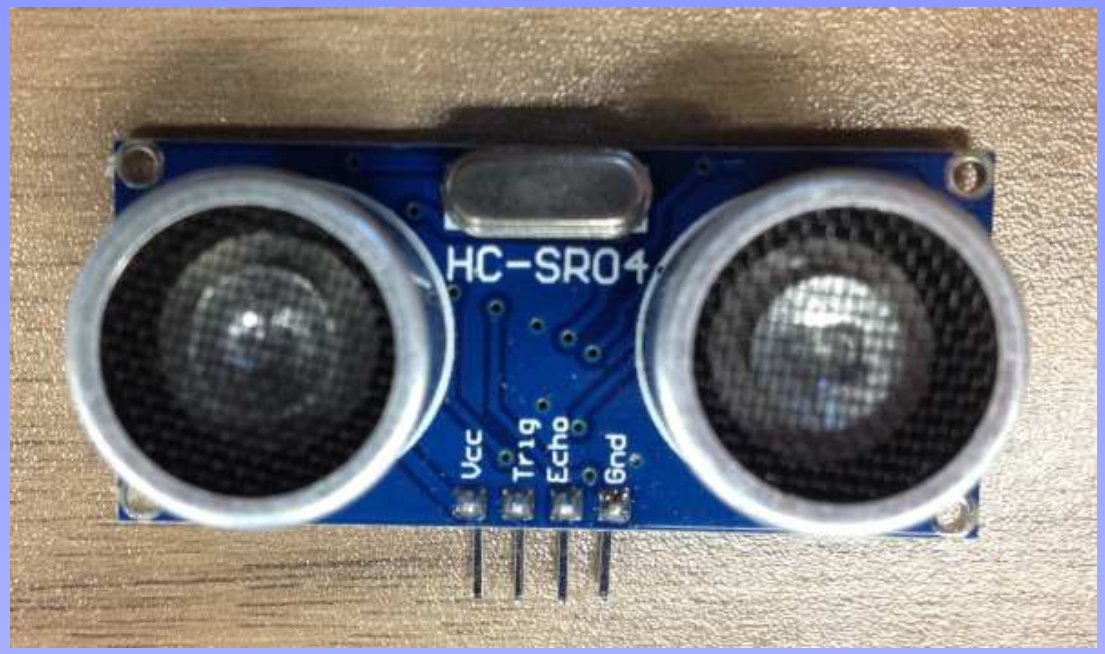
SENSOR ULTRASONIDOS

O

módulo ultrasónico modelo HC-SR04.



* Es un sensor de distancia que nos permite enviar pulsos ultrasónicos y escuchar el eco de retorno.

*Trabaja emitiendo sonidos a altas frecuencias (sonidos que no pueden ser captados por el oído humano). Cuando esas ondas chocan con un objeto rebotan y viajan de regreso al sensor, de esta manera el sensor puede calcular la distancia a la que se encuentra el objeto.

*Funcionan sobre los 40 kHz.

*Tiene un rango de distancias sensible entre 3cm y 3m.

*Les influye la temperatura ambiente, la humedad y los materiales en los que reflejan.

PARTES: consta de 4 pines

VCC – corriente. Conectado a la salida de 5 voltios de la placa

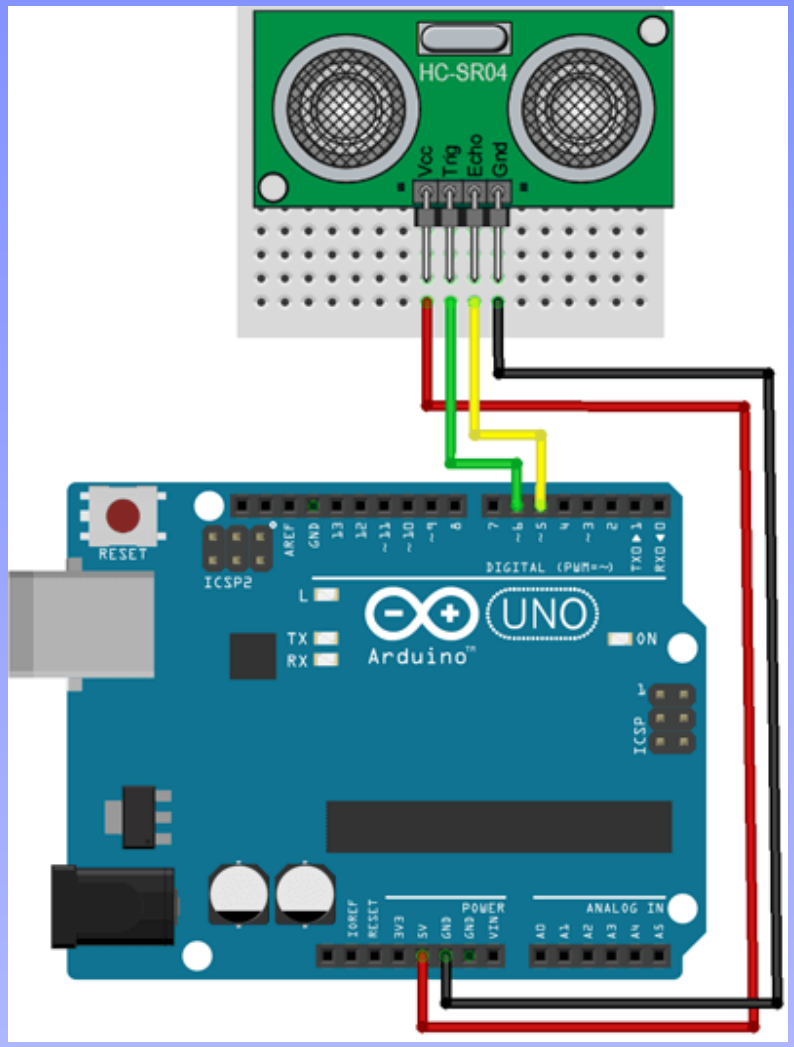
GND – masa o tierra. Conectado a dicho pin en la placa.

TRIG – conectado al pin digital de la placa encargado de enviar el pulso ultrasónico.

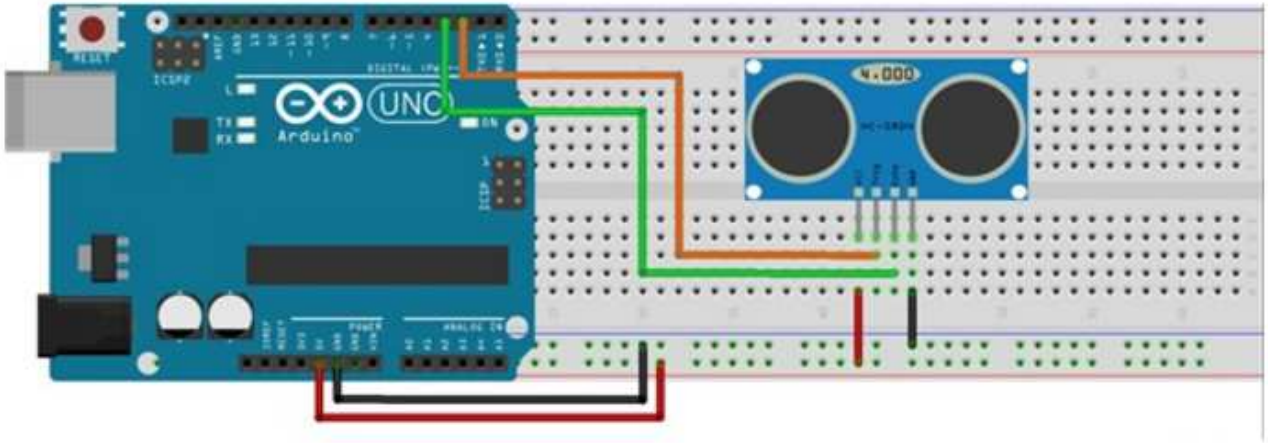
ECHO – conectado al pin digital que recibirá el eco de dicho pulso

ESQUEMA DE MONTAJE:

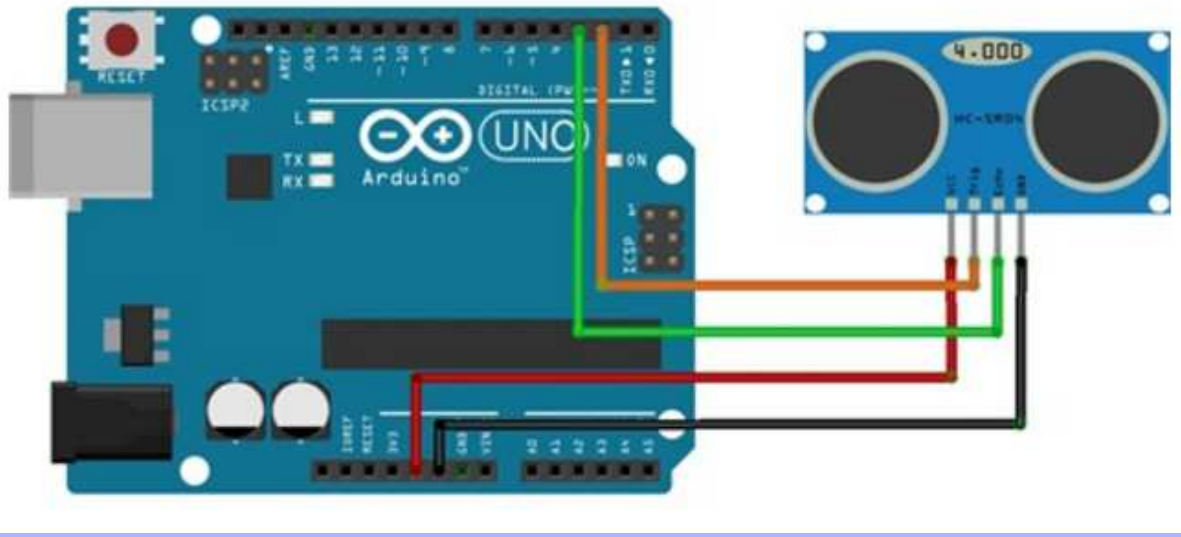
La conexión a los pines 5 y 6 digitales o analógicos, puede cambiar a cualquier otro pin deseado.



MONTAJE DEL SENSOR



También puedes conectar el modulo directamente al Arduino sin usar el protoboard.



¿ Qué recibimos en el sensor?

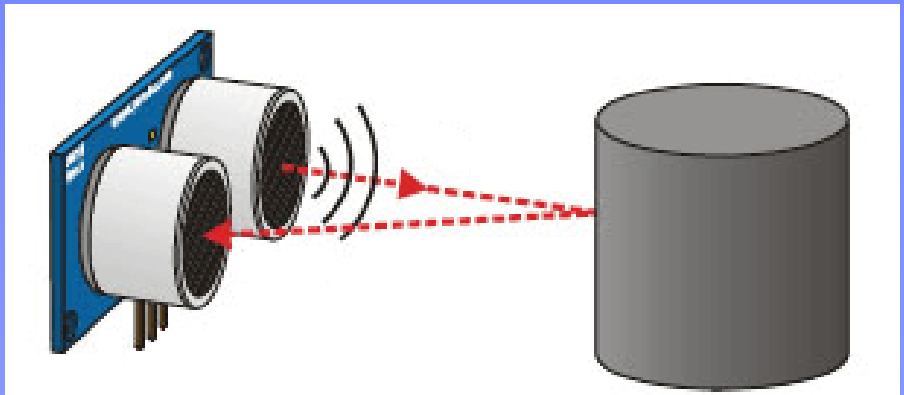
El pulso rebota en los objetos cercanos y es reflejado hacia el sensor, que dispone de un micrófono adecuado para esa frecuencia. Recibimos por tanto el tiempo que transcurre entre el envío y la recepción del ultrasonido.

¿Cómo funciona?

Mide el tiempo entre el envío y la recepción de un pulso sonoro.

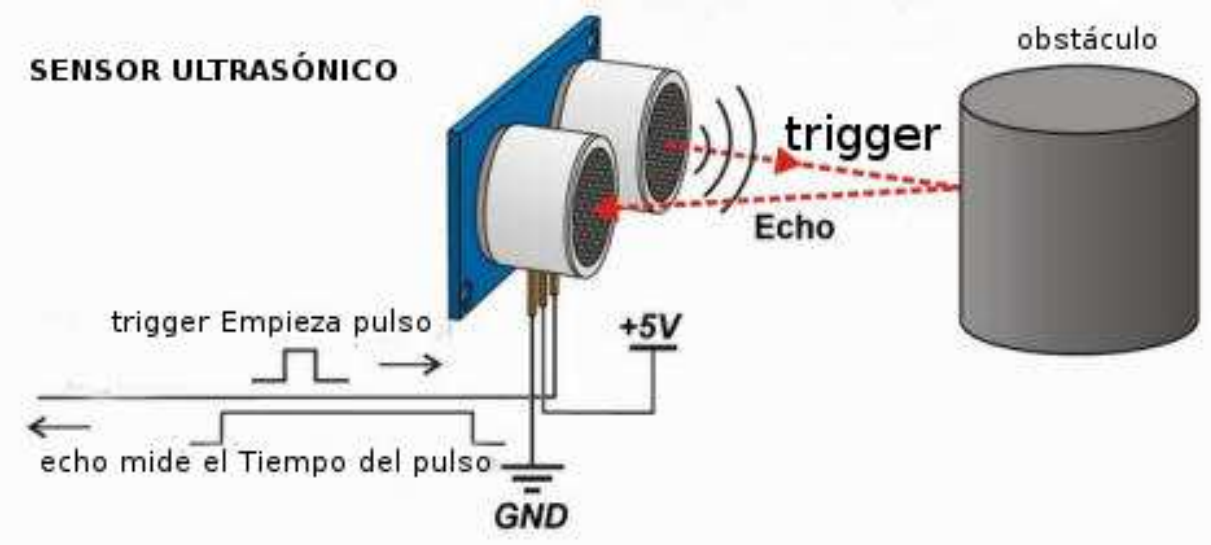
Velocidad del sonido 343 m/s
29 microsegundos por cm

$$Distancia(cm) = \frac{Tiempo(\mu s)}{29.2 \cdot 2}$$



Tiempo = 2 * (Distancia / Velocidad)

Distancia = Tiempo · Velocidad / 2



CÓDIGO BÁSICO PARA EL SENSOR:

Trabajamos 3 partes:

- Creamos un disparo
- Captamos un pulso de salida que retorna al sensor.
- Usamos esa información para calcular la distancia al objeto.

- 1.

- Hacer el disparo

```
/* Hacer el disparo */
digitalWrite(PIN_TRIG, LOW);
delayMicroseconds(2);
digitalWrite(PIN_TRIG, HIGH); // Flanco ascendente
delayMicroseconds(10);      // Duracion del pulso
digitalWrite(PIN_TRIG, LOW); // Flanco descendente
```

* ESTE PRIMER PULSO DE DISPARO ES EL PULSO DE ACTIVACIÓN DEL SENSOR PARA QUE ÉSTE COMIENCE A REALIZAR LA MEDICIÓN.

2 y 3 paso:

- Recepcion del pulso eco de respuesta

```
/* Recepcion del eco de respuesta */
duracion = pulseIn(PIN_ECO, HIGH);
```

- Cálculo de la distancia efectiva

```
distancia = (duracion/2) / 29;
```

* RECIBIMOS ESE ECO DE RESPUESTA CON LA FUNCIÓN pulseIn , QUE NOS VA A PERMITIR MEDIR LA CONGITUD DE UN PULSO DE ENTRADA. (nº de pin ECO, valor)

- **pulseIn:** RECIBE EL PULSO ALTO DE ENTRADA POR EL PIN ECO Y NOS DEVUELVE LA DURACIÓN EXACTA DE ESE PULSO EN MICROSEGUNDOS. Esa duración la almacenados en la variable a la que también hemos llamado – duración –

- Nos referimos a duración = tiempo.

```
duracion = pulseIn(PIN_ECO, HIGH);
```

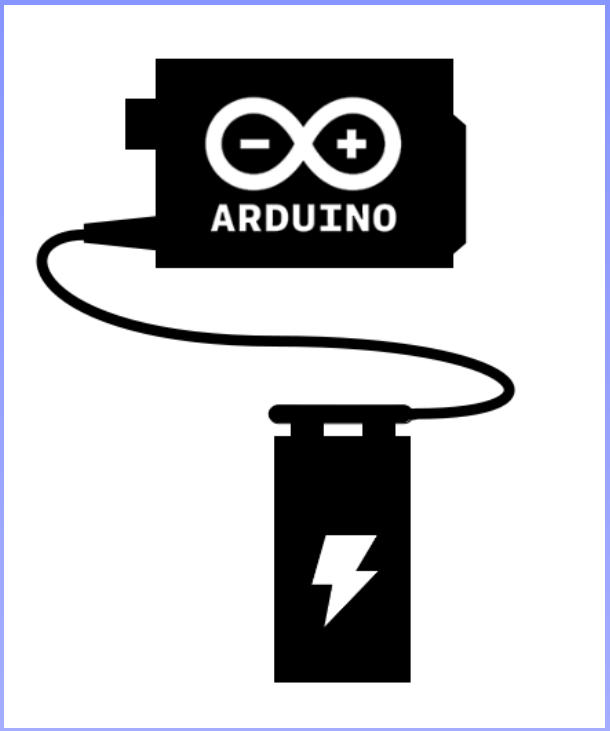
- Partimos de la fórmula de v=d/t

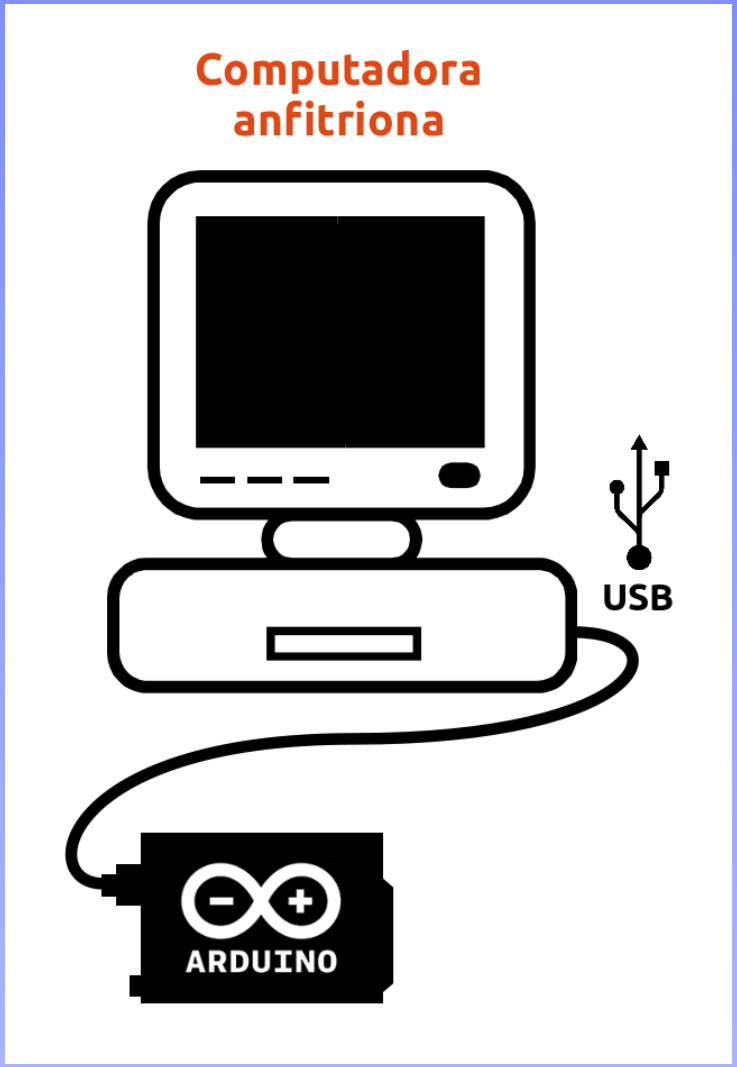
$$Distancia(cm) = \frac{Tiempo(\mu s)}{29.2 \cdot 2}$$

CÓDIGO CONEXIÓN BÁSICA DEL SENSOR

```
const int Trigger = 12;   //Pin digital 12 para el Trigger del sensor
const int Echo = 13;   //Pin digital 13 para el Echo del sensor

void setup() {
  //Inicialización de la comunicación serial
  Serial.begin (9600);
  //Inicialización de pines digitales
  pinMode(Trigger,OUTPUT);
  pinMode(Echo,INPUT);
}

void loop() {

 long tiempo, distancia;

 /* Hacer disparo*/
 digitalWrite(Trigger,LOW);
 delayMicroseconds(2);
 digitalWrite(Trigger,HIGH); //flanco ascendente
 delayMicroseconds(10);   //duracion de pulso
 digitalWrite(Trigger,LOW);    //flanco descendente

 /* Recepcion de respuesta*/
 tiempo = pulseIn (Echo, HIGH);

 /* Calculo de la distancia */
distancia = (tiempo/2) / 29;

/* imprime resultados*/
Serial.print(distancia);
Serial.println("cm");
delay (500);  //retardo para disminuir la frecuencia de las lecturas

}
```

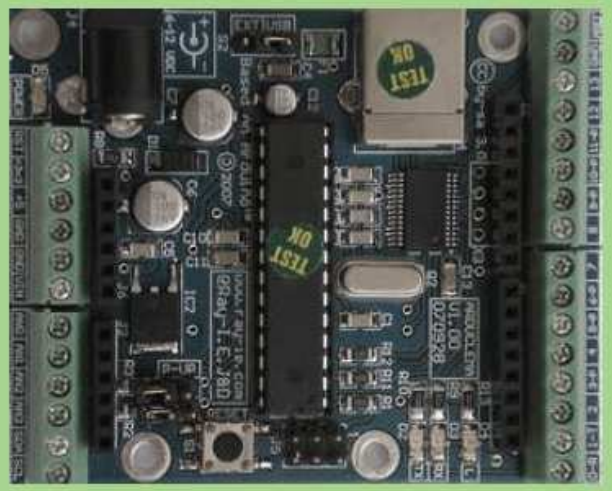
COMUNICACIONES CON EL EXTERIOR



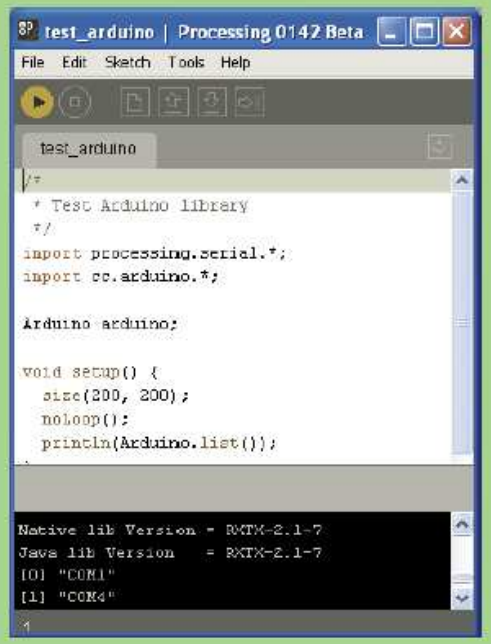


COMUNICACIÓN DE ARDUINO CON PC O DISPOSITIVO.

(Interacción PC entorno físico)



+



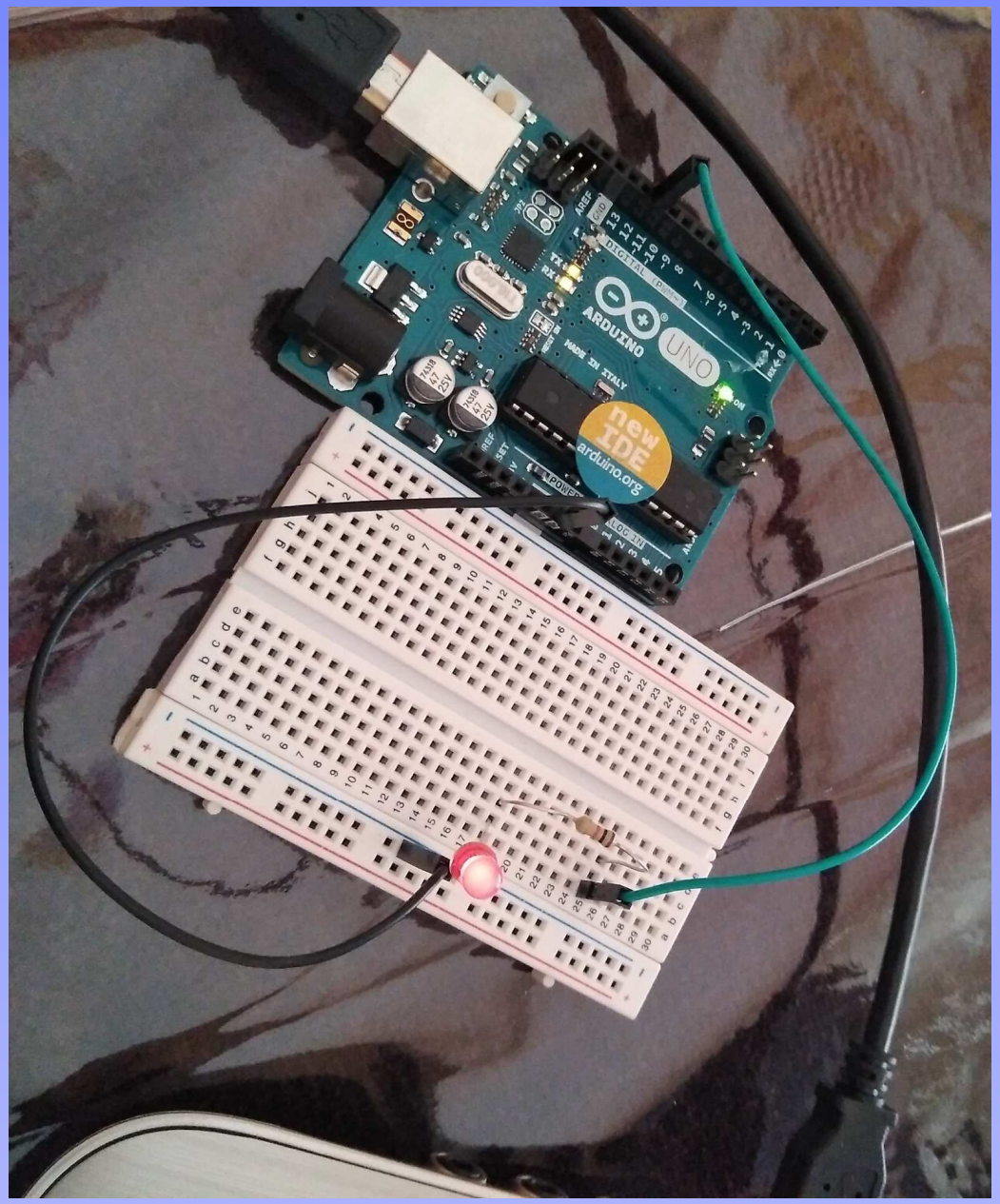
COMUNICACIÓN

INTERACCIÓN PROCESSING Y ARDUINO:
encendido y apagado de led y ratón

File Edit Sketch Tools Help

sketch_LEDrat_n

```
import processing.serial.*; //Importamos las librerias necesarias
import cc.arduino.*;

Arduino arduino; // Crea el objeto Arduino

int ledPin = 12; // Designa el numero de PIN para el LED

void setup() { //Configura el puerto y las señales con las que va a trabajar

 size(200, 200);
 arduino = new Arduino(this, Arduino.list()[0], 57600); // Configura el puerto como [0]

 arduino.pinMode(ledPin, Arduino.OUTPUT); // Configura el PIN12 como salida
 arduino.digitalWrite(ledPin, Arduino.HIGH); //Enciende el LED
}
void draw() { //Dibuja una ventana de interacción
 if (mousePressed == true) { //pregunta si se ha pulsado el botón del ratón
 arduino.digitalWrite(12,Arduino.LOW); // Si se ha pulsado apaga el LED
 } else {
 arduino.digitalWrite(12,Arduino.HIGH); // Si no esta pulsado enciende el LED
 }
}
```



ESQUEMA DEL MONTAJE

- ACTIVIDAD INTERACCION PROCESSING Y ARDUINO

- ACTIVIDAD CONEXIÓN DEL SENSOR, IMPRESIÓN DE DISTANCIAS.