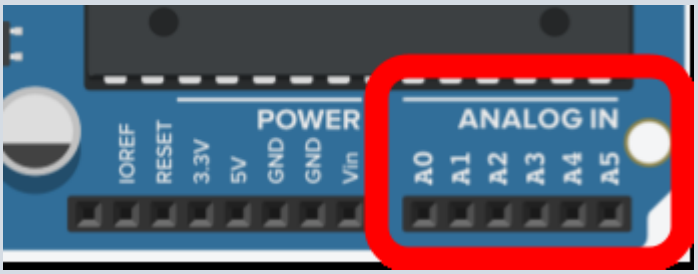
[GitHub Episodio 3 / Estación 3]

Tema 1. Entradas analógicas

Hasta ahora hemos aprendido a usar las funciones digitales: “digitalRead()” y “digitalWrite()” las cuales son capaces de distinguir o establecer (respectivamente) dos valores de voltaje: HIGH y LOW. ¡Pero estas dos funciones ven el mundo en blanco y negro, ahora aprenderemos a ver colores!

Una entrada analógica es un tipo de señal que es interpretada mediante voltaje que ingresa en uno de los 6 pines ANALOG IN de la placa Arduino, a partir del uso de la función analogRead(pin). Estos pines siempre son entradas.



Leer HIGH o LOW es cosa de digitalRead(). Ahora seremos capaces de leer valores intermedios. Seremos capaces de usar sensores que puedan medir: temperatura, humedad, cantidad de luz del ambiente o posición mecánica de un potenciómetro. Todas las variables mencionadas anteriormente tienen la particularidad de que se miden con números y no somos capaces de expresarlo con dos valores como el 1 y 0 de las funciones digitales. Por ejemplo: ¿Qué temperatura sería HIGH… y LOW?

Abrir debate

38. Actividad prioritaria: Pensando en el celular: ¿Qué sensores digitales tiene? ¿Qué sensores analógicos contiene? ¿El celular es una computadora como lo es Arduino? indicar diferencias. Realizar puesta en común. Llegar a acuerdos

Para leer un pin analógico NO es necesario inicializarlo como INPUT en el setup() ya que siempre son entradas (pero se puede hacer), se utiliza principalmente en el loop() de la siguiente manera:

```
analogRead(nro_pin);
```

Donde:

nro_pin: es el número entero del pin analógico, pueden ser: A0, A1, A2, A3, A4 o A5.

Esta función usada para leer sensores analógicos sólo tiene un parámetro que es el pin a leer, lo que leamos debemos guardarlo en una variable entera como muestra el siguiente ejemplo:

```
int lecturaHumedad = analogRead(nro_pin);
```

Se recomienda siempre guardar el valor de lectura del pin ANALOG IN en una variable entera, SIEMPRE hacer esto con las entradas.

En el ejemplo anterior el valor leído es guardado en lecturaHumedad, el cual está comprendido entre 0 y 1023, representa indirectamente la tensión que existe en ese pin. Para “0” son 0 voltios y para “1023” son +5 voltios.

Si lecturaHumedad vale 0 significa que el sensor de humedad está entregando al Arduino 0 voltios. Si lecturaHumedad vale 1023, significa que el sensor está dando +5v al Arduino. Por lo que si el valor leído es 512 el voltaje del sensor sería +2.5v. Entonces 256 es 1.25 voltios. Y así sucesivamente.

El microcontrolador interpreta con 1024 números voltajes entre 0 y 5v. Por lo que si dividimos 5 por 1024 conocemos cuánto vale una unidad guardada en lecturaHumedad.

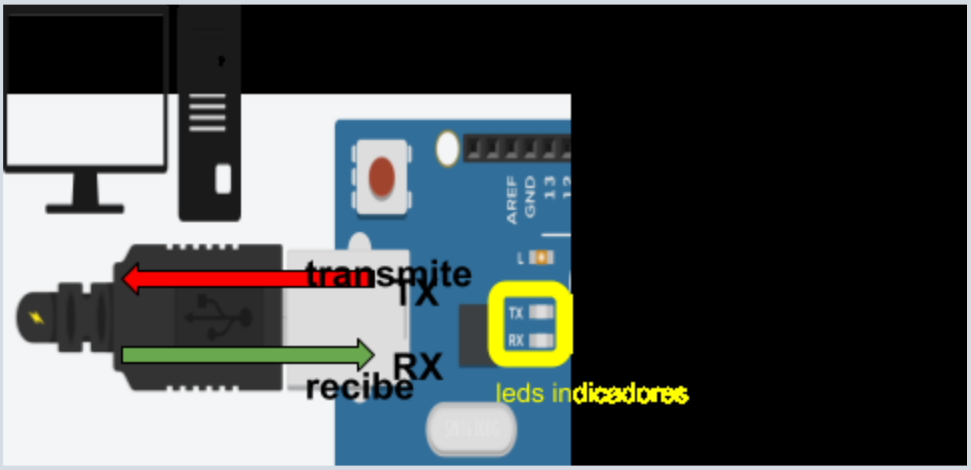
5v / 1024 = 0.004882v = 4.88 milivoltios

por lo que cada unidad de medida son 0.00488v. Si multiplicamos esa cantidad por 1024 el resultado es 5v. Entonces:

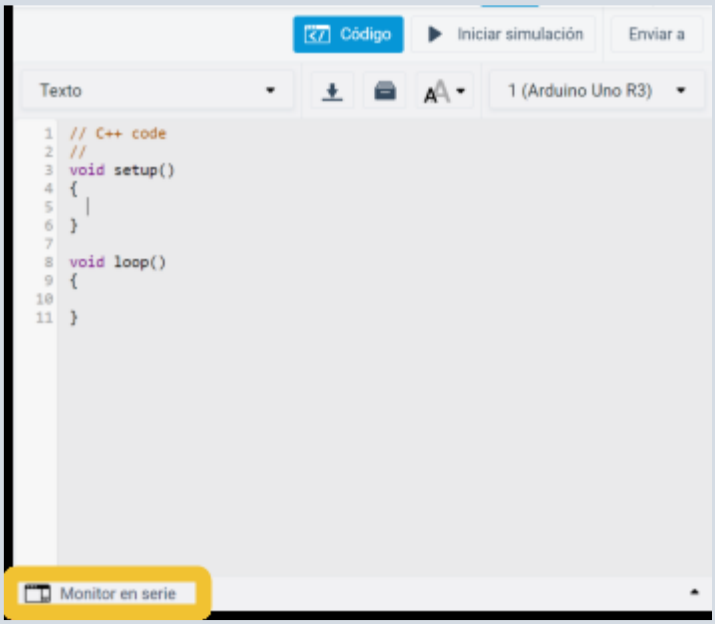
voltajeEntrada = lecturaHumedad * 0.00488

Para poder visualizar los datos de lectura es necesario que Arduino nos envíe la información mediante el cable USB hacia nuestro ordenador. Arduino es capaz de comunicarse mediante comunicación serial con otros dispositivos (otro Arduino u ordenador).

La función Serial.print() envía información desde Arduino al ordenador al que está conectado.



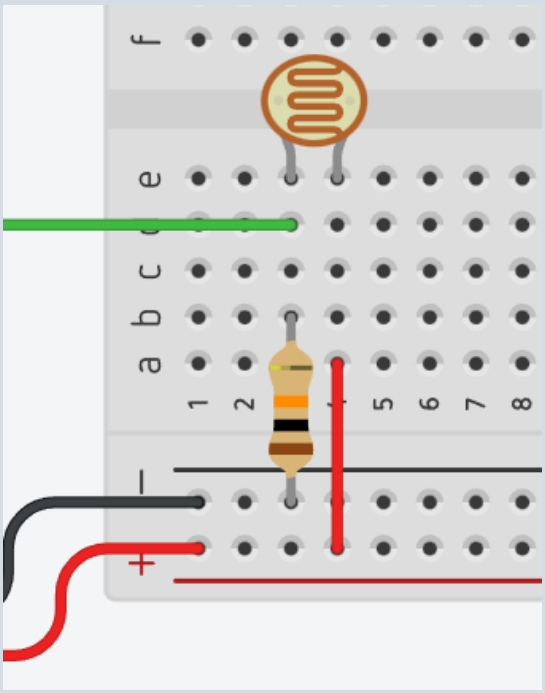
El mensaje se envía con comunicación Serial y se recibe en el ordenador con el 'Monitor Serie'



Si dentro del parámetro de Serial.print() escribimos una palabra entre comillas, mostrará este texto en el monitor serie tal y como está escrito, en este caso "Lectura de Humedad: ". Si le da una variable como parámetro, el monitor serie mostrará el valor de esa variable, como se ve a continuación:

Serial.print(sensorHumedad), el monitor serie podrá mostrar cualquier valor comprendido entre 0 y 1023.

En el siguiente link encontrarán un ejemplo con la conexión de un sensor de luz. Podrán ver el valor leído por Arduino y el cálculo de voltaje. Probar modificar el valor de la resistencia de 10kΩ para ver qué sucede.

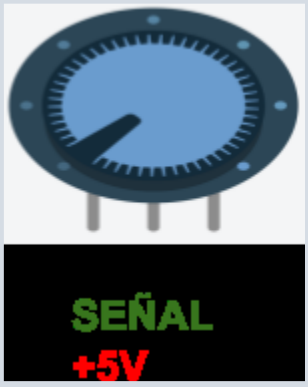


Ejemplo con LDR

Actividades:

39. Actividad complementaria: Llegó el momento del potenciómetro! ¿Qué es? ¿Dónde se usa…? Les proponemos armar una lista de dispositivos de uso diario que usan potenciómetros. Hay 1 punto por dispositivo y 2 puntos por originalidad (dispositivo solamente mencionado por una persona). Tienen 3 minutos! Una vez finalizados los 3 minutos compartir respuestas y contar el puntaje.

La forma de conectarlo es sencilla:

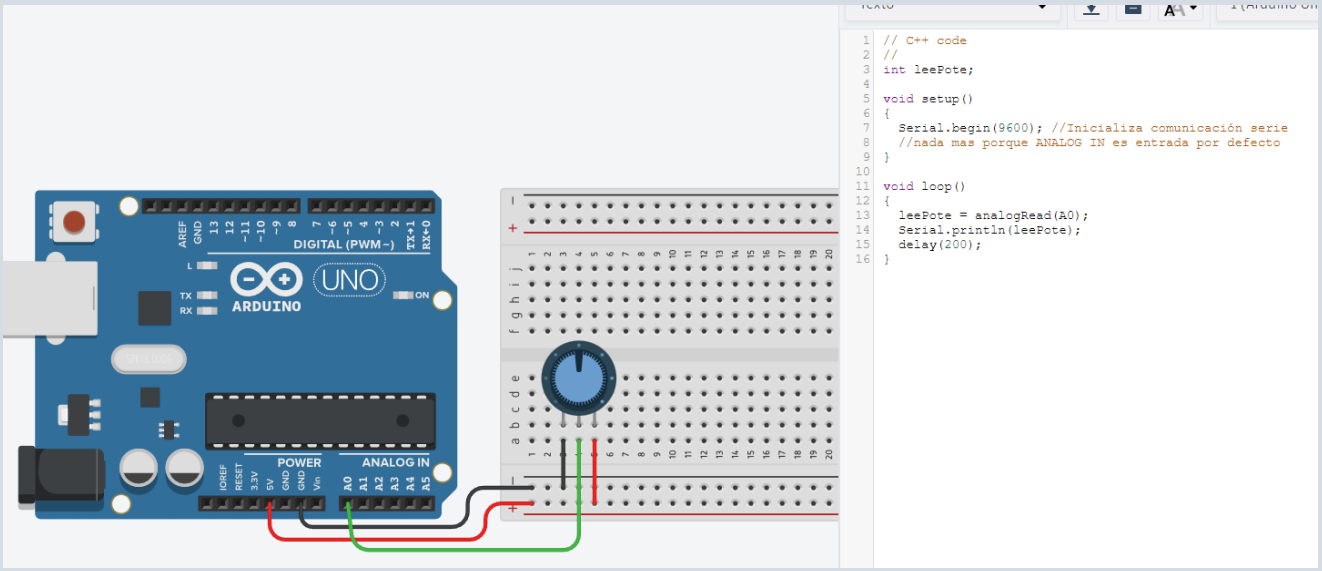


Los potenciómetros tienen 3 pines o patitas, hay que conectarlo de manera tal que el conector izquierdo vaya al GND, el derecho al +5v y el conector del medio es la señal analógica que conectamos a cualquier ANALOG IN de Arduino.

Básicamente es una resistencia que varía en función de la posición mecánica del dial, al moverlo aumentamos o disminuimos el voltaje que sale por SEÑAL. Comportamiento: A mayor resistencia, menor voltaje de salida. A menor resistencia, mayor voltaje de salida.

Actividad:

40. Realizar la siguiente conexión y conectar un voltímetro para medir la tensión entre la señal y el GND. Comparar voltaje vs valor leído y sacar conclusiones grupales. Imagen con link:



Respuesta:

En la función setup(), se inicia la comunicación serie a una velocidad de baudios de 9600 utilizando Serial.begin(9600). Esta configuración permite enviar datos desde el Arduino a través del puerto serial hacia una computadora u otro dispositivo.

En la función loop(), se lee el valor analógico del pin A0 utilizando analogRead(A0) y se almacena en la variable leePote.

A continuación, se imprime el valor de leePote utilizando Serial.println(leePote). Esto enviará el valor leído a través del puerto serial hacia el dispositivo conectado.

Finalmente, se agrega un retraso de 200 milisegundos utilizando delay(200) antes de repetir el ciclo.

En cuanto al valor que se lee del pin A0, eso dependerá de la configuración física y el valor presente en el potenciómetro o sensor conectado a ese pin. El rango de valores posibles será de 0 a 1023, ya que analogRead() devuelve un valor de 10 bits (de 0 a 1023) que representa el voltaje analógico en el pin A0 en relación con la referencia de voltaje del Arduino.

En Arduino, la función analogRead(pin) se utiliza para leer valores analógicos de los pines designados como entradas analógicas, como en tu caso el pin A0. Sin embargo, es importante tener en cuenta que Arduino utiliza una conversión analógico-digital (ADC) de 10 bits

La resolución de 10 bits significa que el ADC puede representar valores en un rango de 0 a 1023. Esto se debe a que el ADC divide la referencia de voltaje (que por defecto es de 5V en la mayoría de las placas Arduino) en 1024 pasos discretos. Cada paso representa un valor de voltaje pequeño que el ADC puede medir.

Por lo tanto, cuando utilizas analogRead(A0), el ADC realiza una medición de voltaje en el pin A0 y devuelve un valor entero entre 0 y 1023. Este valor representa la relación proporcional entre el voltaje medido y la referencia de voltaje del Arduino.

Por ejemplo, si el potenciómetro o el sensor conectado a A0 está proporcionando la mitad del voltaje de referencia (2.5V en una placa Arduino de 5V), la función analogRead(A0) devolverá aproximadamente 512, que es la mitad de 1023.

Es importante tener en cuenta que, para interpretar correctamente los valores leídos, es posible que necesites realizar una conversión adicional para adaptarlos a las unidades o el rango de valores específicos de tu aplicación.

Cuando utilizas analogRead() para leer un valor analógico en Arduino, el resultado que obtienes es un valor entero en el rango de 0 a 1023, que representa la proporción entre el voltaje medido y la referencia de voltaje del Arduino.

Sin embargo, en muchos casos, es necesario realizar una conversión adicional para interpretar correctamente ese valor en términos de unidades específicas o un rango de valores deseado.

Por ejemplo, si estás midiendo la temperatura con un sensor y deseas obtener la temperatura en grados Celsius, necesitarás aplicar una fórmula de conversión específica. Esto generalmente implica conocer las características de tu sensor y cómo se relaciona el valor de analogRead() con la temperatura real.

Aquí hay un ejemplo hipotético para ilustrar la conversión de valores leídos:

Supongamos que tienes un sensor de temperatura que proporciona una salida analógica proporcional a la temperatura en grados Celsius. El sensor tiene una referencia de voltaje de 5V y su salida analógica está linealmente relacionada con la temperatura en un rango de 0 a 100 grados Celsius.

Primero, debes establecer una correspondencia entre los valores de analogRead() y la temperatura real en grados Celsius. Para este ejemplo, digamos que tienes una función convertToCelsius() que realiza la conversión adecuada:

```
float convertToCelsius(int rawValue) {
  // Realiza la conversión específica de tu sensor
  float temperature = (float)rawValue * 100 / 1023; // Ejemplo hipotético
  return temperature;
}
```

Luego, dentro de tu programa, puedes llamar a esta función para obtener la temperatura en grados Celsius utilizando el valor leído:

```
int sensorValue = analogRead(A0);
float temperatureCelsius = convertToCelsius(sensorValue);
```

Con esta conversión adicional, ahora puedes interpretar el valor leído de analogRead() en términos de grados Celsius.

Es importante tener en cuenta que la fórmula de conversión y los parámetros específicos dependerán del sensor que estés utilizando y las características de tu aplicación. Es posible que necesites consultar la hoja de datos del sensor o realizar calibraciones adicionales para obtener una conversión precisa.

En resumen, realizar una conversión adicional implica comprender la relación entre el valor leído con analogRead() y la magnitud o rango de valores que deseas medir en tu aplicación, y aplicar una fórmula de conversión específica para adaptarlos correctamente.

Si deseas realizar una conversión para representar los valores leídos de 0 a 5 voltios en lugar de 0 a 1023, puedes refactorizar la función convertToCelsius de la siguiente manera:

```
float convertToVolts(int valorLeido) {
  // Realiza la conversión específica para representar valores de 0 a 5 voltios
  float voltage = (float) valorLeido * 5.0 / 1023.0;
  return voltaje;
}
```

En este caso, la función convertToVolts toma un valor leído valorLeido y lo convierte en un valor de voltaje en el rango de 0 a 5 voltios. Se utiliza una simple proporción para realizar esta conversión, multiplicando el valor leído por 5 y dividiéndolo por 1023.

Puedes llamar a esta función dentro de tu programa para obtener el valor de voltaje correspondiente:

```
int valorSensor = analogRead(A0);
float voltaje = convertToVolts(valorSensor);
```

De esta manera, el valor “voltaje” contendrá el voltaje correspondiente a la lectura analógica en el rango de 0 a 5 voltios.

Recuerda que es importante conocer las especificaciones y características de tu sensor para realizar una conversión precisa y ajustar la fórmula según sea necesario.

Antes de seguir ejercitando vamos a conocer cómo son las salidas analógicas.

Tema 2. Salidas analógicas

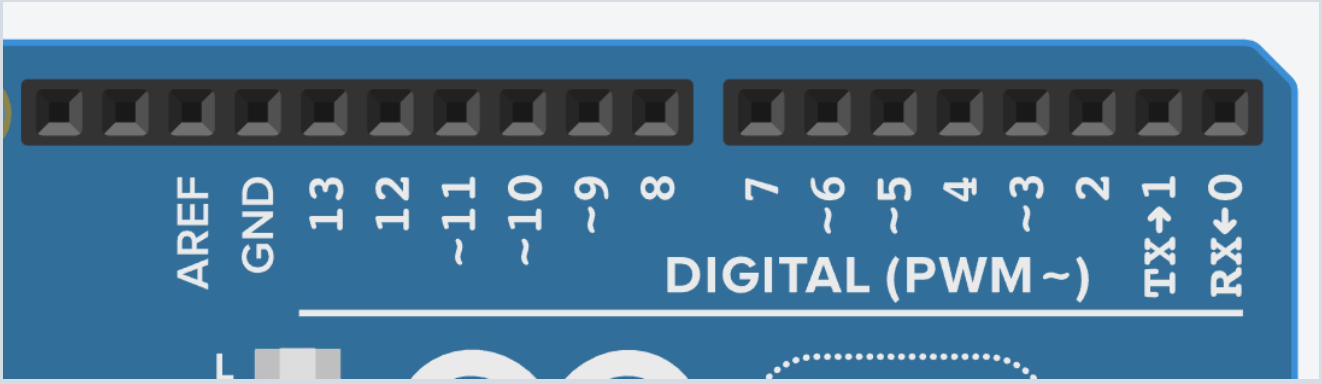
Los pines digitales de Arduino que contienen el símbolo ~ “seño” son capaces de ser utilizados como salidas analógicas. ¿Pero cómo un pin digital, puede ser una salida analógica?

Recordemos que lo digital tiene dos valores posibles, HIGH (1) o LOW (0), existe una técnica llamada PWM que a partir de estos valores digitales podemos simular una salida analógica. El proceso es encender y apagar tan rápido como se desee un pin de arduino para que la salida en promedio sea un voltaje entre 0v y 5v.

VIDEO EXPLICATIVO (hasta min 3:52)

SOLO LOS PINES CON “~” PUEDEN SER USADOS COMO SALIDAS ANALÓGICAS.

Los pines 3, 5, 6, 7, 10 y 11 son los disponibles para hacer uso de la función analogWrite().



Para hacer uso de una salida analógica primero se debe declarar el pin como salida en el setup():

pinMode(nro_pin,OUTPUT);

Luego se debe hacer uso de la función analogWrite() dentro del loop:

analogWrite(nro_pin,valor);

Donde:

nro_pin: es el número entero del pin declarado como salida en el setup que queremos encender o apagar.

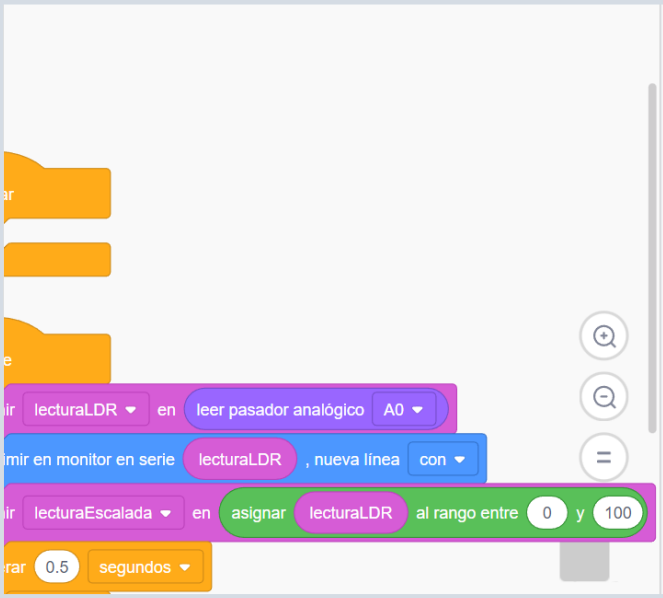
valor: es un número entre 0 y 255. Mientras más pequeño es valor, más pequeño es el voltaje de salida en el pin.

Esta función es usada para generar una señal entre 0v y 5v mediante valores que podemos elegir entre 0 y 255, dónde 0 es 0v y 255 es 5v. Por lo que 127 será la mitad del voltaje 2.5v.

Actividad:

Analizar el siguiente ejemplo sobre entradas y salidas analógicas en la herramienta de simulación Tinkercad.

Uso de función matemática **map():** re-escala hacia un mínimo y máximos buscados, a partir de un mínimo y un máximo del sistema a escalar. Es una “regla de tres simple”.




```
// C++ code
//
int lecturaLDR = 0;

int lecturaEscalada = 0;

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);
}

void loop()
{
  lecturaLDR = analogRead(A0);
  Serial.println(lecturaLDR);
  lecturaEscalada = map(lecturaLDR, 0, 1023, 0, 100);
  delay(500); // Wait for 500 millisecond(s)
}
```

map() es el bloque “Asignar ***** al rango **** y ****”

Sintaxis:

map(valor, desdeMin, desdeMax, hastaMin, hastaMax)

Parámetros

valor: el número a mapear. Suele ser el valor leído del sensor.

desdeMin: el límite inferior del rango actual del valor.

desdeMax: el límite superior del rango actual del valor.

hastaMin: el límite inferior del rango objetivo del valor.

hastaMax: el límite superior del rango objetivo del valor.

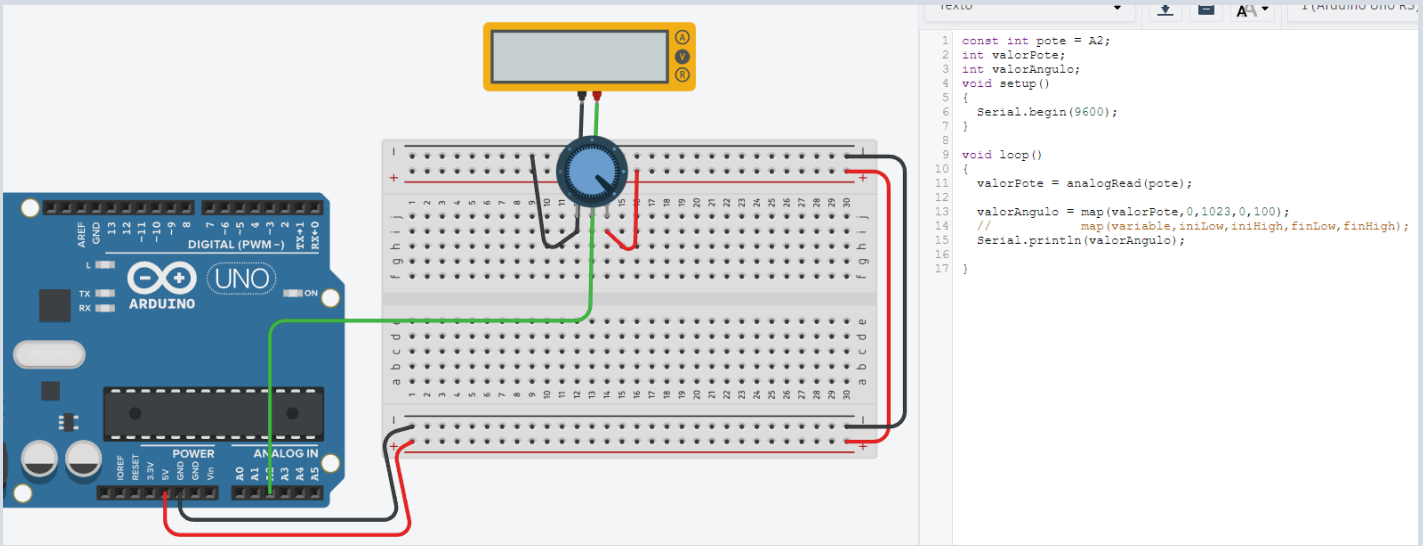
Este tipo de escalamiento es muy utilizado en sistemas de automatización. Tengo que conocer los mínimos y máximos de medición (prueba y error) y conocer hacia dónde queremos a escalar, para eso es necesario hacer pruebas en espacio donde va a estar nuestro dispositivo o robot. Suele usarse para la medición de luz, temperatura, sonido o cualquier variable física que necesitemos expresar en otra unidad, por ejemplo en el caso de la medición de temperatura, el sensor envía una señal a Arduino para que sea interpretada como un número de 0 a 1023. Imaginemos que sabemos que cuando leemos 250 la temperatura es 20°C y la temperatura máxima son 80°C siendo la señal 1023.

Usamos: map(lectura,250,1023,20,80); el resultado de map debe ser guardado en una variable entera, quedando:

lecturaTemp = analogRead(A0);

centigrados = map(lecturaTemp,250,1023,20,80);

Ejemplo con uso de map().

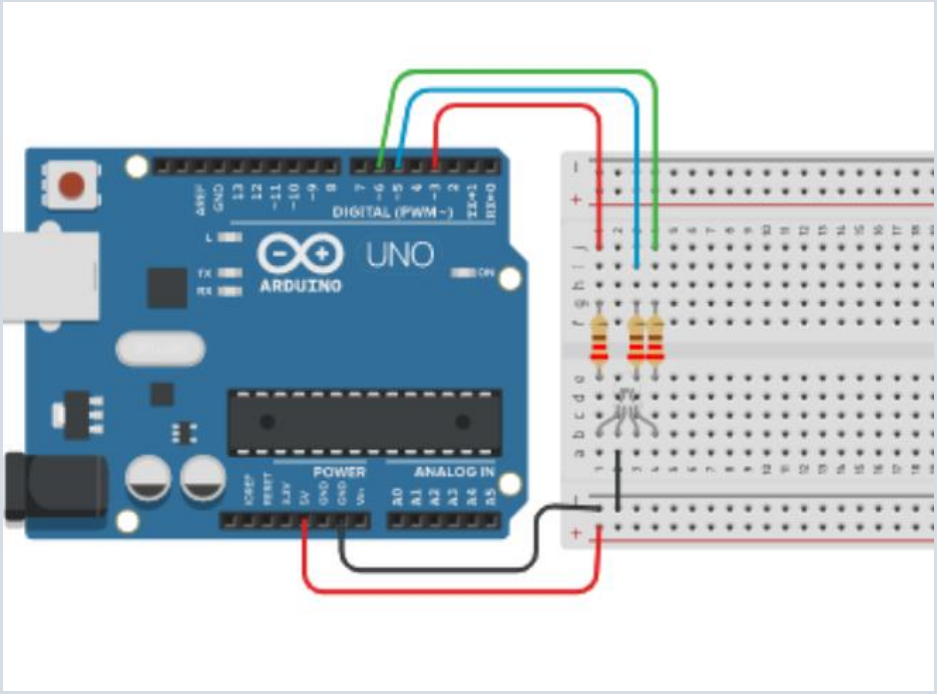


Tema 2.1-ANEXOS: otros temas para consultar en caso de ser necesario.

LED RGB

Video Explicativo para PWM y LED RGB. (de 0 hasta minuto 8 y desde minuto 13)

Un LED RGB (Red Green Blue) es un led multicolor compuesto de tres leds en uno: de colores Rojo, Verde y Azul. Seguramente has visto leds RGB porque son muy comunes hoy en día, más que nada en electrodomésticos. Cada led puede controlarse independientemente y con la mezcla de luces generan nuevos colores, los leds RGB son controlados por señales analógicas para poder llegar a todos los espectros de colores. Ejemplo:



Actividades:

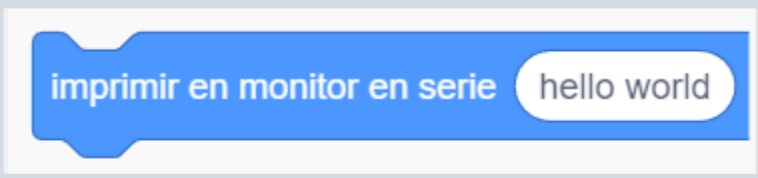
41. Actividad prioritaria: Realizar un programa que haga parpadear lentamente un led (de una intensidad mínima a una máxima) cada 750 milisegundos. Se recomienda usar bucle de repetición for.

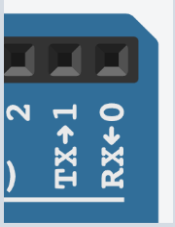
42. Actividad complementaria: Realizar un programa que mediante la lectura de 3 potenciómetros controlar los colores de un led RGB

Comunicación Serial

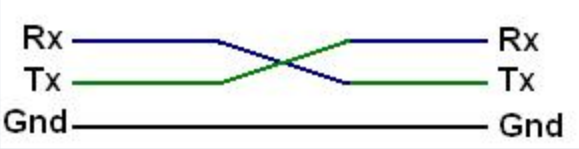
La comunicación serial es la forma que tiene Arduino de transmitir información con otros dispositivos, ya sean otros arduinos, un bluetooth u otro ordenador. Es utilizada para enviar mensajes a través del puerto USB al PC, para poder visualizar los datos que leen los sensores y evaluar si el programa está ejecutándose como buscamos.

Es el bloque de salida:

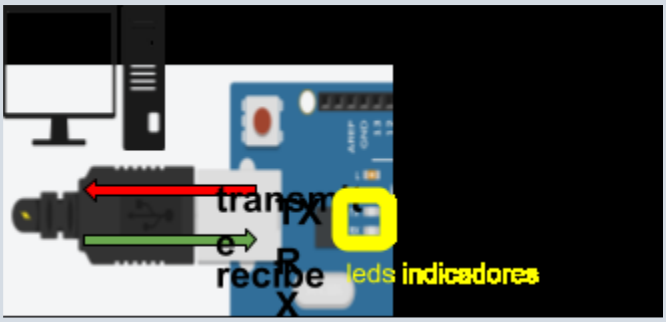




Los pines que se usan para esto, están marcados con las siglas (0) RX, por el que se enviarán los datos y (1) TX, por el que se recibirán los datos. Cuando se usa la comunicación serie, los pines digitales 0 y 1 no pueden ser usados al mismo tiempo. Estos pines están conectados directamente al puerto USB.



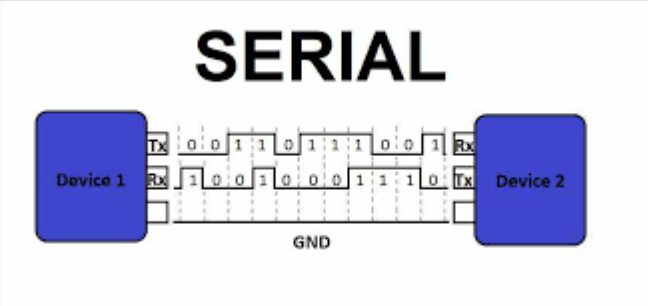
Para poder realizar una comunicación serie de una manera correcta, hay que conectar el pin de recepción (RX) al pin de transmisión del dispositivo externo, y el pin de transmisión (TX) al pin de recepción de dicho dispositivo y las tierras entre ellas.



Para que haya una comunicación entre computadoras se han creado una serie de reglas. Estas reglas garantizan que la comunicación será exitosa y sin errores y se le llaman protocolos de comunicación.

El protocolo USB (Bus Serie Universal) y Ethernet son las más conocidas hoy en día. La comunicación serial de Arduino es asincrónica, ya que es más sencilla y dispone de menos cables para transferencia de datos. Asincrónico significa que los relojes de los microcontroladores a comunicar, no están sincronizados (cuentan el tiempo en distinta sincronía) y es el más común o más utilizado.

Reglas de la comunicación serie



El protocolo serie asincrónico tiene una serie de reglas integradas: mecanismos que ayudan a garantizar transferencias de datos sólidas y sin errores. Estos mecanismos, que obtenemos para evitar la señal del reloj externo, son:

> ★ Velocidad en baudios: la velocidad de transmisión indica que tan rápido se enviarán o recibirán los datos. Se expresa en bits por segundos (bps). Las velocidades en baudios pueden ser casi cualquier valor dentro de lo que permite el hardware. El único requisito es que ambos dispositivos funcionen a la misma velocidad. Una de las velocidades en baudios más comunes es de 9600 bps. Otras velocidades en baudios «estándar» son 1200, 2400, 4800, 19200, 38400, 57600 y 115200. En Tinkercad, al ser un simulador, la velocidad de transferencia es la que se establece en el setup por defecto.

Para poder usar la comunicación serial es necesario inicializarla y configurar la velocidad de la comunicación serial. Se hace mediante una función llamada Serial.begin(velocidad) la cual puede ser traducida como “iniciar Serial”, dentro del setup() y la velocidad es un número entero expresada en bits por segundos (bps) 9600 es la

más usada y usaremos esta velocidad a lo largo de todo el episodio. Como se realizó en el ejemplo de la sección de analogRead().

A continuación veremos este ejemplo para conocer cómo usar las funciones principales de Serial.

Existen más funciones en la biblioteca Serial de Arduino, se pueden visualizar en la referencia de arduino Serial.

Haremos uso de las siguientes funciones (métodos) para visualización y recepción de datos:

- Serial.begin() : Establece la velocidad de datos en bits por segundo (baudios) para la transmisión de datos en serie. Inicializa la comunicación serie.

- Serial.available(): responde true (1) si no está transmitiendo datos y la comunicación está disponible. Si no está disponible, es un false (0).

- Serial. print(): imprime datos para ser leídos por humanos, es como el print de Python.

- Serial.println(): es igual que el print() solo que al final lleva un \n o salto de línea, de ahí viene el "ln" de "\n".

- Serial.read(): Espera un dato de entrada para ser leído y guardado en la memoria de Arduino.

    a. Ver EJEMPLO Serial.read().

    b. Ver otro ejemplo con Serial.read().

    c. Ver ejemplo de Serial.read() con uso de switch case.

Actividades E3/E4:

43. Actividad complementaria: Un productor de maíz guarda sus granos en un silo que tiene un medidor de altura para conocer la capacidad actual del silo. El problema es que el sensor se comunica de forma serial y envía letras en vez de números.

El sensor envía una X si el silo está vacío, envía una Y cuando el silo está al 25%, envía una Z cuando el silo está al 50%, envía una A cuando el silo está al 75% y una B cuando el silo está al 100%. Ayudar al productor de granos a realizar un indicador lumínico para conocer cuál es la capacidad actual del silo en función de los datos enviados por el monitor serie (hacer de cuenta que ustedes son el sensor que mide y envía las letras XYZAB).

Se puede usar este circuito como base.

La función millis()

Existe una función del microcontrolador de Arduino que permite hacer uso del temporizador del micro. Los temporizadores tienen la capacidad de contar el tiempo y la función millis() nos devuelve un valor de tiempo en el instante que la llamamos.

EXPLICACIÓN DE LA FUNCIÓN MILLIS PARA MULTITAREAS CON EJEMPLO || TAREAS SIMULTANEAS CON ARDUINO

Uso de la función millis() para hacer un blink (parpadeo) de un LED.

Tema 4. Sensores y actuadores

Los sensores son transductores encargados de transformar distintos tipos de energía en energía eléctrica.

Los actuadores son transductores capaces de transformar energía eléctrica en otros tipos de energía.

Estos sensores y actuadores se utilizan en conjunto para resolver problemas, los encontramos en todos los dispositivos electrónicos. Para ampliar dejamos aplicaciones biotecnológicas en distintas áreas:

Aplicaciones robótica en biotecnología:

- Mano biónica
- Prótesis robóticas
- Robots para operaciones quirúrgicas
- Nanotecnología

Ejemplos donde se usan sensores/actuadores en medicina:

- Rayos X
- Eco Doppler (Ultrasonido) / Ecografías
- Neurociencia
- BPM corazón y presión sanguínea
- Diagnóstico por imágenes.
- Medidor de pulso (BPM) automático.

Ejemplos donde se usan sensores/actuadores en agro:

- Micro Oxigenadores de vino
- Medidor de PH para bodegas.
- Densimetros para vino
- Medidor de nivel de aforo de agua, con SMS te manda el nivel.
- Sensores de temperatura para las piletas de vinificación para fermentación alcohólica
- Detección de ciertos organismos benéficos y dañinos usando análisis de sonido (agro)
- Sensado de una “Bio Variable” con Arduino.

Ejemplos donde se usan sensores/actuadores en biología:

- Incubadora de bacterias/huevos (mantiene temp en una consigna de Temp dada)

- Control motor o servo para enfoque en microscopios.
- Biología molecular

Actividades E2/E3:

44. Actividades prioritarias: Analizar los ejemplos anteriores y entre todos responder:

    - ¿Conocías todas las aplicaciones?
    - Agregar un ejemplo por categoría.
    - Agregar una categoría nueva

En las siguientes secciones se hará estudio de diversos sensores y actuadores con el fin de aplicarlos en soluciones biotecnológicas en el futuro.

Servomotores:

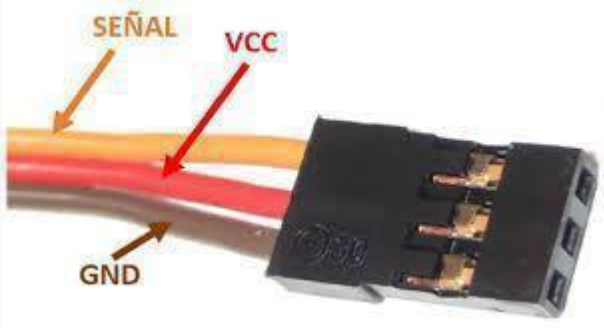
Un servomotor es un actuador o motor capaz de moverse un cierto ángulo, solo puede moverse de 0 a 180° y realizar movimientos lentos. No es capaz de girar como un motor cualquiera, se lo utiliza para controlar sistemas que necesiten precisión o movimientos angulares reducidos. Suele usarse en aeromodelismo, automodelismo .

Conectar un servo a Arduino es sencillo. El servo dispone de tres cables, dos de alimentación (GND y Vcc) y uno de señal.

El color de estos cables suele tener dos combinaciones:

- Marrón (GND), Rojo (Vcc) y Naranja (señal)
- Negro (GND), Rojo (Vcc) y Blanco (señal)

Por un lado, alimentamos el servo mediante el terminal GND (Marrón / Negro) y Vcc (Rojo).



Control de un servo con Arduino

Controlar un servomotor con arduino es muy sencillo ya que se hace a partir de la biblioteca "Servo.h" que viene incluida en el IDE Arduino por defecto. Lo primero que hay que hacer en el código es incluir la biblioteca del servo para Arduino:

```
#include <Servo.h>
```

El segundo paso es colocarle nombre al servo que se va a controlar (fuera del setup() y fuera del loop()), se hace de la siguiente manera:

```
Servo robotito; //Se crea un objeto servomotor, entonces de ahora en más todo o que vayamos a hacer con este servo lo llamaremos "robotito" (o el nombre que uds quieran)
```

El tercer paso es indicar en setup() dónde vamos a conectar el terminal de "señal" de mi servo con la función "Nombre_del_servo.**attach(nro_pin)",** por ejemplo en el pin 10.

```
void setup(){
robotito.attach(10); //conectamos el servo al pin 10.
}
```

Ya están todas las condiciones dadas para controlar el servomotor, simplemente en el loop() se debe colocar "nombre_del_servo.**write(angulo)**",

El rango angular de un servomotor es de 0 a 180 grados.

Solución 1: Ver ejemplo en tinkercad

Solución 2:

```
#include <Servo.h> //Se incluye librería para controlar servomotores
Servo miServo; // Se crea un objeto servomotor, entonces de ahora en más todo
//lo que vayamos a hacer con este servo lo llamaremos "miServo" (o el nombre que uds quieran)
int pos = 0; // posicion del servo
void setup() {
miServo.attach(10); // vincula el servo al pin digital 10
}
void loop() {
//varia la posicion de 0 a 180, con esperas de 15ms
for (pos = 0; pos <= 180; pos += 1)
{
miServo.write(pos);
delay(15);
}
```

```
//varia la posicion de 0 a 180, con esperas de 15ms

for (pos = 180; pos >= 0; pos -= 1)

{

miServo.write(pos);

delay(15);

}

}
```

Actividades (E2/E3):

45. Actividad prioritaria: Se desea controlar un servomotor con dos pulsadores, al activar el pulsador 1 hará que el servomotor se mueva en sentido horario y activar el pulsador 2 hará que se mueva en sentido antihorario. Si ningún pulsador está activado el servomotor estará quieto.

Desarrollar el código y conexión en Tinkercad para el problema planteado anteriormente.

46. Actividad complementaria: Se tiene una cámara capaz de reconocer los 5 productos de la línea de producción de una empresa de alimentos. La cámara cuenta con una inteligencia artificial y según el color y forma del empaque, los clasifica en 5 letras: A, B, C, D y E. Cada producto debe ser enviado por un camino diferente, para esto existe un motor que direcciona el empaque hacia su camino de salida.

Para resolver este problema, haremos de cuenta que somos la cámara y sabemos que letra enviar en cada caso. Si la letra recibida por el monitor serial es una “A”, se moverá un servomotor a 45°, si la letra recibida es una “B”, el servomotor se mueve a 75°, si la letra recibida es una “C”, el servomotor se mueve a 105°, si la letra recibida es una “D”, el servomotor se mueve a 135° y si la letra recibida es una “E”, el servomotor se mueve a 165°.

47. Actividad prioritaria: Realizar la conexión de tres leds (verde amarillo y rojo), conectar un botón y modificar el programa anterior para que si el botón se presiona durante un segundo, se encienda un led verde, si se presiona durante dos segundos se enciende el led amarillo y al pulsarlo durante tres segundos también enciende el rojo. Si se deja de presionar el botón se apagan todos los leds.

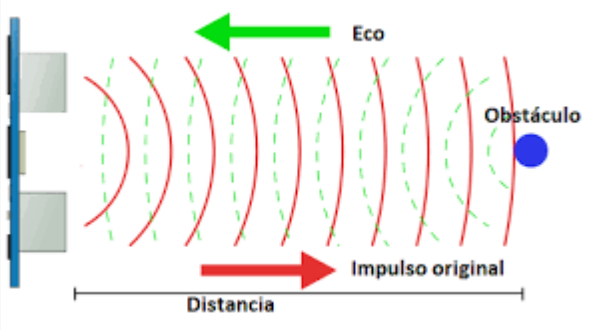
48. Actividad prioritaria: Realizar un programa que guarde un dato cada 1 hora. Ayudarse con este link.

Sensor De Distancia

El sensor HC-SR04 es un sensor de distancia de bajo costo, su uso es muy frecuente en la robótica, utiliza transductores de ultrasonido para detectar objetos en un rango de 2cm a 450cm.

Su funcionamiento consiste en emitir un sonido (pulso) ultrasónico por uno de sus transductores (TRIGGER), esperar que el sonido rebote de algún objeto presente frente a él y por último capta el eco con el segundo transductor.

La distancia es proporcional al tiempo que demora en llegar el eco.



Actividad ejemplo

49. Actividad prioritaria: Utilizando Tinkercad, conectar el sensor de distancia a una placa Arduino e imprimir en el monitor serial la distancia que va midiendo. El programa puede estar en bloques o en código.

Explicación y resolución:
https://www.youtube.com/watch?v=iAOedaQWpyw&ab_channel=MendozaFuturaContenidos

Para utilizar el sensor con código C++, primero configuramos los pines y la comunicación serial a 9600 baudios.

```
int TRIG = 3;

int ECHO = 4;

float distancia;

void setup()

{

pinMode(TRIG, OUTPUT);

pinMode(ECHO, INPUT);
```

```
Serial.begin(9600);
}
```

Ahora en el bucle void loop() empezamos enviando un pulso de 10us al Trigger del sensor

```
digitalWrite(TRIG, HIGH); //Enciende emisor
delayMicroseconds(10); //Enviamos un pulso de 10us
digitalWrite(TRIG, LOW); //Apaga emisor, el resultado es un pulso.
```

Seguidamente recibimos el pulso de respuesta del sensor por el pin Eco, para medir el pulso usamos la función pulseIn(pin, valor)

```
t = pulseIn(ECHO, HIGH); //espera un pulso de entrada en ALTO
```

La variable t, tiene el tiempo que dura en llegar el eco del ultrasonido, el siguiente paso es calcular la distancia entre el sensor ultrasónico y el objeto.

Partimos de la siguiente fórmula:

Velocidad = distancia recorrida / tiempo

$$Velocidad = \frac{distancia\ recorrida}{tiempo}$$

Donde Velocidad es la velocidad del sonido 340m/s, pero usaremos las unidades en cm/us pues trabajaremos en centímetros y microsegundos, tiempo es el tiempo que demora en llegar el ultrasonido al objeto y regresar al sensor, y la distancia recorrida es dos veces la distancia hacia el objeto, reemplazando en la fórmula tenemos:

$$\frac{340m}{s} x \frac{1s}{1000000us} x \frac{100cm}{1m} = \frac{2\boldsymbol{d}}{\boldsymbol{t}}$$

$$\boldsymbol{d}(cm) = \frac{\boldsymbol{t}(us)}{59}$$

en código:

```
d = t/59;
```

Finalmente enviamos de manera Serial el valor de la distancia y terminamos poniendo una pausa de 100ms, que es superior a los 60ms recomendado por los datos técnicos del sensor.

A continuación se muestra el código completo del programa: VER Simulación EN TINKERCAD

```
int TRIG = 3;
int ECHO = 4;
float distancia;
```

```
void setup()
{
pinMode(TRIG, OUTPUT);
pinMode(ECHO, INPUT);
Serial.begin(9600);
}
void loop()
{
//Solo hay que llamar la función medirDistancia()
// ECHO en pin 4
//TRIG en pin3
distancia = medirDistancia();
}
float medirDistancia(){
float t; //tiempo que demora en llegar el eco
float d; //distancia en centimetros
digitalWrite(TRIG, HIGH);
delayMicroseconds(10); //Enviamos un pulso de 10us
digitalWrite(TRIG, LOW);
t = pulseIn(ECHO, HIGH); //espera un pulso de entrada en ALTO
d = t/59;
/*
Serial.print("Distancia: ");
Serial.print(d); //Enviamos serialmente el valor de la distancia
Serial.print("cm");
Serial.println();
delay(100); //Hacemos una pausa de 100ms IMPORTANTE, recomendado por el fabricante
del sensor
*/
return d;
}
```

Actividades E3:

50. Actividad complementaria: Utilizando un sensor ultrasónico y un botón, crear un sistema que pueda guardar hasta 5 medidas de distancia. Cada vez que se pulsa el botón, se toma una medida, se la guarda e imprime en el monitor serial. Puede guardar solo 5 medidas y al tocar el botón por sexta vez, el contador se resetea ○ Opcional: Guardar medidas ilimitadas y reiniciar el contador al presionar el botón durante 2.5 segundos.

51. Actividad prioritaria: A partir del siguiente circuito, programar un Arduino para que con la distancia medida con un sensor ultrasónico, encienda 8 leds. Escalar los valores de distancia para que si la distancia es pequeña, haya pocos o ningún led prendido y si la distancia es grande, que se enciendan todos los leds.

Tema 4.1 Otros sensores

Se explican a través de videos y circuitos ejemplos los siguientes sensores:

Fotodiodos

Video explicativo

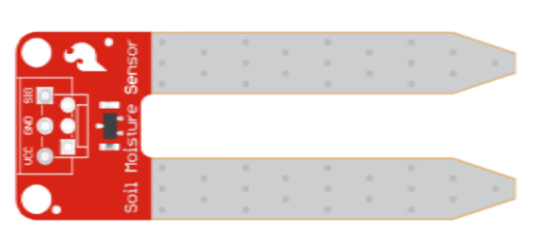
Circuito ejemplo

Sensor de Fuerza

Video explicativo

Circuito ejemplo

Sensor de Humedad del Suelo



Video explicativo

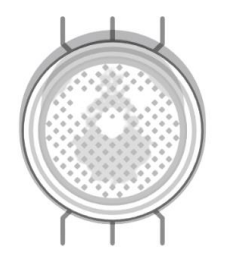
Circuito ejemplo

Sensor de Inclinación

Video explicativo

Circuito ejemplo

Sensor de gas

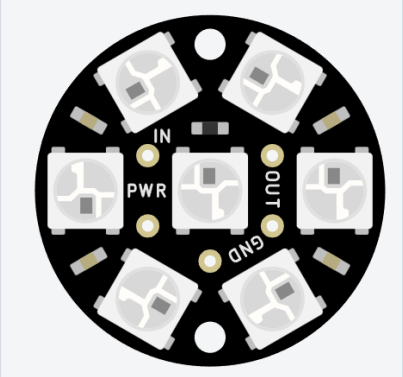


Video explicativo

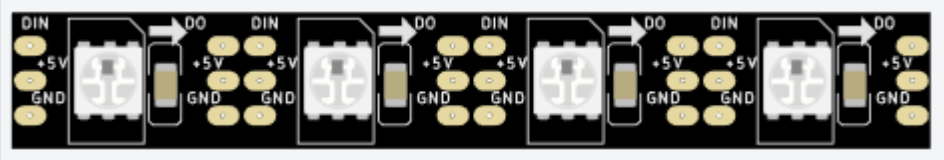
Circuito ejemplo

NeoPixel

NeoPixel son módulos leds RGB dispuestos de diferentes formas, capaces de encenderse y hacer secuencias lumínicas muy interesantes. Son usados con la biblioteca de Adafruit NeoPixel.

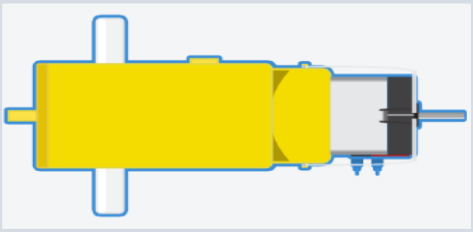


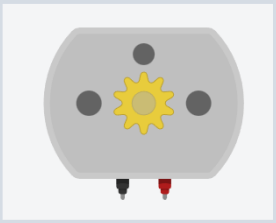
Curso de Arduino en TinkerCad - Estructura de control For y Neopixel



Motor de CC

Un motor de corriente contínua es capaz de transformar la electricidad en energía mecánica o de movimiento. Son sencillos de usar, según una polaridad dada, giran hacia un lado o el opuesto.

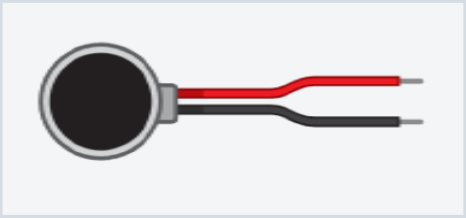




Motor de vibración

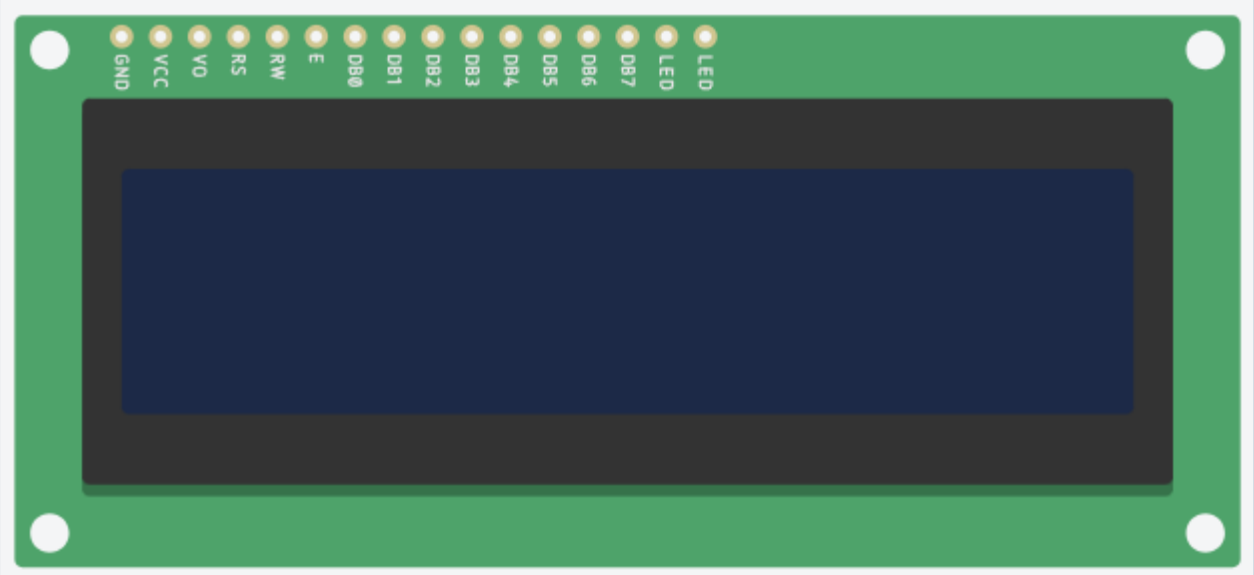
Es un motor de corriente contínua pero con una masa desbalanceada en su salida, lo que produce una vibración al girar.

Al ser más pequeño consume menos corriente.



LCD 16x2

Simulando un LCD en Tinkercad con Arduino

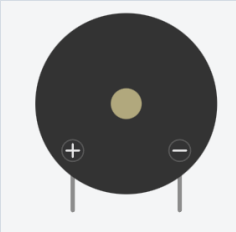


Altavoz

El altavoz tal vez es el actuador más común de todos, es utilizado para reproducir sonidos en distintas frecuencias. En la sintaxis de Arduino se dispone de la función tone() que permite generar una salida sonora en el pin indicado a una frecuencia deseada.

tone(pin, frequency)

tone(pin, frequency, duration)



Actividades E3/E4:

52. Actividad prioritaria: Contador vacuno: En un tambo quieren contar las vacas que ingresan y salen del comedero. Realizar un autómata (programa de arduino) que a partir de dos sensores PIR cuente ascendente o de forma descendente el ingreso y egreso de vacas respectivamente. Usar función millis() para que si el sensor capta durante 2 segundos haga el conteo ascendente o descendente. El monitor serie imprime el conteo de vacas.

El tambero pidió un croquis y la lista de materiales para el proyecto, ¿se lo podemos hacer?

53. Actividad complementaria: Alarma de gases tóxicos: hay un vecino que es gasista, el problema es que perdió el olfato! Se busca hacer un dispositivo para saber si hay presencia de gas. Solo se dispone de un sensor de gas y un buzzer.

54. Actividad prioritaria: Se desea realizar un sistema automatizado de para un invernadero. El invernadero cuenta con dos sensores, uno de luz para conocer si es de día o de noche y otro de humedad.

Cuenta con un motor DC que mueve una bomba de agua para regar las plantas. También cuenta con una pantalla LCD para mostrar en todo momento el valor de temperatura y la incidencia de luz.

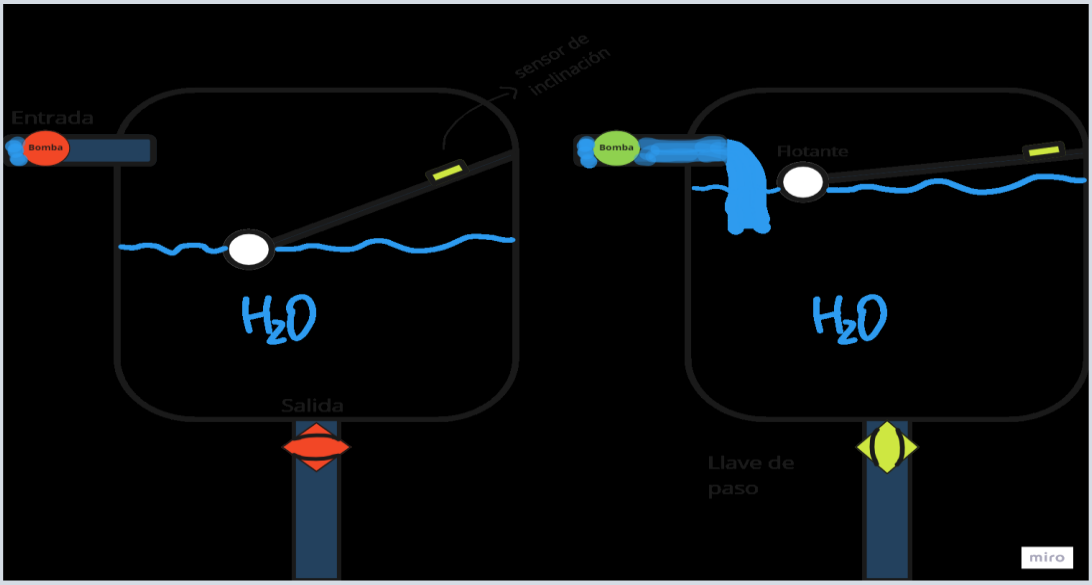
El control funciona de la siguiente manera:

○ Todas las noches riega hasta que la humedad sea del 80%.

○ Si la humedad es superior al 50% y es de día, no riega.Si la humedad es menor al 50% y es de día, riega hasta el 60%.

○ Si la humedad es menor al 20%, se riega a pesar de ser de día

55. Actividad complementaria: El mismo invernadero dispone de un sensor de inclinación y desea automatizar el llenado del tanque que provee el agua a las plantas. El tanque dispone de una bomba que permite el flujo de entrada de agua a través de un caño en la parte superior del tanque. En la parte inferior está la salida de agua para riego que es controlada por una válvula eléctrica(simular con un botón -regar y cerrado-). Se tiene que diseñar un programa que mientras no haya agua en el tanque, no se pueda regar y active la bomba de entrada hasta que haya suficiente agua. Mientras haya agua puede regarse las veces que se solicite.



56. Actividad complementaria: Hilario es un productor de ajos del este de Mendoza, dispone de un tractor y al cosechar va cargando la producción en acoplados. El problema es que al hacer marcha atrás siempre se atropella los acoplados. Se busca hacer un indicador de luces en el tablero del tractor (se puede usar el circuito 8Leds) que se active cuando se ponga marcha atrás. En el indicador se ve la escala de distancia que mide un sensor ultrasónico en la parte trasera del tractor.

57. Actividad prioritaria: En una avícola se les desea dar de comer a varias gallinas a la vez. Se desea desarrollar un programa que le dé de comer a las gallinas en el momento que un grupo de ellas lo pidan. La consigna que dejaron es que si hay más de 10 gallinas (aproximadamente) esperando a comer frente al comedero, el comedero se abre automáticamente a través de un motor. Se dispone de un sensor de fuerza (el peso es una fuerza vertical), el cual está dispuesto bajo una plataforma.

Cada gallina pesa 1.3kg, en promedio.

58. Actividad complementaria: Se desea automatizar el cultivo de champiñones para aumentar la producción y calidad del producto.

“La temperatura ideal para el cultivo de estos hongos es de 12º a 14º C, con una humedad ambiental del 75-80%. Así y todo, crecerán sin inconvenientes aún si la temperatura es de entre 8-18º C, es decir de fría a media. ” -podés conseguir el manual básico para el cultivo del champiñón en fruticola.com-.

Se dispone de un sensor de humedad ambiente (haremos de cuenta que el sensor de humedad de suelo de Tinkercad sirve para ambiente) y un sensor de temperatura TMP36. Los actuadores son: un motor que representa el calentador, un motor vibrador simulando ser el humidificador que es el encargado de aportar

humedad al ambiente y una pantalla para LCD para mostrar de forma ordenada los datos de temperatura, humedad y el estado de los motores (calentador y humidificador).

Desarrollar el programa y circuito electrónico con la solución.

59. Actividad complementaria: Investigación: COMPLETAR LA CONSIGNA con datos reales de empresas o pymes locales Y RESOLVER EL PROBLEMA. Contar a qué se dedica la empresa y colocar una breve descripción de la misma.

El gerente de la empresa "_____" produce "_____", y busca una manera de poder configurar una temperatura en todo el galpón para que sus trabajadores no tengan frío en invierno ni calor en verano. El ambiente es grande pero el sistema de calefacción funciona perfectamente y tiene dos modos de trabajo: "caliente" (1) y "frío" (0). Se busca mantener la temperatura en un rango de 15°C a 45°C. El gerente establece un valor de temperatura deseado mediante un potenciómetro, configurado en este rango de temperaturas. El sistema de control debe ser diseñado de tal manera que mantenga la temperatura en el valor deseado 3°C. Es decir ± que si establecemos el potenciómetro en 20°C, y la temp es 15°C, el sistema debería calentar hasta llegar a 17°C. Si la temperatura es 29°C, está enfriando, va a enfriar hasta los 23°C, ideal si llega a 20°C (se aconseja debatir en grupo como debería ser el movimiento del slider de temperatura para poder simular el comportamiento real que tendría este fenómeno).

3 Transductores (sensores): uno es un potenciómetro que establece la temperatura del que tiene un rango de 15°C a 45°C, el segundo es un sensor de temperatura que mide la temperatura del la planta de producción (galpón) y el tercero un botón que prende o apaga en cualquier momento el sistema.

Objetivo: realizar un autómata de control de temperatura y mostrar por monitor serie de forma ordenada los datos que consideren que son relevantes para el gerente.

Tema 5- Producción Web

En esta etapa del episodio aprenderemos los lineamientos básicos sobre la producción web para aplicarlo en el desarrollo de una página web para presentar el proyecto final del Episodio 3. La idea de este episodio no es desarrollar programación web, por ahora solo es una introducción al desarrollo web.

Para la creación de una página web hay que abordar varios ejes, ya que en ella podemos ver que el contenido está ordenado de una manera, la tipografía cambia según la sección de la página, las imágenes pueden ser visualizadas de una manera especial, los títulos son llamativos, entre otras cosas. Si nos ponemos a pensar, hay páginas que funcionan muy rápido y hay algunas que andan muy lento. El diseño o producción web se encarga de pensar en todo esto y es muy importante para que el funcionamiento completo de tu web sea rápido, óptimo y sobre todo funcional.

El diseño web implica trabajo relacionado con el layout y diseño de páginas online, así como la producción de contenido, aunque generalmente se aplica a la creación de sitios web.

El desarrollo web está basado en dos partes: el front end y el back end.

En diseño de software el front end es la parte del software que interactúa con los usuarios y el back end es la parte que procesa la entrada desde el front end.

La separación del sistema en front ends y back ends es un tipo de abstracción que ayuda a mantener las diferentes partes del sistema separadas. La idea general es que el front end sea el responsable de recolectar los datos de entrada del usuario, que pueden ser de muchas y variadas formas, y los transforma ajustándose a las especificaciones que demanda el back end para poder procesarlos, devolviendo generalmente una respuesta que el front end recibe y expone al usuario de una forma entendible para este. La conexión del front end con el back end es un tipo de interfaz.

Front end: interfaz gráfica e interacción con usuario

Back end: procesamiento de datos y respuestas

Para el front end suelen utilizarse lenguajes de programación que permitan realizar gráficos de manera sencilla. Más información aquí. La interfaz de cualquier sitio web o aplicación contiene programación que pone ciertas características a disposición de los usuarios.

Los más utilizados son:

1. HTML

2. CSS

3. JavaScript

Para back end suelen usarse lenguajes creados para el procesamiento de datos e información, los más utilizados son:

1. PHP.

2. Python.

3. Javascript.

Con el fin de crear una página web de calidad estética y funcional proponemos los siguientes links con información (pueden usar las webs que los facilitadores deseen):

● Aprende qué es el diseño web y lo que hace un profesional de esta área

Link:https://rockcontent.com/es/blog/diseno-web/#:~:text=El%20dise%C3%B1o%20web%20es%20un,lenguajes%20de%20marcado%20como%20HTML.

● Cómo Crear una Página Web con Google Sites 2022✓Paso a Paso

Link
https://www.youtube.com/watch?v=7JN0hi6yV0Q&ab_channel=Programaci%C3%B3nF%C3%A1cil%2CSEOyMarketing

● Producción de contenido para web.

https://rockcontent.com/es/blog/produccion-de-contenido-para-web/

Actividades E3/E4:

Crear una página web estilo presentación para tu mascota, para vos, o una empresa familiar. Puedes usar Google Sites o cualquier otro motor con el que estés familiarizado. Ambos facilitadores te ayudarán en este fin.

GitHub Episodio 3 / Estación 3

Fernando Del Pozzi Mendoza Futura