ECOLAB

Autora:

Ana Lucía Vico Martínez

Huerto inteligente

- ¿Qué hace nuestro huerto?
  - Muestra por pantalla los datos más importantes
  - Mide muchos valores con los sensores que pueden afectar al cuidado de las plantas

¿Te animas a probarlo?

Huerto inteligente

INSPIRACIÓN DEL PROYECTO:

¿No te ha pasado alguna vez que…

- se te olvida regar una planta?
- la planta empieza a ponerse fea y no sabes por qué?

Muchas veces cuidamos plantas “a ojo”, sin saber realmente qué necesitan.

EcoLab nace de la idea de… ¿Y si pudiéramos crear un sistema que ayudase a cuidar una planta automáticamente?

Gracias a sensores y programación, nuestro huerto puede:

- detectar cuándo la tierra está seca
- medir la luz y la temperatura
- avisarnos si algo va mal

PASOS A SEGUIR

- **Prueba de Componentes:**

  Probaremos cada componente por separado para entender cómo funciona.

- **Diseño del funcionamiento del huerto:**

  Crearemos la lógica y el código que hará que nuestro huerto se automatice.

- **Montaje:**

  Juntaremos todas las piezas para dar forma a nuestro huerto.

- **Añadir Funcionalidades Extra:**

  Pensaremos en funciones nuevas y las añadiremos para personalizar nuestro huerto.

Oficina de Software Libre Universidad de Granada

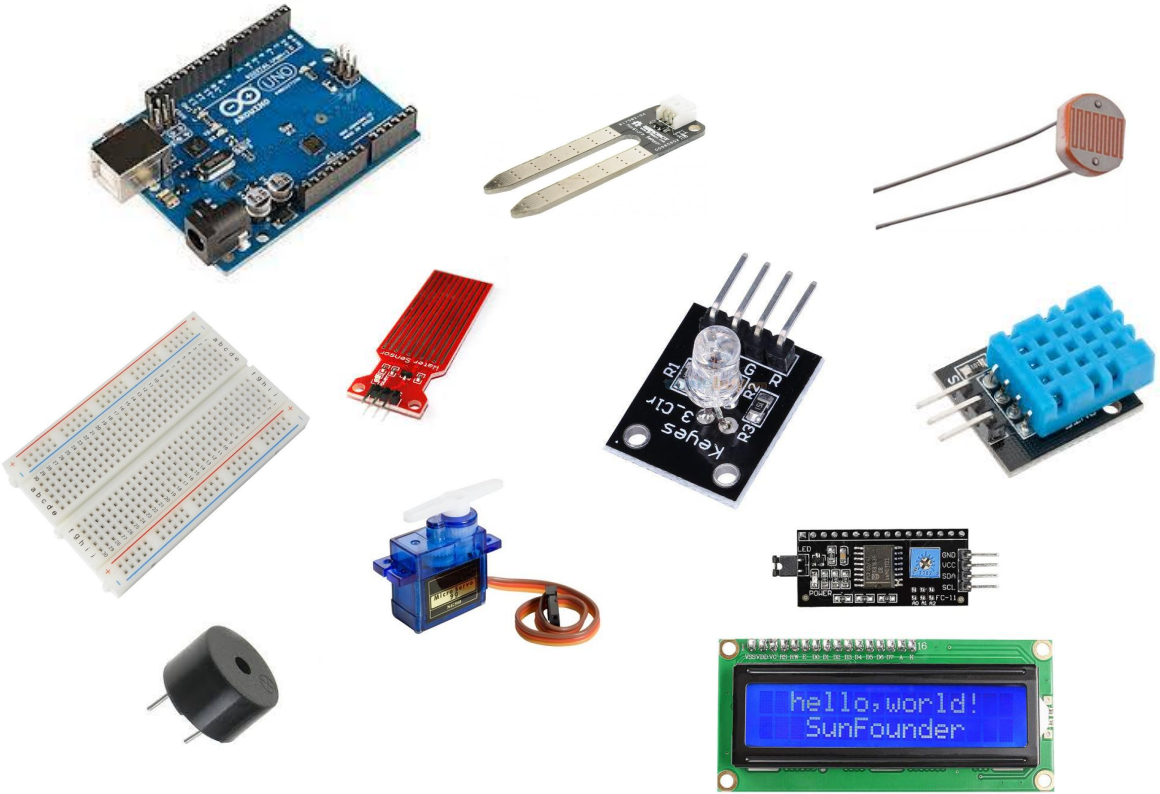
UNIVERSIDAD DE GRANADA

Hardware necesario para

EL HUERTO INTELIGENTE

Huerto inteligente

- ARDUINO UNO
- Pantalla LCD
- Sensores:
  - Fotoresistencia
  - Temperatura y humedad
  - Humedad de suelo
  - Lluvia
  - Nivel de agua
- Led RGB
- Servomotor
- Zumbador

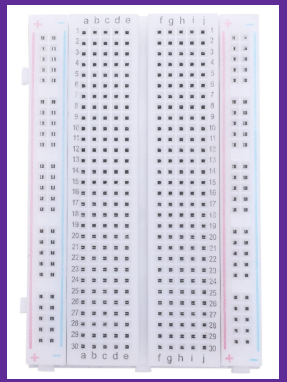


Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

Huerto inteligente

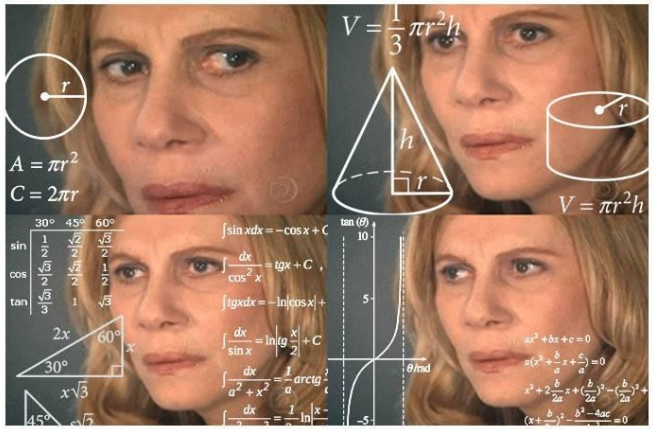
PROTOBOARD



PROTOBOARD

Por ahora la protoboard en la prueba de componentes no va a ser necesaria, pero es muy importante para el montaje del proyecto, porque conectar toooodos los elementos a la vez no sería posible sin ella.

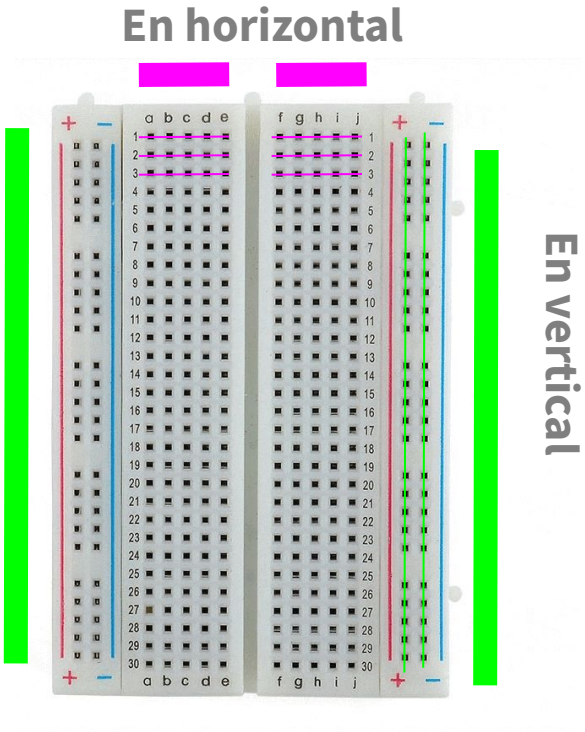
¿Cómo hacemos esto? ¿Todos los elementos tienen que conectarse al GND y 5V a la vez, pero no hay suficientes huecos para todos?



PROTOBOARD

¡¡¡No hay problema si utilizamos la **protoboard!!!**

Todas las filas del medio están conectadas horizontalmente y las dos columnas de los extremos están conectadas verticalmente. Normalmente, en los extremos pondremos la carga positiva y negativa, (5V y GND) y en medio para poder conectar otras cosas



Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

Huerto inteligente

Leds

LED

Un **LED** (Light Emitting Diode) es un componente electrónico que emite luz cuando se le aplica una corriente eléctrica.

- Los LEDs son conocidos por su eficiencia energética y larga vida útil.
- Pueden emitir luz en diferentes colores, dependiendo del material semiconductor utilizado en su construcción.

LED

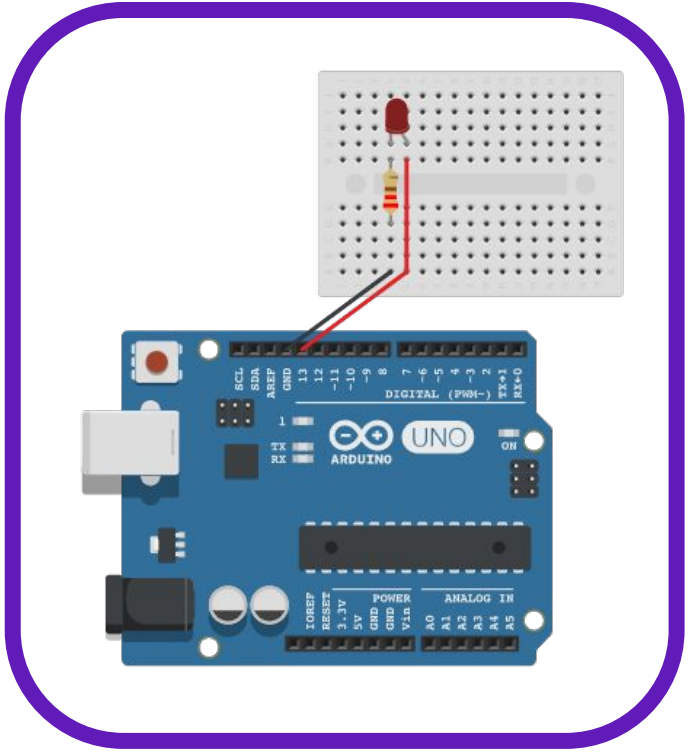
1. Para conectar un led necesitamos:

- El arduino UNO
- La protoboard
- Un led del color que queramos
- Una resistencia de 220Ω

Conexiones del LED:

- La patilla más larga se conecta con el **pin 13.** ■
- La patilla más corta se conecta **a la resistencia** y esta a **tierra** ■

(La resistencia limita la corriente que pasa a través del LED para evitar que se queme)



LED

Este led lo hemos conectado al pin 13. Por lo que queremos hacer el código para:

- Encender el led,
- Esperar un segundo,
- Apagar el led,
- Esperar un segundo.

Y esto se hace con:

pinMode(Numero_de_pin, tipo_de_elemento)

- Número de pin al que está conectado el led ___
- Tipo de elemento: **Output** emite, Input recibe información
  - ¿Que hace una luz, **emitir** o recibir? ____

Para que se encienda el led hay que “escribir” sobre él, utilizando:

- **digitalWrite**(Numero_de_pin, intensidad)
- Siendo **HIGH**, encendido y **LOW** apagado.

```
// Decidimos el pin del LED
int ledPin = 13;

void setup() {
     // Configurar el pin del LED
     pinMode(_______, _______);
}
void loop() {
     // Encender el LED
     digitalWrite(_______, _______);
     delay(1000); // Esperar un
segundo
     // Apagar el LED
     digitalWrite(ledPin, _______);
     delay(1000); // Esperar un
segundo
}
```

Huerto inteligente

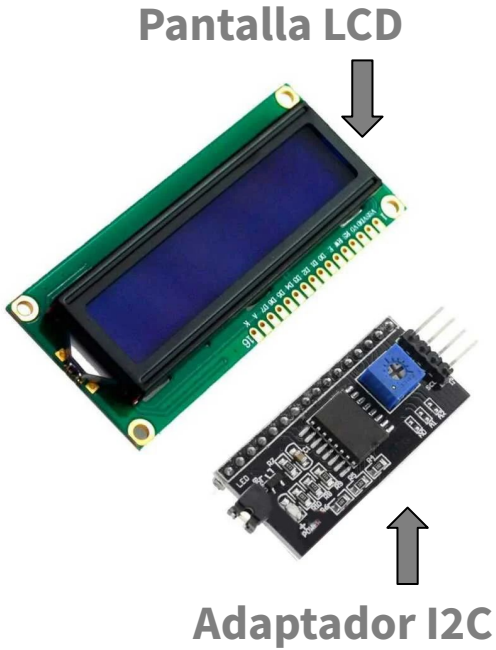
Pantalla LDC

Pantalla LCD

Una pantalla LCD (Liquid Crystal Display) es como una pantallita que muestra texto. En este proyecto usamos una LCD de 16 columnas y 2 filas, lo que significa que puede mostrar hasta 32 caracteres.

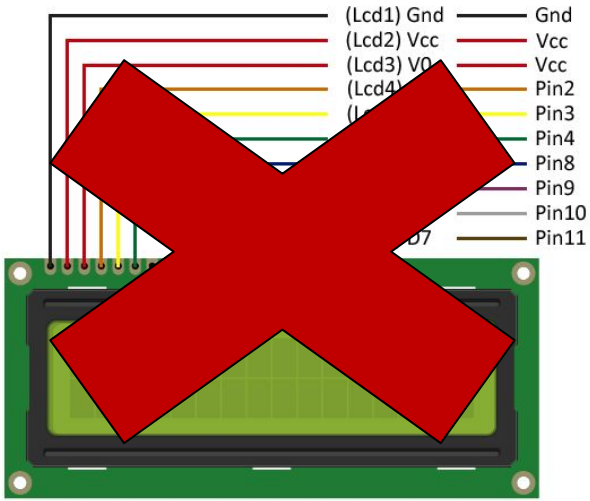
Sirve para mostrar mensajes como "¡Bienvenida!", “Planta OK”, "Necesita Aguar", etc., así que es super útil para que te diga lo que está pasando en cada momento

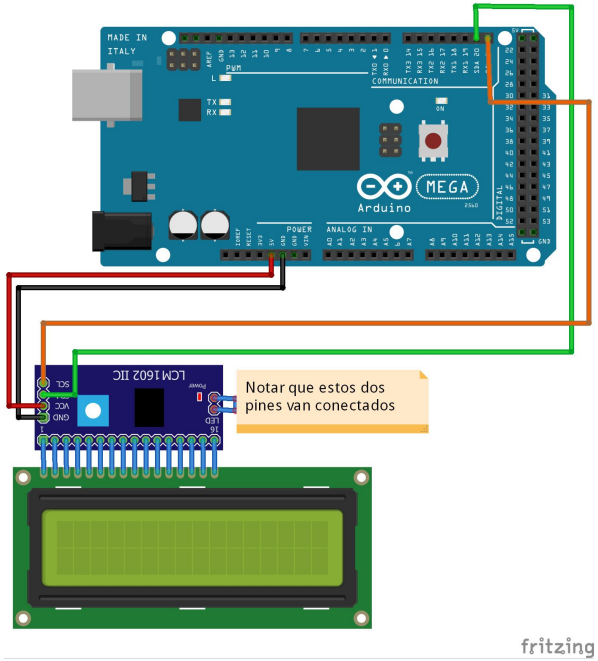
- Además, nuestra pantalla tiene un adaptador I2C, para que solo necesitemos 4 cables.



Pantalla LDC

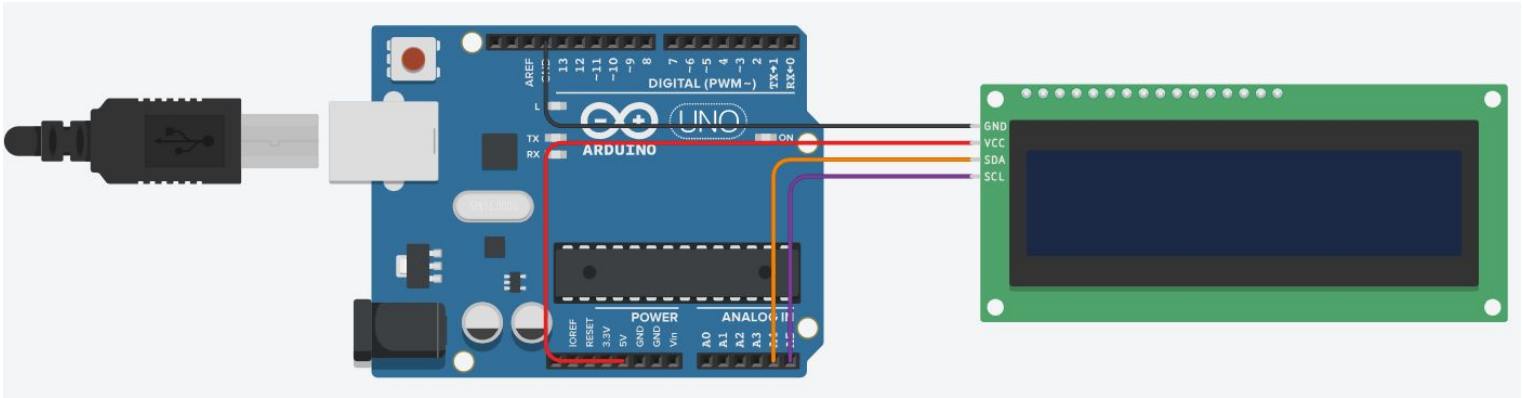
- La pantalla que vamos a utilizar: 16x02
  - con un controlador para no malgastar pines.





Juguemos con la pantalla

- Tiene 4 pines:
  - **SCL** y **SDA** conexión analógica
  - **+5V** proporciona energía a los leds
  - **GND** Toma de tierra



Pantalla LCD

Para utilizar la pantalla LCD hace falta incluir las librerías necesarias:

- Hay **dos formas de incluir librerías** a la biblioteca de nuestro arduino.
  - Si tenemos el archivo comprimido con extensión .zip
  - Buscarlas en el repositorio de Arduino

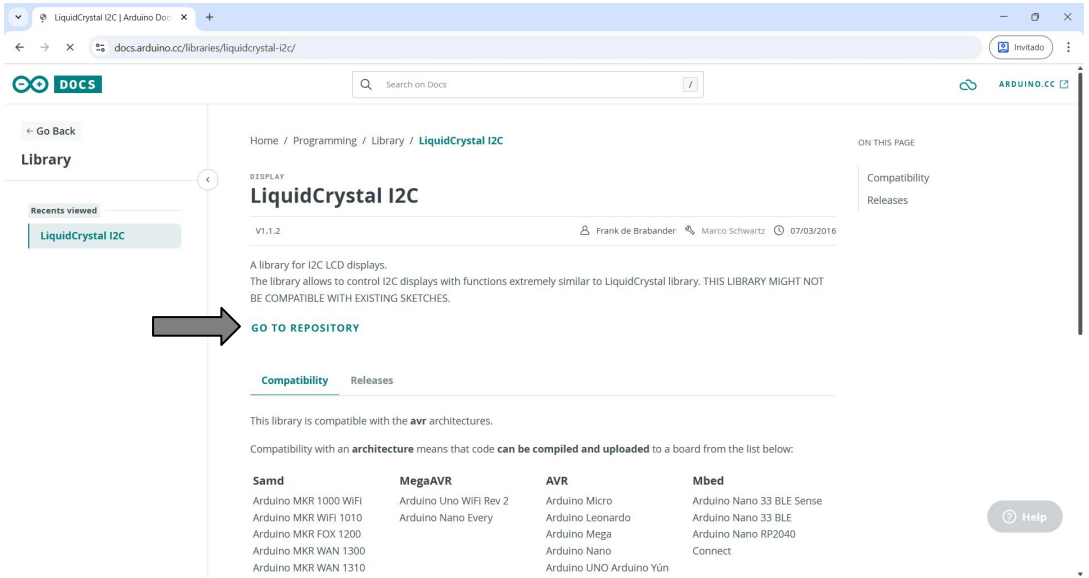
- Para la pantalla tenemos el archivo comprimido con el nombre **Arduino-LiquidCrystal-I2C.zip**
- La **descargamos** y la incluimos de la 1º forma.

Pantalla LCD

1. Incluir en el IDE la librería LiquidCrystal.h

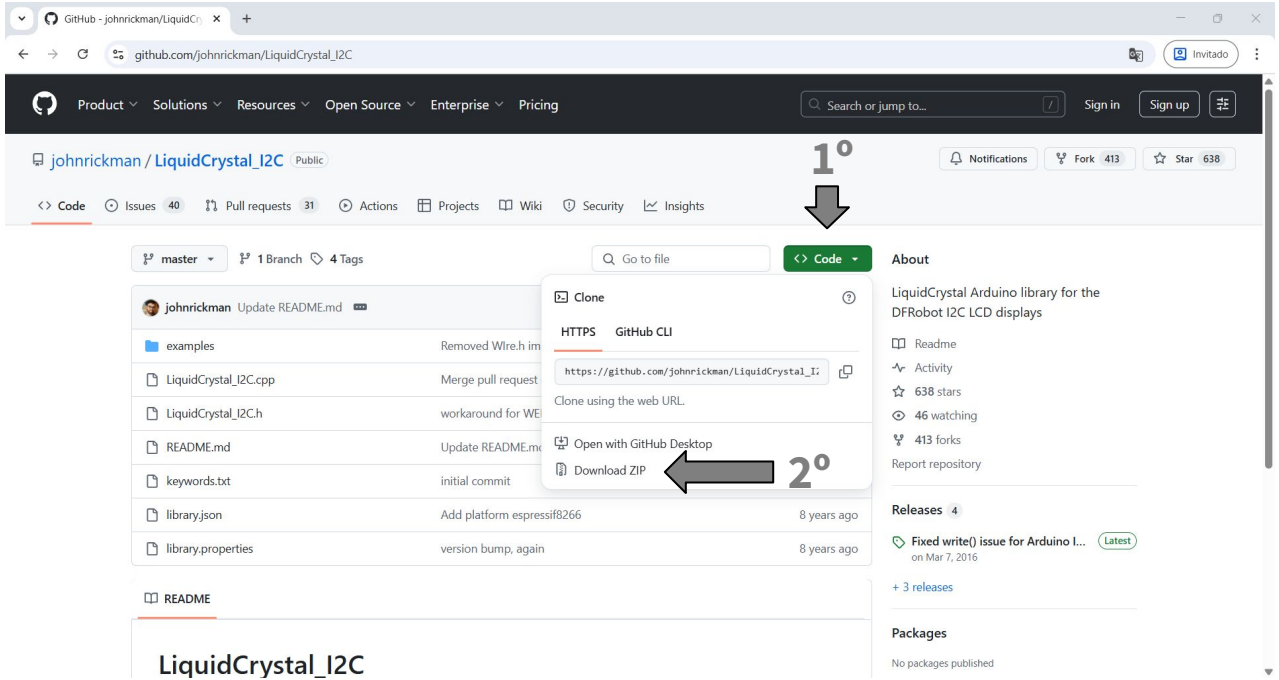
Vamos al enlace y descargamos la librería haciendo clic en download

https://www.arduino.cc/reference/en/libraries/liquidcrystal-i2c/



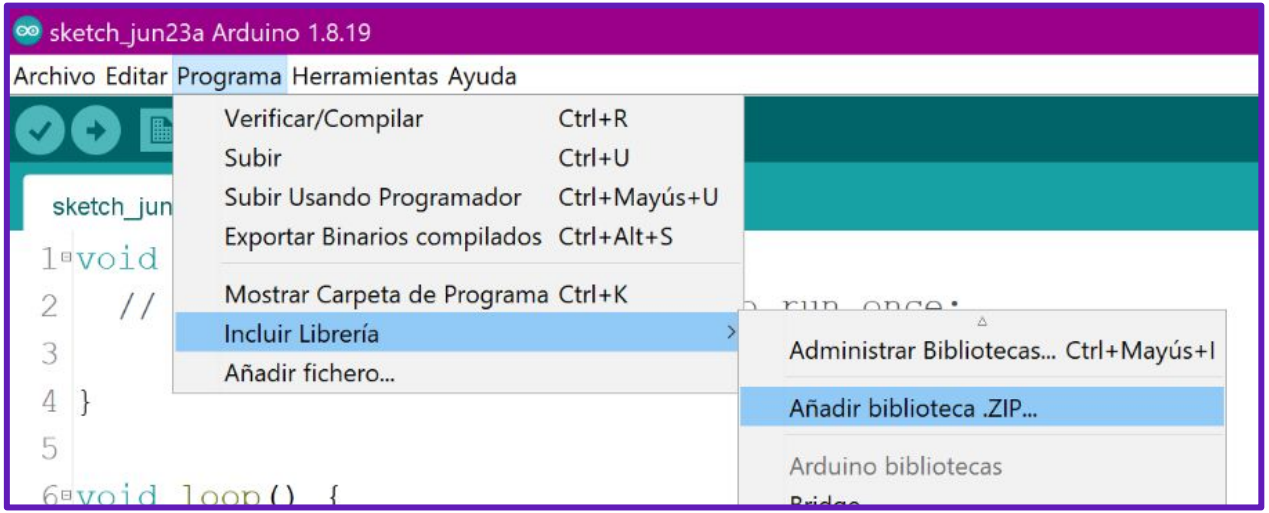
PANTALLA LCD

1. Incluir en el IDE la librería LiquidCrystal.h



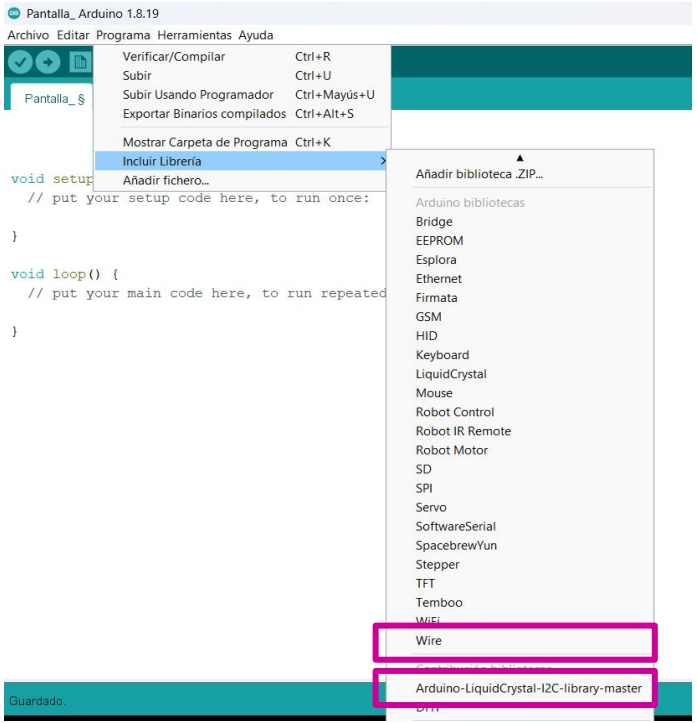
Pantalla LCD

1. Incluir en el IDE la librería LiquidCrystal.h



Pantalla LCD

2. **Para poder usar la librería tenemos que añadirla en el archivo. Vamos a añadir también wire.**



Pantalla LCD

3. Declaramos un objeto basándonos en nuestra pantalla

- Para ello utilizaremos la siguiente **función de la librería LiquidCrystal_I2C** que acabamos de incluir.
  - **LiquidCrystal_I2C lcd(0x27, 16, 2);**
- Se inicializa con:
  - 0x27 porque se inicializa en esa dirección
  - 16 = número de caracteres por línea
  - 2 = número de líneas

- Vale pero, ¿qué tenemos que hacer y cómo? Vaya locura no entendí nada…
  - Poco a poco, paso a paso noseque

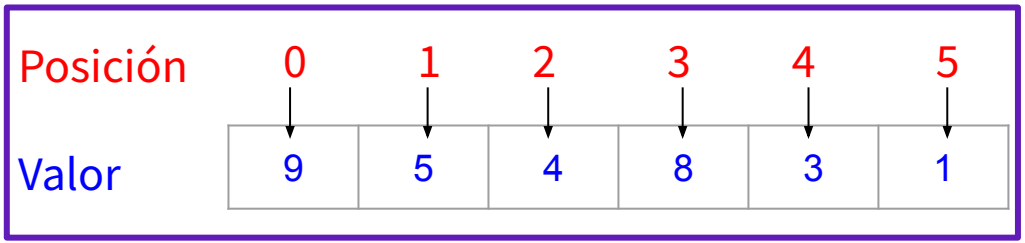
4. Aprendemos a usar la librería para utilizar cada punto de nuestra pantalla.

- ❏ Antes de empezar a usar las funciones de la librería necesitamos saber cómo usar las posiciones de cada línea.

¿Qué es un vector?

4.1 ¿Qué es un vector?

- El vector tiene **posiciones**
- La primera posición **siempre empieza en 0**
- **Cada posición** tiene **un valor** (en la imagen, la posición 0 tiene valor 9, la posición 4 tiene valor 3…)
- El tamaño máximo del vector es el número de posiciones que tiene. En este ejemplo el vector tiene **6 posiciones que van del 0 al 5**.
- En nuestra pantalla, habrá dos vectores, que irán del… **0 al 15**



Pantalla LDC

4. Aprendemos a usar la librería para mostrar mensajes por pantalla:

- Para poder utilizar nuestra pantalla vamos a tener que aprender las funciones de la librería:
  - lcd.begin() se utiliza para inicializar la comunicación con la pantalla LCD.
  - lcd.backlight() enciende la retroiluminación de la pantalla LCD.
  - lcd.setCursor(0, 0) establece la posición del cursor en la columna 0 y la fila 0.
  - lcd.print("Hola") muestra el texto "Hola" en la pantalla LCD.
  - lcd.clear() borra el contenido de la pantalla LCD.

¿Creéis que podríais hacer que la pantalla muestre en la primera línea “Hola”?

Pantalla LDC

4. Aprendemos a usar la librería para mostrar mensajes por pantalla:

- lcd.begin()
- lcd.backlight()
- lcd.setCursor(0, 0)
- lcd.print("Hola")
- lcd.clear()

pantalla

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);
  // Inicia el LCD en la dirección 0x27, con 16 caracteres y 2 líneas

void setup()
{
}
```

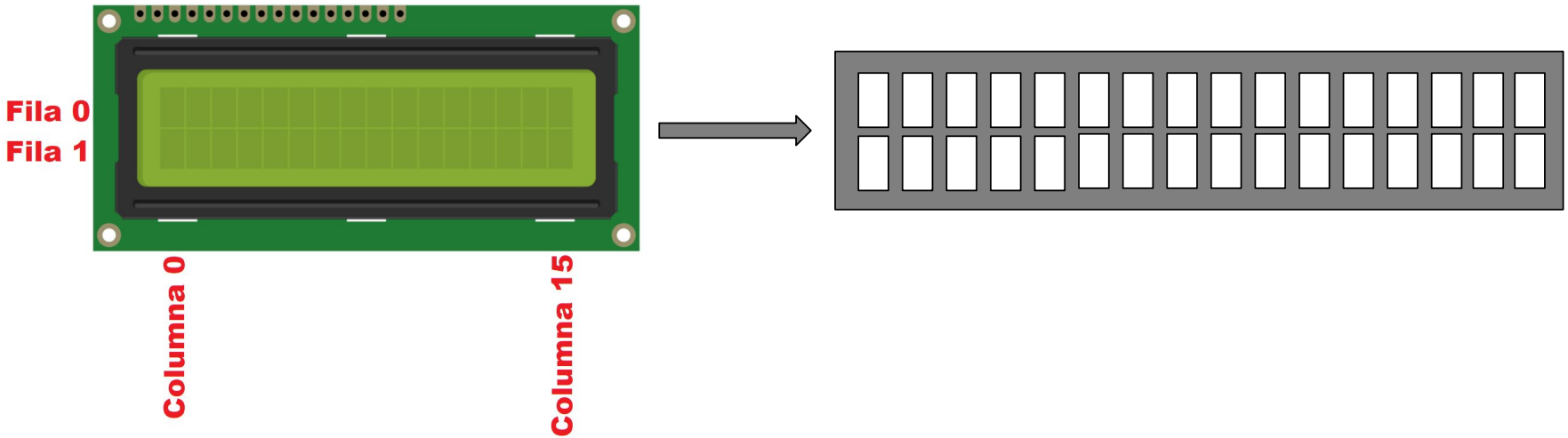
```
loop{
      inicio la pantalla
      luz de fondo
      empiezo a escribir en línea
      añado el texto que quiera
      espero 2 segundos
      limpio pantalla
}
```

Pantalla LCD

4. Aprendemos a usar la librería para mostrar mensajes por pantalla:

   EXTRA:

- ¿Molaría poder poner iconos un poco más complicados?
- Vamos a entrar más en profundidad en cómo se muestran las letras en pantalla:

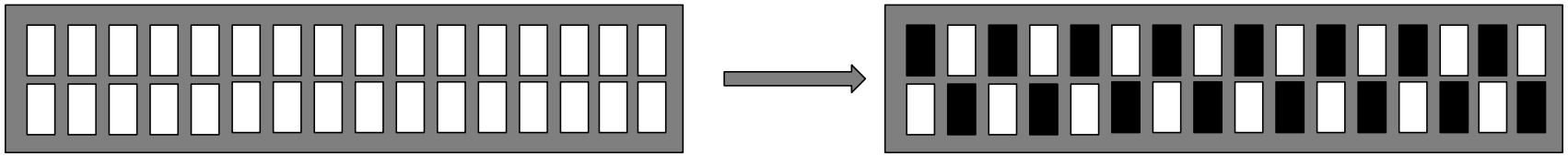


Pantalla LCD

4. Aprendemos a usar la librería para mostrar mensajes por pantalla:

   EXTRA:

- Si quisiéramos poner una especie de ajedrez, ¿Qué cuadrados colorearemos?



Oficina de Software Libre Universidad de Granada

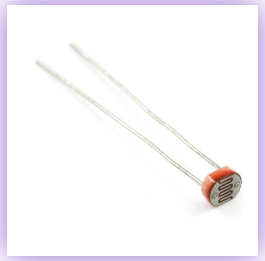
UNIVERSIDAD DE GRANADA

Huerto inteligente

Los sensores

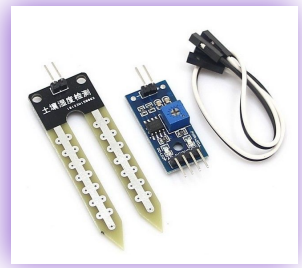
Sensores

Lluvia

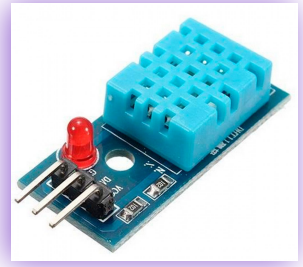


Fotoresistencia

Nivel de agua



Humedad de tierra



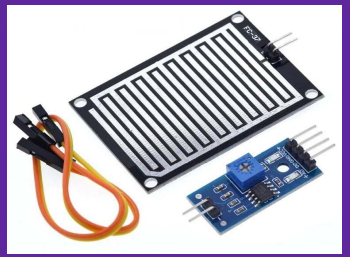
Humedad y Temperatura

Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

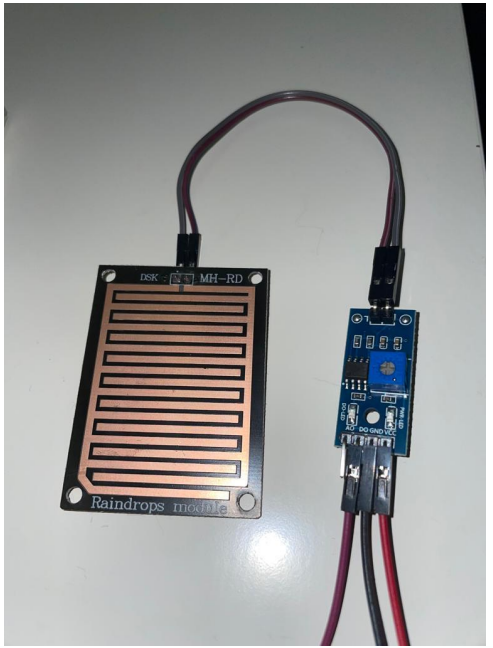
Conectando el hardware de Huerto inteligente

Sensor de Lluvia



Sensor de lluvia

- Este sensor detecta la presencia de lluvia…
- **¿Cómo?**
  - **Por la variación de conductividad**



Sensor de lluvia

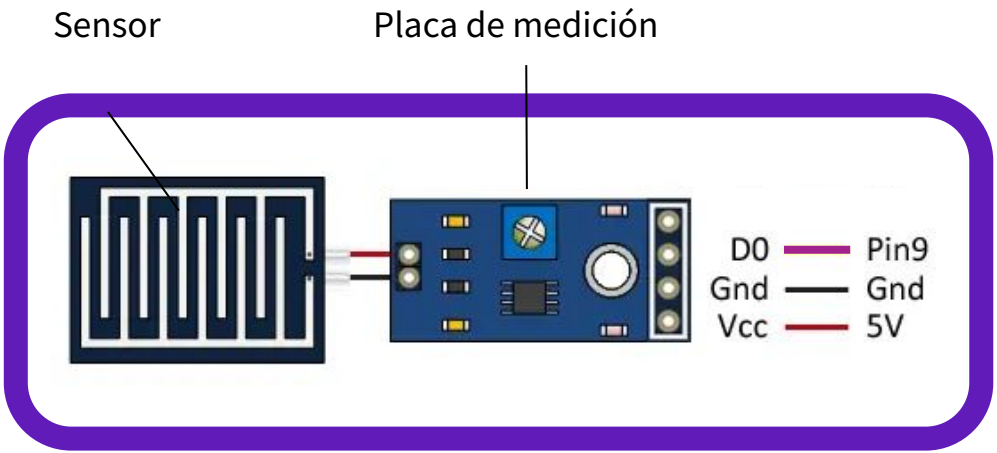
Pero … ¿qué es la **conductividad**?

Es la **propiedad física** que mide la capacidad de un material o medio para **transmitir la corriente eléctrica o el calor**

Sensor de lluvia

Tiene dos partes:

- El **sensor** que detecta la presencia de agua
- La **placa de medición** que convierte la detección en señales eléctricas

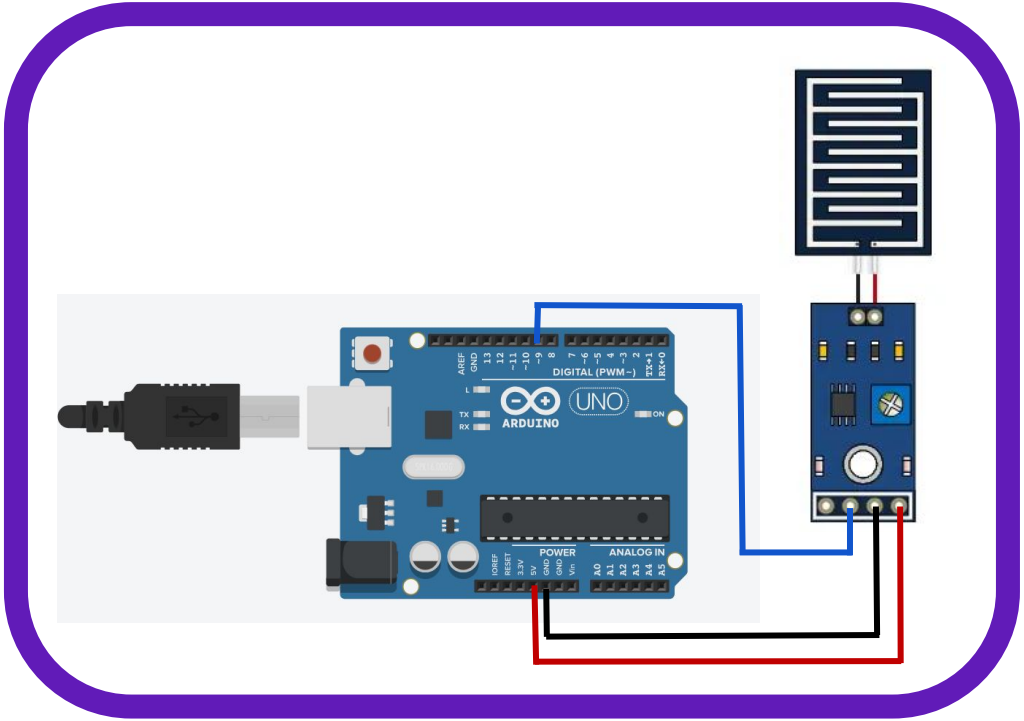


Conectamos el sensor a la placa de medición, y la placa de medición con el arduino

Sensor de lluvia

1. Conectamos el sensor al arduino

   - ❑ **Pin gnd - GND**
   - ❑ **Pin DATA - 6**
   - ❑ **Pin vcc - VCC**



Sensor de lluvia

2. Queremos usar el pin en el que vamos a conectarlo… en este caso el Pin 9, esto se realiza en el setup().

```
const int pin = 6;

void setup() {
        Serial.begin(9600);   //iniciar puerto serie
        pinMode(pin, INPUT);  //definir pin como entrada
}
```

¿Y cómo sabemos si hay agua o no?

Sensor de lluvia

Entonces hay que utilizar **CONDICIONALES.**
En programación, los condicionales son como las decisiones.
por ejemplo:

```
si (tienes hambre) {
    come una fruta;
} sino si (tienes sed) {
    bebe agua;
} sino {
    descansa y diviértete;
}
```

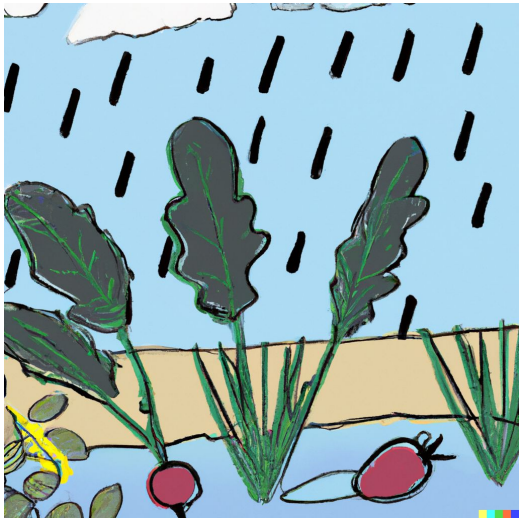
¿Tiene sentido?

Sensor de lluvia

- Pero en programación no podemos usar “si”, “si no” … hay que hacerlo en inglés…

  ¿Cómo se dice “**Si** tienes hambre, come algo” en inglés?

  If you are hungry, eat something



- Pues esto es exactamente igual:

```
if (está lloviendo){
muestra por pantalla que está lloviendo
}
```

Sensor de lluvia

Y ahora nos quedarían los **símbolos de comparación.**

- `>` Mayor que
- `<` Menor que
- `==` Igual a
- `!=` Distinto de

3 es … que 6
6 es … que 6

```
int edad = 15;
if (edad > 18) {
   // Acción si la edad es mayor a 18
   imprimir "Eres mayor de edad"
} else {
   // Acción si la edad no es mayor a 18
   imprimir "Eres menor de edad"
}
```

Sensor de lluvia

3. Utilizamos el sensor:

Cosas que tenéis que tener en cuenta para poder hacer vuestra función loop():

- Para poder guardar lo que dice el sensor : **“digitalRead(pin)”**
- ¿Os acordáis cómo funcionaba este sensor? ¿**LOW** es que hay lluvia? ¿O es **HIGH**?
- Para escribir por pantalla se utiliza: **Serial.println("mensaje");**
- No puede estar leyendo toooodo el rato por lo que para que espere 1 segundo: **delay(1000);**

sensorLluvia

```
const int sensorPin = 6;

void setup() {
  Serial.begin(9600);   //iniciar puerto serie
  pinMode(sensorPin, INPUT);  //definir pin como entrada
}

void loop(){
```

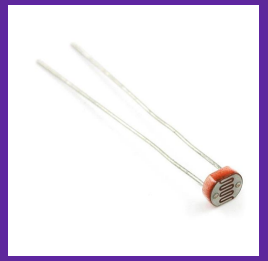
```
Crea la variable que guardará el valor del sensor
si hay lluvia{
     enseño por la consola que está lloviendo
}
espero un tiempo
```

Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

Conectando el hardware de Huerto inteligente

Sensor Fotoresistencia



Sensor fotoresistencia

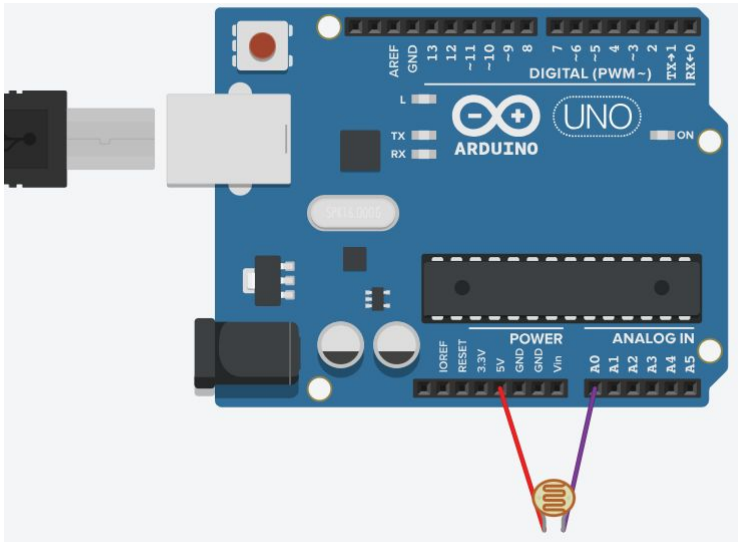
Sirve para detectar la cantidad de luz que hay en el entorno.

- Cuando hay **poca luz**: paso de **poca corriente eléctrica**
- Cuando hay **mucha luz**: paso de **mayor corriente eléctrica**

Sensor fotoresistencia

1. Conectamos el sensor al arduino

   - ❑ S - **Pin A0**
   - ❑ VCC - **5V**



Sensor fotoresistencia

2. Queremos iniciar el pin en el que vamos a conectarlo… en este caso el Pin A0, esto se realiza en el setup().

```
const int pin = A0;

void setup() {
        Serial.begin(9600);   //iniciar puerto serie
}
```

Sensor fotoresistencia

3. Utilizamos el sensor:

Vamos a mostrar un mensaje diciendo la cantidad de luz que hay actualmente en el loop():

- **“analogRead(pin)”**
- **Serial.println("mensaje");**
- **delay(1000);**

fotoresistencia

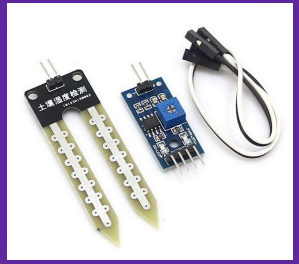
```
const int pinFotoresistencia = A0;

void setup() {
  Serial.begin(9600);
}

void loop() {
Crea variable que almacene valor del sensor

imprime por terminal un mensaje
imprime por terminal el valor obtenido

Espera un segundo
}
```

Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

Conectando el hardware de Huerto inteligente

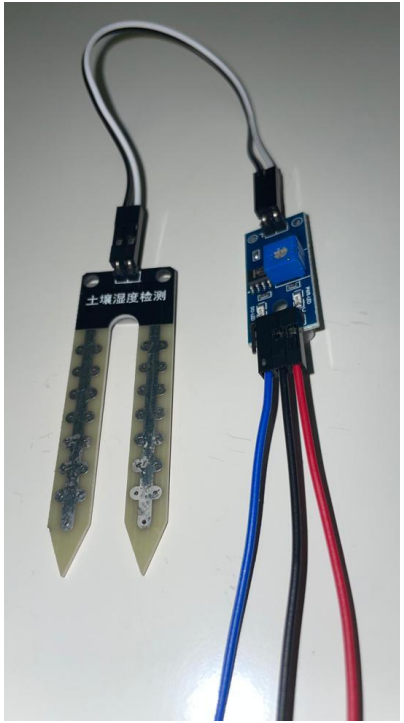
Sensor de Humedad de la tierra



Sensor de Humedad del suelo

Sirve para detectar la humedad que hay en el suelo.

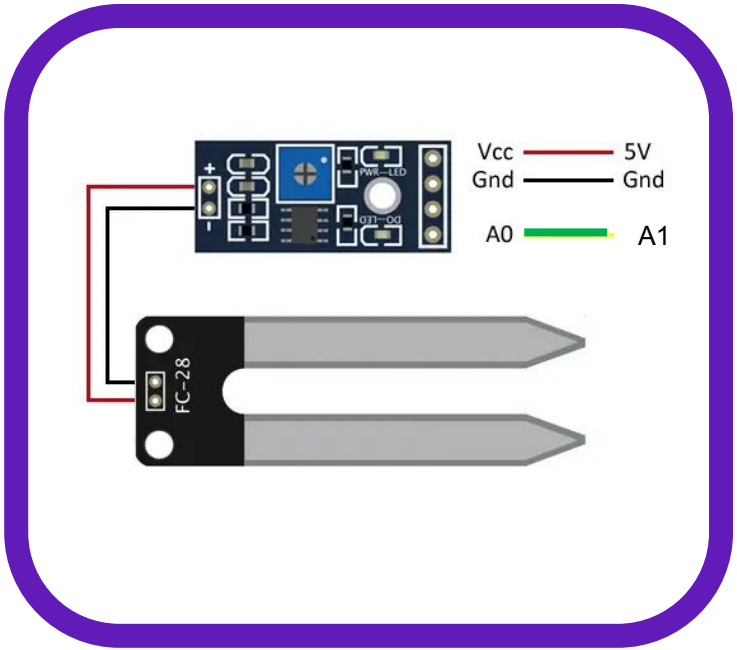
- Los valores van desde **0 sumergido en agua**, a **1023 en un suelo muy seco**.
- Un suelo **ligeramente húmedo** daría valores típicos de **600-700**.
- Un **suelo seco** tendrá valores de **800-1023**.



Sensor de Humedad del suelo

Tiene dos partes:

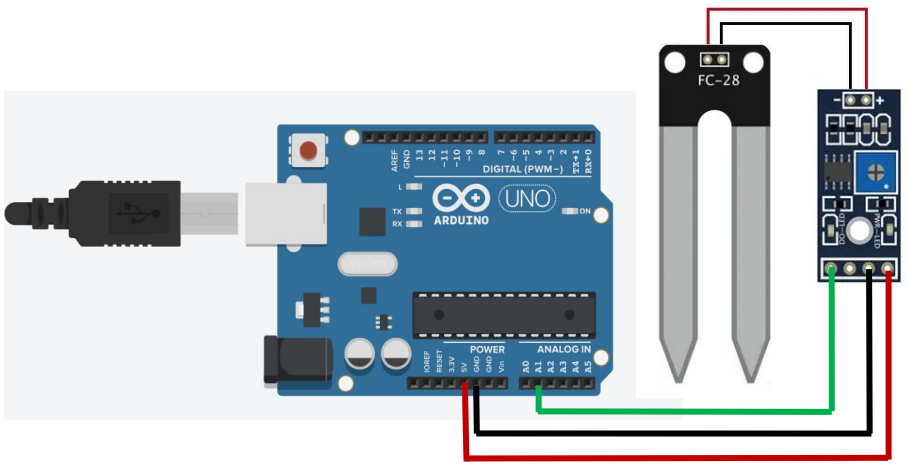
- El **sensor** que detecta la presencia de agua
- La **placa de medición** que convierte la detección en señales eléctrica



Sensor de Humedad del suelo

1. Conectamos el sensor al arduino

   - ❑ S - **Pin A1**
   - ❑ GND - **Tierra**
   - ❑ VCC - **5V**



Sensor de Humedad del suelo

2. Utilizamos el sensor:

Vamos a mostrar un mensaje diciendo la cantidad de luz que hay actualmente en el loop():

- **“analogRead(pin)”**
- **Serial.println("mensaje");**
- **delay(1000);**

sensorHumedadSuelo §

```
const int sensorPin = A0;

void setup() {
   Serial.begin(9600);
}

void loop()
{
```

```
Crea variable que almacene valor del sensor

Cuando el valor de la humedad sea superior a 800{
     imprime por terminal un mensaje
}

Espera un segundo
```

```
}
```

Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

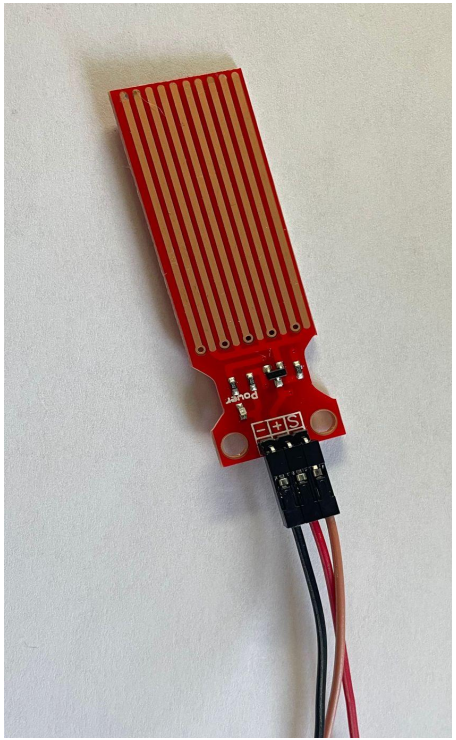
Conectando el hardware de Huerto inteligente

Sensor de Nivel de agua

Sensor de Nivel de agua

Funciona prácticamente igual que el sensor de lluvia.

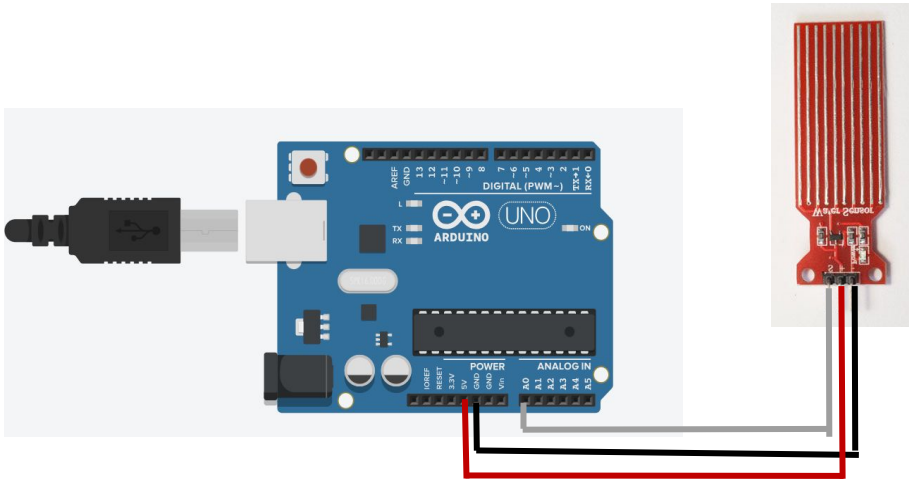
- **Sin agua** en contacto con las placas, el circuito está **abierto** y no hay flujo de corriente.
- **Con agua** en contacto con las placas, se **cierra el circuito** y si hay flujo de corriente.



Sensor de Nivel de agua

1. Conectamos el sensor al arduino

   - ❑ S - **Pin A2**
   - ❑ GND - **Tierra**
   - ❑ VCC - **5V**



Sensor de Nivel de agua

2. Utilizamos el sensor:

Vamos a mostrar un mensaje diciendo la cantidad de luz que hay actualmente en el loop():

- **“analogRead(pin)”**
- **Serial.println("mensaje");**
- **delay(1000);**

SensorNivelAgua §

```
const int sensorPin = A2;

void setup() {
  Serial.begin(9600);
}
```

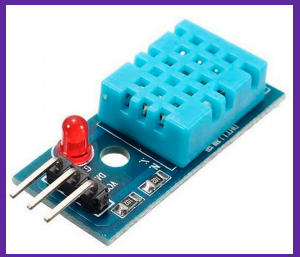
```
Crea la variable que almacene el valor del sensor

Imprime por consola un mensaje
Imprime por consola el valor obtenido

Espera 1 segundo
}
```

Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

Conectando el hardware de Huerto inteligente

Sensor de Humedad y temperatura



Pantalla LDC

- Hay **dos formas de incluir librerías** a la biblioteca de nuestro arduino.
  - Si tenemos el archivo comprimido con extensión .zip
  - Buscarlas en el repositorio de Arduino

- Para la pantalla tenemos el archivo comprimido con el nombre **DHT_sensor_library.zip**
- La **descargamos** y la incluimos de la 1º forma.

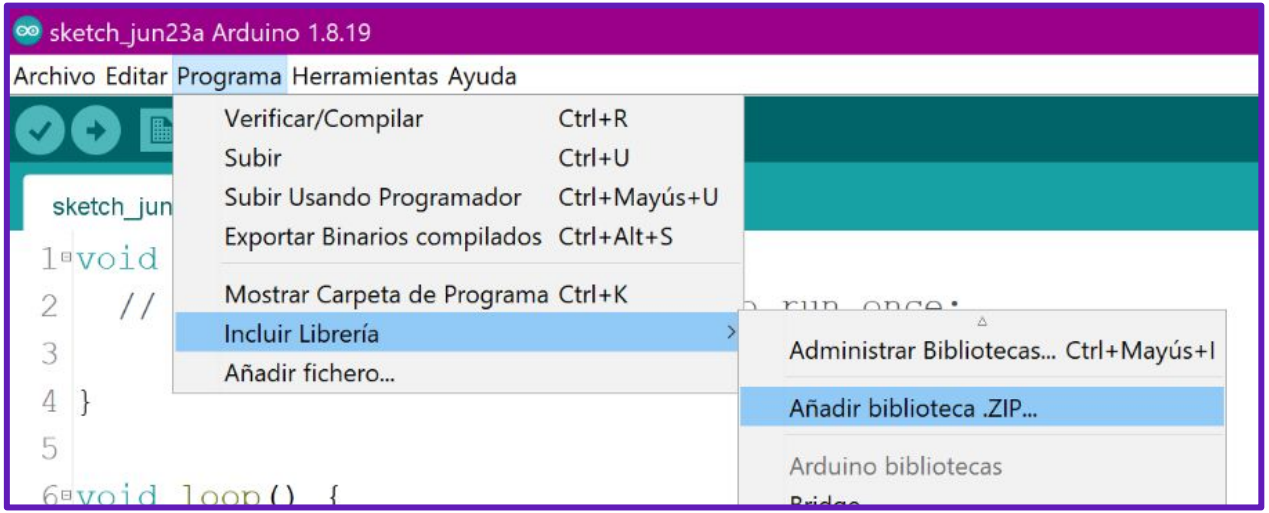
1. Incluir en el IDE la librería DHT_sensor_library.h

Vamos al enlace y descargamos la librería haciendo clic en download

https://downloads.arduino.cc/libraries/github.com/adafruit/DHT_sensor_library-1.4.3.zip

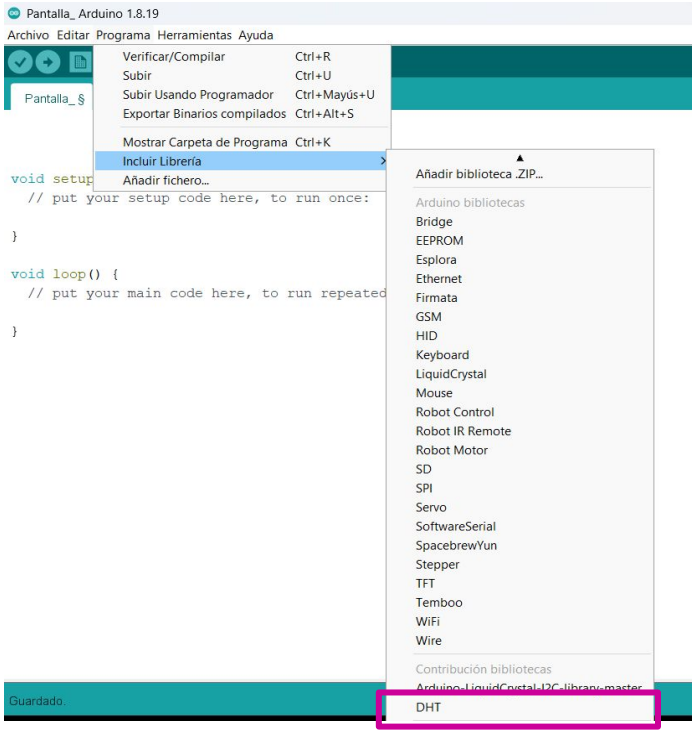
Sensor Humedad y Temperatura

1. Incluir en el IDE la librería **DHT_sensor_library.h** ✓



Sensor Humedad y Temperatura

2. Para poder usar la librería tenemos que añadirla en el archivo.




```
sketch_jul03a §
#include <DHT.h>
#include <DHT_U.h>

#include <dht.h>

void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Sensor Humedad y Temperatura

3. Declaramos el tipo de sensor que vamos a utilizar

- Existen dos tipos de sensores DHT:
  - 11 - Es el más sencillo, peores características técnicas pero más económico. Se utiliza para proyectos que requieran mediciones pero no necesitan gran precisión.
  - 22 - Es más complejo, pero no llega a ser en absoluto un sensor de alta precisión. Se puede utilizar en proyectos reales de monitorización o registro.

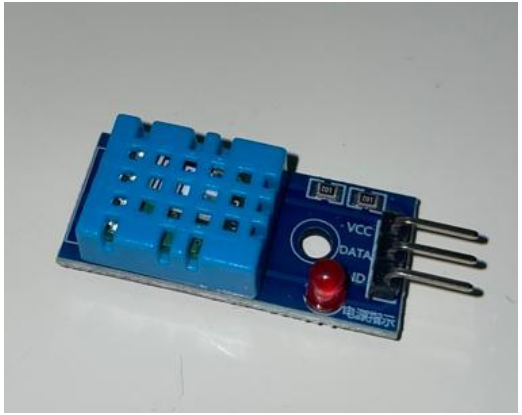
Utilizaremos el 11 por lo que lo definimos, y creamos un objeto de esta librería:

```
#include <DHT.h>
#include <DHT_U.h>

#define DHTTYPE DHT11   // DHT 11

const int DHTPin = 5;

DHT dht(DHTPin, DHTTYPE);
```

Sensor Humedad y Temperatura

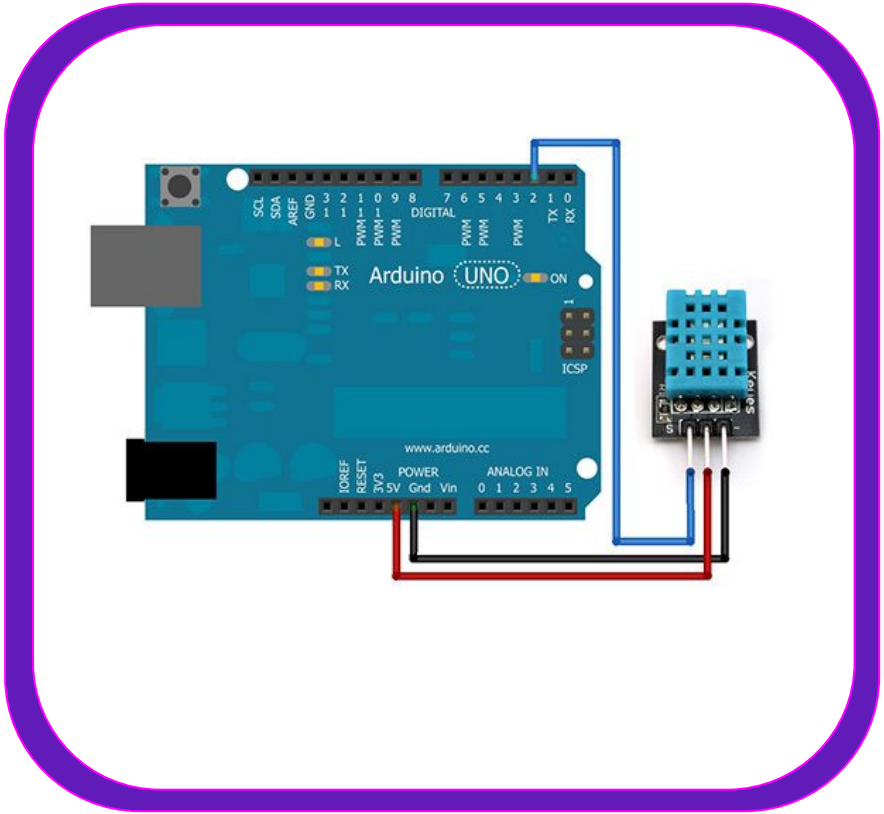
Sensor de Humedad:

- Medición de temperatura: 0 a 50 grados, precisión de 2ºC
- Medición de humedad: 20 a 80%, precisión del 5%.
- Frecuencia de muestreo: 1 muestra por segundo



Sensor Humedad y Temperatura

- Pin gnd - GND
- Pin DATA - D5 (pin digital 5)
- Pin vcc - VCC



Sensor Humedad y Temperatura

2. Utilizamos el sensor:

Vamos a mostrar un mensaje diciendo la cantidad de luz que hay actualmente en el loop():

- **“analogRead(pin)”**
- **Serial.println("mensaje");**
- **delay(1000);**

Funciones de esta librería:

- dht.readHumidity()
- dht.readTemperature()

pruebaHumedad

```
#include <DHT.h>
#include <DHT_U.h>

#define DHTTYPE DHT11   // DHT 11

const int DHTPin = 5;

DHT dht(DHTPin, DHTTYPE);

void setup() {
```

Crea dos variables, una para almacenar humedad y otra temperatura.

Leemos los valores del objeto de la librería y los almacenamos en las variables

Muestra por pantalla un mensaje que identifique qué valor muestras y el valor que marca el sensor

Esperamos 1 segundo

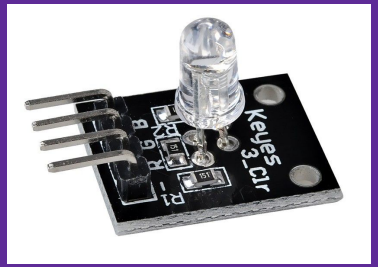
```
}
```

Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

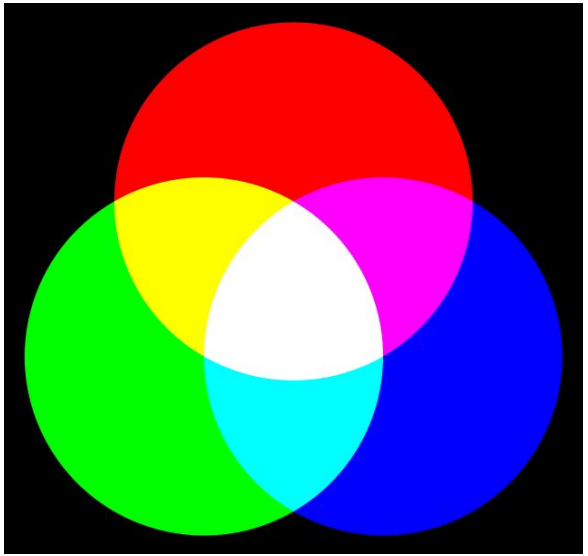
Conectando el hardware de Huerto inteligente

Led RGB



Módulo led RGB

- Un led RGB permite emitir luz de diferentes colores.
- **RGB = Red Green Blue** → Los tres colores primarios
  - Se pueden mezclar con distintas proporciones para generar una gama de colores



Módulo LED RGB

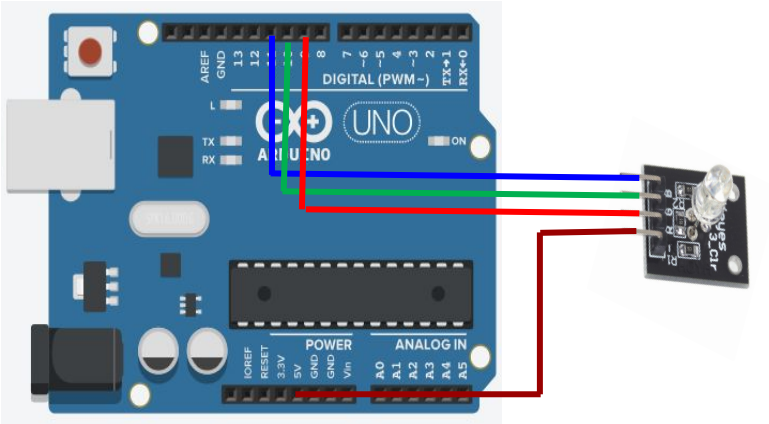
- Tenemos dos opciones de hacer que funcione:
  1. Escritura digital → Solo se pueden poner valores LOW y HIGH
  2. Escritura analógica → Se pueden poner valores en el rango de 0-255

	Digital	Analógico
Ventaja	Mayor disponibilidad de pines. Más flexibilidad para conectar otros componentes al Arduino	Mayor precisión en la mezcla de colores.
Desventaja	Menor precisión en la modulación de la intensidad	Mayor cantidad de pines requeridos

¿Qué preferís? ¿Analógico o digital?

Módulo LED RGB

- Y ¿Cómo se muestran los colores?
  - El módulo LED tiene 4 pines: uno para la corriente positiva, y tres para los colores (RGB).
- R - **9**
- G - **10**
- B - **11**
- VCC - **5V**



Módulo LED RGB

Vamos a encender el pin rojo:

```
// Definir los pines para cada color del módulo LED RGB
const int pinRojo = 9;
const int pinVerde = 10;
const int pinAzul = 11;

void setup() {
  // Configurar los pines como salidas
  pinMode(pinRojo, OUTPUT);
  pinMode(pinVerde, OUTPUT);
  pinMode(pinAzul, OUTPUT);
}

void loop() {
  // Encender el LED en color rojo
  digitalWrite(pinRojo, HIGH);
  digitalWrite(pinVerde, LOW);
  digitalWrite(pinAzul, LOW);

  delay(1000); // Esperar 1 segundo
  //...
}
```

¿Podríais encender el verde y el azul, esperando un segundo entre cada luz?

Módulo LED RGB

¿Y si probamos con analógico?

```
// Definir los pines para cada color del módulo LED RGB
const int pinRojo = 9;
const int pinVerde = 10;
const int pinAzul = 11;

void setup() {
  // Configurar los pines como salidas
  pinMode(pinRojo, OUTPUT);
  pinMode(pinVerde, OUTPUT);
  pinMode(pinAzul, OUTPUT);
}

void loop() {
  // Modulación de intensidad para el color rojo
  for (int i = 0; i <= 255; i++) {
    analogWrite(pinRojo, i);    // Establecer la intensidad del color rojo
    analogWrite(pinVerde, 0);   // Apagar el color verde
    analogWrite(pinAzul, 0);    // Apagar el color azul
    delay(10);
  }
}
```

¿Podríais encender el verde y el azul, esperando un segundo entre cada luz?

Oficina de Software Libre Universidad de Granada

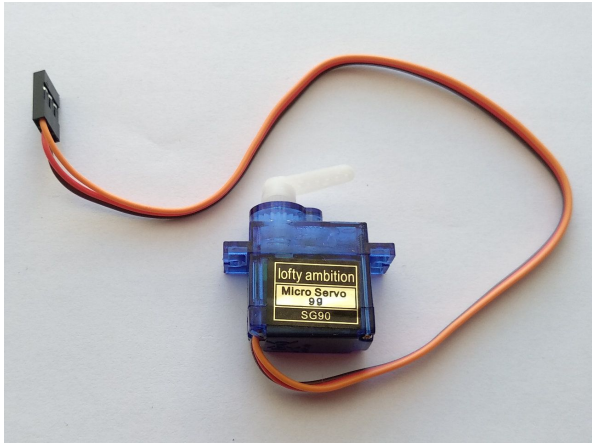
UNIVERSIDAD DE GRANADA

Conectando el hardware de Huerto inteligente

Servomotor

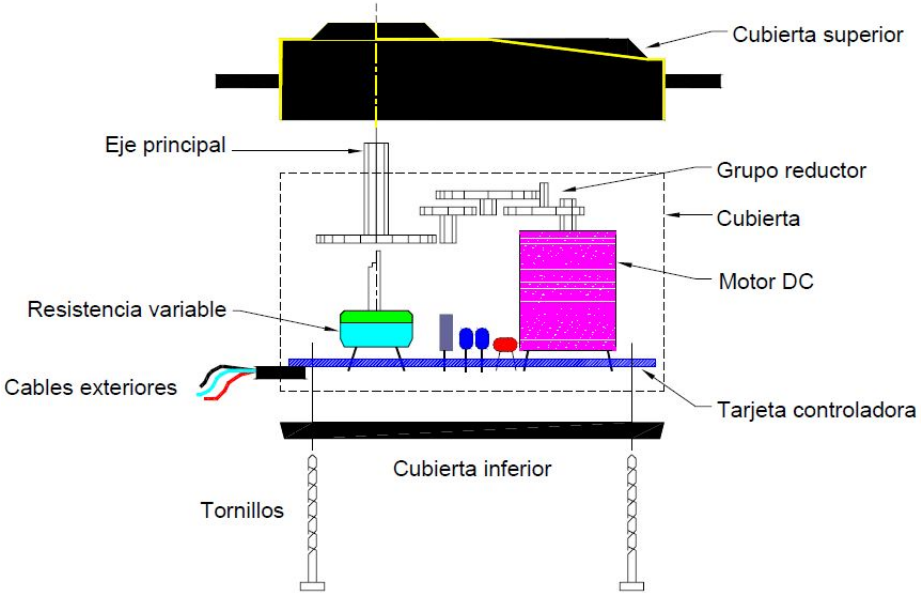
ServoMotor

- Un servomotor es un dispositivo electromecánico que combina un motor, una caja de engranajes y un circuito de control.
- El circuito de control envía una señal al servomotor para indicarle la posición deseada. El motor y la caja de engranajes se encargan de mover el eje del servomotor hasta alcanzar dicha posición.



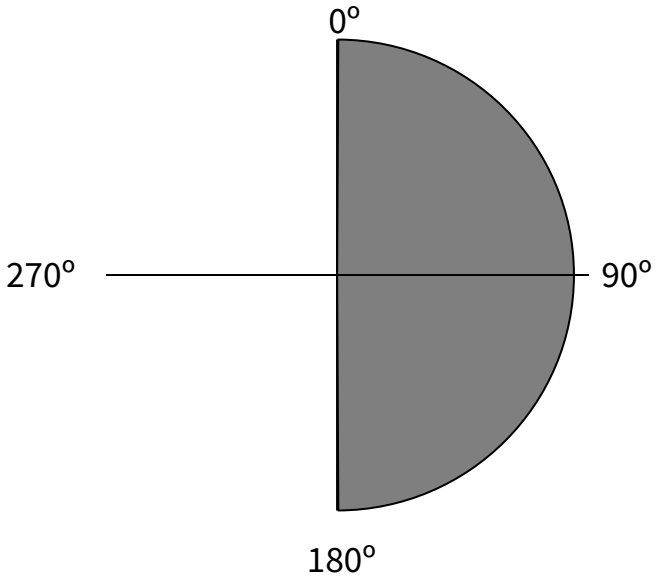
ServoMotor

- Un servomotor consta de un motor, una caja de engranajes, un potenciómetro de retroalimentación y un circuito de control.



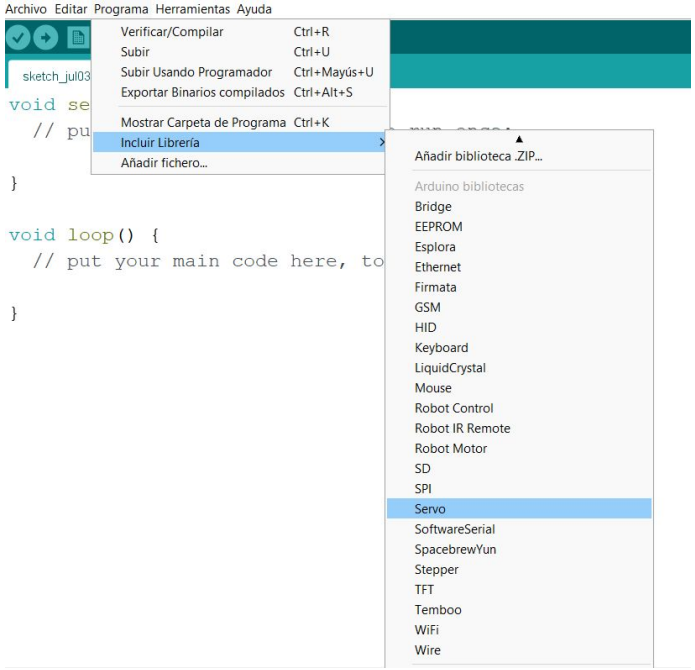
ServoMotor

- Los servomotores tienen un rango de movimiento limitado, que generalmente es de 0 a 180 grados. Algunos servomotores pueden tener un rango de movimiento mayor.



SERVOMOTOR

- Para este servomotor hace falta incluir una librería PERO no es necesario descargar un zip:



SERVOMOTOR

- Y para usar el servomotor hay que asociar el objeto servo al pin del servo:

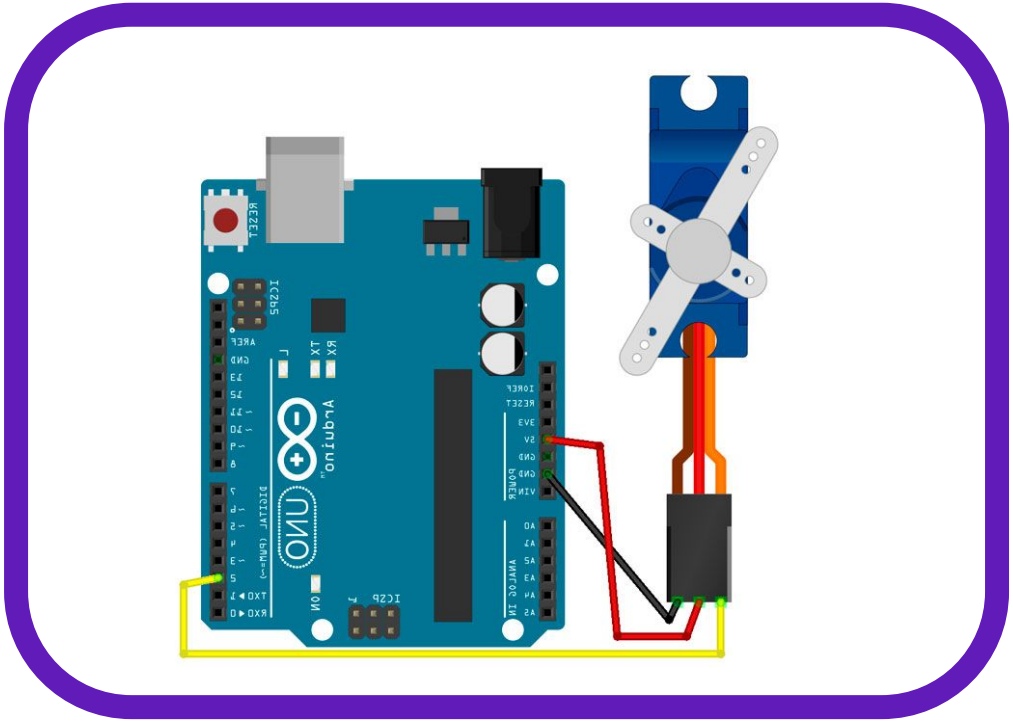
servo §

```
#include <Servo.h>

Servo servoMotor; // Crea un objeto Servo

int servoPin = 8; // Pin al que está conectado el servo

void setup() {
  servoMotor.attach(servoPin); // Asocia el objeto Servo al pin del servo
}
```

ServoMotor

1. Conectamos el motor al arduino

   - I/O - **Pin 8**
   - GND - **Tierra**
   - VCC - **5V**



ZUMBADOR

2. Utilizamos el servo - Vamos a hacer que gire 90 º cada segundo:

- **servoMotor.write(0);**
- **delay(1000);**

servo §

```
#include <Servo.h>

Servo servoMotor; // Crea un objeto Servo

int servoPin = 8; // Pin al que está conectado el servo

void setup() {
  servoMotor.attach(servoPin); // Asocia el objeto Servo al pin del servo
}
```

escribimos el punto 0 en el objeto servo
esperamos 1 seg

. . . vosotras . . .

(Hasta 180)

```
}
```

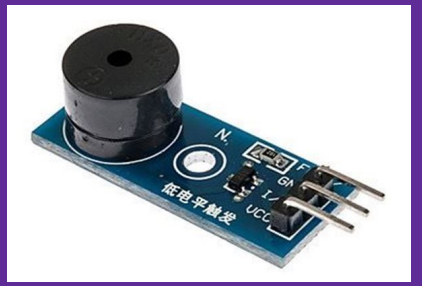
¿Sabríais hacerlo con un for?

Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

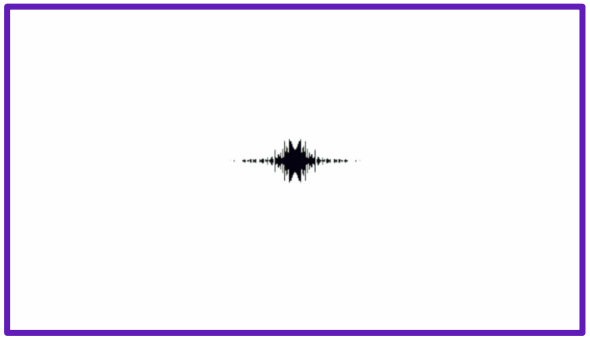
Conectando el hardware de Huerto inteligente

Zumbador



ZUMBADOR

- Un zumbador produce **ruidos o sonidos.**
  - Al enviarle la señal eléctrica al pin del zumbador, este hace que la pieza interna del zumbador vibre, lo que genera ondas sonoras, lo que escuchamos como sonidos.
  - La frecuencia del sonido depende de la señal eléctrica aplicada.
  - Podemos controlar el tono, tiempo y volumen en el que suena el zumbador



ZUMBADOR

Un zumbador es un pequeño componente que emite sonidos o pitidos cuando le llega electricidad. En nuestro proyecto lo usaremos para:

❌ problemas

ZUMBADOR

- Utilizamos la función **tone()** para generar tonos con el zumbador.
  - Esta función toma como parámetros el pin de salida y la frecuencia del tono que deseamos generar.

ZUMBADOR

2. Utilizamos el sensor:
   Lo conectamos al pin 3

- **tone(pin, tonoHz);**
- **noTone(pin)**
- **delay(1000);**

PruebaZumbador

```
const int zumbadorPin = 9;

void setup() {
  // Configurar el pin del zumbador como salida
  pinMode(zumbadorPin, OUTPUT);
}

void loop() {
```

```
ponemos el tono de 1000hz en el pin
esperamos 1 segundo

quitamos el tono del pin
esperamos dos segundos
```

```
}
```

Oficina de Software Libre Universidad de Granada

UNIVERSIDAD DE GRANADA

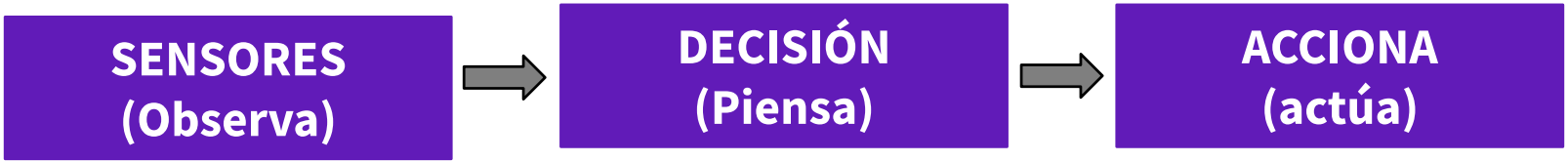
Conectando el hardware de Huerto inteligente

pensemos la lógica del eco lab

Pensando la lógica del EcoLab

La lógica define cómo se comporta nuestro sistema.

Nuestro proyecto:



Pensando la lógica del EcoLab

¿Qué datos puede leer?

Lista:

- Humedad del suelo
- Temperatura
- Luz
- Lluvia
- Nivel de agua

Pensando la lógica del EcoLab

Decisiones a tomar (por ejemplo):

si la tierra está seca → activar aviso

si está lloviendo → no actuar

si no hay agua → mostrar error

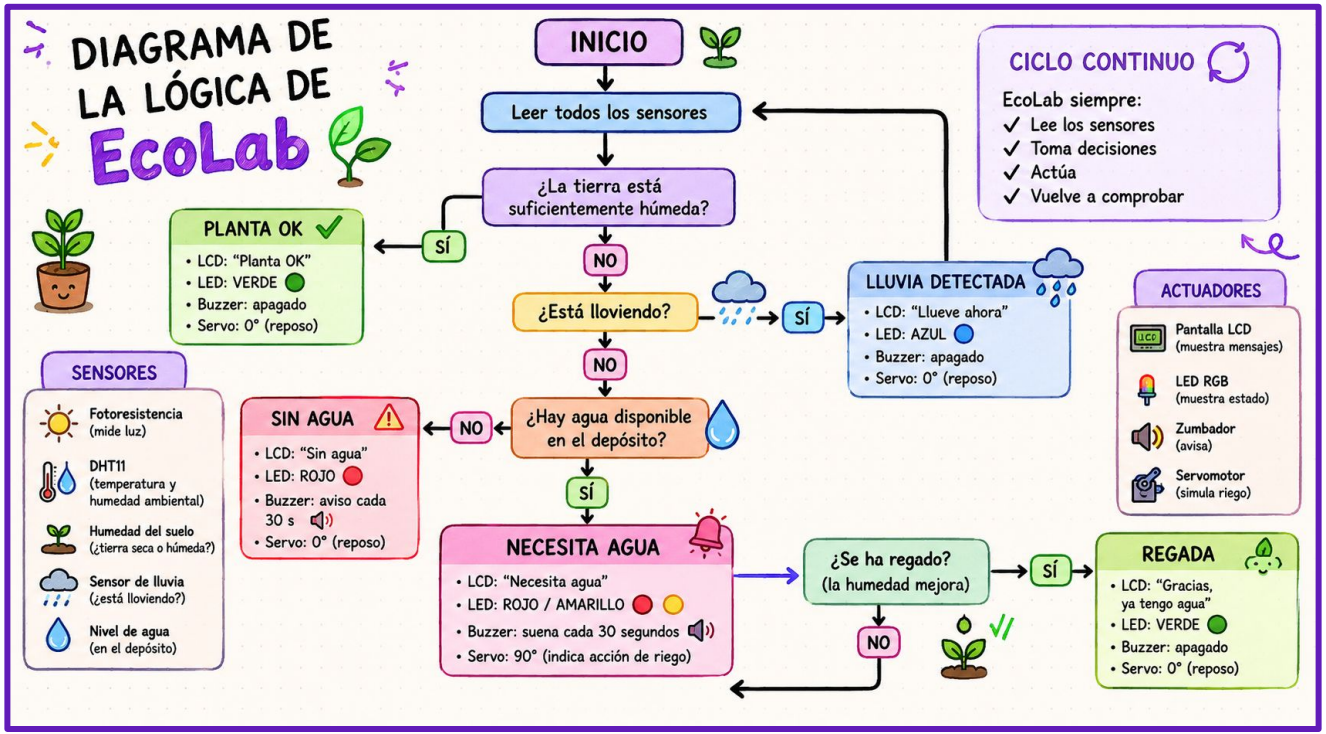
Pensando la lógica del EcoLab

Algunos posibles estados del sistema:

Situación	Qué ocurre
Todo bien	LED Verde
Tierra seca	LED Rojo + Sonido
Simulación de Riego	Servo
Error	Mensaje alerta

Pensando la lógica del EcoLab

Ejemplo:



Oficina de Software Libre Universidad de Granada

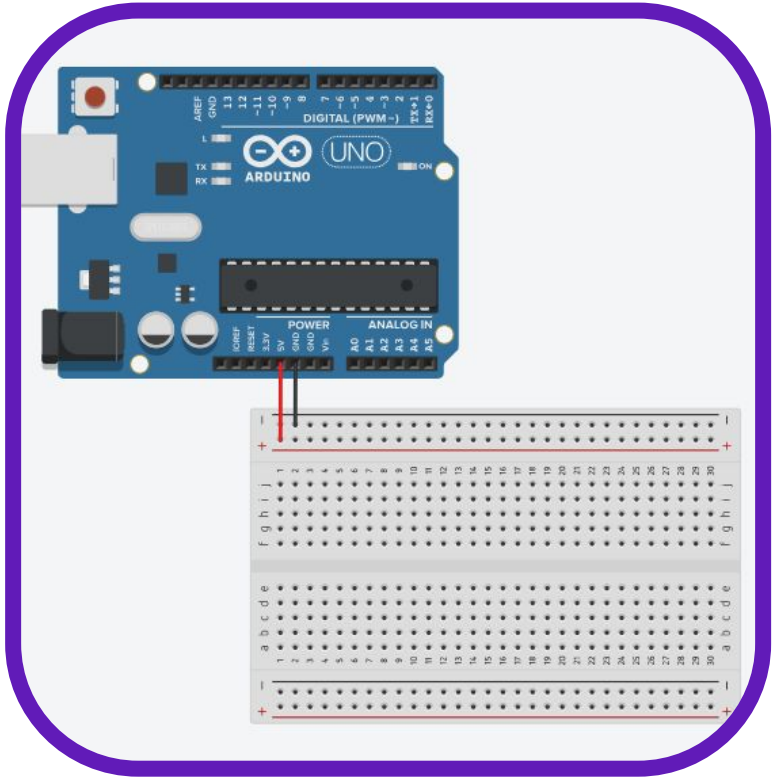
UNIVERSIDAD DE GRANADA

Conectando el hardware de Huerto inteligente

Montemos el eco lab

Montando el EcoLab

1. Protoboard:
   - Coloca la protoboard delante de ti
   - Conecta **5V** del Arduino **a la línea roja (+)** y **GND** a la **línea azul (-)** de la protoboard.
   - Así tendrás alimentación lista para todos los componentes



Montando el EcoLab

Conectamos **toooodos los sensores** en los pines que hemos ido probando durante estos días.

- Fotoresistencia= A0;
- Humedad del suelo= A1;
- Nivel de agua = A2;
- Pantalla:
  - SDA = A4;
  - SCL = A5;
- Humedad y temperatura = 5;
- Sensor lluvia= 6;
- Led RGB:
  - red = 9;
  - green = 10;
  - blue = 11;
- Zumbador = 3;
- Servo = 8;