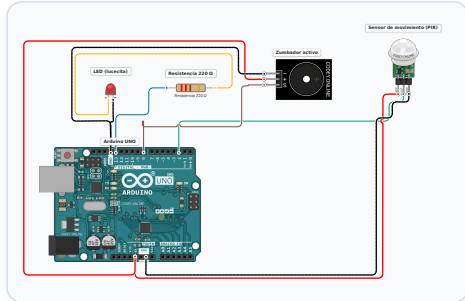
🔧 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/night-light y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ ★ Principiante avanzado ⌚ 20 minutos

Alarma de movimiento

Vas a construir una alarma antirrobo. ¿El sensor de movimiento ve a alguien? Entonces suena el zumbador y parpadea el LED rojo.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Conecta el sensor PIR: OUT a pin 2, VCC a 5V y GND a GND.
2. Conecta el zumbador: S a pin 8, + a 5V y – a GND.
3. Pon el LED rojo con una resistencia de 220 Ω en pin 13, la patita corta a GND.

El código

```
// Alarma de movimiento
// ¿Pasa alguien? Entonces parpadea el LED rojo y suena el zumbador.

const int PIR_PIN = 2;      // OUT del sensor de movimiento en pin 2
const int ZUMBADOR_PIN = 8; // S del zumbador en pin 8
const int LED_PIN = 13;     // LED rojo en pin 13 (con la resistencia)

void setup() {
  Serial.begin(9600);
  pinMode(PIR_PIN, INPUT);
  pinMode(ZUMBADOR_PIN, OUTPUT);
  pinMode(LED_PIN, OUTPUT);
  Serial.println(F("El sensor se prepara... ¡quédate quieto!"));
  delay(10000); // El sensor necesita un momento para acostumbrarse a la habitación
  Serial.println(F("Alarma activada."));
}

void loop() {
  if (digitalRead(PIR_PIN) == HIGH) { // ¡Movimiento detectado!
    Serial.println(F("¡Movimiento! ¡ALARMA!"));
    for (int i = 0; i < 5; i++) { // Pitar y parpadear 5 veces
      digitalWrite(LED_PIN, HIGH);
      digitalWrite(ZUMBADOR_PIN, HIGH);
      delay(150);
      digitalWrite(LED_PIN, LOW);
      digitalWrite(ZUMBADOR_PIN, LOW);
      delay(150);
    }
  }
  delay(100);
}
```

¿Qué hace el código?

- El sensor PIR está en pin 2. Si ve movimiento, el pin está HIGH.
- Al encender, esperamos 10 segundos. Así el sensor se acostumbra a la habitación.
- Si hay movimiento, pitamos y parpadeamos 5 veces.

Prueba esto...

- Añade un botón para encender y apagar la alarma
- Cuenta cuántas veces hubo movimiento y muéstralo en el monitor serie
- Haz que la alarma suene más tiempo y más fuerte la segunda vez que haya movimiento

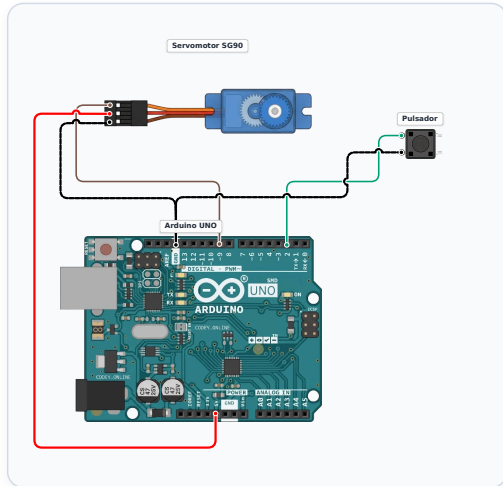
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/motion-alarm y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ ★ Principiante avanzado ⌚ 20 minutos

Barrera

Vas a construir una barrera, como en un parking. Pulas el botón: el servo sube la barrera, espera un poco y se cierra otra vez.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Pon el pulsador sobre la ranura de tu breadboard: una patita a pin 2, una patita del otro lado a GND.
2. Conecta el servo: naranja a pin 9, rojo a 5V y marrón a GND.
3. Pega una pajita o un palito de helado al brazo del servo: ¡esa es tu barrera!

El código

```
// Barrera
// Pula el botón: la barrera (servo) sube, espera un poco y se
// cierra otra vez.

#include <Servo.h>

const int BOTON_PIN = 2;    // Pulsador en pin 2 (el otro lado
// a GND)
const int SERVO_PIN = 9;    // Cable naranja/amarillo del servo
// en pin 9

const int CERRADO = 0;      // Ángulo del servo cuando la
// barrera está cerrada
const int ABIERTO = 90;     // Ángulo del servo cuando la
// barrera está abierta

Servo barrera;

void setup() {
  pinMode(BOTON_PIN, INPUT_PULLUP);

  barrera.attach(SERVO_PIN);
  barrera.write(CERRADO);    // Empieza con la barrera cerrada
}

void loop() {
  if (digitalRead(BOTON_PIN) == LOW) {    // Botón pulsado
    // Girar despacio para abrir, grado a grado
    for (int angulo = CERRADO; angulo <= ABIERTO; angulo++) {
      barrera.write(angulo);
      delay(15);
    }
    delay(3000);    // 3 segundos abierta:
    // Cerrar despacio otra vez
    for (int angulo = ABIERTO; angulo >= CERRADO; angulo--) {
      barrera.write(angulo);
      delay(15);
    }
  }
}
```

¿Qué hace el código?

- Usamos la biblioteca `Servo`. Con `write()` dices a qué ángulo debe ir el servo.
- Después de pulsar el botón, giramos grado a grado de 0 a 90. Así va tranquilo.
- Después de 3 segundos volvemos a 0: la barrera está cerrada.

Prueba esto...

- Haz que la barrera se abra y se cierre más rápido
- Haz que un zumbador pite mientras la barrera se mueve
- Haz que un LED rojo se encienda cuando la barrera esté cerrada y uno verde cuando esté abierta

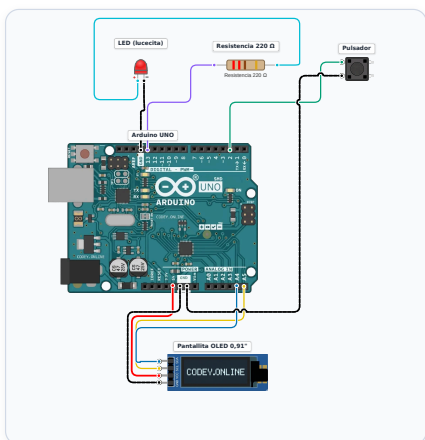
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/barrier y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★ ★ Principiante avanzado ⌚ 25 minutos

Juego de reacción

¿Qué tan rápido eres? Espera hasta que el LED se encienda y pulsa el botón lo más rápido que puedas. La pantalla muestra tu tiempo en milisegundos. ¡Reta a tus amigos!

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Pon el pulsador sobre la ranura del breadboard: una patita al pin 2 y una patita del otro lado a GND.
2. Pon el LED con una resistencia de 220 Ω en el pin 13, con la patita corta a GND.
3. Conecta la pantalla OLED: SDA a A4, SCL a A5, VCC a 5V y GND a GND.

El código

```
// Juego de reaccion
// Espera a que el LED se encienda y pulsa el boton lo mas rapido posible.
// La pantalla muestra que tan rapido fuiste!

#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

const int BOTON_PIN = 2; // Pulsador en pin 2 (otro lado a GND)
const int LED_PIN = 13; // LED en pin 13 (por la resistencia)

Adafruit_SSD1306 pantalla(128, 32, &Wire, -1); // OLED: SDA = A4, SCL = A5

unsigned long mejorTiempo = 0; // Tu tiempo mas rapido hasta ahora

// Pone un texto grande en la pantalla
void mostrar(const __FlashStringHelper* texto, unsigned long numero) {
  pantalla.clearDisplay();
  pantalla.setTextSize(1);
  pantalla.setCursor(0, 0);
  pantalla.print(texto);
  if (numero > 0) {
    pantalla.setTextSize(2);
    pantalla.setCursor(0, 14);
    pantalla.print(numero);
    pantalla.print(F(" ms"));
  }
  pantalla.display();
}

void setup() {
  pinMode(BOTON_PIN, INPUT_PULLUP);
  pinMode(LED_PIN, OUTPUT);
  randomSeed(analogRead(A0)); // Da tiempos de espera realmente al azar

  pantalla.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  pantalla.setTextColor(SSD1306_WHITE);
}

void loop() {
  mostrar(F("Pulsa el boton..."), 0);
  while (digitalRead(BOTON_PIN) == HIGH); // Espera hasta que pulses el boton
  while (digitalRead(BOTON_PIN) == LOW); // ...y lo sueltes otra vez
  mostrar(F("Listo? Mira LED!"), 0);

  delay(random(2000, 5000)); // Un tiempo secreto entre 2 y 5 segundos

  if (digitalRead(BOTON_PIN) == LOW) { // Pulsaste antes? Tramposo!
    mostrar(F("Muy pronto! Otra."), 0);
    delay(2000);
    return;
  }

  digitalWrite(LED_PIN, HIGH); // AHORA!
  unsigned long start = millis();
  while (digitalRead(BOTON_PIN) == HIGH); // Esperando tu pulsacion
  unsigned long tiempo = millis() - start;
  digitalWrite(LED_PIN, LOW);

  if (mejorTiempo == 0 || tiempo < mejorTiempo) {
    mejorTiempo = tiempo;
    mostrar(F("NUEVO RECORD!"), tiempo);
  } else {
    mostrar(F("Tu tiempo:"), tiempo);
  }
  delay(3000);
}
```

¿Qué hace el código?

- Después de pulsar el botón, esperamos un tiempo secreto entre 2 y 5 segundos (`random()`).
- Luego el LED se enciende y ponemos en marcha un cronómetro con `millis()`.
- Cuando pulsas, calculamos cuántos milisegundos tardaste. Guardamos tu mejor tiempo.

Prueba esto...

- Haz un juego para dos jugadores, cada uno con su propio botón
- Haz que un zumbador pite cuando haya un nuevo récord
- Juega 5 rondas y luego muestra tu tiempo medio

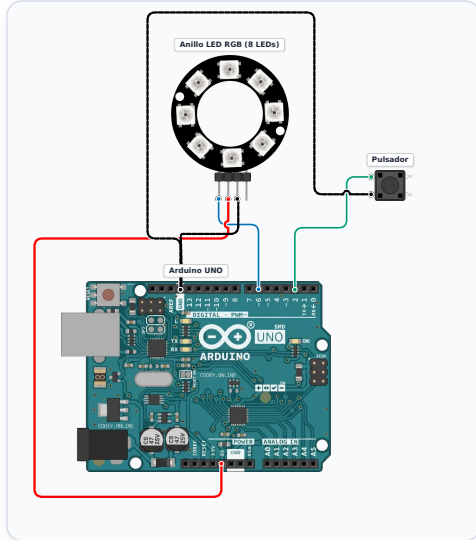
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/reaction-game y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

☆☆ Principiante avanzado ⌚ 15 minutos

Dado electrónico

Vas a hacer un dado de luz. Pulsa el botón: las luces ruedan un momento, y luego ves cuánto has sacado. ¡De 1 a 6 luces!

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Pon el pulsador sobre la ranura del breadboard: una patita a pin 2, y una patita del otro lado a GND.
2. Conecta el anillo LED: DI a pin 6, 5V a 5V y GND a GND.

El código

```
// Dado electrónico
// Pulsa el botón: el anillo LED rueda un momento y luego deja
// encendidas de 1 a 6 luces.

#include <Adafruit_NeoPixel.h>

const int BOTON_PIN = 2; // Pulsador en pin 2 (otro lado a GND)
const int ANILLO_PIN = 6; // DI del anillo LED en pin 6
const int NUM_LEDS = 8;

Adafruit_NeoPixel anillo(NUM_LEDS, ANILLO_PIN, NEO_GRB + NEO_KHZ800);

// Enciende varias luces en un color
void mostrarNumero(int numero, uint32_t color) {
  anillo.clear();
  for (int i = 0; i < numero; i++) {
    anillo.setPixelColor(i, color);
  }
  anillo.show();
}

void setup() {
  pinMode(BOTON_PIN, INPUT_PULLUP);
  randomSeed(analogRead(A0)); // Hace que el dado sea de verdad al
  azar
  anillo.begin();
  anillo.setBrightness(50);
  anillo.show();
}

void loop() {
  if (digitalRead(BOTON_PIN) == LOW) {
    // Un momento "rodando": números que cambian rápido, cada vez un
    poco más lento
    for (int i = 0; i < 15; i++) {
      mostrarNumero(random(1, 7), anillo.Color(0, 80, 255));
      delay(40 + i * 10);
    }
    int tirada = random(1, 7); // Un número del 1
    al 6
    mostrarNumero(tirada, anillo.Color(0, 255, 60)); // El número
    real en verde
    delay(500);
  }
}
```

¿Qué hace el código?

- `random(1, 7)` elige un número al azar del 1 al 6.
- Primero mostramos números que cambian rápido en azul. Así parece que el dado rueda.
- Después `mostrarNumero()` muestra el número real en verde.

Prueba esto...

- Da a cada número su propio color
- Haz que un zumbador pite un momento cuando el dado se pare
- Haz un dado del 1 al 8

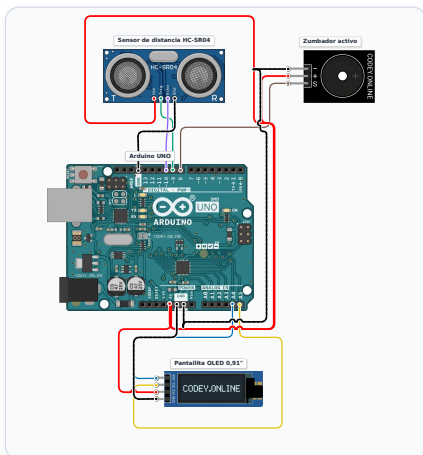
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/dice y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★★★★ Reto ⌚ 30 minutos

Sensor de aparcamiento deluxe

¡La versión deluxe del sensor de aparcamiento! Pita más rápido cuando te acercas, y la pantalla OLED muestra la distancia en centímetros. Con una barra que se hace más larga cuando te acercas.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Conecta el HC-SR04: Trig al pin 9, Echo al pin 10, Vcc a 5V y Gnd a GND.
2. Conecta el zumbador: S al pin 8, + a 5V y – a GND.
3. Conecta la pantalla OLED: SDA a A4, SCL a A5, VCC a 5V y GND a GND.
4. Tres piezas necesitan 5V y GND. Usa las tiras + y – de tu breadboard.

El código

```
// Sensor de aparcamiento deluxe
// Como el sensor normal, pero ahora también ves la distancia en la pantalla OLED.

#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

const int TRIG_PIN = 9; // Trig del HC-SR04 en pin 9
const int ECHO_PIN = 10; // Echo del HC-SR04 en pin 10
const int ZUMBADOR_PIN = 8; // S del zumbador en pin 8

Adafruit_SSD1306 pantalla(128, 32, &Wire, -1); // OLED: SDA = A4, SCL = A5

// Mide la distancia en centímetros
long medirDistancia() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  long tiempo = pulseIn(ECHO_PIN, HIGH, 30000);
  if (tiempo == 0) return 999; // Sin echo: nada cerca
  return tiempo / 58;
}

// Pone la distancia grande en la pantalla, con una barra debajo
void mostrarDistancia(long distancia) {
  pantalla.clearDisplay();
  pantalla.setTextSize(2);
  pantalla.setCursor(0, 0);
  if (distancia >= 999) {
    pantalla.print(F("----"));
  } else {
    pantalla.print(distancia);
    pantalla.print(F(" cm"));
  }
  // Barra: cuanto más cerca, más larga
  int largo = map(constrain(distancia, 0, 100), 0, 100, 128, 0);
  pantalla.fillRect(0, 24, largo, 8, SSD1306_WHITE);
  pantalla.display();
}

void setup() {
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(ZUMBADOR_PIN, OUTPUT);
  pantalla.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  pantalla.setTextColor(SSD1306_WHITE);
}

void loop() {
  long distancia = medirDistancia();
  mostrarDistancia(distancia);

  if (distancia < 5) {
    digitalWrite(ZUMBADOR_PIN, HIGH); // Muy cerca: un tono largo
    delay(100);
  } else if (distancia < 50) {
    digitalWrite(ZUMBADOR_PIN, HIGH); // Pitido corto
    delay(50);
    digitalWrite(ZUMBADOR_PIN, LOW);
    delay(distancia * 10); // Más cerca = más pitidos
  } else {
    digitalWrite(ZUMBADOR_PIN, LOW);
    delay(100);
  }
}
```

¿Qué hace el código?

- `medirDistancia()` mide la distancia en centímetros con el HC-SR04.
- `mostrarDistancia()` pone el número grande en la pantalla y dibuja una barra con `fillRect()`.
- El zumbador pita más rápido cuanto menor es la distancia.

Prueba esto...

- Muestra "¡STOP!" en la pantalla si estás a menos de 5 cm
- Muestra la distancia en metros si algo está a más de 1 metro
- Añade el anillo LED que cambia de verde a rojo

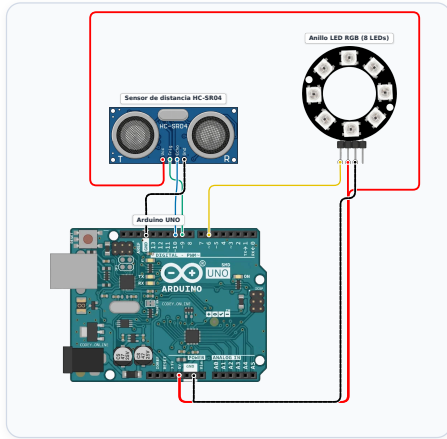
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/parking-sensor-deluxe y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★★★★ Reto ⌚ 25 minutos

Medidor de distancia con barra de luz

Vas a hacer una barra de luz que muestra lo cerca que está algo. Lejos: ninguna lucecita. Cuanto más cerca, más lucecitas se encienden. Primero verde, luego naranja y muy cerca rojo.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Conecta el HC-SR04: Trig al pin 9, Echo al pin 10, Vcc a 5V y Gnd a GND.
2. Conecta el anillo LED: DI al pin 6, 5V a 5V y GND a GND.

El código

```
// Medidor de distancia con barra de luz
// Cuanto más cerca estás, más lucecitas se encienden: verde, luego
naranja, luego rojo.

#include <Adafruit_NeoPixel.h>

const int TRIG_PIN = 9;    // Trig del HC-SR04 en el pin 9
const int ECHO_PIN = 10;   // Echo del HC-SR04 en el pin 10
const int RING_PIN = 6;    // DI del anillo LED en el pin 6
const int NUM_LEDS = 8;
const int DISTANCIA_MAX = 80; // Más lejos que esto (en cm): todas las
luzes apagadas

Adafruit_NeoPixel anillo(NUM_LEDS, RING_PIN, NEO_GRB + NEO_KHZ800);

long medirDistancia() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  long tiempo = pulseIn(ECHO_PIN, HIGH, 30000);
  if (tiempo == 0) return 999;
  return tiempo / 58;
}

void setup() {
  Serial.begin(9600);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  anillo.begin();

  anillo.setBrightness(40);
  anillo.show();
}

void loop() {
  long distancia = medirDistancia();
  Serial.print(F("Distancia: "));
  Serial.print(distancia);
  Serial.println(F(" cm"));

  // ¿Cuántas lucecitas deben encenderse? Cerca = 8, lejos = 0
  int cantidad = map(constrain(distancia, 0, DISTANCIA_MAX), 0,
DISTANCIA_MAX, NUM_LEDS, 0);

  for (int i = 0; i < NUM_LEDS; i++) {
    if (i < cantidad) {
      if (i < 3) anillo.setPixelColor(i, anillo.Color(0, 255, 0));
      // Verde
      else if (i < 6) anillo.setPixelColor(i, anillo.Color(255, 120, 0));
      // Naranja
      else anillo.setPixelColor(i, anillo.Color(255, 0, 0));
      // Rojo
    } else {
      anillo.setPixelColor(i, 0);
      // Apagado
    }
  }
  anillo.show();
  delay(100);
}
```

¿Qué hace el código?

- `medirDistancia()` mide la distancia con el HC-SR04.
- `map()` convierte la distancia en un número de lucecitas: de 0 a 8.
- Las primeras 3 lucecitas son verdes, las siguientes 3 naranjas y las últimas 2 rojas.

Prueba esto...

- Haz que todas las lucecitas rojas parpadeen si estás a menos de 10 cm
- Añade el zumbador que pita en rojo
- Haz el medidor más sensible: todas las lucecitas encendidas a 40 cm en vez de 80 cm

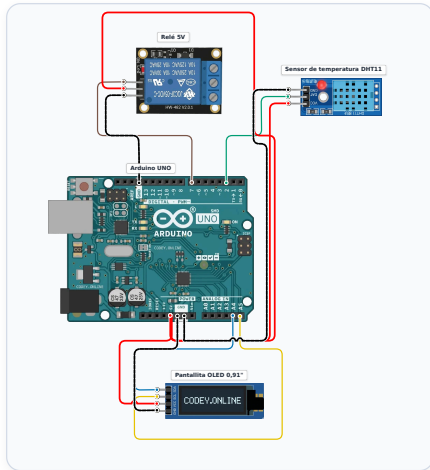
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/distance-light-bar y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★★★★ Reto ⌚ 30 minutos

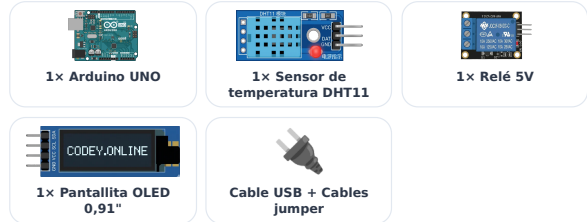
Ventilador inteligente

Vas a construir un termostato. Si hace más de 25 grados, el relé hace clic y se enciende. Así tu Arduino puede encender, por ejemplo, un ventilador pequeño con una batería. En la pantalla ves la temperatura y si el ventilador está encendido.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Conecta el DHT11: DAT al pin 2, VCC a 5V y GND a GND.
2. Conecta el relé: S al pin 7, + a 5V y - a GND.
3. Conecta la pantalla OLED: SDA a A4, SCL a A5, VCC a 5V y GND a GND.
4. ¿Quieres encender algo de verdad? Usa solo una batería o bajo voltaje en COM y NO, nunca el enchufe.

El código

```
// Ventilador inteligente (termostato)
// ¿Hace demasiado calor? Entonces el relé se enciende, por ejemplo para un ventilador.
// En la pantalla ves la temperatura y si el ventilador está encendido.

#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <DHT.h>

const int DHT_PIN = 2; // DAT del DHT11 en pin 2
const int RELE_PIN = 7; // S del relé en pin 7
const float ENCENDER_SOBRE = 25.0; // Sobre esta temperatura se enciende el ventilador
const float APAGAR_BAJ0 = 24.0; // Bajo esta temperatura se vuelve a apagar

DHT dht(DHT_PIN, DHT11);
Adafruit_SSD1306 pantalla(128, 32, &Wire, -1); // OLED: SDA = A4, SCL = A5

bool ventiladorEncendido = false;

void setup() {
  Serial.begin(9600);
  pinMode(RELE_PIN, OUTPUT);
  digitalWrite(RELE_PIN, LOW); // Empieza con el relé apagado
  dht.begin();
  pantalla.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  pantalla.setTextColor(SSD1306_WHITE);
}

void loop() {
  float temperatura = dht.readTemperature();

  if (!isnan(temperatura)) {
    // Dos límites, para que el relé no haga clic sin parar
    if (temperatura > ENCENDER_SOBRE) ventiladorEncendido = true;
    if (temperatura < APAGAR_BAJ0) ventiladorEncendido = false;
    digitalWrite(RELE_PIN, ventiladorEncendido ? HIGH : LOW);
  }

  pantalla.clearDisplay();
  pantalla.setTextSize(2);
  pantalla.setCursor(0, 0);
  if (isnan(temperatura)) {
    pantalla.print(F("DHT11 ?"));
  } else {
    pantalla.print(temperatura, 1);
    pantalla.print((char)247);
    pantalla.print(F("C"));
  }

  pantalla.setTextSize(1);
  pantalla.setCursor(0, 22);
  pantalla.print(ventiladorEncendido ? F("Ventilador: SI") : F("Ventilador: no"));
  pantalla.display();

  Serial.print(F("Temperatura: "));
  Serial.print(temperatura);
  Serial.println(ventiladorEncendido ? F(" C -> SI") : F(" C -> no"));

  delay(2000);
}
```

¿Qué hace el código?

- El DHT11 mide la temperatura. Por encima de 25 grados encendemos el relé.
- Solo por debajo de 24 grados el relé se vuelve a apagar. Con esos dos límites no hace clic todo el tiempo encendiendo y apagando.
- La pantalla OLED muestra la temperatura y "Ventilador: SI" o "no".

Prueba esto...

- Añade dos botones para subir y bajar la temperatura límite
- Haz que un LED se encienda mientras el ventilador esté encendido
- Haz que el ventilador también se encienda si la humedad del aire pasa del 70%

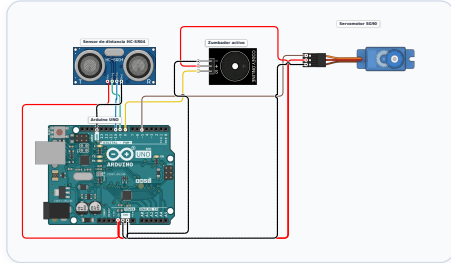
🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/thermostat y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.

★★★★ Reto ⌚ 30 minutos

Barrera automática

¡La versión inteligente de la barrera! El sensor de distancia ve que viene un coche. Entonces el zumbador pita y la barrera se abre sola. ¿El coche se ha ido? Entonces se vuelve a cerrar.

Esquema de conexión



¿Qué necesitas?



Así lo construyes

1. Conecta el HC-SR04: Trig al pin 9, Echo al pin 10, Vcc a 5V y Gnd a GND.
2. Conecta el servo: naranja al pin 5, rojo a 5V y marrón a GND.
3. Conecta el zumbador: S al pin 8, + a 5V y - a GND.
4. Pega una pajita o un palito de helado al servo como barrera y pon el sensor delante.

El código

```
// Barrera automática
// ¿Viene un coche (a menos de 15 cm)? Entonces el zumbador pita y la
// barrera se abre.

#include <Servo.h>

const int TRIG_PIN = 9;    // Trig del HC-SR04 en pin 9
const int ECHO_PIN = 10;   // Echo del HC-SR04 en pin 10
const int SERVO_PIN = 5;   // Cable naranja/amarillo del servo en pin 5
const int ZUMBADOR_PIN = 8; // S del zumbador en pin 8
const int CERCA = 15;      // Desde esta distancia (cm) se abre la barrera

const int CERRADO = 0;     // Ángulo cuando la barrera está cerrada
const int ABIERTO = 90;    // Ángulo cuando la barrera está abierta

Servo barrera;

long medirDistancia() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  long tiempo = pulseIn(ECHO_PIN, HIGH, 30000);
  if (tiempo == 0) return 999;
  return tiempo / 58;
}

void setup() {
  Serial.begin(9600);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(ZUMBADOR_PIN, OUTPUT);
  barrera.attach(SERVO_PIN);
  barrera.write(CERRADO);
}

void loop() {
  long distancia = medirDistancia();

  if (distancia < CERCA) {
    Serial.println(F("Coche visto! Barrera abierta."));
    digitalWrite(ZUMBADOR_PIN, HIGH); // Pitido corto
    delay(100);
    digitalWrite(ZUMBADOR_PIN, LOW);

    barrera.write(ABIERTO);
    delay(1000);
    // Quedarse abierta mientras el coche siga ahí
    while (medirDistancia() < CERCA) {
      delay(200);
    }
    delay(2000); // Esperar un poco más hasta que
    // el coche se vaya
    barrera.write(CERRADO);
    Serial.println(F("Barrera cerrada."));
  }
  delay(100);
}
```

¿Qué hace el código?

- El HC-SR04 mide la distancia todo el tiempo. ¿Menos de 15 cm? Entonces hay un coche.
- El zumbador pita un momento y el servo gira a 90 grados: abierto.
- Mientras el coche está allí, la barrera queda abierta. Luego esperamos 2 segundos y se cierra.

Prueba esto...

- Cuenta cuántos coches han pasado y muéstralo en el monitor serie
- Añade un LED rojo y uno verde como semáforo junto a la barrera
- Haz que la barrera solo se abra si también se pulsa un botón

🚀 Construye este proyecto con Codey: abre codey.online/es/starter-kits/uno/projects/automatic-barrier y haz clic en HAZ ESTO. Entonces el código y el esquema de conexión estarán listos enseguida.