APRENDE
Y DIVIÉRTETE

Arduino

Módulo

8

Semana	Tiempo sugerido	Temas/ Subtemas	Aprendizaje esperados	Eje -Ámbitos-Ambientes Sociales de Aprendizaje
24	70-90 minutos	¿Qué es Arduino?	Comprende los conceptos de intensidad de corriente, diferencia de potencial, fuentes. Conoce la historia del led. Reconocer el concepto de conductividad	Número, Algebra y Variación. Análisis de Datos. Lúdico y Literario, Académico y Formación
25	70-90 minutos	Principios básicos	Escribir un primer programa en Arduino. Reconocer los diferentes tipos de puertos que puede utilizar. Reforzar sus conocimientos en el idioma inglés.	Número, Algebra y Variación. Análisis de Datos. Lúdico y Literario, Académico y Formación
26	70-90	Práctica con ultrasónico	Utiliza Arduino conectando el sensor ultrasónico para obtener mediciones.	Número, Algebra y Variación. Análisis de Datos
27	minutos	Practica con Sensor de Humedad	Aprende el funcionamiento del sensor de humedad con Arduino.	Número, Algebra y Variación. Análisis de Datos
28	70-90	Practica con Sensor de Gas MQ-9	Aprende el funcionamiento de un sensor de gas en Arduino.	Número, Algebra y Variación. Análisis de Datos

24. ¿Qué es Arduino?

Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Reconoce un lenguaje de programación de alto nivel.	Arduino IDE	¿Qué es Arduino? ¿Qué es Arduino IDE? El origen Anatomía de un Arduino	Conocer qué es Arduino. Reconocer y explorar el entorno de desarrollo de Arduino. Reforzar los conocimientos en el idioma inglés.

24. ¿Qué es Arduino?

Arduino es toda una plataforma para creación de electrónica de código abierto, la cual está basada en hardware y software libre, flexible y fácil de utilizar para los creadores y desarrolladores.

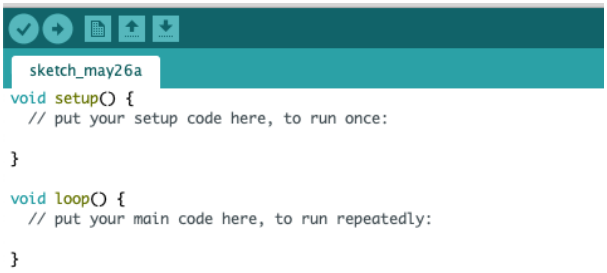
Este sistema te permite jugar para crear diferentes tipos de microordenadores desde una sola placa a los que la comunidad de creadores puede darles diferentes tipos de uso.

Para entender este concepto, primero tienes que entender los conceptos de hardware libre y el software libre. El hardware libre son los dispositivos cuyas especificaciones y diagramas son de acceso público, es decir, cualquier persona desde su cómodo colchón de su hogar puede fabricarlo.

El software libre son los programas informáticos cuyo código es accesible por cualquiera para que quien quiera pueda utilizarlo y modificarlo.

¿Qué es Arduino IDE?

Un IDE es un Entorno de Desarrollo Integrado Interactivo, (Integrated Development Environment). Es un espacio de programación con el que cualquiera puede crear aplicaciones para las placas Arduino, de manera que se les puede dar todo tipo de utilidades.



Sin embargo, nunca se pensó que esta placa se llegaría a convertir en el líder mundial de tecnologías DIY (Do It Yourself). Inicialmente fue un proyecto creado para economizar la creación de proyectos escolares dentro del instituto, y después Banzi tuvo la intención de ayudar a su escuela a evitar la quiebra de la misma con las ganancias que produciría vendiendo sus placas dentro del campus a un precio de1 euro.

Conoce más…

Arduino fue inventado en el año 2005 por el entonces estudiante del instituto IVRAE Massimo Banzi, quien pensaba en hacer Arduino por una necesidad de aprendizaje para los estudiantes de computación y electrónica del mismo instituto, ya que desde entonces, comprar una placa de micro controladores eran bastante caro y no ofrecían el soporte

El origen

Debido a su origen el primer prototipo de Arduino fue fabricado en el mismísimo instituto IVRAE.

Inicialmente estaba basado en una simple y común placa de circuitos eléctricos, donde estaban conectados un micro controlador sencillo junto con resistencias de voltaje, además, sólo se podían conectar sensores simples como leds u otras resistencias. Y aún no contaba con el soporte de algún lenguaje de programación para controlarla.

Años después, se integró al equipo Hernando Barragán, estudiante de la Universidad de Colombia que se encontraba haciendo su tesis (como cualquier otro), y tras enterarse de este proyecto, contribuyó al desarrollo de un entorno para la programación del procesador.

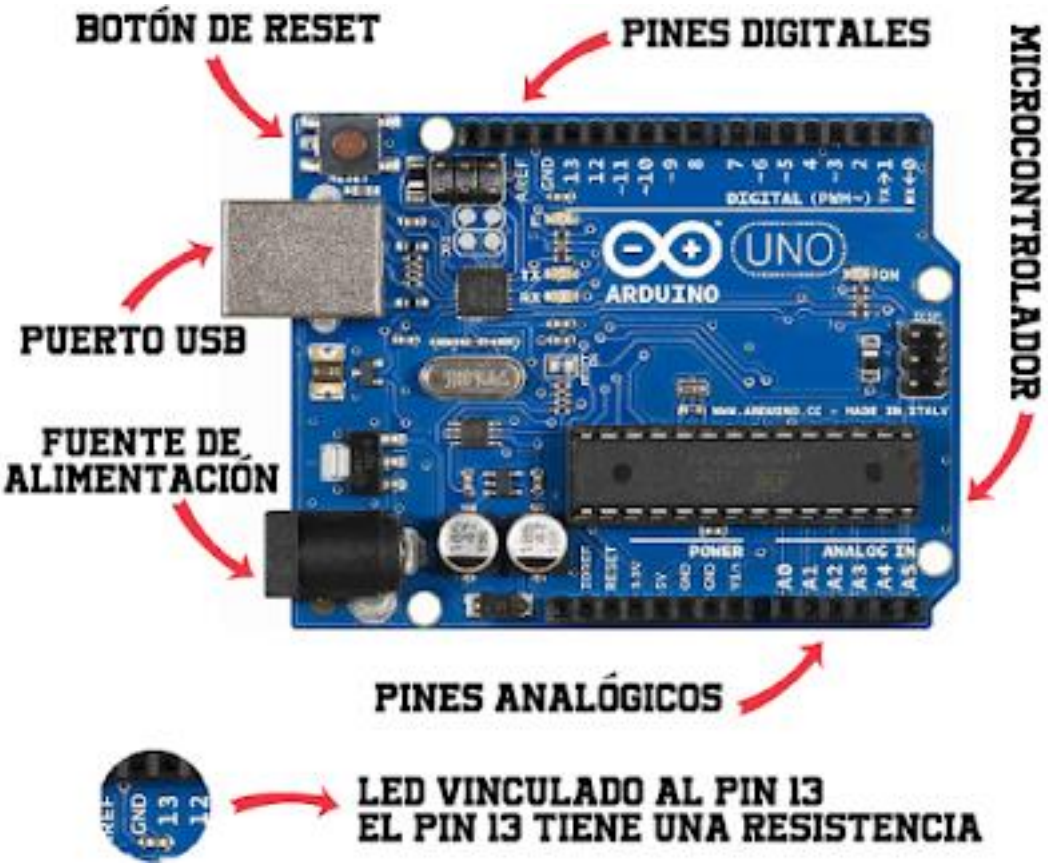
Tiempo después, se integró al "Equipo Arduino" el español David Cuartielles, experto en circuitos, quien ayudó Banzi a mejorar la interfaz de hardware agregando los microcontroladores y memoria al lenguaje de programación para manipular esta plataforma.

Más tarde, Tom Igoe, un estudiante de Estados Unidos que se encontraba

haciendo su tesis, se interesó en el proyecto y fue a visitar las instalaciones del Instituto IVRAE para averiguar en que estaban trabajando. Tras regresar a su país natal, recibió un e-mail donde Massimo lo invitó a trabajar con su equipo. Una vez aceptada la invitación agregó puertos USB para poder conectarla a un ordenador.

Y lo demás es historia…

Anatomía de un Arduino



25. Principios básicos

Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Interactúa con el lenguaje de programación de Arduino. Desarrolla competencias necesarias para hacer su primer programa.	Arduino IDE	Tu “Hola mundo en Arduino” (Tu primer programa)	Escribir un primer programa en Arduino. Reconocer los diferentes tipos de puertos que puede utilizar. Reforzar sus conocimientos en el idioma inglés.

Tu “Hola mundo en Arduino”

(Tu primer programa)

Es una costumbre en el mundo de la programación que el primer programa que se crea es el llamado: “Hello World” que es imprimir la frase en un dispositivo de visualización o terminal. Para este caso y aprovechar al máximo el hardware de Arduino vas a encender un led.

¿Sabías…?

Al comienzo del proyecto Wiring, tanto Hernando como Massimo, se reunían a menudo en un bar de Ivrea, llamado «Bar di Re Arduino». Por este motivo, Massimo, decidió llamar a su proyecto Arduino.

Instrucciones.

Primero abre el IDE de Arduino:

Después te diriges a Archivo -> Ejemplos -> 01. Basics -> Blink

En la pantalla se desplegará lo siguiente:

d

“Void setup” son las funciones que se inician al correr el programa una sola vez.

Ocuparas el pin como salida al cual llamados LED-BUILTIN.

“Void loop” es donde se encuentran los bucles que se correrán durante todo el tiempo que dure el programa.

En este caso podemos dos estados intermitentes, apagado/LOW y encendido/HIGH. Con un delay o retraso de 1,000 milisegundos que es igual a 1 segundo.

En la esquina superior izquierda se encontrará la palomita de verificar. Al hacer clic el IDE verificará que tu código no tenga errores, esto al compilar dicho programa.

Una vez que pasa la prueba continúa con cargar el archivo a la placa. Para este caso hazlo por medio del cable USB. Y seleccionamos “subir” que se encuentra localizado al lado de “verificar”.

Una vez que está cargado el programa en la placa ésta deberá parpadear intermitentemente cada segundo. Y es así como has construido tu primer programa en Arduino.

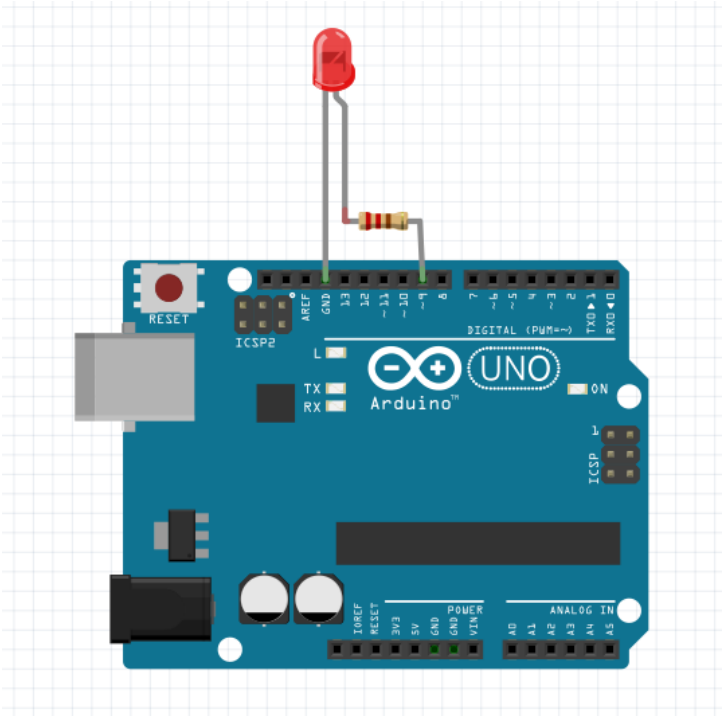
Enciende un led

Define en tu código el pin9 como un entero y después como salida.

Haz lo que en el programa anterior y la diferencia radica en el hardware:

El led va a conectado de su pata más larga a la resistencia de 330 Ohms y de la resistencia al pin 9 de salida.

La otra pata del led va directamente a GND quedando así:



Ahora procederás a realizar los mismos ejercicios con los sensores que utilizaste conectándolos para la Raspberry, recuerda que un Arduino se podrán observar variables analógicas y habrá un mundo de posibilidades. Si quieres saber más de los sensores lee la introducción teórica de los módulos anteriores.

26. Práctica con Sensor Ultrasónico

Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Utiliza Arduino conectando el sensor ultrasónico para obtener mediciones.	Arduino IDE Sensor Ultrasónico	Conectar un sensor ultrasónico a Arduino	Escribir un primer programa en Arduino utilizando sensor Ultrasónico. Reconocer los diferentes tipos de puertos que puede utilizar.

Practica con Sensor Ultrasónico

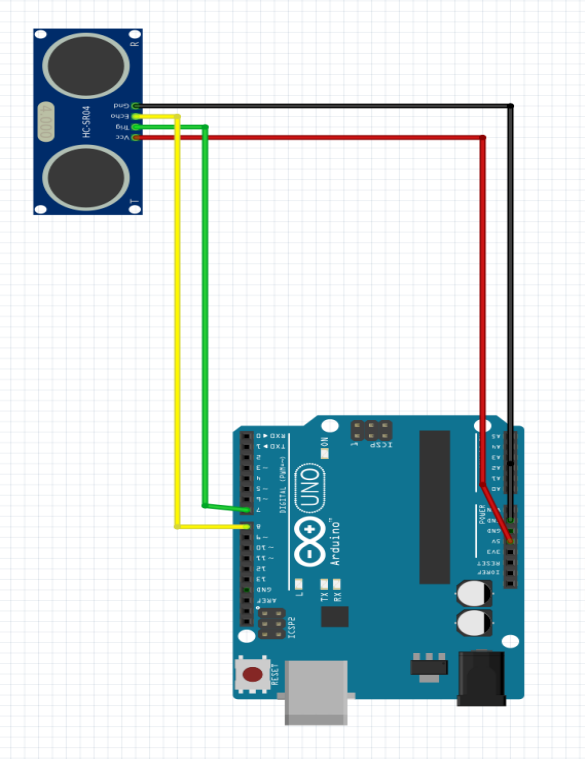
Para la práctica se requieren los siguientes componentes:

- Sensor HC-SR04
- Cables conectores.

Es muy importante conectar los componentes de forma correcta para evitar todo tipo de problemas.

Conectar a Arduino

El sensor HC-SR04 se conecta al Arduino de la siguiente manera:



//Como el eco viaja de ida y de regreso, dividimos entre dos.

El código quedará de la siguiente manera:

Archivo Editar Sketch Herramientas Ayuda

distancia

```
// trigger y echo del sensor
const int tri = 7;
const int eco = 8;

void setup() { // inicializar puerto serial y poines del arduino
  Serial.begin(9600);
  pinMode(tri, OUTPUT);
  pinMode(eco, INPUT);}

void loop()
{ long tiempo, cm;

  digitalWrite(tri, HIGH);
  delayMicroseconds(10);
  digitalWrite(tri, LOW);

  // of the ping to the reception of its echo off of an object.
  tiempo = pulseIn(eco, HIGH);

  // para convertir el tiempo en distancia
  // la velocidad del sonido es 340 m/s o 29 microsegundos por centimetro.
  // el eco viaja 2 veces la distancia, por lo tanto:

  cm = tiempo/29/2;

  Serial.print(cm);
  Serial.print(" cm");
  Serial.println();
  delay(500);
}
```

27. Práctica de sensor de Humedad

Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Aprende el funcionamiento del sensor de humedad con Arduino.	Arduino IDE Sensor de humedad Cables	Conectar sensor de humedad.	Conectar un sensor de humedad en Arduino. Reconocer los diferentes tipos de puertos que puede utilizar.

27. Práctica de sensor de Humedad

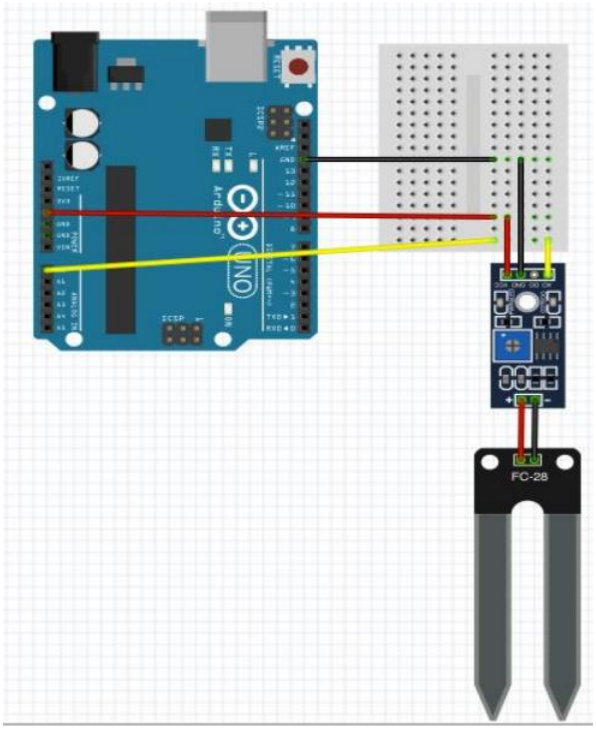
++++++++++++++++++++++++++++++++

Para la práctica se requieren los siguientes componentes:

- Un protoboard.
- Cables conectores.
- Sensor de humedad FC-28
- Potenciómetro

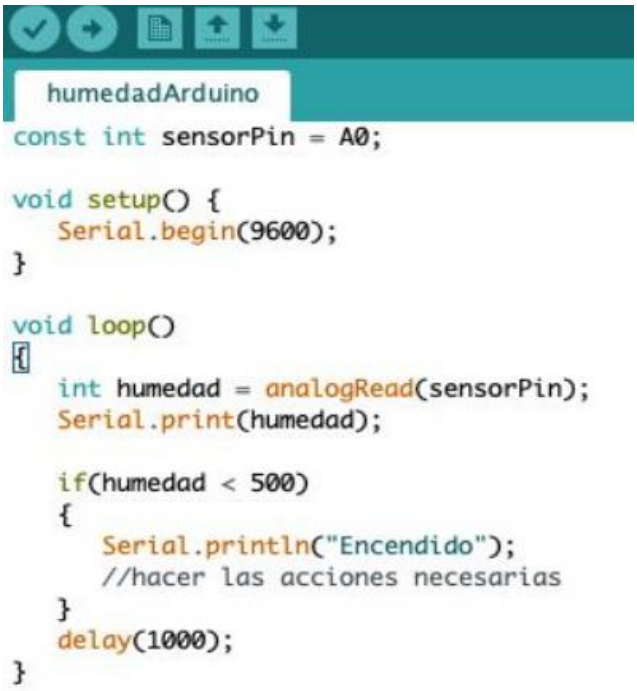
Es muy importante conectar los componentes de forma correcta para evitar todo tipo de problemas

Conectar dispositivos en Arduino Uno de la siguiente manera:



Como vemos es mucho más sencillo que en la Raspberry, la diferencia radica en que hay menos pines y los módulos para operar los sensores son innecesarios.

El código en Arduino queda así, como vemos utilizamos el puerto analógico A0. Sin más.



28. Práctica Sensor de gas MQ-9

Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Aprende el funcionamiento de un sensor de gas en Arduino.	Arduino IDE Sensor de gas MQ-9	¿Cómo se utiliza en la detección de gas?	Conocer como conectar un sensor de gas en Arduino. Reconocer los diferentes tipos de puertos que puede utilizar.

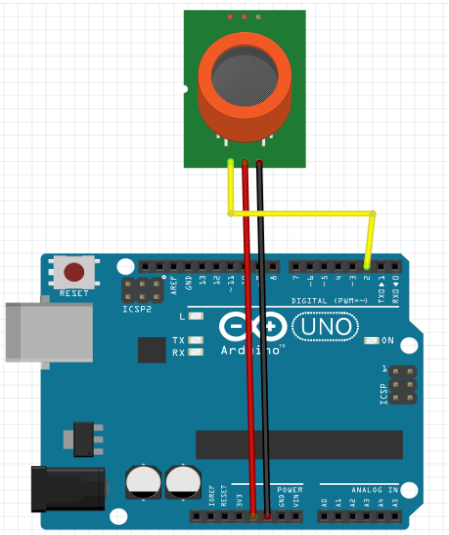
28. Práctica Sensor de gas MQ-9

Para la práctica se requieren los siguientes componentes:

- MQ-9.
- Cables conectores.

Es muy importante conectar los componentes de forma correcta para evitar todo tipo de problemas.

Para el caso de que se ocupa un Arduino Uno en vez de Raspberry PI, nuestro esquema de conexión quedaría así:



¿Bonito no lo creen? Y bastante parecido a las conexiones en Raspberry Pi.

El código para este último queda como:

MQ-9-lecturaDigital

```
const int MQ_PIN = 2;
const int MQ_DELAY = 2000;

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


void loop()
{
  bool state= digitalRead(MQ_PIN);

  if (!state)
  {
    Serial.println("Deteccion");
  }
  else
  {
    Serial.println("No detectado");
  }
  delay(MQ_DELAY);
}
```

Aquí declaramos que el PIN 2 como el de la señal.

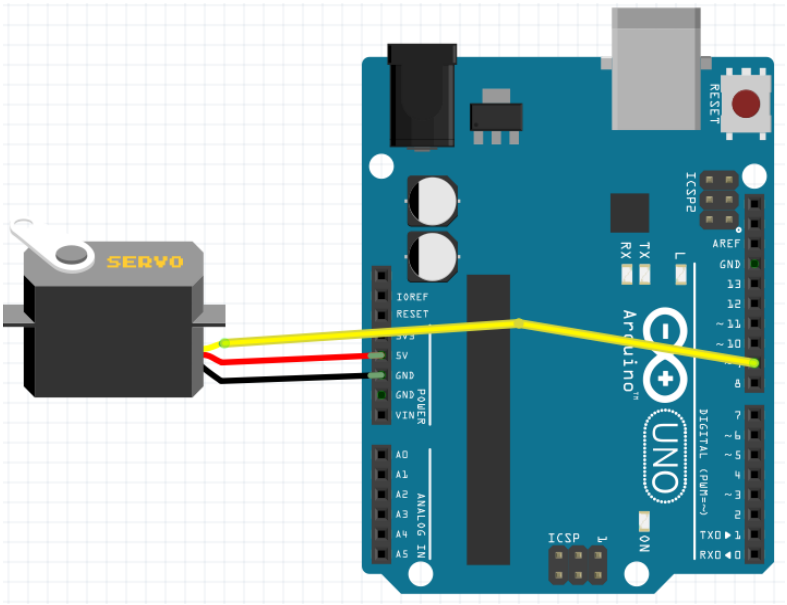
29. Práctica con Motores1: Servomotor.

Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Interactúa con el lenguaje de programación de Arduino. Desarrolla competencias necesarias para conectar un motor.	Arduino IDE Servomotor	Hacer funcionar un motor.	Conectar tu primer motor con servomotor Reconocer los diferentes tipos de puertos que puede utilizar.

29. Práctica con Motores1: Servomotor

++++++++++++++++++++++++++++++++++



Para la práctica se requieren los siguientes componentes:

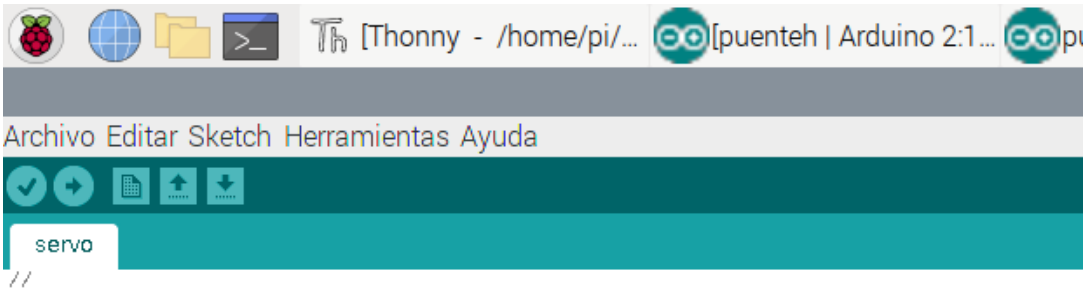
- Un servo
- Cables conectores.

Es muy importante conectar los componentes de forma correcta para evitar todo tipo de problemas.

MOTORES 1

Así se conecta el servo al Arduino:

Y así se ve el código en Arduino:




```
//

#include <Servo.h>

Servo servo;       // crea un objeto servo, se pueden tener hasta 8
int   p = 0;       // variable para almacenar su posicion

void setup()
{
 servo.attach(9);  // conecta servo al pin 9
}


void loop()
{
 for(p = 0; p < 180; p += 1)       // va de 0 a 180 grados, en pasos de 1 grado
  { servo.write(p); delay(15);}    // escribir la posicion en servo y espera 15 ms

 for(p = 180; p >= 1; p -= 1)      // va de 0 a 180 grados, en pasos de 1 grado
  { servo.write(p); delay(15);}    // escribir la posicion en servo y espera 15 ms

}
```

//Incluimos la librería disponible en Arduino llamada servo y definimos el pin 9 como el de la señal para nuestro motor a pasos (servo).

30. Práctica Controlar Motores 2: Puente H

Aprendizajes esperados

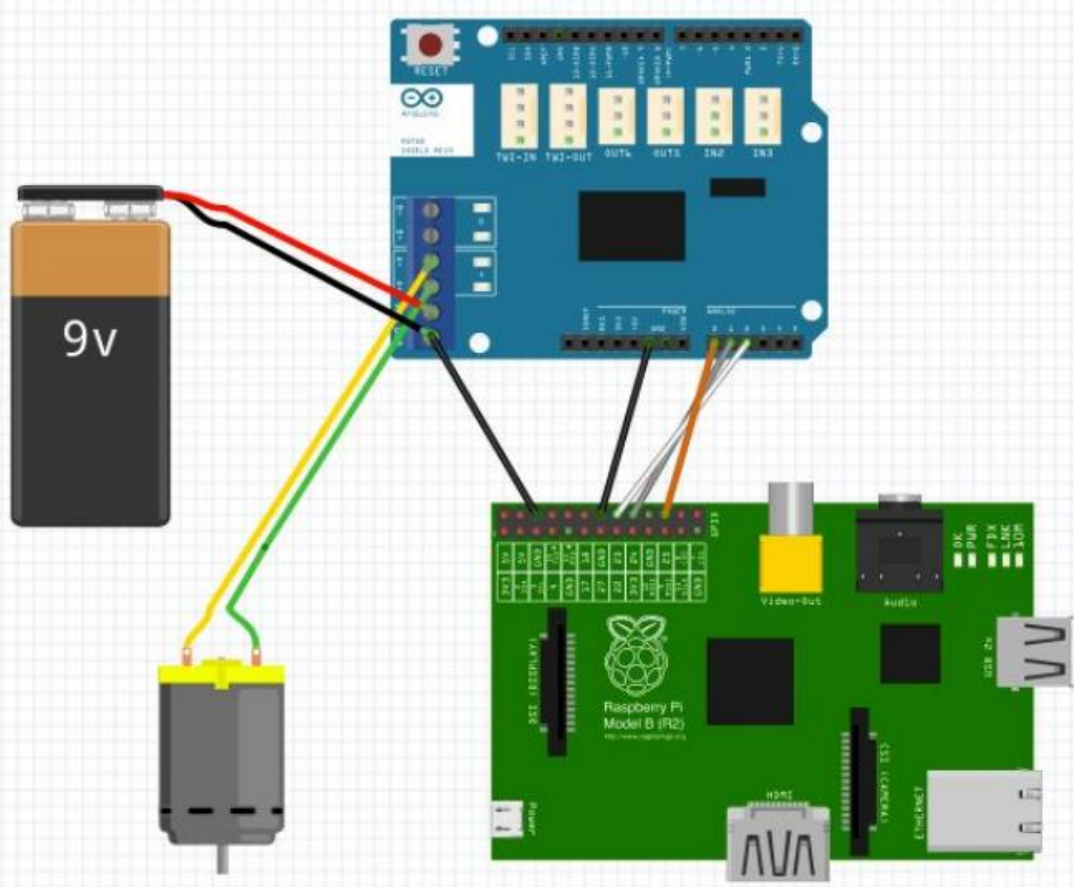
Habilidades	Medio	Contenido	Finalidad
Interactúa con el lenguaje de programación de Arduino. Desarrolla competencias necesarias para diferenciar un servomotor y un puente H	Arduino IDE Pila 9v Motor DC 12v	Conectar tu primer motor en puente H.	Escribir un primer programa en Arduino con motor. Reconocer los diferentes tipos de puertos que puede utilizar.

30. Práctica Controlar Motores 2: Puente H

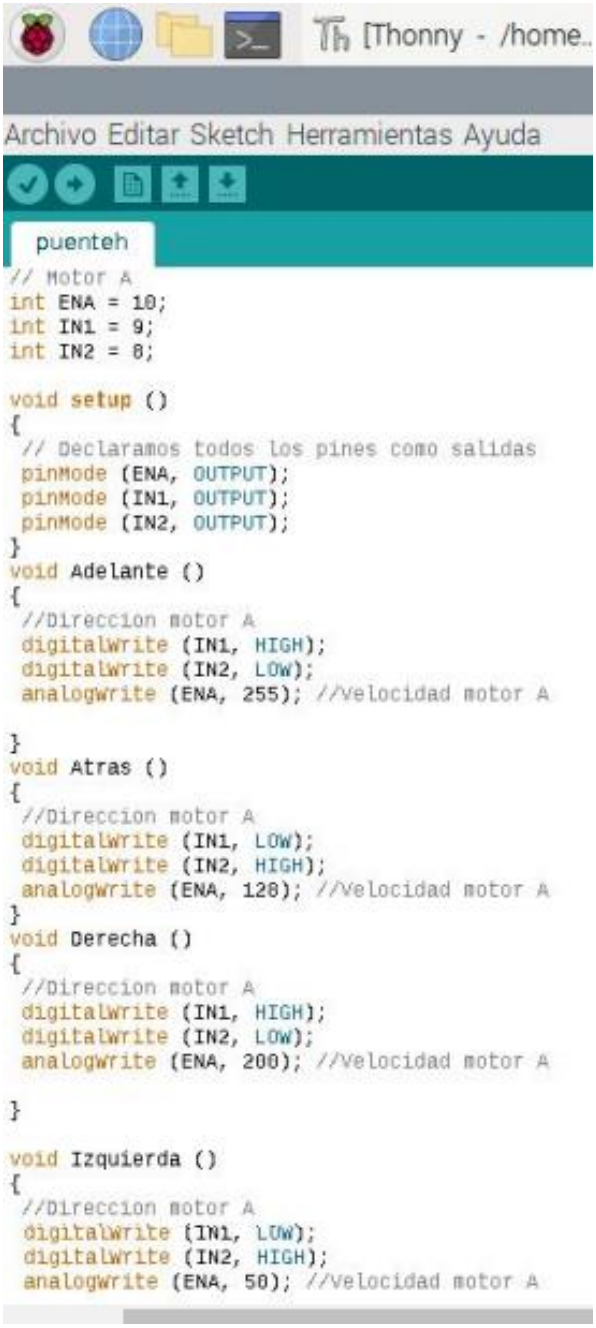
++++++++++++++++++++++++++++++++++
Para la práctica se requieren los siguientes componentes:

- Pila 9v
- Motor DC 12v
- Cables conectores.

Es muy importante conectar los componentes de forma correcta para evitar todo tipo de problemas.



El código de Arduino se verá así:




```
// Motor A
int ENA = 10;
int IN1 = 9;
int IN2 = 8;

void setup ()
{
 // Declaramos todos los pines como salidas
 pinMode (ENA, OUTPUT);
 pinMode (IN1, OUTPUT);
 pinMode (IN2, OUTPUT);
}
void Adelante ()
{
 //Direccion motor A
 digitalWrite (IN1, HIGH);
 digitalWrite (IN2, LOW);
 analogWrite (ENA, 255); //Velocidad motor A

}
void Atras ()
{
 //Direccion motor A
 digitalWrite (IN1, LOW);
 digitalWrite (IN2, HIGH);
 analogWrite (ENA, 128); //Velocidad motor A
}
void Derecha ()
{
 //Direccion motor A
 digitalWrite (IN1, HIGH);
 digitalWrite (IN2, LOW);
 analogWrite (ENA, 200); //Velocidad motor A

}

void Izquierda ()
{
 //Direccion motor A
 digitalWrite (IN1, LOW);
 digitalWrite (IN2, HIGH);
 analogWrite (ENA, 50); //Velocidad motor A
```

//Como se observa, declaramos las funciones, que fungirán como las direcciones de nuestro motor desde el puente h.


```
 //Direccion motor A
 digitalWrite (IN1, LOW);
 digitalWrite (IN2, HIGH);
 analogWrite (ENA, 128); //Velocidad motor A
}
void Derecha ()
{
 //Direccion motor A
 digitalWrite (IN1, HIGH);
 digitalWrite (IN2, LOW);
 analogWrite (ENA, 200); //Velocidad motor A

}

void Izquierda ()
{
 //Direccion motor A
 digitalWrite (IN1, LOW);
 digitalWrite (IN2, HIGH);
 analogWrite (ENA, 50); //Velocidad motor A

}
void Parar ()
{
 //Direccion motor A
 digitalWrite (IN1, LOW);
 digitalWrite (IN2, LOW);
 analogWrite (ENA, 0); //Velocidad motor A

}
void loop ()
{
 Adelante ();
 delay (5000);
 Atras ();
 delay (3000);
 Derecha ();
 delay (2000);
 Izquierda ();
 delay (2000);
 Parar ();
 delay (4000);
}
```

//Seguimos con una secuencia de direcciones para hacer una función de inicio con retrasos entre ellos, vistos como delays.