

ARDUINO

ARDUINO DESDE 0

CON ARDUINOBLOKS



[Descripción breve](#)

He realizado un resumen de los tutoriales de Arduino realizados por el canal de YouTube
Aprendemanía

Pere Manel Verdugo Zamora
www.peremanelv.com

1.- ¿Cómo usar ARDUINO?

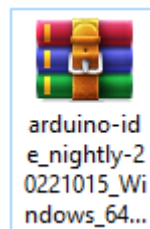
SW para programar Arduino

Existe muchos SW (Programas) para poder programar Arduino. Pero los podemos resumir en 2 grandes categorías:

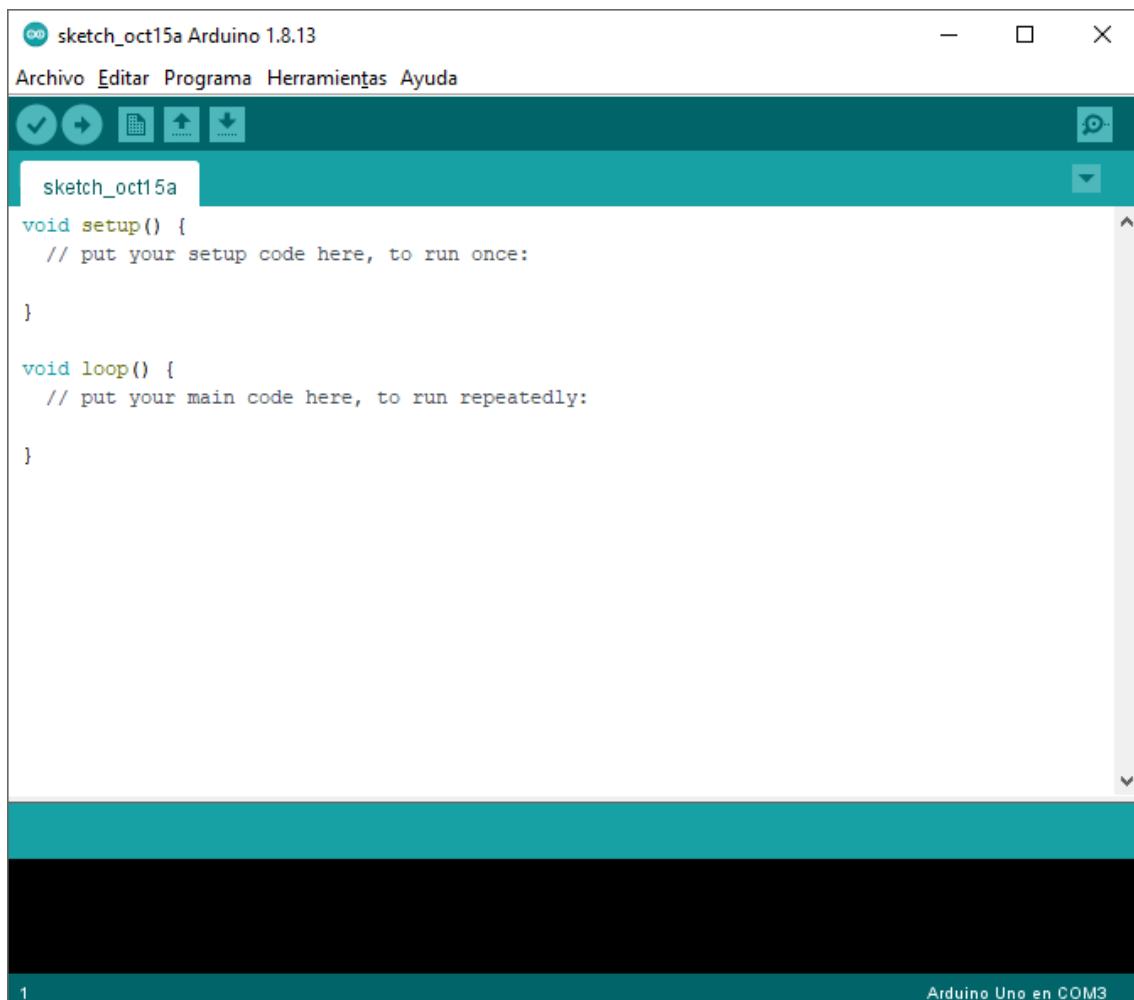
1. **ENTORNO DE DESARROLLO OFICIAL (IDE).** Contamos con un IDE para casi todas las plataformas (Windows, Linux, Mac). Un IDE un lugar donde podemos escribir nuestras aplicaciones (POR COMANDOS), descargarlas al Arduino y ejecutarlas o depurarlas desde allí. El entorno de desarrollo es gratuito y descargable desde aquí.

<https://www.arduino.cc/en/software>

Una vez descargado el software:

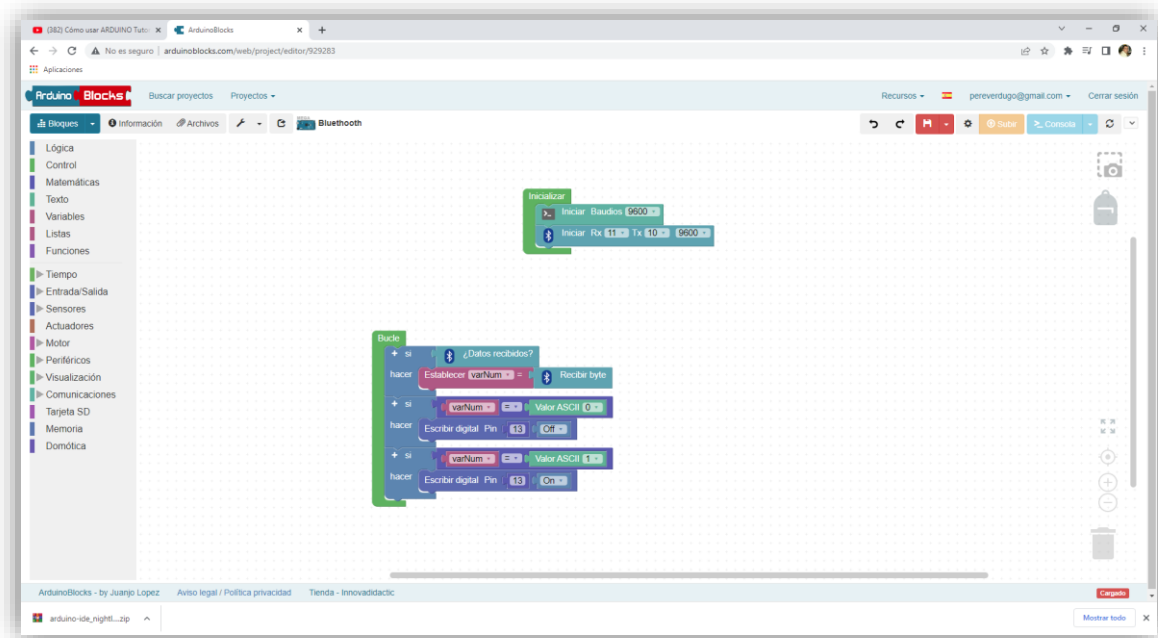


Lo descomprimes y lo instalas.



2. **OTROS ENTORNOS DE DESARROLLO MÁS SENCILLOS.** Para poder programar por BLOQUES como por ejemplo: ArduinoBlocks o MBlock. Nosotros vamos a trabajar con **ARDUINOBLOCKS.**

Trabajaremos On-Line. <http://www.arduinoblocks.com/web/site/login>

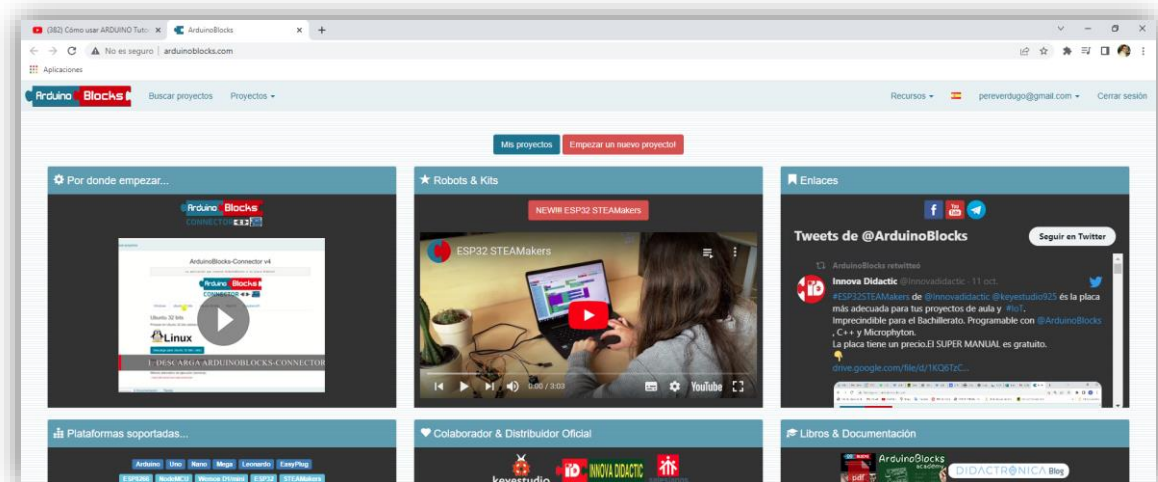


ArduinoBlocks

ArduinoBlok es una plataforma **Online de uso libre y gratuito** donde podemos programar nuestra placa Arduino de forma visual, a través de bloques, sin necesidad de conocer el lenguaje C++/Processing que utiliza Arduino IDE.

Este proyecto está orientado a jóvenes desde Primaria hasta Bachillerato y ha sido desarrollado por Juanjo López, profesor de FP, al cual desde Aprendermanía queremos agradecer enormemente su esfuerzo y contribución por el desarrollo de esta magnífica herramienta para poder programar de forma sencilla Arduino y otros microcontroladores similares.

Al proyecto accedemos a través de la página: <http://www.arduinoblocks.com/>



Registro nuevo Usuario

Podemos comenzar a programar nuestro Arduino con ArduinoBlocks directamente entrando a la página Web de la Plataforma y dando el botón **Probar ahora**, sin necesidad de registrarnos. Sin embargo nos vamos a perder algunas de las funcionalidades de la plataforma:

- Guardar tus proyectos en la nube de ArduinoBlocks.
- Añadir información al proyecto: descripción, componentes, utilizando imágenes, etc.
- Añadir archivos adjuntos relacionados con el proyecto: esquemas, fotos archivos para impresión 3D, aplicaciones, etc.
- Compartir proyectos con el resto del mundo.
- Importar proyectos compartidos por otros usuarios.
- Importar y exportar proyectos en archivo para publicar en web o compartir.
- Exportar el código .ino para utilizarlo en Arduino IDE.
- Utilizar la consola serie desde el propio navegador

Por lo tanto debemos, para empezar a usar ArduinoBlocks y tener acceso a todas sus funcionalidades debemos de registrarnos para tener nuestra cuenta de usuario. Para ello debemos pinchar en **“Iniciar sesión”** y a continuación en la opción **“nuevo usuario”**. Entonces rellenaremos el formulario que nos aparecerá.

A continuación revisa tu bandeja de entrada del correo electrónico y **recibirás un correo con un enlace para confirmar y activar tu cuenta de usuario**. A partir de ese momento puedes iniciar sesión y crear tus proyectos dentro de tu cuenta de ArduinoBlocks.

Empezar un nuevo Proyecto

En ArduinoBlocks puedo trabajar con proyectos de 2 formas:

1. CREAR NUEVOS PROYECTOS.

Para empezar un nuevo proyecto damos al botón “Empezar Nuevo Proyecto” o en el menú Proyecto seleccionamos “Nuevo Proyecto”. Tenemos entonces la siguientes opciones:

- Iniciar un nuevo proyecto que sólo será accesible para el usuario. Posteriormente se puede compartir al resto de la comunidad si se desea.
- Iniciar un proyecto como profesor. Esta funcionalidad se solo para profesores.
- De este forma nos unimos a un proyecto planteado por el profesor. Nosotros realizaremos el proyecto como si de un proyecto personal se tratara, pero el profesor podrá supervisar y valorar nuestro trabajo.

2. CONTINUAR TRABAJANDO A PARTIR DE MIS PROYECTOS CREADOS ANTERIORMENTE.

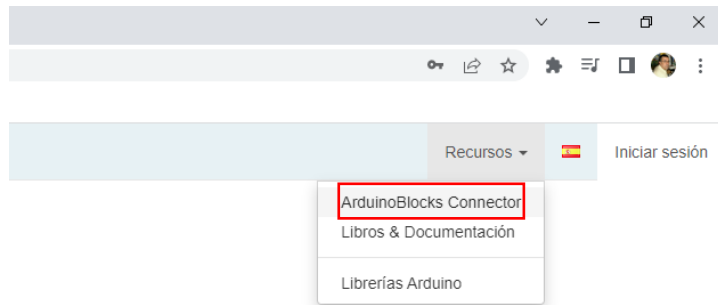
ArduinoBlocks Conector

Para poder cargar los programas desde Arduino-blocks en la placa Arduino.

1. Hay que tener instalado el programa ArduinoBlocks-connector.
2. Hay que estar ejecutando ArduinoBlocks-connector.

Instalación de ArduinoBlocks Conector.

La instalación de ArduinoBlocks-connector solo hay que realizarla una vez por ordenador, pero el archivo habrá que tenerlo abierto cada vez que se quiera cargar el programa. Podemos acceder al área de descarga directamente desde la página principal:



Buscamos la versión correspondiente a nuestro sistema operativo, la descarga y la instalación en nuestro ordenador.



Una vez lo hayamos descargado procederemos a su instalación, solo lo tenemos que hacer una vez.

Conectar y detectar Arduino

Al hacer doble clic sobre el icono anterior, se abrirá la siguiente pantalla en el escritorio. Simplemente hay que dejarla abierta (se puede minimizar).

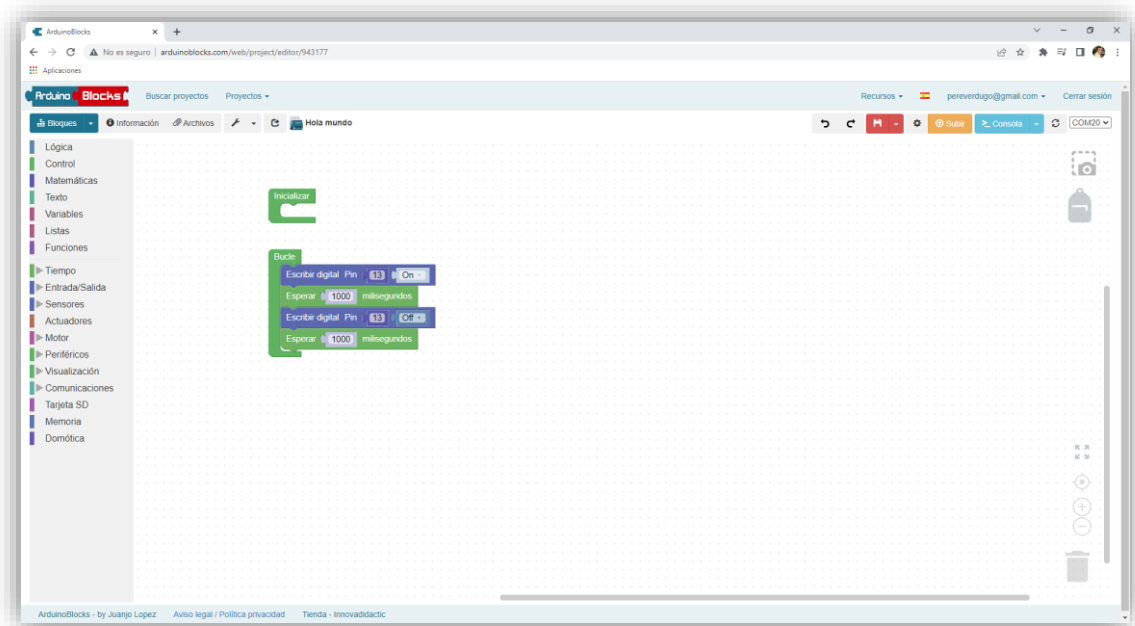


NO DEBEMOS OLVIDAR SIEMPRE QUE VAYAMOS A TRABAJAR CON ARDUINOBLOCK, CARGAR ESTE PROGRAMA EN NUESTRO ORDENADOR Y NO CERRARLO SINO MINIMIZARLO.

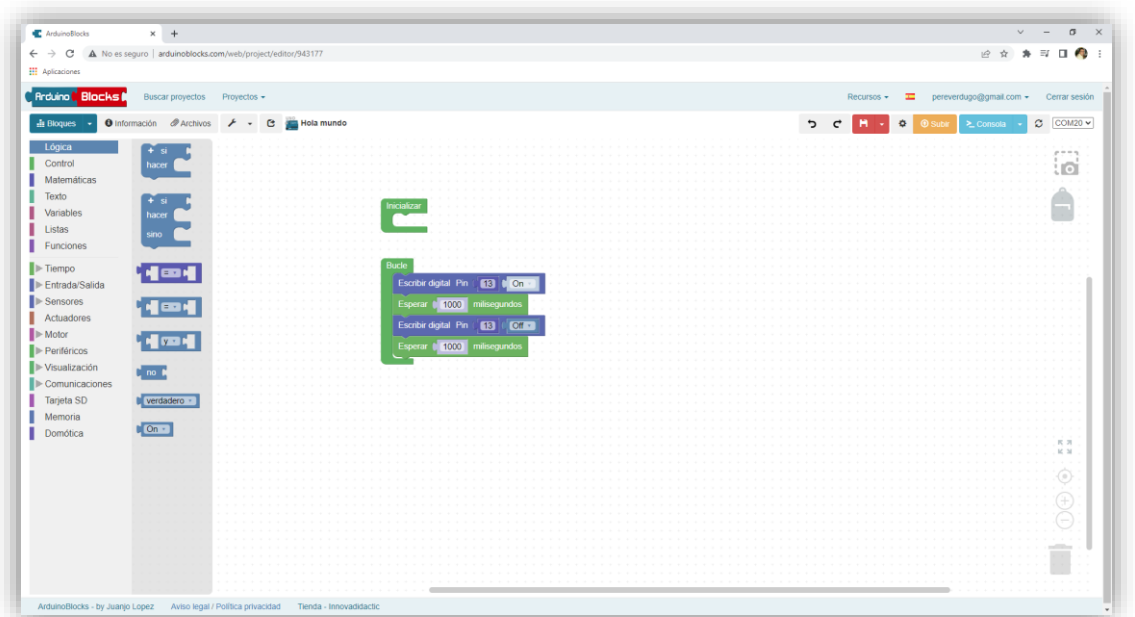
Nuestro primer proyecto

Vamos a seguir los siguientes pasos para la construcción de nuestro primer programa en Arduino con ArduinoBlocks:

1. Crear un proyecto nuevo.
2. Interface de Trabajo.
3. Programa Encender/Apagar Led integrado Arduino (Pin 13)
4. Conectar Cable Arduino al Ordenador.
5. Cargar el programa de ArduinoBlocks a nuestro ordenador.



En la parte izquierda encontramos todos los bloques organizado por colores, si seleccionamos uno.



Nos muestra los bloques pertenecientes a este grupo.

Hay que seleccionar el bloque que necesitamos y lo arrastramos a la ventana del centro, que es donde programaremos.



Hay bloques que tienen un signo más para poder configurar el bloque.

En la parte inferior derecha tenemos las opciones para modificar el zoom y además tenemos la papelera para eliminar aquellos bloques que no deseamos.



En la parte superior izquierda en el apartado bloques podemos ver el código que tendríamos que utilizar si trabajáramos con el editor de Arduino.

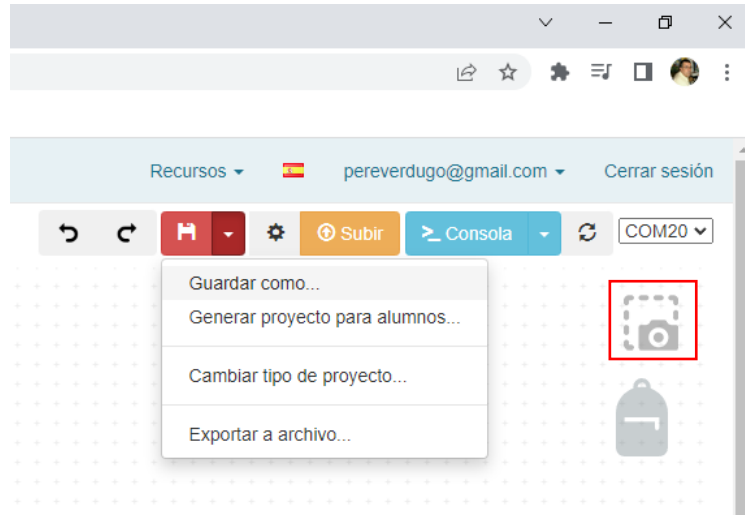
```
Código Arduino Copiar

1 void fnc_dynamic_digitalWrite(int _pin, int _e){
2   pinMode(_pin,OUTPUT);
3   digitalWrite(_pin,_e);
4 }
5
6 void setup()
7 {
8
9
10 }
11
12
13 void loop()
14 {
15
16   fnc_dynamic_digitalWrite(13, HIGH);
17   delay(1000);
18   fnc_dynamic_digitalWrite(13, LOW);
19   delay(1000);
20
21 }
```

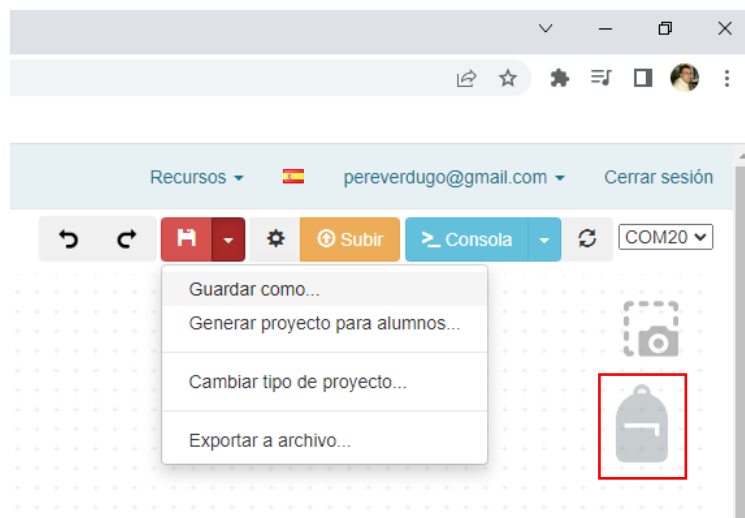
Lo que hace el programa es traducir la programación en bloque en el código oficial de Arduino, que es que finalmente sube a nuestra placa.

También lo puedo descargar, para cargarlo en el IDE oficial de Arduino y desde la aplicación oficial de Arduino poderla subir.

El proyecto se va guardando automáticamente pero si nos queremos asegurar.



Con el botón seleccionado podemos hacer una fotografía de nuestra pantalla con el código y poderla compartir.



La mochila, podemos pasar un conjunto de bloques para poder tener acceso desde otro proyecto.

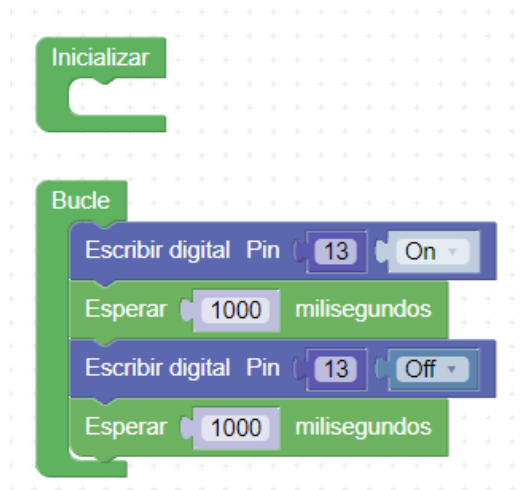
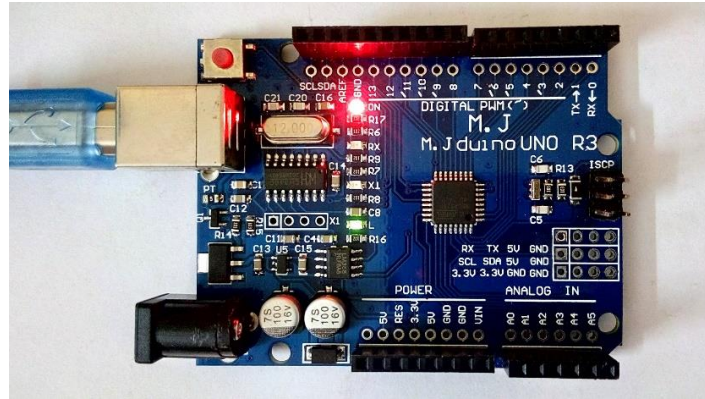
Vamos a realizar nuestro primer proyecto para esto vamos a comentar un concepto que hay que tener en cuenta.



El primer bloque se llama Iniciar, todas las instrucciones que añadimos a este bloque se ejecutarán primero y solo una vez, en cambio el bloque Bucle se ejecutará en un bucle infinito.

Estos dos bloques aparecen siempre por defecto al iniciar un nuevo proyecto.

Nuestro primer proyecto consiste en encender un led que está integrado en la placa de Arduino y es el Pin 13.



En este ejemplo solo vamos a utilizar el bloque de Bucle.

Al poner Escribir digital Pin 13 en On le estamos pasando 5 voltios, esto hará que se encienda la luz.

Hacemos una espera de 1000 milisegundos, un segundo para que podamos ver con un intervalo de un segundo como se enciende y se apaga la luz.

Al poner Escribir digital Pin 13 en Off dejamos que no pasen los 5 voltios.

Volvemos a hacer una espera de 1 segundo y el ciclo empieza de nuevo.

Este es el código Original.

```
1 void fnc_dynamic_digitalWrite(int _pin, int _e){
2   pinMode(_pin,OUTPUT);
3   digitalWrite(_pin,_e);
4 }
5
6 void setup()
7 {
8
9
10 }
11
12
13 void loop()
14 {
15
16   fnc_dynamic_digitalWrite(13, HIGH);
17   delay(1000);
18   fnc_dynamic_digitalWrite(13, LOW);
19   delay(1000);
20
21 }
```

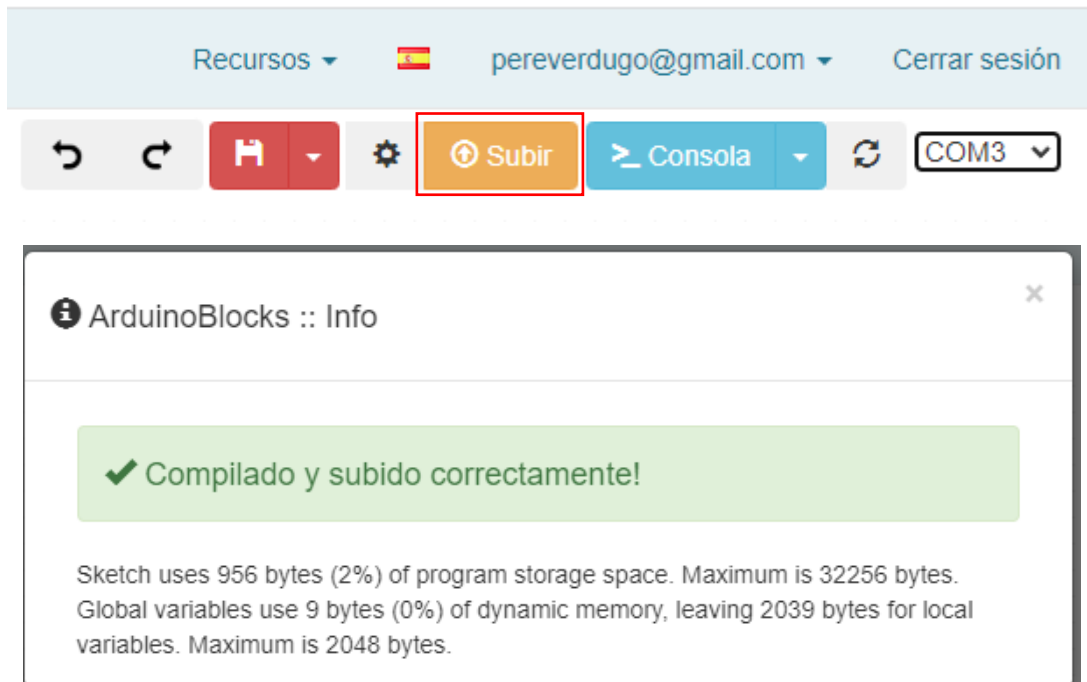
Vamos a subir el proyecto a la placa de Arduino.

Con el cable USB conectamos la placa a nuestro ordenador.

Recuerda que tienes que estar ejecutándose ArduinoBlocks connector.



Una vez haya detectado el puerto USB en donde tenemos conectada la placa seleccionaremos el botón Subir.

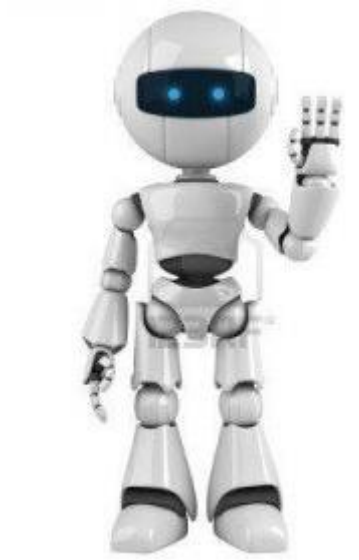


El programa ya se ha subido a la placa de Arduino, ahora podemos ver como la luz que tiene la placa integrada en el Pin 13 se enciende y se apaga a intervalos de 1 segundo.

2.- ¿Qué es la Robótica?

La rama de la Tecnología encargada del diseño, construcción y uso de los Robots se llama ROBÓTICA.

Se puede definir un robot como una máquina capaz de “comprender” e interactuar con su entorno.



Los robot **no tienen porque tener siempre una forma humana**, sino que cualquier dispositivo programable que comprenda el entorno e interactúe con el es un sistema automático o también denominado Robot:

La forma en la que el robot interactúa físicamente con su entorno puede ser a través de diversas formas.

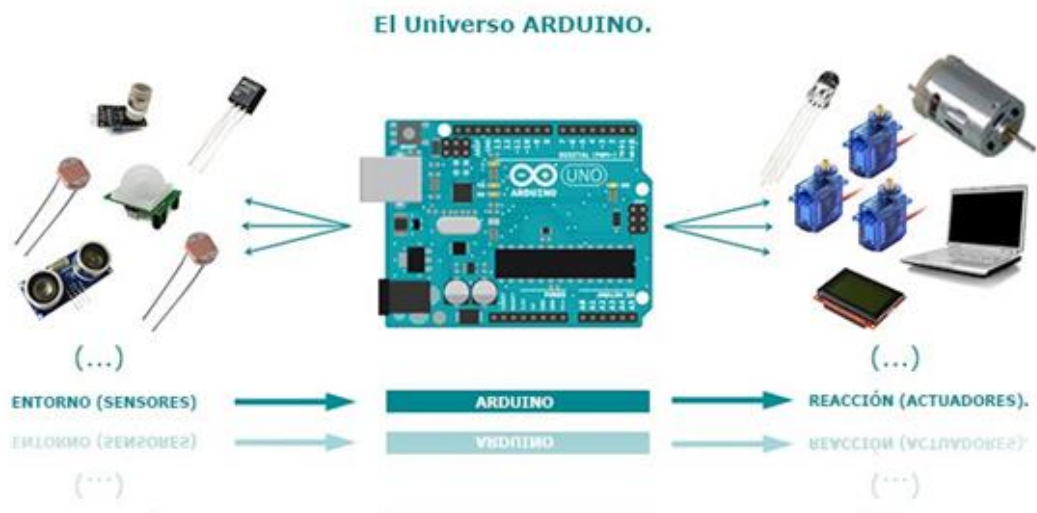
- Movimiento
- Sonido
- Luz/Imágenes

¿Qué disciplinas intervienen en la Robótica?

El gran atractivo que ofrece la Robótica frente a otras disciplinas es que es un terreno en el que convergen mucho otras Tecnologías:

- La Mecánica
- La Electrónica
- La informática

Nosotros nos vamos a centrar en la parte de la ELECTRÓNICA Y LA INFORMÁTICA.



¿Cómo funciona un Robot?

Un sistema Automático o Robótico funciona siguiendo los siguientes principios:

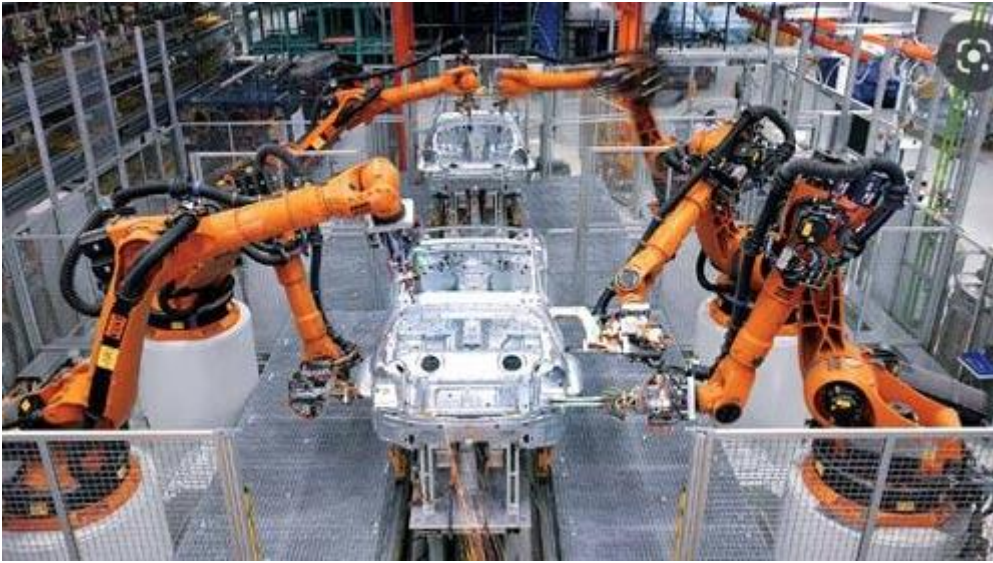
1. Está capacitado para ejecutar una o varias tareas según un **PROGRAMA PREESTABLECIDO**.
2. Es capaz de **captar cambios en las variables del proceso y modificar la secuencia de acciones en función de su Entorno**.
3. Puede ser **REPROGRAMADO** de nuevo.



Objetivo de la Robótica.

La Robótica va unida a la construcción de “artefactos” que trataban de materializar un deseo humano de crear seres a su semejanza y que al mismo tiempo lo descargasen de trabajos tediosos o peligrosos.

Pero el verdadero objetivo de la Robótica es el de **sustituir a las personas en labores peligrosas, pesadas, repetitivas o que requieren una mayor precisión**, difícilmente alcanzable por el ser humano.



Futuro / Peligros Robótica

1. Pérdida de puestos de trabajo.



2. Poder que cedemos a las máquinas.



LA SOLUCIÓN PASA POR LA INCORPORACIÓN Y EL DESARROLLO CONTROLADO DE LA ROBÓTICA A TRAVÉS DE LA COLABORACIÓN CON LOS HUMANOS.

A diferencia de los Robots “tradicionales”, los **COBOTS** lo hacen interactuando con las personas que, técnicamente, se convierten en su compañeros de trabajo.



3.- ¿Qué es Arduino?

Arduino es una PLATAFORMA ELECTRONICA/PROGRAMADA “open-source” o de **código abierto** cuyos principios son contar ELEMENTOS:

1. **FÁCILES DE USAR.**
2. **DE BAJO COSTE.**
3. **LIBRES.**

Es decir, que promete ser una forma sencilla de realizar proyectos de electrónica programada (ROBÓTICA) para cualquier persona.

“FÁCIL Y SENCILLO... PARA TODA LA FAMILIA”

Elementos que componen la Plataforma Arduino

La Plataforma Arduino no está formada SOLO POR LA PLACA ARDUINO, sino que Arduino es UN PUZZLE DE 3 PIEZAS, que lo componen los siguientes Elementos:

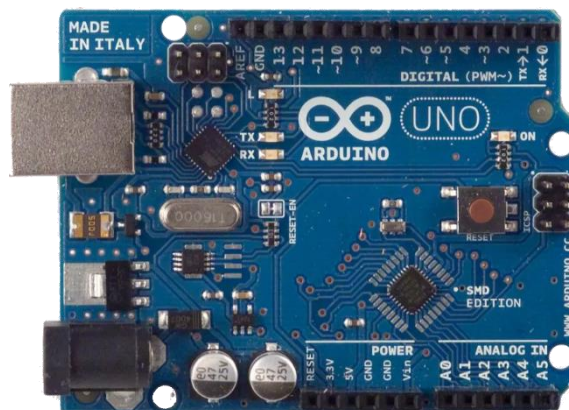
1. **HARDWARE. Componentes FÍSICOS.**
2. **SOFTWARE. Programas.**
3. **COMUNIDAD ALREDEDOR. Lo más importante y el gran éxito de Arduino.**



HARDWARE

1. **PLACA ARDUINO.** Arduino es una PLACA PROGRAMABLE **IDEAL** para iniciarse en pequeños proyectos domésticos de ELECTRÓNICA Y ROBÓTICA.

Esto significa que es capaz de recibir información del entorno (SENSORES) y realizar acciones (ACTUADORES, MOTORES...), según un programa que introducimos con un ordenador, y que puede ejecutar de forma autónoma.



2. **SHIELDS.** Las shields son placas de circuitos modulares que se montan una encima de otra para dar funcionalidad extra a un Arduino. Esta Shields son apilables de forma que nos permite ampliar el hardware/capacidades de Arduino.



3. **COMPONENTES ELECTRÓNICOS.** Muchos componentes electrónicos que podemos conectar a nuestro Arduino.

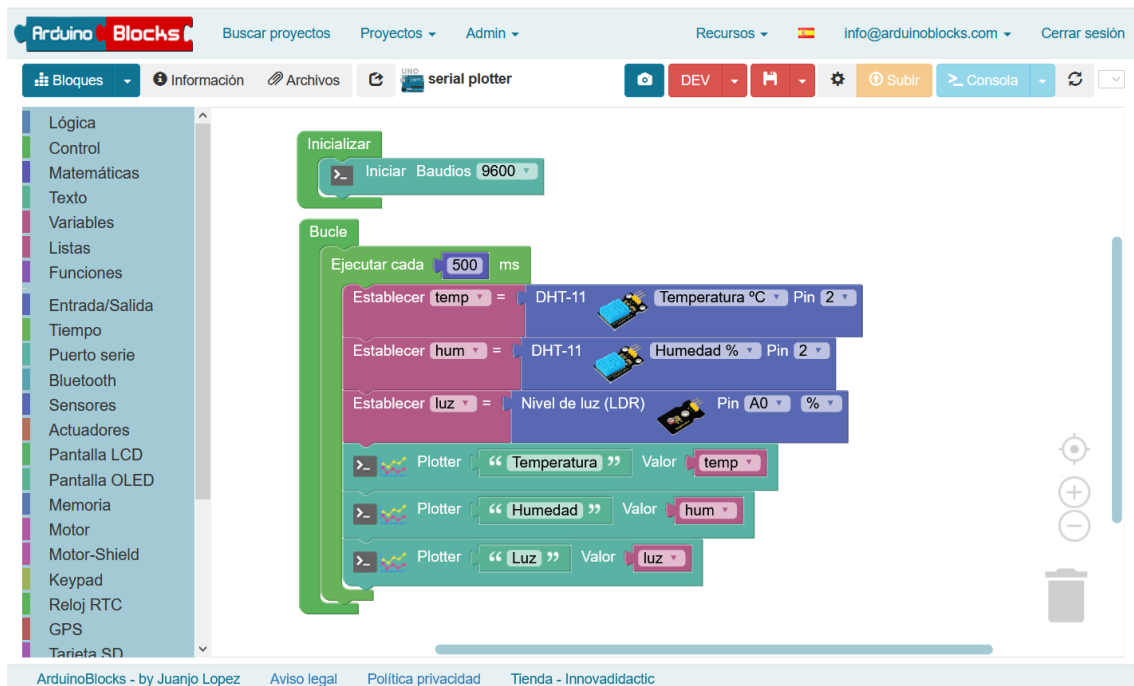


SOFTWARE

1. **ENTORNO DE DESARROLLO OFICIAL (IDE).** Contamos con un IDE para casi todas las plataformas (Windows, Linux, Mac). Un IDE un lugar donde podemos escribir nuestras aplicaciones (POR COMANDOS), descargarlas al Arduino y ejecutarlas o depurarlas desde allí. El entorno de desarrollo es gratuito y descargable desde aquí.

```
MegaDuino_1.3  CheckForExt  CounterPercent  Display  Fonts.h  Lo
#define VERSION "MegaDuino 1.3"
//
// Original firmware written by Andrew Beer, Duncan Edwards, Rafael
//
// MegaDuino Firmware is an adaptation of the MaxDuino firmware by
//
```


2. **OTROS ENTORNOS DE DESARROLLO MÁS SENCILLOS.** Para poder programar por BLOQUES como por ejemplo. ArduinoBlocks o MBlock.



COMUNIDAD

Comunidad alrededor. Mucha gente que trabaja con ella y comparte todo sus conocimientos, experiencias, etc. a través de Internet.



El programa mensual de educ@conTIC en el que hablamos, en directo, sobre el uso de las TIC en las aulas.

28oct
20h

Robótica educativa y programación en el aula. David Cuartielles



No hay duda. La robótica y la programación están de moda, y por eso, hemos dedicado el mes de octubre a este tema en educ@conTIC.

El miércoles 28 de octubre a las 20:00, hora peninsular, hablaremos en directo sobre este tema con David Cuartielles, co-fundador del proyecto Arduino, responsable de educación para Arduino/Genuino y profesor en diseño interactivo en la Facultad de Arte y Comunicación en la Universidad de Malmo, Suecia.

Presenta: Carlos Magro.

@educalab @educ@contic #directTIC

competencias recursos conocimiento
locentes formación actividades profesorado

educ@conTIC
el uso de las TIC en las aulas

Modelos Arduino

Existen muchos modelos de placas Arduino en función de:

- Potencia de procesamiento.
- Cantidad de pines.
- Memoria Interna.

- Sistemas de comunicación.
- Tamaño.
- ...

NOSOTROS VAMOS A UTILIZAR EL MODELO DE PLACA ARDUINO UNO EN SU VERSIÓN ACTUAL.



Placas Oficiales VS No Oficiales

- **OFICIALES.** Son aquellas placas oficiales manufactureras por las compañías oficiales autorizadas en Europa o Estados Unidos (Arduino y Genuino).
- **NO OFICIALES O COMPATIBLES.** Son placas compatibles con Arduino pero no pueden estar registradas bajo el nombre de Arduino. Por supuesto son diseñadas y fabricadas por otras compañías ajenas. Esto no significa que sean “PIRATAS”, puesto que el desarrollo del proyecto Arduino es libre y en abierto, pero lo que no deberían hacer las empresas que las fabrican es venderlas como si fueran originales o con el nombre y logotipo original. Estas frecuentemente utilizan un nombre que integra el sufijo “duino” para identificarlas, como por ejemplo Freeduino.



4.- ¿Qué kit de Arduino Comprar?

Orientaciones: la decisión es vuestra...

En este punto se trata de indicaros algunas orientaciones generales sobre como **adquirir el Kit** que mejor se ajuste a vuestras necesidades para realizar los curso de Robótica y al mejor precio posible.



En ningún momento se tratará de indicaros un producto en concreto te tengáis que comprar, **NO LLEVAMOS COMISIÓN CON NINGUNA MARCA/EMPRESA NI PRODUCTO.**



Así que sabéis que la decisión final es vuestra...

Compatibles VS Original.

La primera decisión que tenemos que tomar es si queremos adquirir Kit Original frente al Compatible.

Desde aquí **recomendamos la opción de algún Kit compatible** que normalmente suele venir con más componentes electrónicos y además nos costará e todo a la mitad que el Original.

Arduino Starter Kit



A la izquierda vemos el kit Arduino Original y a la derecha se muestra un modelo compatible.

¿Dónde Comprar?

Finalmente si decidimos adquirir un Kit compatible son muchas las opciones disponibles y lugares donde puedes adquirirlos.

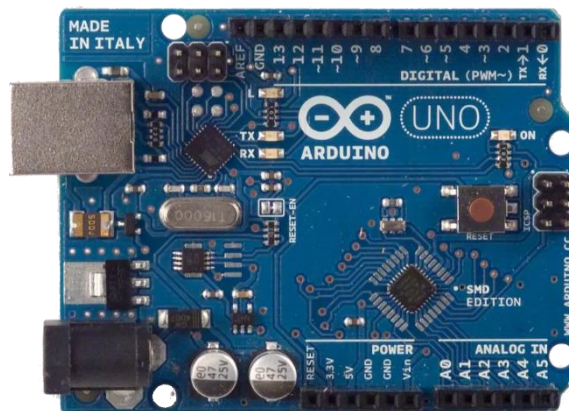
- Tiendas especializadas en robótica. Suelen ser más caros.
- Grandes tiendas genéricas online, AMAZON, PCComponentes... Opción más equilibrada relación calidad/precio.
- Tiendas de origen chino, Aliexpress... Está opción es la más económica pero puede presentar el problema del tiempo hasta que nos llegue el envío y problema con la garantía y devolución.



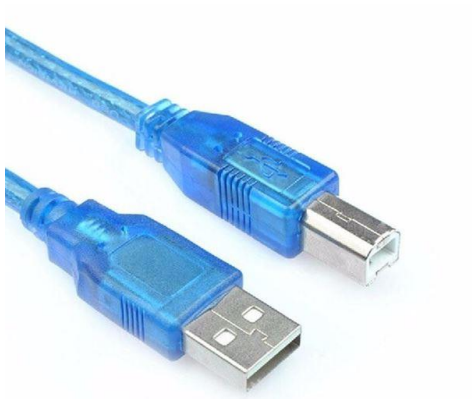
Dispositivos Recomendados

A continuación vamos a realizar una enumeración de todos los componentes básicos, que como mínimo deberían de incluir el Kit Arduino que compremos. Además se indicarán algunas recomendaciones importantes sobre cada uno de ellos.

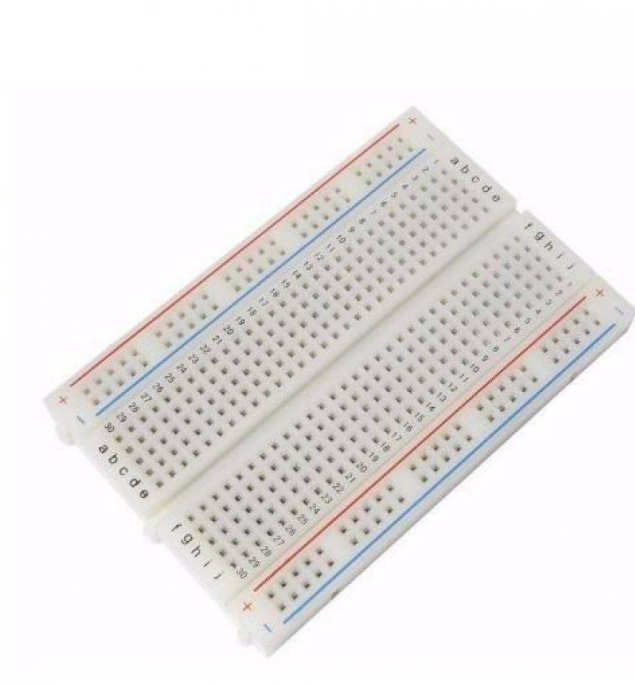
- **Placa Arduino Uno.**



- **Cable USB.**



- **Placa Prototipado**



- **Cables.**



- Leds



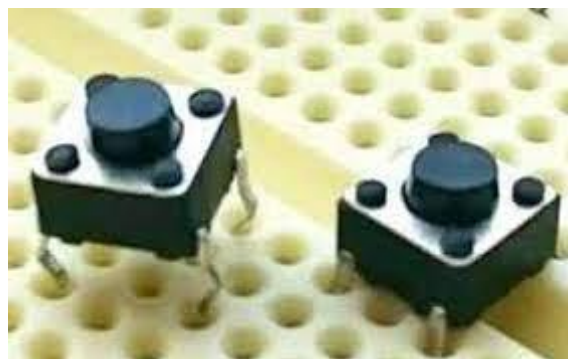
- RGB



- Resistencias



- Pulsadores



- Potenciómetro



- Sensor ultrasónico



- Buzzer Pasivo



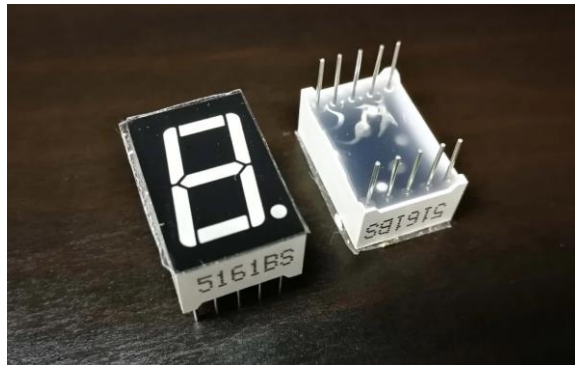
- Servo



- Pantalla LCD



- Display 7 segmentos



- Joystick



- Sensor de temperatura y humedad



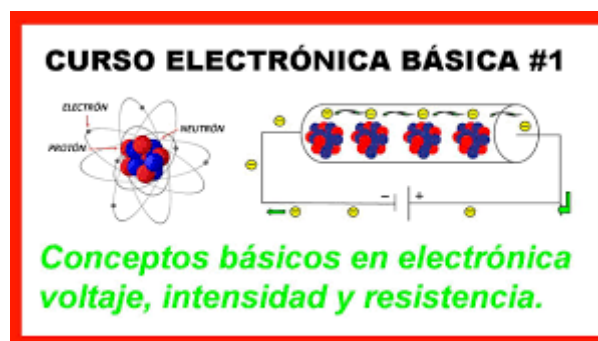
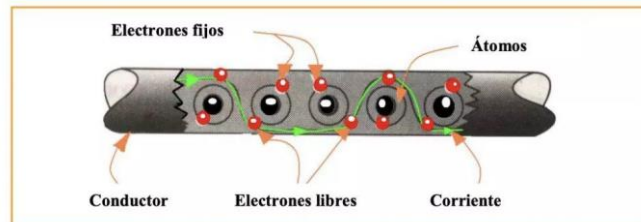
... y a partir de aquí todo lo que venga de más pues mucho mejor.

5.- Introducción Electricidad

¿Qué es la corriente Eléctrica?

La corriente eléctrica es el flujo de carga eléctrica que recorre un material. Se debe al movimiento de las cargas en el interior del mismo.

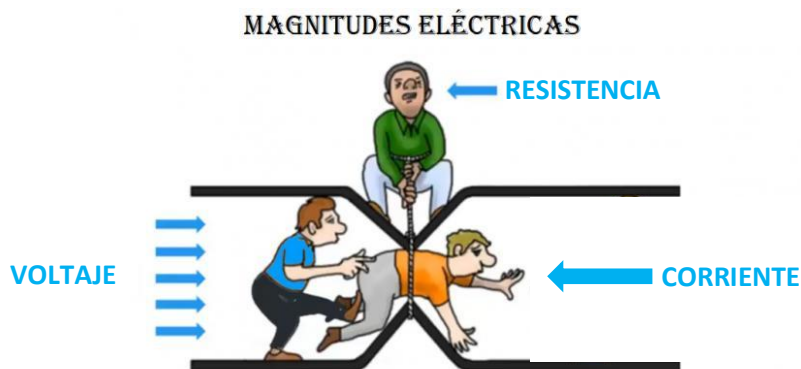
Llamamos corriente eléctrica al movimiento ordenado de los electrones a través de un conductor.



Magnitudes Eléctricas

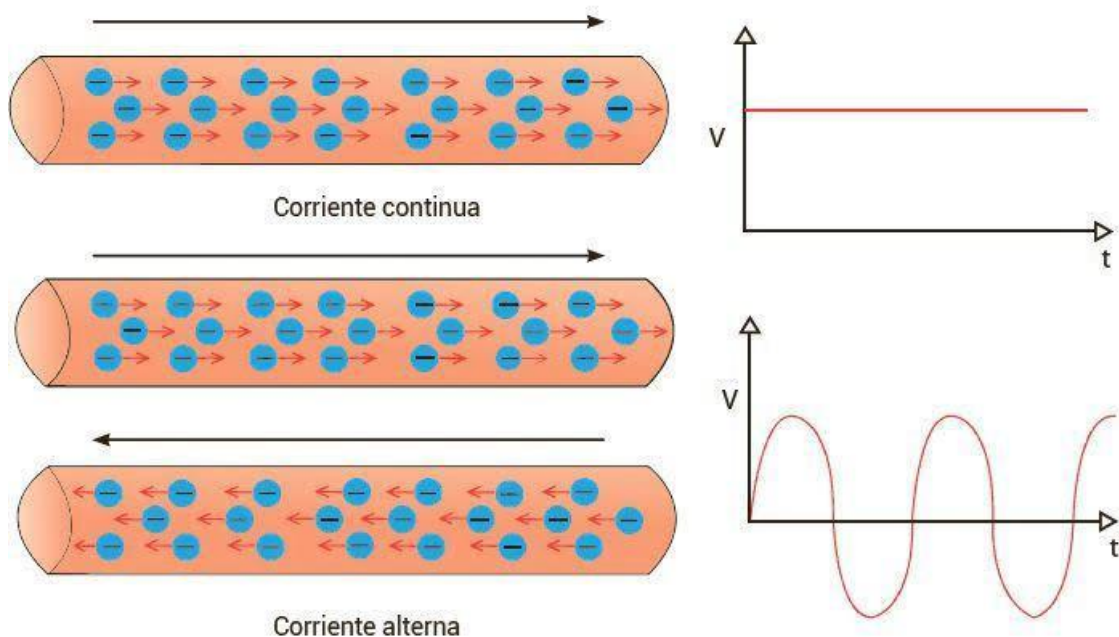
- **VOLTAJE.** La fuerza o presión capaz de obligar a los electrones libres de un conductor a moverse en una determinada dirección.
 - El voltaje se mide en voltios y se representa con la letra V.
 - El voltaje lo aplica las Baterías o Fuentes de alimentación.
 - Tiene que tener dos lados: uno polo positivo (+) y un polo negativo (- o GND)
- **INTENSIDAD.** Es la capacidad de electricidad que pasa o circulan por un conductor en una determinada unidad de tiempo.
 - La intensidad se mide en Amperios y se representa con la letra I.
- **RESISTENCIA.** Es la oposición que presentan los materiales al paso de la corriente eléctrica.

RELACION ENTRE LAS MAGNITUDES ELÉCTRICAS



Corriente Continua Vs Alterna

- **Corriente Continua (CC).** Las cargas eléctricas se desplazan siempre en el mismo sentido y el Voltaje mantiene la misma polaridad. Por ejemplo los voltajes suministrados por baterías y pilas son continuos.
- **Corriente Alterna (CA).** Las cargas eléctricas se mueven primero en un sentido y luego en otro y el voltaje cambia alternativamente de polaridad. Este cambio es muy rápido (se produce 50 cambios de sentido en un segundo). Es el tipo de corriente que llega a nuestras viviendas.



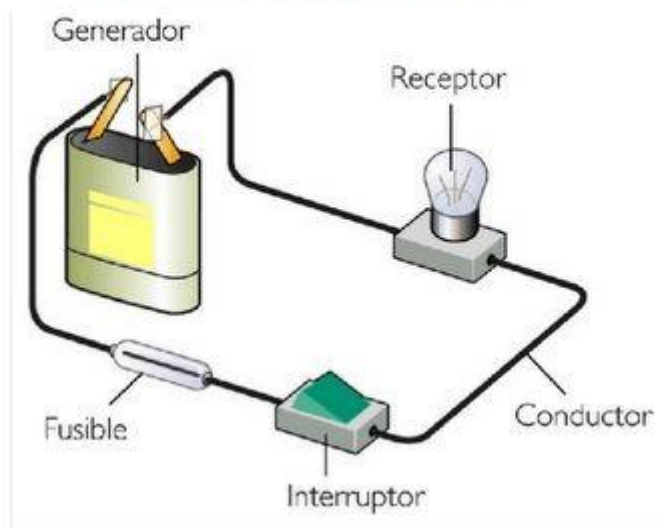
Circuito Eléctrico

Cuando los electrones realizan un recorrido cerrado decimos entonces que tenemos un CIRCUITO ELÉCTRICO. Los componentes de un circuito eléctrico son los siguientes:

1. **GENERADORES (GENERADORES ELÉCTICOS)**
 - a. Pilas
 - b. Baterías
 - c. Fuentes de Alimentación
2. **CONECTORES**
 - a. Cables
3. **RECEPTORES**
 - a. Lámparas
 - b. Motores
 - c. Altavoces
4. **ELEMENTOS DE CONTROL**
 - a. Interruptores
 - b. Pulsadores
5. **PROTECTORES**
 - a. Fusibles

EJEMPLO DE CIRCUITO ELÉCTRICO:

Partes de un Circuito Eléctrico

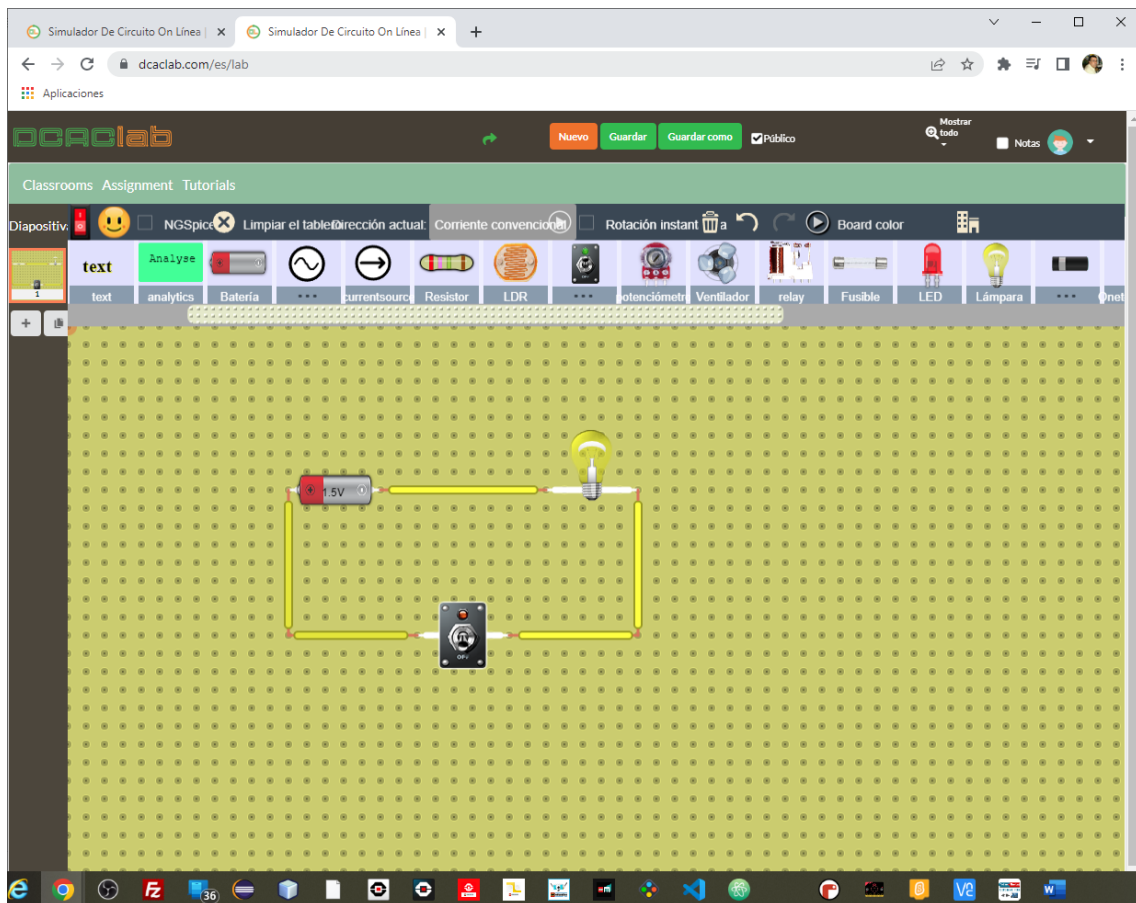


PROGRAMA PARA LA SIMULACION DE CIRCUITOS ELÉCTRICOS.

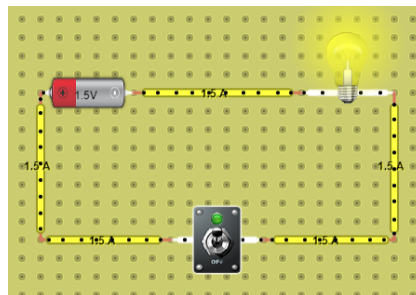
Para la simulación de circuitos podemos ir a la siguiente página web: <https://dcaclab.com/es/lab>

Lo primero que tenemos que hacer es registrarnos.

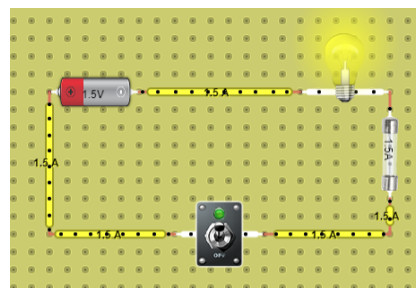
La imagen muestra una captura de pantalla de un navegador web que muestra la página de inicio de DCACLab. El título principal es "¡ENSEÑAR ELECTRONICA E IMPLEMENTAR EXPERIMENTOS NUNCA FUE TAN FACIL!". El texto describe el simulador como una herramienta para estudiantes y profesores que permite construir circuitos de CC/CA con componentes como baterías, resistencias y cables. Incluye un botón de "START SIMULATION!" destacado con un recuadro rojo. En la parte inferior, se menciona que miles de estudiantes y profesores alrededor del mundo utilizan DCACLab. A la derecha del texto principal hay una imagen de un aula con estudiantes trabajando en computadoras y un profesor presentando en una pizarra.



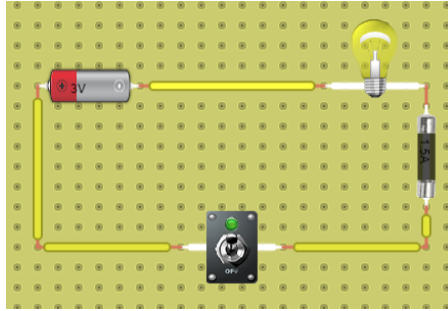
Cuando activamos el interruptor.



Ahora le vamos a agregar un fusible.

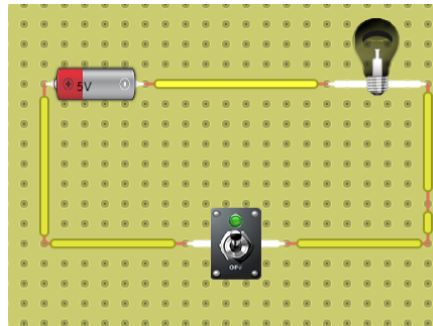


Vamos a aumentar la pila a 3 Voltios.



Podrás observar que el fusible se ha fundido.

Vamos a quitar el fusible y la pila la aumentamos a 5 voltios.



Se ha fundido la bombilla.

Ley de Ohm

La intensidad (I) de la corriente eléctrica que circula por un circuito es directamente proporcional al Voltaje aplicado (V) e inversamente proporcional a la resistencia del mismo. Para que los resultados de esta formula sean coherentes todas las unidades deben estar en las magnitudes básicas: Amperios, Voltios y Ohmios.

$$I = \frac{V}{R}$$

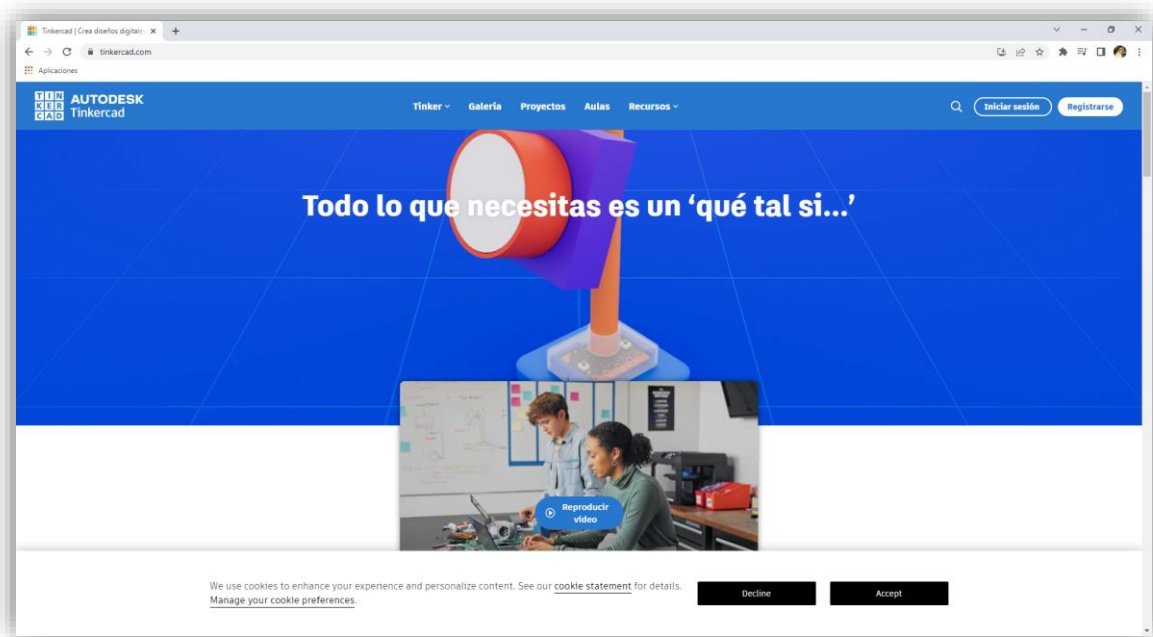
Primer circuito Eléctrico con Arduino

Encender un Led utilizando la salida de 5V del Arduino.

- Placa de Prototipado
- Led
- Cable
- Resistencia

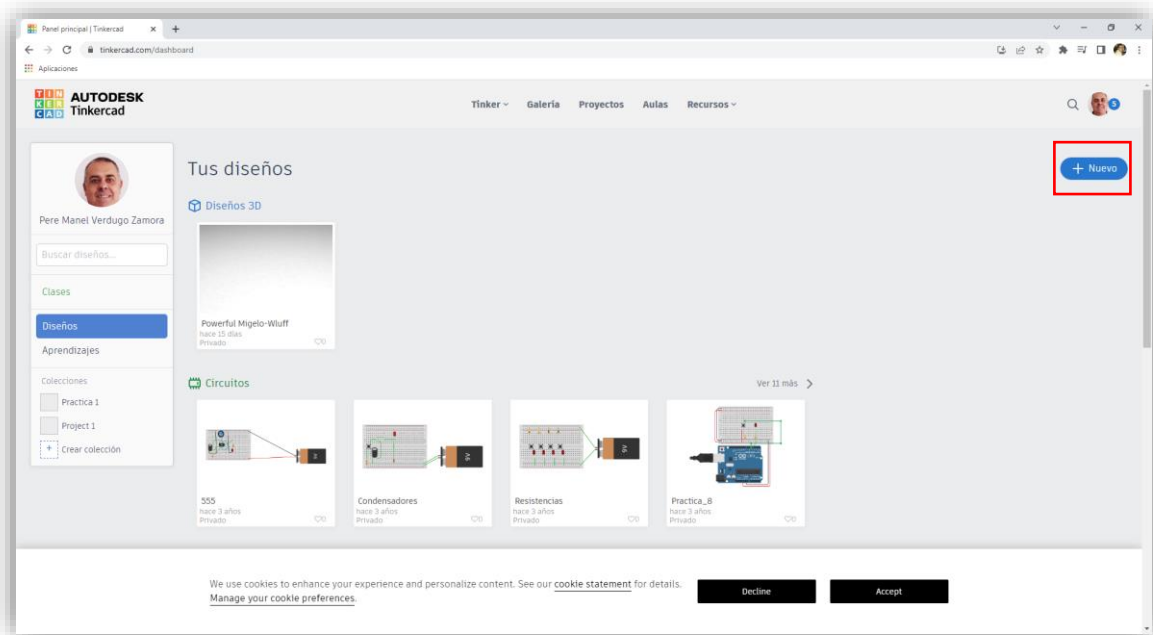
Placa de Prototipado:

Para diseñar el proyecto vamos a entrar en el siguiente enlace: <https://www.tinkercad.com/>



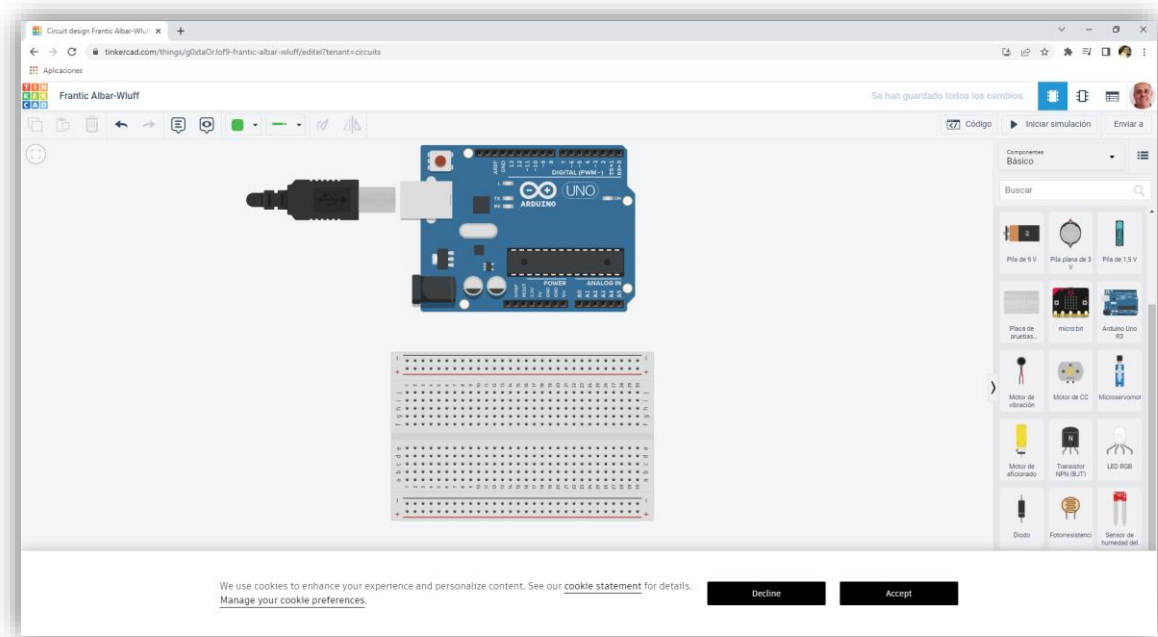
Lo primero que tenemos que hacer es registrarse.

Yo he accedido por mi cuenta de Google.

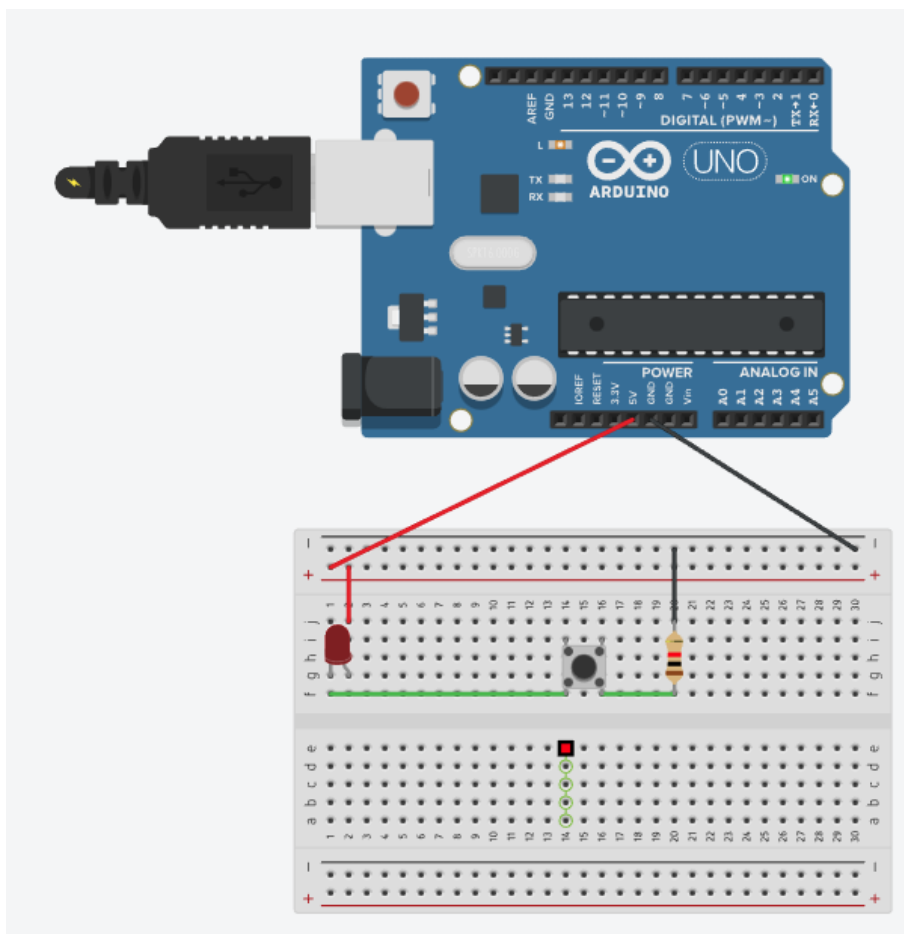


Vamos a crear un nuevo proyecto.

Seleccionaremos Circuito.



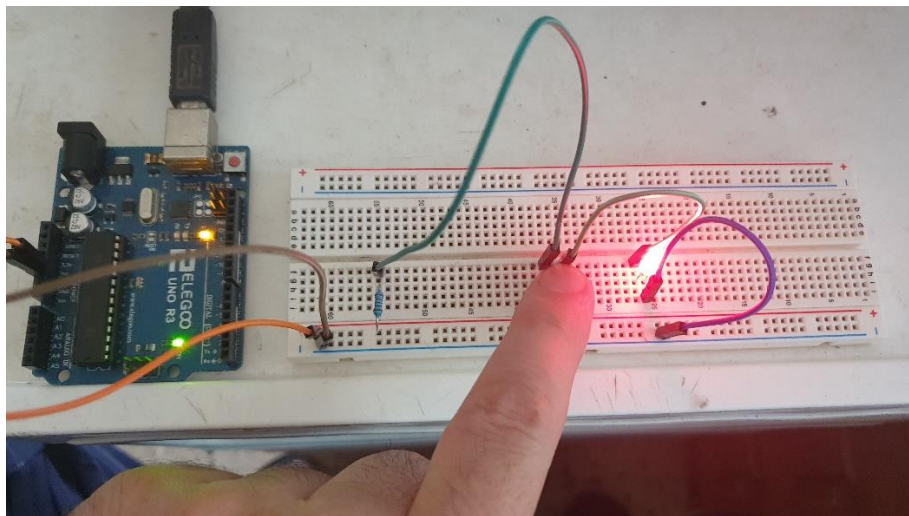
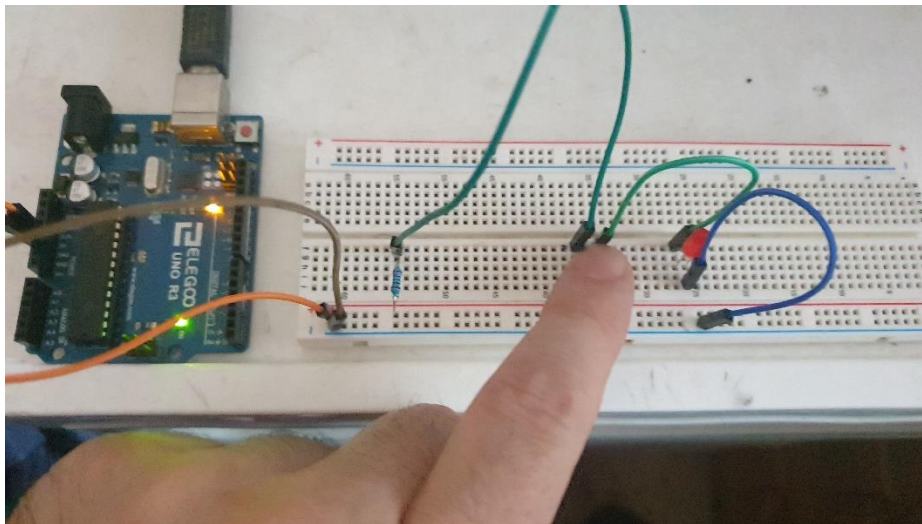
Vamos a realizar los siguientes pasos:



La resistencia tiene que ser de $220\ \Omega$.

Resistencia	
Nombre	1
Resistencia	220 Ω

Vamos a llevarlo a la práctica:



6.- Placa Arduino Uno

PINES

Arduino dispone de un montón de elementos de conexión a través de una serie de PINES HEMBRA sobre los que tendremos que pinchar MACHO par realizar la conexión. Estos Pines tienen la siguiente funcionalidad:



- **Entradas y salidas digitales (2,3):** Están situadas en la parte de arriba de la placa, van del 0 hasta el 13, este último pin lleva un LED INTERNO INCLUIDO. La señal digital puede estar o encendida o apagada (LOW o HIGH). Los pines cero y uno se pueden utilizar para cargar el programa en la placa. Por ejemplo, se utilizan para parpadear un LED o; como entrada, un pulsador.
- **Salidas analógicas (2,3):** Son los pines 11, 10, 9, 5 y 3, si os fijáis tiene una raya curva al lado, se denomina salidas PWM (Pulser Width Modulation) que realmente son salidas digitales que imitan salidas analógicas, modificando la separación entre los diferentes pulsos de la señal. La señal PWM puede dar diversos valores hasta 255, se utiliza, por ejemplo para variar la intensidad de un LED o hacer funcionar un servo. Hay que decir que estos pines funcionan como salidas o entradas digitales o como salidas analógicas.
- **Entradas analógicas (12):** Son los pines A0, A1, A2, A3, A4 y A5 (analog in). Se utiliza para que entre una señal de un sensor analógico, tipo un potenciómetro o un sensor de temperatura, que dan un valor variable. También se pueden utilizar como pines digitales.
- **Pines de alimentación (11):**
 - GND: Son los pines a tierra de la placa, el negativo.
 - 5v: Por este pin suministra 5v.
 - 3,3v: Por este pin se suministra 3,3v.
 - Vin: Voltaje de entrada, por este pin también se puede alimentar la placa.
 - RESET: Por este pin se puede reiniciar la placa.

Conectando un cable en el Pin RESET y el GND se realiza un reset.

Otros Componentes de la Placa



1.- Botón de Reset. Sirve para inicializar nuevamente el programa cargado en el microcontrolador de la placa.

- EL ÚLTIMO PROGRAMA CARGADO EN EL ARDUINO SIEMPRE PERMANECE EN EL MISMO AUNQUE EL ARDUINO SE APAGUE.** Cuando Arduino vuelva a recibir corriente este volverá a cargar y ejecutar dicho programa.
- Cuando deje de responder el Arduino uno es el botón de encendido o apagado para que vuelva a restablecerse.

4.- Puerto USB. Utilizado tanto para:

- Conectar con un ordenador y transferir o cargar los programas al microcontrolador.
- Dar electricidad al Arduino.

5.- Chip de interface USB, es el encargado de controlar la comunicación con el puerto USB. Los últimos Arduino compatibles ya vienen con el Chip de interface USB original de Arduino pero aún hay algunos que vienen con un compatible (CH340), el cual para su funcionamiento necesita la instalación de un driver tal y como se indica en la siguiente página Web:



Arduino Uno compatible con Chip interface USB CH340 Compatible con el original

6.- Reloj oscilador. Es el elemento que hace que el Arduino vaya ejecutando las instrucciones. Es el encargado de marcar el ritmo al cual se debe ejecutar cada instrucción del programa.

7.- Led de encendido. Es un pequeño LED que se ilumina cuando la placa esta correctamente alimentada.

8.- Microcontrolador. Este es el cerebro de cualquier placa Arduino. Es el procesador que encarga de ejecutar las instrucciones de los programas.

9.- Regulador de tensión. Este sirve para controlar la cantidad de electricidad que se envía a los pines, con lo que asegura que no se estropee lo que conectamos en dichos pines.

10.- Puerto de corriente continua. Este puerto es el que se usa para darle electricidad a la placa si no se usa alimentación USB.

13.- LED Interno. Conectado directamente al PIN 13.

TX, RX. Se encienden cuando Arduino envía o recibe información del exterior a través del puerto Serie. Por ejemplo cuando estamos cargando nuestro programa en Arduino.

Son las luces de parpadeo cuando estamos cargando un programa.

Características Técnicas

Microcontrolador Atmega328

Voltaje de operación 5V

Pines de entrada – salida digital. 14 (6 pueden usarse como salida de PWM)

Pines de entrada analógica. 6

Memoria Flash 32 KB (0,5 KB ocupador por el bootloader) SRAM 2 KB EEPROM 1 KB

Frecuencia de reloj 16 MHz

7.- Encender un Led. SALIDAS DIGITALES

CONCEPTOS

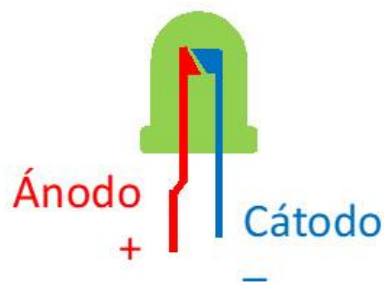
¿Qué es un Led?

Un led no es una bombilla, es un **diodo emisor de luz**, es decir, **un tipo particular de diodo que emite** luz al ser atravesado por una corriente eléctrica.

Los diodos, tienen polaridad, es decir, **solo dejar pasar corriente en un sentido**. Por tanto, tenemos que conectar correctamente este dispositivo.

¿Cómo Funciona un Led?

La pata larga del Led debemos de conectarla al lado positivo (ánodo) de la fuente de voltaje (+5v) mientras que la corta al lado negativo o GND (cátodo).



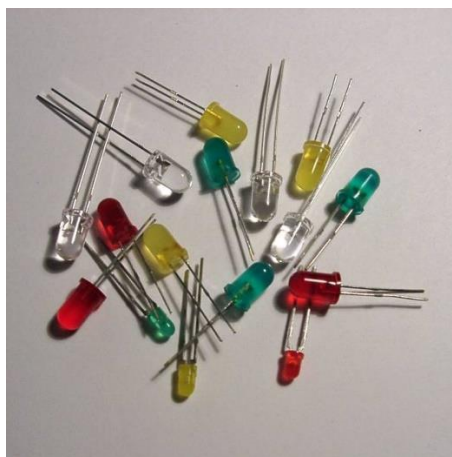
A veces es difícil distinguir la pata más larga de la corta pero en este caso también saber porque la pata larga tiene una pequeña curvatura.

Si conectamos el led al revés, no circulará la corriente. Cuando se conecta un led al revés, éste se comporta como si fuera un interruptor abierto.

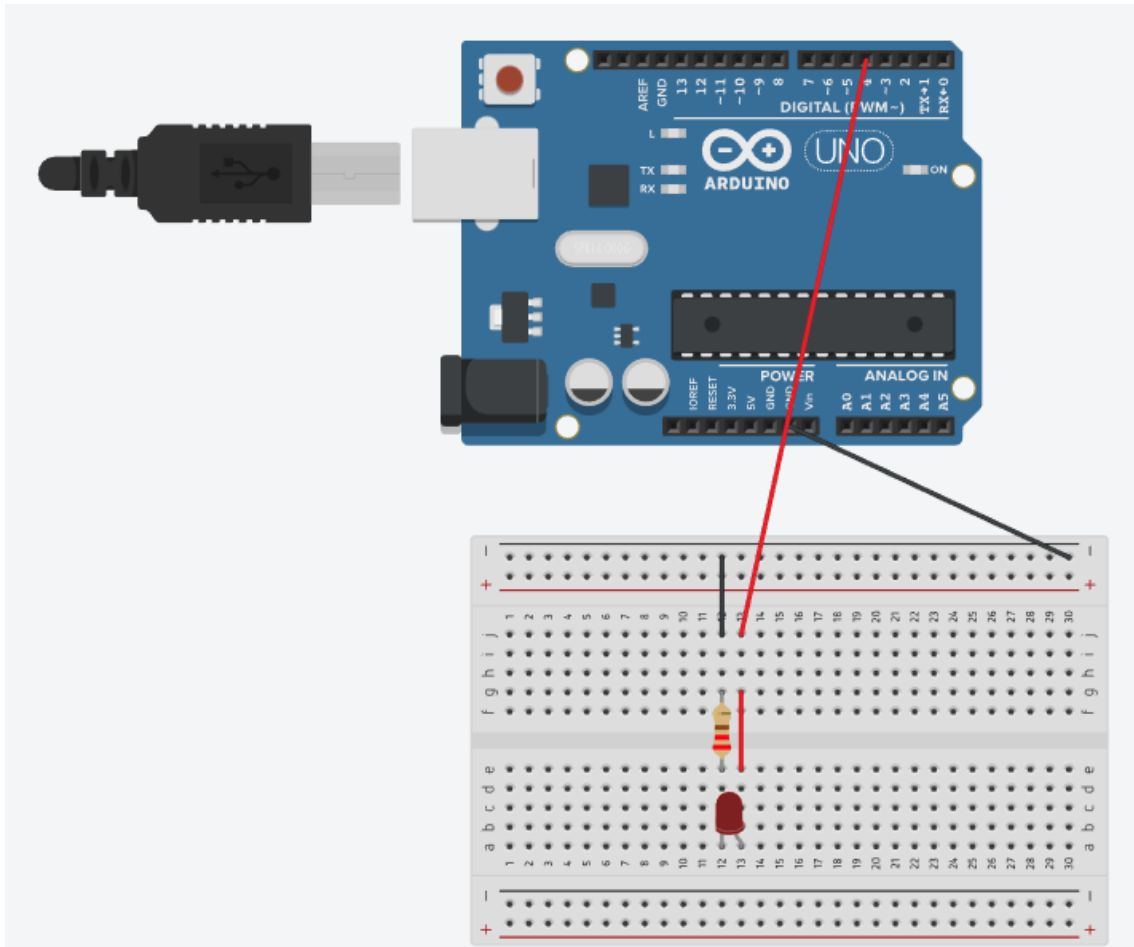
MUY IMPORTANTE. No podemos conectar directamente un led a la salida del Arduino porque este se funde, al tener demasiada intensidad. Para reducir esta intensidad debemos de conectar una resistencia para que este no se funda. El valor de la resistencia depende de cada tipo de led (depende del color su voltaje de funcionamiento e intensidad son diferentes), pero en principio podremos utilizar resistencias de **220 Ohmios** para nuestras prácticas.

Tipos de Leds

Existen diferentes tipos de leds por tamaño como por colores. En los Kits de Arduino suelen venir leds de 5 mm y de diferentes colores.

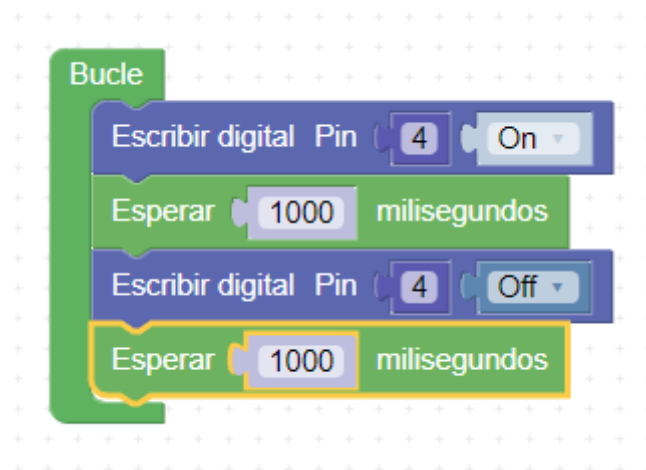


CIRCUITO ELECTRÓNICO



PROGRAMACIÓN

En bloques:



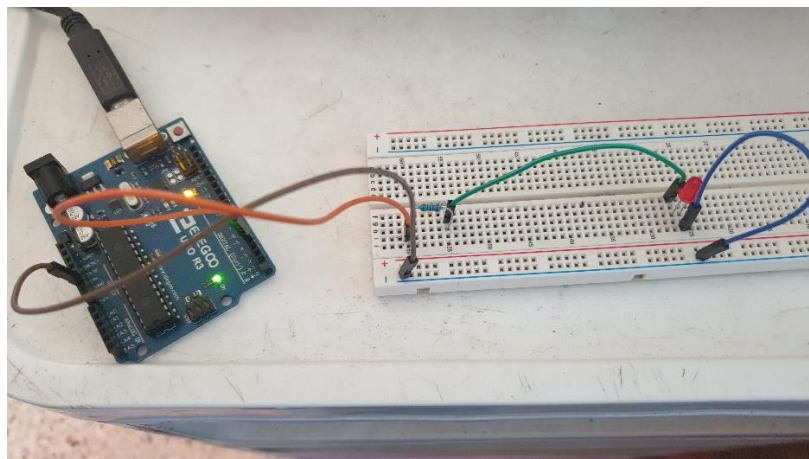
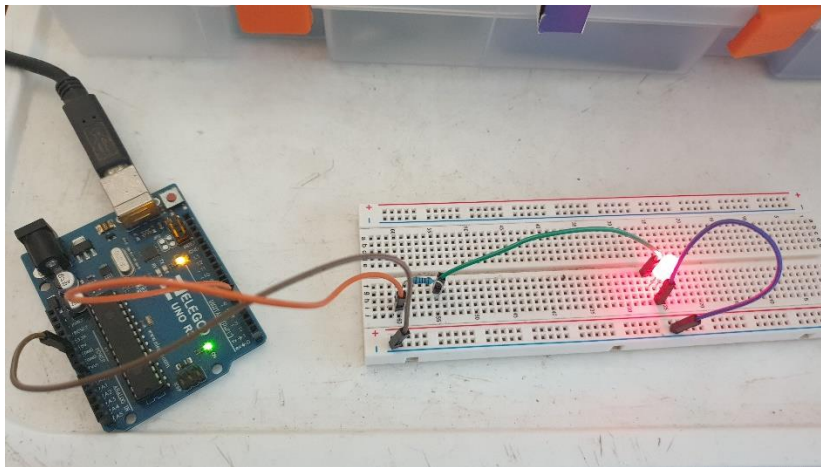
Con código:

```
1 void fnc_dynamic_digitalWrite(int _pin, int _e){
2     pinMode(_pin,OUTPUT);
3     digitalWrite(_pin,_e);
4 }
5
6 void setup()
```



```
7 {  
8  
9  
10 }  
11  
12  
13 void loop()  
14 {  
15  
16   fnc_dynamic_digitalWrite(4, HIGH);  
17   delay(1000);  
18   fnc_dynamic_digitalWrite(4, LOW);  
19   delay(1000);  
20  
21 }
```

PRUEBAS



El Led se enciende y se apaga con un intervalo de 1 segundo.

8.- Led en Serie VS Paralelo

CONCEPTOS

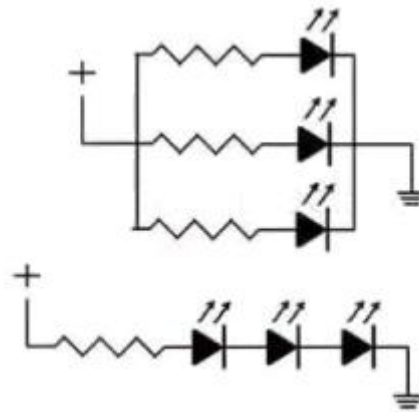
Conectar varios Leds a una Salida Digital

El objetivo de esta práctica es poder conectar varios leds a una única y misma salida digital.

Opciones: Series VS Paralelo

Tenemos dos opciones para poder conectar varios Leds a una única salida Digital.

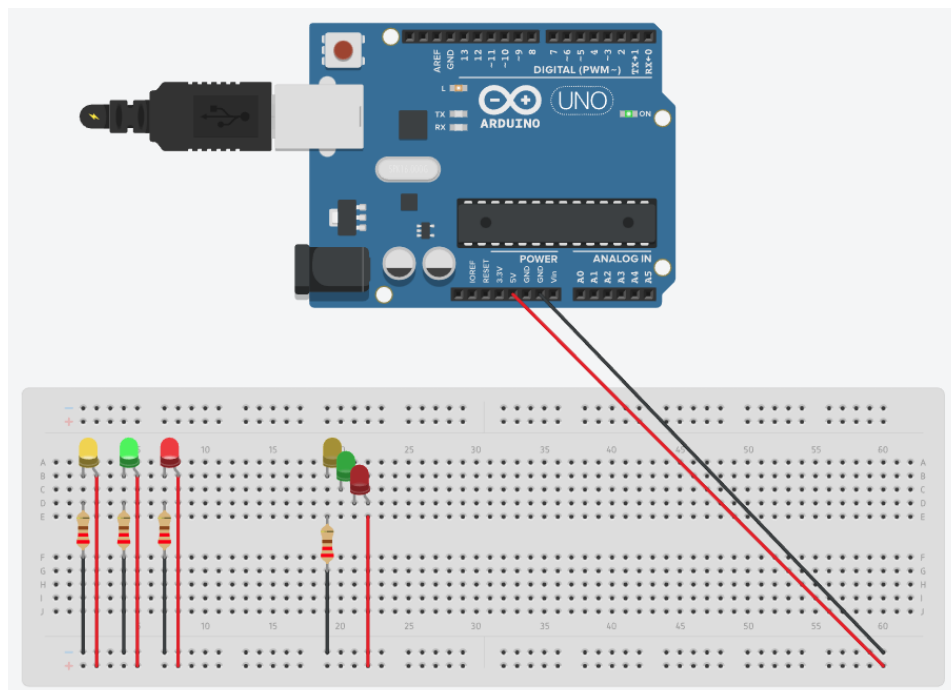
CONEXIÓN EN PARALELO. Tal y como vemos en la parte superior del dibujo, la conexión en paralelo hace que todo el voltaje llegue a los 3 leds a la vez. Para esta configuración del circuito necesitamos utilizar una resistencia para cada led que situemos en el circuito.



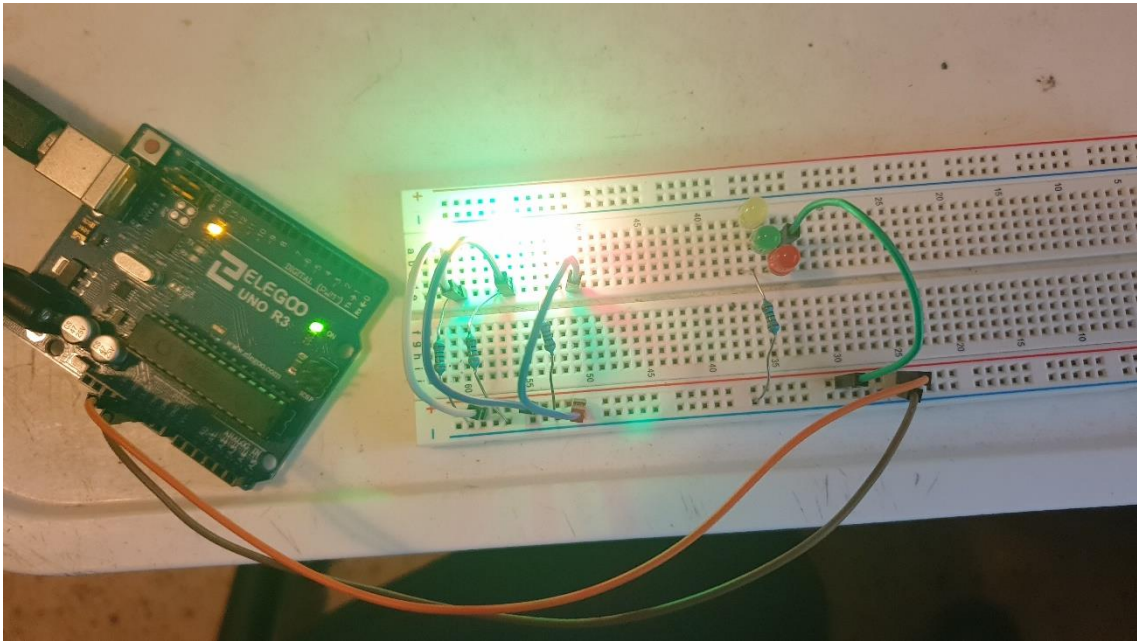
CONEXIONES EN SERIE. Tal y como vemos en la parte inferior del dibujo, la conexión en serie hace que todo el voltaje se reparta entre cada uno de los leds. Para esta configuración del circuito solo necesitamos utilizar una resistencia para todos los Leds.

¿QUÉ CONFIGURACIÓN CREES QUE ES LA MEJOR?

CIRCUITO ELECTRÓNICO



PRUEBAS



La conexión en paralelo de la parte izquierda funciona correctamente, en cambio la conexión en serie, al tener que dividir el voltaje y la resistencia entre los 3 Leds no llega lo suficiente para poderlas encender.

Podemos decir que la conexión en paralelo es más eficiente.

9.- Led intensidades. SALIDAS ANALÓGICAS PWM

CONCEPTOS

¿Salida Analógica?

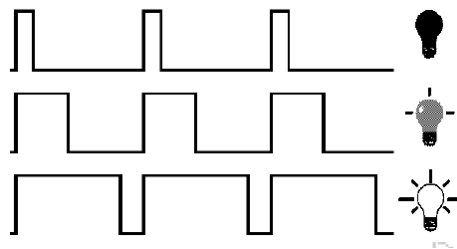
Hasta ahora hemos visto las salidas digitales que solo pueden tomar 2 valores 0 o 5V. Sin embargo, en ocasiones no será suficiente con una señal digital (ON/OFF), si no que necesitaremos proporcionar un valor analógico de tensión, por ejemplo, para regular la intensidad de iluminación de un LED, o variar la velocidad de un motor.

Pero hemos dicho que Arduino no tiene salidas analógicas...

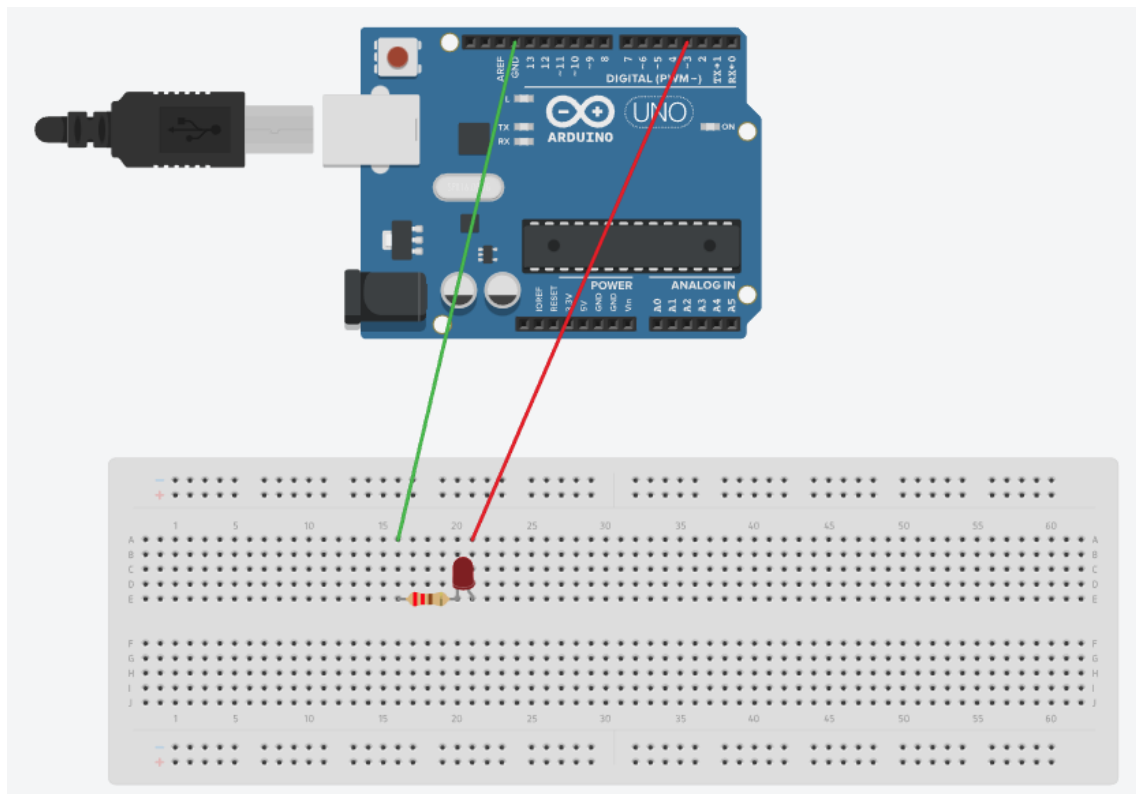
¿Qué es PWM?

Lo primero que tenemos que entender es que la mayoría de automatismos (y Arduino no es una excepción) **no son capaces de proporcionar una auténtica salida analógica**.

Para salvar esta limitación y simular una salida analógica la mayoría de los automatismos emplean un **“truco”, que consiste en activar una salida digital durante un tiempo y mantenerla apagada durante el resto**. El promedio de la tensión de salida, a lo largo del tiempo, será igual al valor analógico deseado. Este “truco” se hace con la técnica que se **denomina modulación de ancho de pulso (PWM)**.

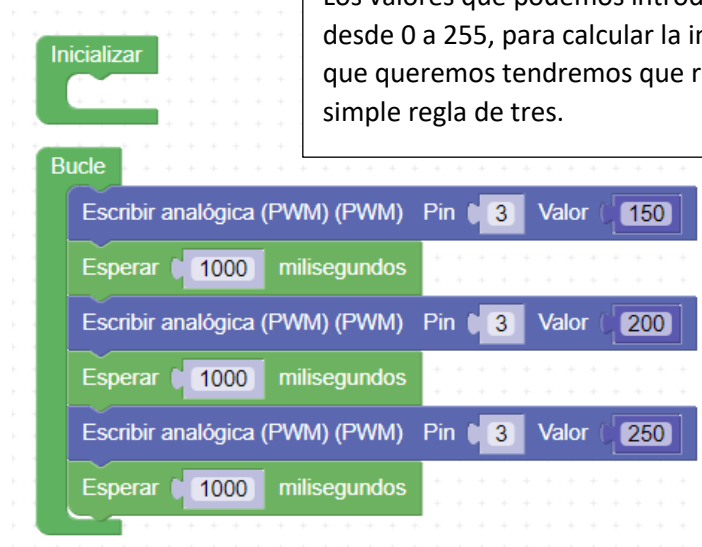


CIRCUITO ELECTRÓNICO



PROGRAMACIÓN

Programación por bloques

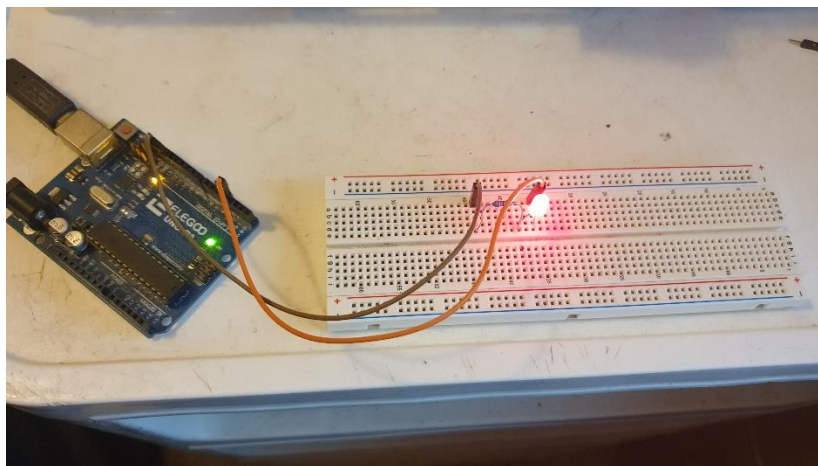


Los valores que podemos introducir son desde 0 a 255, para calcular la intensidad que queremos tendremos que realizar una simple regla de tres.

Programación por código:

```
1 void fnc_dynamic_analogWrite(int _pin, int _e){
2   pinMode(_pin,OUTPUT);
3   analogWrite(_pin,_e);
4 }
5
6 void setup()
7 {
8
9
10 }
11
12
13 void loop()
14 {
15
16   fnc_dynamic_analogWrite(3, 150);
17   delay(1000);
18   fnc_dynamic_analogWrite(3, 200);
19   delay(1000);
20   fnc_dynamic_analogWrite(3, 250);
21   delay(1000);
22
23 }
```

PRUEBAS



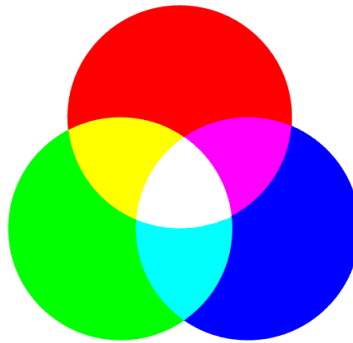
10.- Led RGB. SALIDAS DIGITALES Y ANALÓGICAS

CONCEPTOS

¿Qué es un Led RGB?

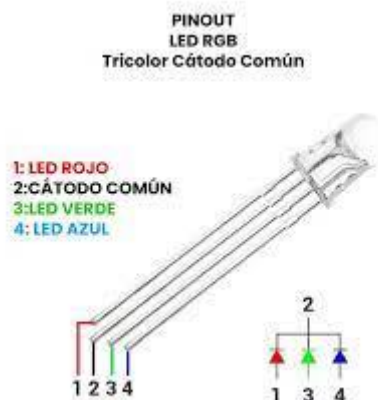
Un LED RGB (Red-Green-Blue) es en realidad la **unión de tres LEDs de los colores básicos**, en un encapsulado común, compartiendo el Ground (cátodo es otro nombre más para el negativo).

En función de la tensión que programamos a cada pin podemos conseguir la mezcla de color que deseemos con relativa sencillez.



¿Cómo funciona en Led RGB?

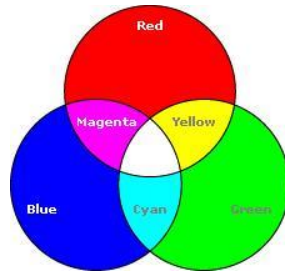
Los Leds RGB que vamos a utilizar tienen la configuración en **CÁTODO COMÚN**. Para alimentar este módulo debemos conectar el cátodo a cualquier pin negativo (también llamado masa o GND) de nuestra placa y los pines de los 3 sub-pixeles a otras salidas digitales, cerrando así los circuitos.



¿Cómo obtener diferentes colores?

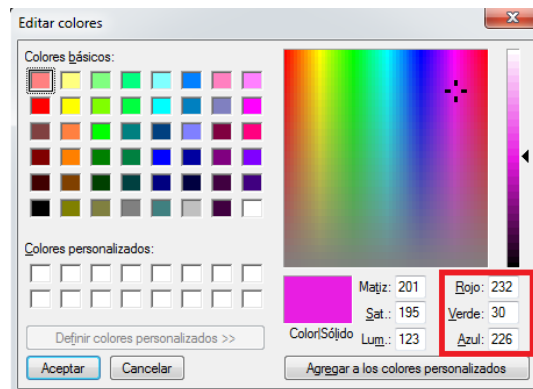
COMBINACIONES DE COLORES ON/OFF: SALIDAS DIGITALES PARA PINES RGB.

Si conectamos los pines R, G, B a salidas digitales, solo puedes encender o pagar cada uno de los canales R, G y/o B. Sin embargo, como aprendimos en el capítulo anterior, podemos variar la intensidad de un LED, en este caso podríamos variar la intensidad de cada canal de color. Aportando mayor o menor intensidad en cada canal por separado.



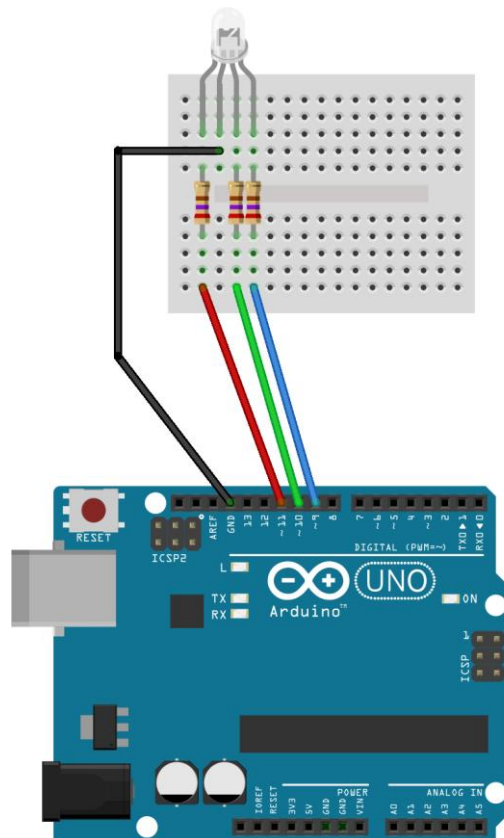
COMBINACIONES DE COLORES POR INTENSIDAD: SALIDAS ANALÓGICAS PARA PINES RGB.

Si utilizamos una señal PWM para controlar la intensidad de cada color, podremos elegir entre 0 y 255 intensidades para el Rojo, entre 0 y 255 para el Verde y entre 0 y 255 para el Azul.



Tipos de Leds RGB

1.- **SIN ENCAPSULAR.** Sin resistencias incorporadas. Tenemos que añadir una resistencia para cada uno de los pines de colores que conectaremos en la salida Digital del Arduino.

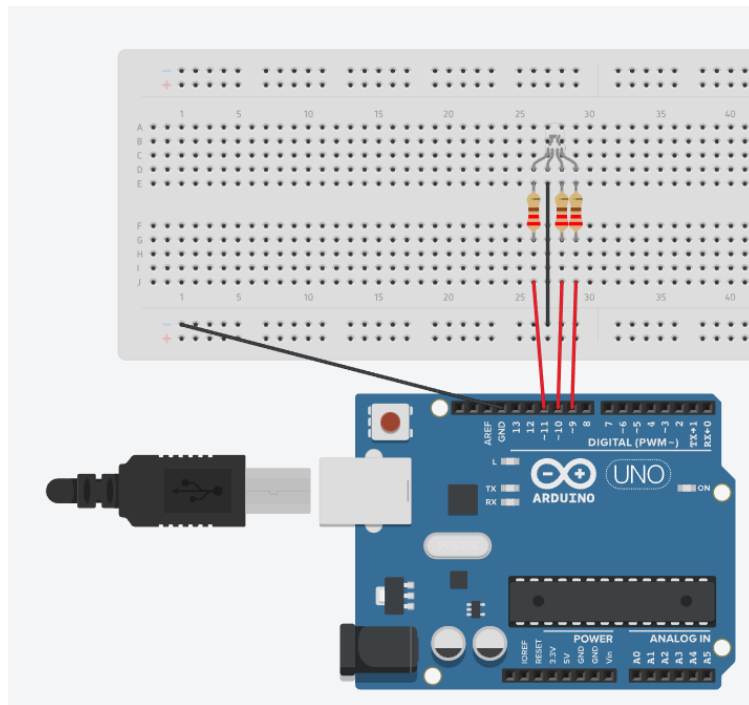


2.- **ENCAPSULADO.** Con resistencias incorporadas. No es necesario añadir ninguna resistencia ya que el encapsulado incorpora las resistencias para cada uno de los 3 pines de colores que conectaremos a una salida digital del Arduino.



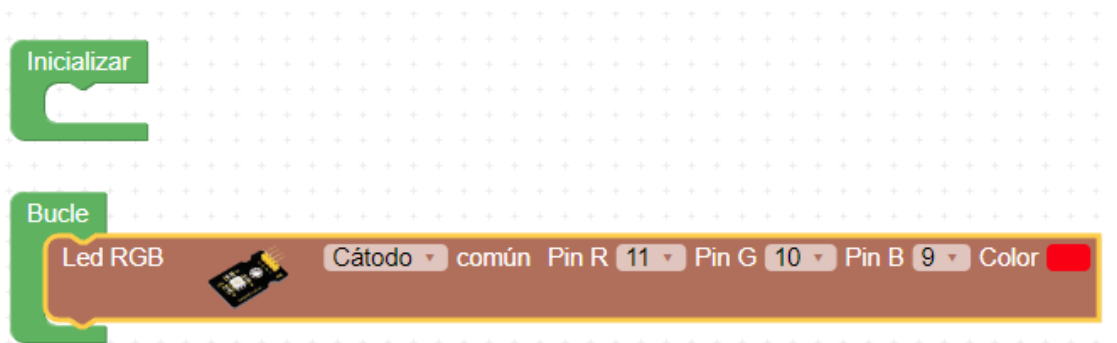
Banggood

CIRCUITO ELECTRÓNICO



PROGRAMACIÓN

Programación por bloques:



Programación por código:

```

1 void setup()
2 {
3     pinMode(11, OUTPUT);
4     pinMode(10, OUTPUT);
5     pinMode(9, OUTPUT);
6
7 }
8
9
10 void loop()
11 {
12
13     analogWrite(11,255);
14     analogWrite(10,0);
15     analogWrite(9,0);
16
17 }

```

Otra forma de programación en bloques:



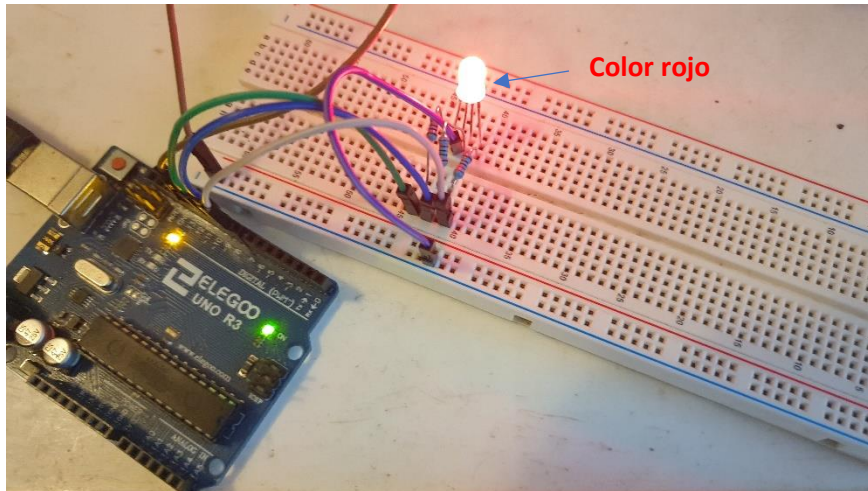
Programación por código:

```

1 void fnc_dynamic_analogWrite(int _pin, int _e){
2     pinMode(_pin,OUTPUT);
3     analogWrite(_pin,_e);
4 }
5
6 void setup()
7 {
8
9
10 }
11
12                                     255
13 void loop()
14 {
15
16     fnc_dynamic_analogWrite(11, 255);
17     fnc_dynamic_analogWrite(10, 0);
18     fnc_dynamic_analogWrite(9, 0);
19
20 }

```


PRUEBAS



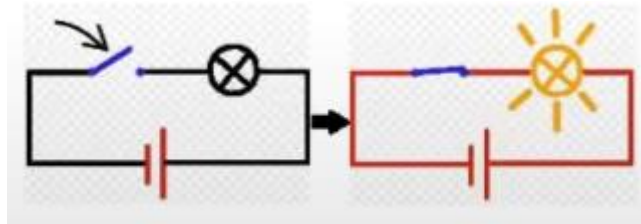
11.- Pulsador interruptor. ENTRADAS DIGITALES

CONCEPTOS

¿Qué es un Pulsador?

Un pulsador es un dispositivo electrónico que nos permite cortar y abrir el paso de corriente en un determinado punto de un circuito.

En realidad un pulsador se comporta como un interruptor salvo que el pulsador es necesario que lo mantengamos todo el tiempo oprimido para que una los dos cables a los que lo conectamos.



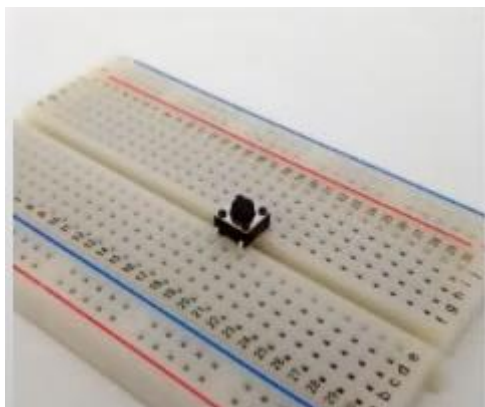
¿Cómo Funciona un Pulsador?

Un pulsador es un dispositivo electrónico que permanece abierto, es decir no une los dos cables que le vamos a conectar mientras este no sea pulsado, en el momento en que pulsamos el pulsador lo que hará internamente es unir los dos cables que estamos conectando.

Los pulsadores que vamos a utilizar suelen tener 4 pines para poder pincharlos en nuestra placa de prototipado pero tal y como hemos dicho solo necesitamos usar dos. Los pines están conectados dos a dos, a su vez las otras patas están separadas a forma de pulsador.



Como en muchos pulsadores no se puede ver que pares de pines están conectados y cuales hacen de pulsador, la mejor forma es poner en la posición que permita pincharlo en medio de la placa del prototipado, entre la separación intermedia que separa los pines de conexión, tal y como se muestra en la siguiente imagen:



Entradas Digitales: ejemplo con un Pulsador.

Arduino dispone de PINES DIGITALES para poder sacar Voltaje 0 o 5V. Pero dichos pines digitales también permiten el proceso inverso, es decir, que seamos nosotros externamente los que le suministremos voltaje al Arduino y este lo lea.

Poner un pulsador en medio de un circuito para cortar o abrir el circuito es intuitivo.

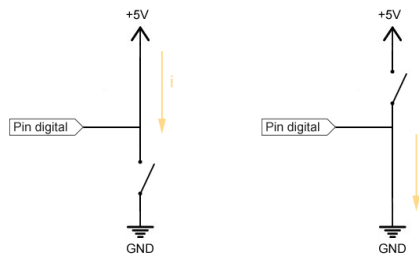
Lo que vamos a hacer en esta práctica/tutorial es utilizar un pulsador para hacer una lectura de corriente en Arduino, **ENTRADA DIGITAL**, de forma que:

1. El voltaje cambiará en función de si tenemos pulsado o no el pulsador.
2. Enviaremos ese voltaje al Arduino a través de una **ENTRADA DIGITAL**.
3. El Arduino leerá dicho voltaje.
4. Y en función de lo que lea Arduino hará una cosa u otra.

Montaje Pull-Up on Pull-Down

1. PRIMERA APROXIMACIÓN.

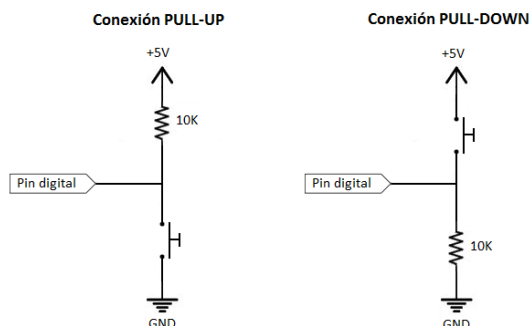
La primera idea que se nos puede ocurrir para hacer el circuito que indicamos en el punto anterior es montar cualquiera de los siguientes circuitos. Pero estos circuitos tienen un gran problema y es que provocarán un **COROTCIRCUITO**.



En un circuito electrónico nunca se puede juntar el polo positivo (+5V) y negativo (GND) de un generador directamente, siempre necesitamos que hay un componente por medio que consuma la electricidad (led, motor, resistencia, etc).

2. CIRCUITO CORRECTO.

Tenemos 2 posibilidades para poder conectar un pulsador a una entrada digital de Arduino.



1.- PULL-UP. En el circuito de la izquierda si el pulsador están sin pulsar (abierto), al no circular corriente por el circuito el valor el "Pin Digital" será 5V, mientras que si se pulsa (cerrado), en el circuito circulará corriente y el "Pin Digital" tendrá un valor de 0V (GND).

2.- PULL-DOWN. En el circuito de la derecha el proceso es inverso. Si el pulsador está sin pulsar el valor del "Pin Digital" será 0V (GND), mientras que si lo pulsamos será de 5V.

Para estos circuitos tenemos que poner una resistencia de gran valor 4,7K O 10K Ohmios.

Tipos de Pulsadores

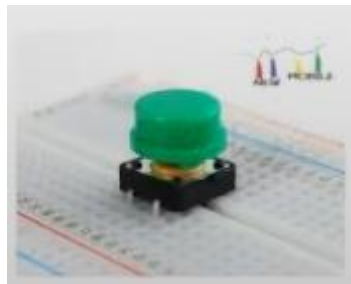
Existen pulsadores de diferentes tamaño:



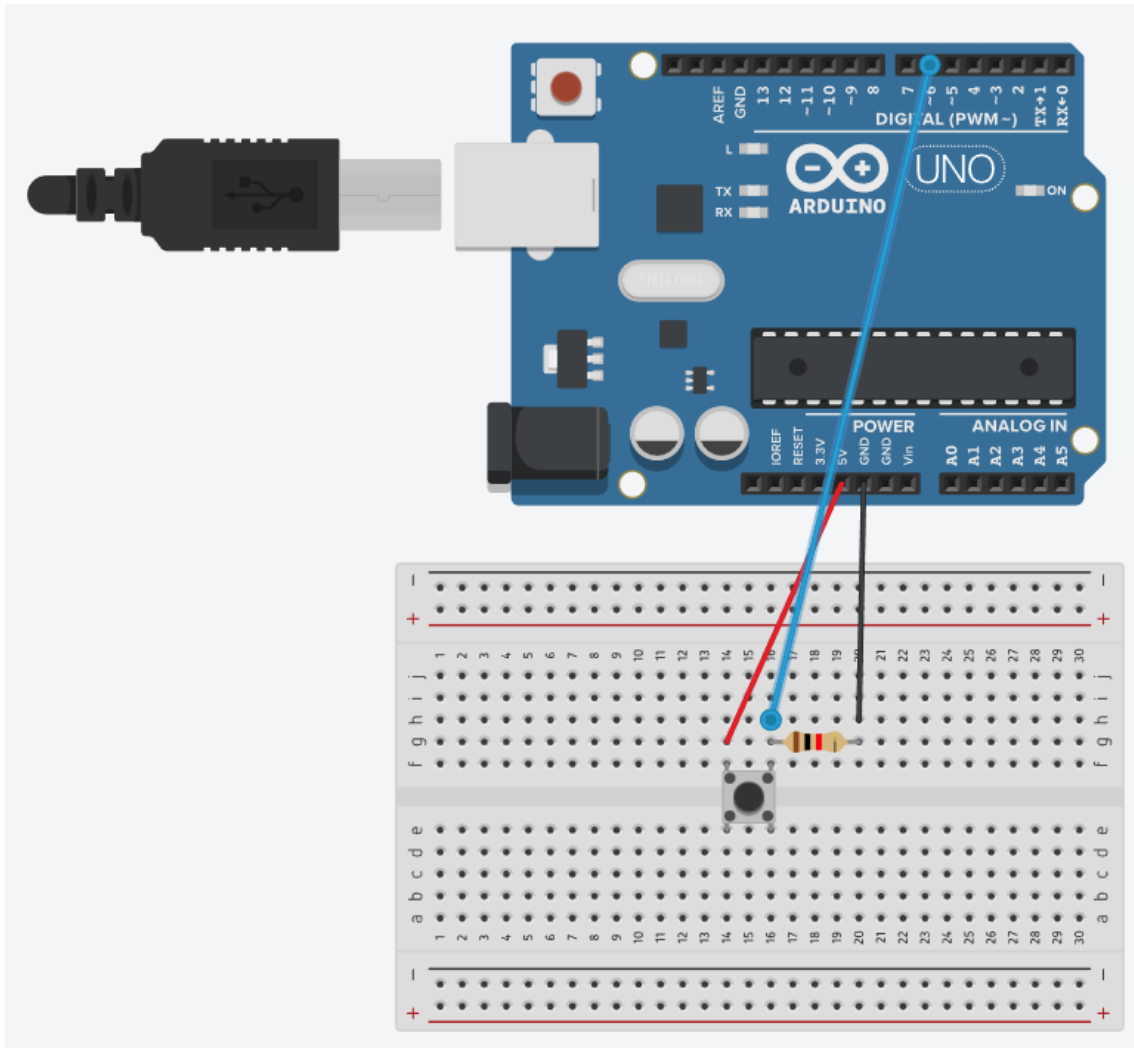
Además nos podemos encontrar otros encapsulados, con directamente solo los 3 pines que vamos a utilizar y la resistencia ya integrada.



Por último algunos pulsadores suelen venir acompañados de un plástico para poder insertar en la parte superior, para que su aspecto y manejo a la hora de pulsarse sea mejor.

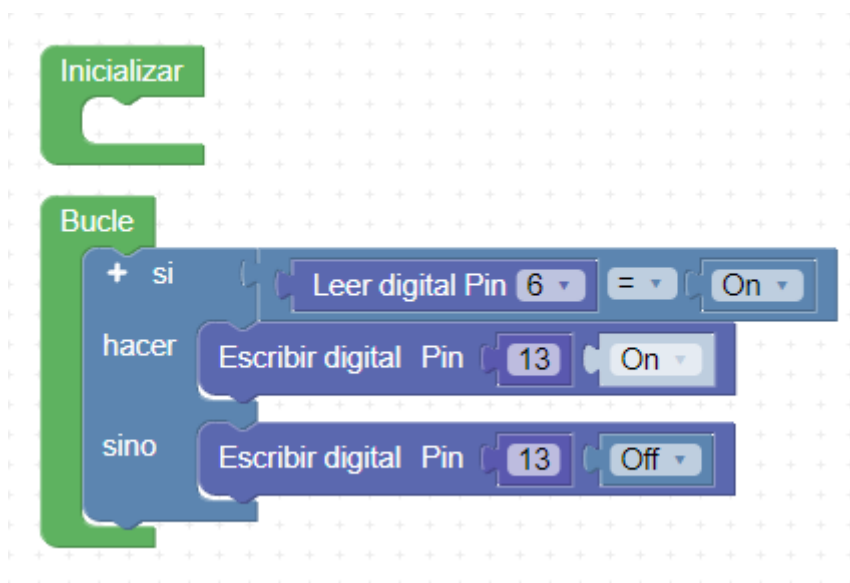


CIRCUITO ELECTRÓNICO



PROGRAMACIÓN

Programación por bloques:



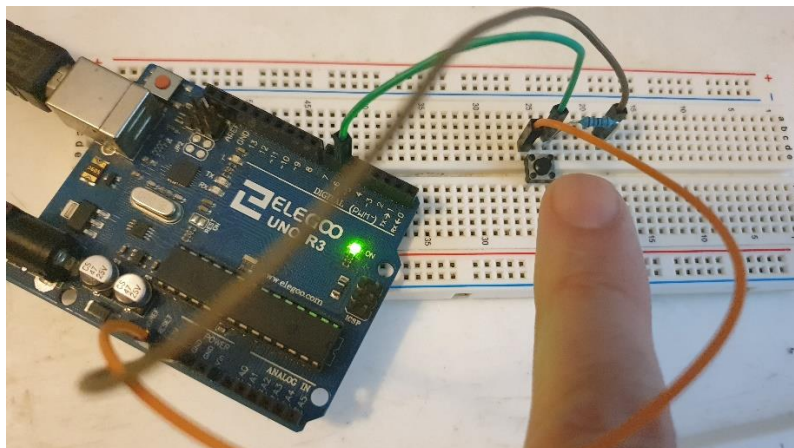
Programación por código:

```

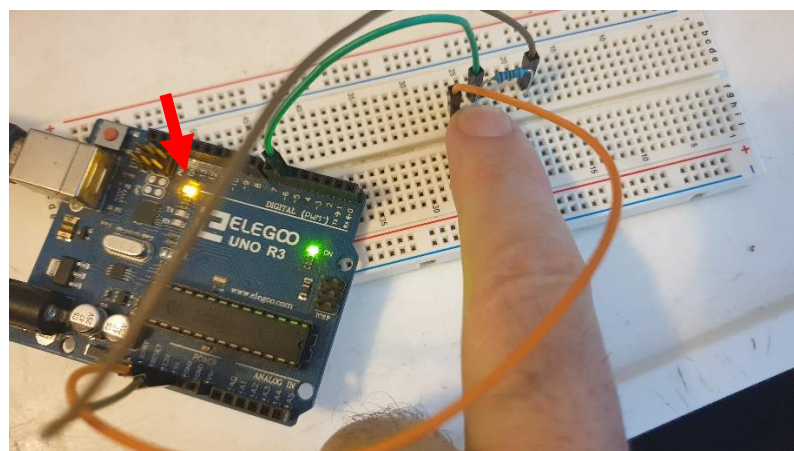
1 void fnc_dynamic_digitalWrite(int _pin, int _e){
2   pinMode(_pin,OUTPUT);
3   digitalWrite(_pin,_e);
4 }
5
6 void setup()
7 {
8   pinMode(6, INPUT);
9
10 }
11
12
13 void loop()
14 {
15
16   if ((digitalRead(6) == true)) {
17     fnc_dynamic_digitalWrite(13, HIGH);
18   }
19   else {
20     fnc_dynamic_digitalWrite(13, LOW);
21   }
22 }
23 }

```

PRUEBAS



Cuando no presionamos el interruptor.



Cuando presionamos el interruptor.

12.- Potenciómetro. ENTRADAS ANALÓGICAS

CONCEPTOS

¿Qué es un Potenciómetro?

Un potenciómetro es un componente electrónico que nos permite aplicar una RESISTENCIA VARIABLE en un circuito. A efectos prácticos lo que nos permite es tener una lectura de un voltaje VARIABLE que va a estar comprendido entre 0 y un máximo de voltaje que apliquemos al mismo.

El potenciómetro en realidad se puede sustituir por una resistencia, pero la ventaja de potenciómetro es que su valor de esa resistencia es variable entre 0 y el máximo que indica el valor del potenciómetro (10K Ohmios) en función de la posición donde esté situado el mismo.

¿Cómo Funciona un Potenciómetro?

Tiene 3 pines. En los pines del extremo ponemos la alimentación +5V y GND (se puede poner en la posición que queramos) y en el pin del centro tendremos la lectura del voltaje variable, en función de la posición del potenciómetro y consecuentemente de la resistencia que el mismo está aplicando al circuito.



Entradas Analógicas

Las entradas analógicas permiten leer corriente (voltajes) que introducimos en el Arduino. A diferencia de las Entradas Digitales que ya vimos en un capítulo anterior con el Pulsador, los voltajes en una entrada Analógica pueden tomar cualquier valor (por eso se dice Analógico) entre 0 Voltios y 5 Voltios.

En esta practica vamos a utilizar el PIN LECTURA del Pulsador para enviar dicho valor a una ENTRADA ANALÓGICA del Arduino y que este proceda a su lectura.

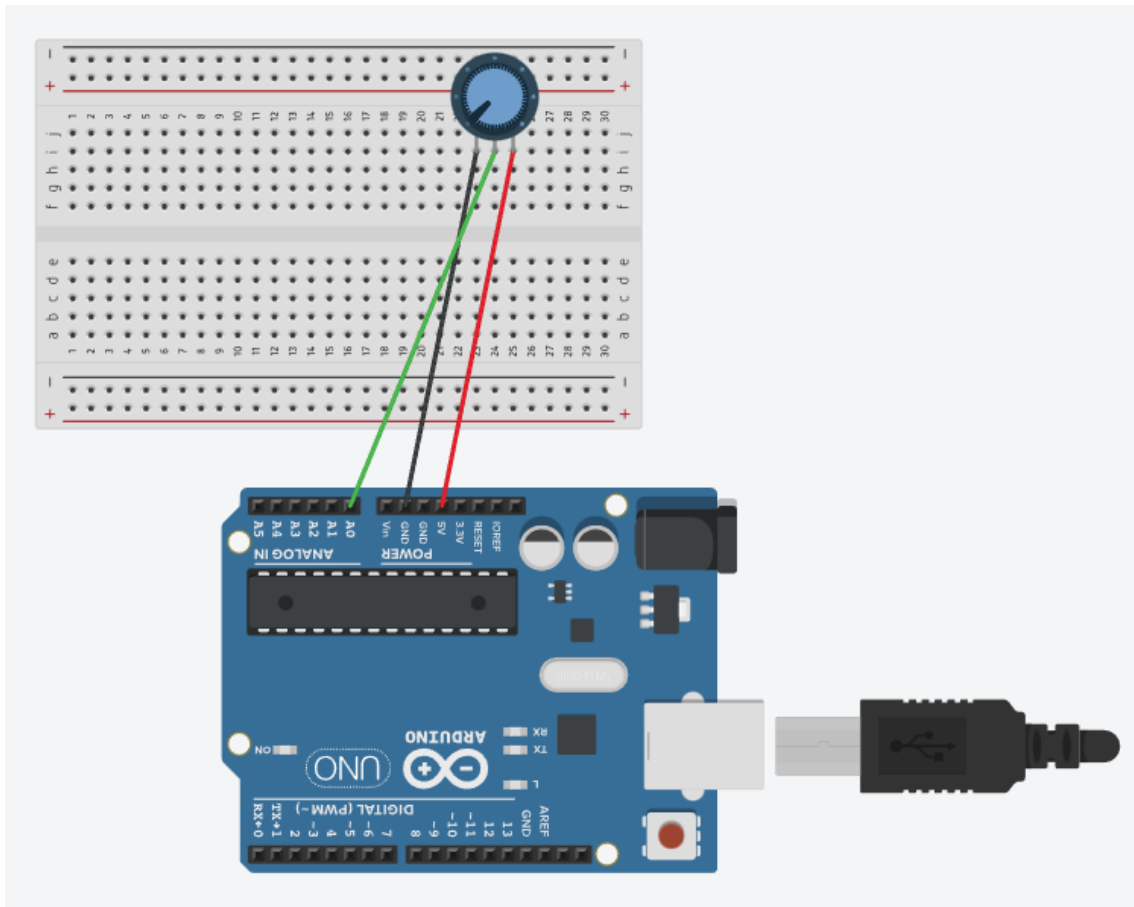
La lectura que realiza Arduino de este valor analógico no la pasa directamente al programa con un número decimal entre 0 y 5V, como cabría suponer, sino que la transforma en un número entero entre 0 y 1023.

Tipos de Potenciómetros

Existen diferentes tipos de Potenciómetros, pero todos coinciden en que tiene que tener 3 pines, los dos extremos para suministrar la corriente y el del medio para hacer la LECTURA.

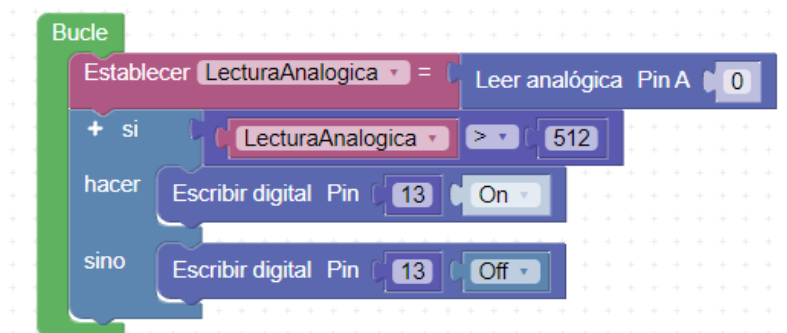


CIRCUITO ELECTRÓNICO



PROGRAMACIÓN

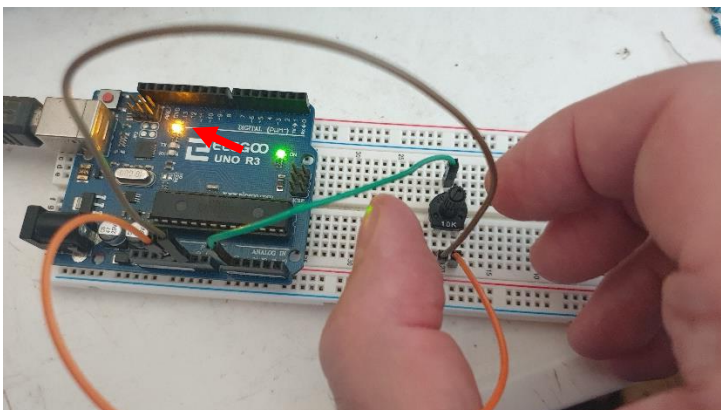
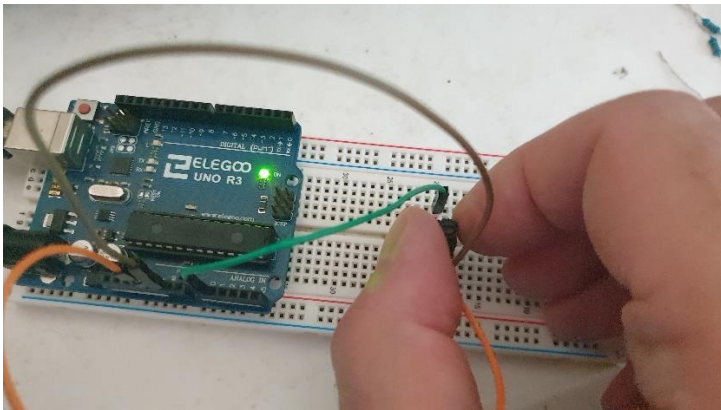
Programación por bloques:



Programación por código:

```
1 double LecturaAnalogica;
2
3 int fnc_dynamic_analogRead(int _pin){
4     pinMode(_pin,INPUT);
5     return analogRead(_pin);
6 }
7
8 void fnc_dynamic_digitalWrite(int _pin, int _e){
9     pinMode(_pin,OUTPUT);
10    digitalWrite(_pin,_e);
11 }
12
13 void setup()
14 {
15
16
17 }
18
19
20 void loop()
21 {
22
23     LecturaAnalogica = fnc_dynamic_analogRead(0);
24     if ((LecturaAnalogica > 512)) {
25         fnc_dynamic_digitalWrite(13, HIGH);
26     }
27     else {
28         fnc_dynamic_digitalWrite(13, LOW);
29     }
30 }
31 }
```

PRUEBAS



Según el voltaje que se le pase, la luz del pin 13 se encenderá o no.

El potenciómetro se aconseja conectarlo en la mitad de la placa con en el ejemplo del interruptor.

13.- Comunicación Serie Consola

CONCEPTOS

¿Qué es la Comunicación Serie?

Arduino incorpora una **conexión serie** que permite conexión con el Pc (o con otros muchos dispositivos). Esta es la misma conexión que se utiliza para subir programas al Arduino a través del cable USB al Ordenador.



¿Qué es la Consola de Arduino?

La consola de Arduino es una funcionalidad que utilizando la comunicación Seria que hemos comentado en el punto anterior nos va a permitir:

1. Enviar información desde Arduino al PC y poder verla en pantalla.
2. Que nosotros a través del teclado podamos enviar información desde el PC al Arduino.

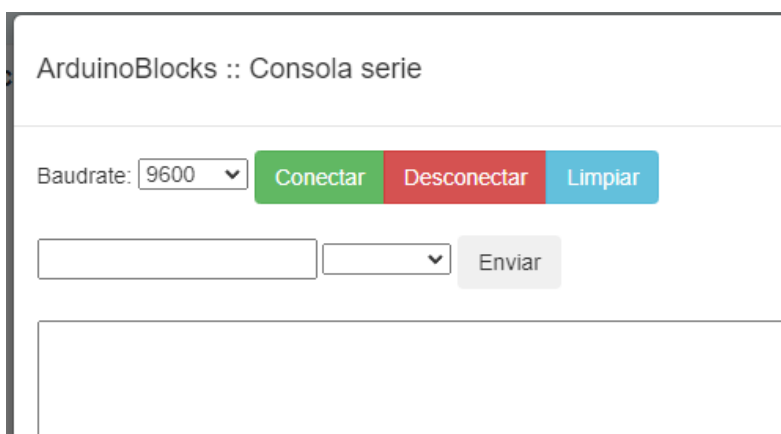
Por lo tanto a través de la consola vamos a poder comunicarnos e intercambiar información dentro de nuestros programas entre Arduino y nosotros.

¿Cómo usar la Consola de Arduino?

El uso de esta consola depende del programa con el que vayamos a programar el Arduino. Pero en líneas generales lo pasos serían los siguientes:

1. Conectar y sincronizar la comunicación entre el Arduino y la Consola.
2. Enviar información a través del tecla al Arduino.
3. Recibir información del Arduino y mostrarlo en la pantalla de la Consola.

Esta es una imagen de la consola de Arduino para el programa ArduinoBloks con el que estamos programando en este tutorial.



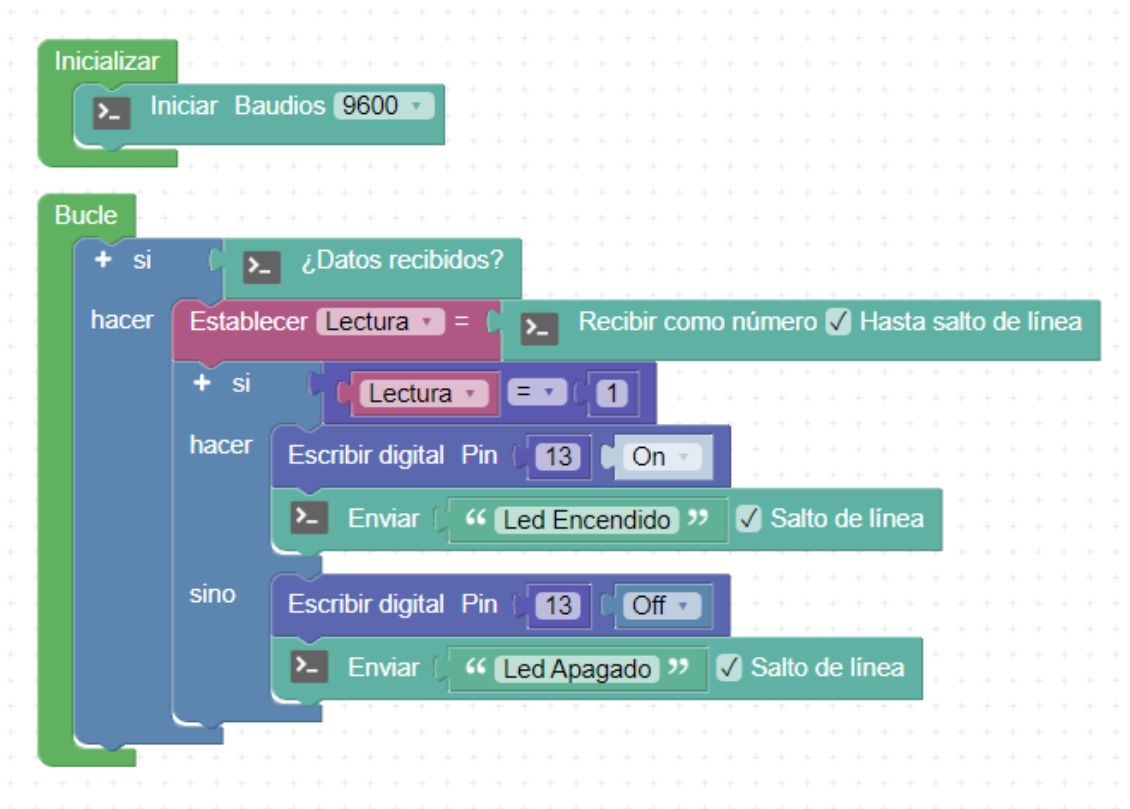
Utilizar la consola para encender/apagar Led

Haciendo uso de la consola vamos a realizar una práctica en la que vamos a: Encender o pagar un led en función de un número que vamos a escribir en la consola de Arduino:

1. El Arduino cuando reciba un valor encenderá o pagará dicho led, en función de su valor.
2. A continuación Arduino se comunicará con la consola para enviar un mensaje de Texto de: "Led Encendido/Apagado" que se mostrará en la pantalla de la Consola.

PROGRAMACIÓN

Programación en bloques:



Programación por código:

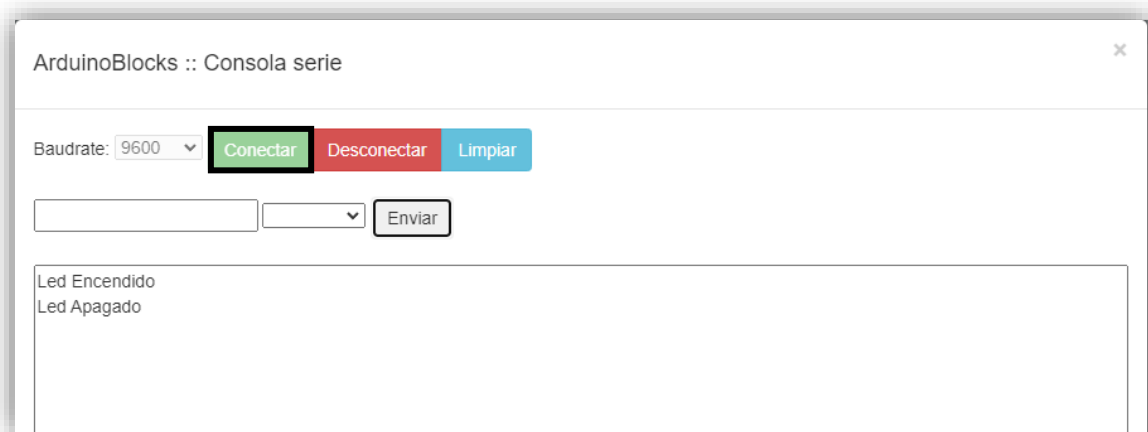
```
1 double Lectura;
2
3 void fnc_dynamic_digitalWrite(int _pin, int _e){
4     pinMode(_pin,OUTPUT);
5     digitalWrite(_pin,_e);
6 }
7
8 void setup()
9 {
10
11
12     Serial.begin(9600);
13     Serial.flush();
14     while(Serial.available(>0)Serial.read();
15
16 }
```

```

17
18
19 void loop()
20 {
21
22     if ((Serial.available()>0)) {
23         Lectura = atof((Serial.readStringUntil('\n')).c_str());
24         if ((Lectura == 1)) {
25             fnc_dynamic_digitalWrite(13, HIGH);
26             Serial.println(String("Led Encendido"));
27         }
28         else {
29             fnc_dynamic_digitalWrite(13, LOW);
30             Serial.println(String("Led Apagado"));
31         }
32     }
33 }
34
35 }

```

PRUEBAS



Ejecutamos la consola y nos conectamos en el cuadro de texto vamos a introducir el número 1 o el número 2 indistintamente, si recibe el Arduino un 1 encenderá el led del pin 13 y nos enviará un mensaje de “Led Encendido” si introducimos otro número que no sea el uno apagar el led que tenemos conectado en el pin 13 y enviará un mensaje de “Led Apagado”.

14.- Sensor Ultrasonidos. SENSORES

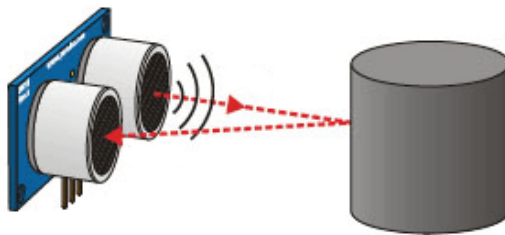
CONCEPTOS

¿Qué es un Sensor de distancia por Ultrasonidos?

Son detectores de proximidad que detectan objetos a distancias que van desde pocos centímetros hasta varios metros (4 como alcance máximo). Depende de los modelos.

¿Cómo Funciona un Sensor de Ultrasonidos?

El sensor emite un sonido (denominado ultrasonidos no apreciable por el oído humano), y mide el tiempo que la señal tarda en regresar. Estos reflejan en un objeto, el sensor recibe el eco producido y lo convierte en señales eléctricas.



La forma de conectar este dispositivo para poder utilizarlo en nuestros circuitos es muy sencillo tal y como muestra la siguiente figura:

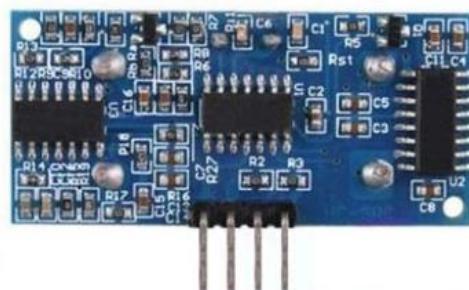


Las conexiones del sensor a nuestro Arduino serían las siguientes:

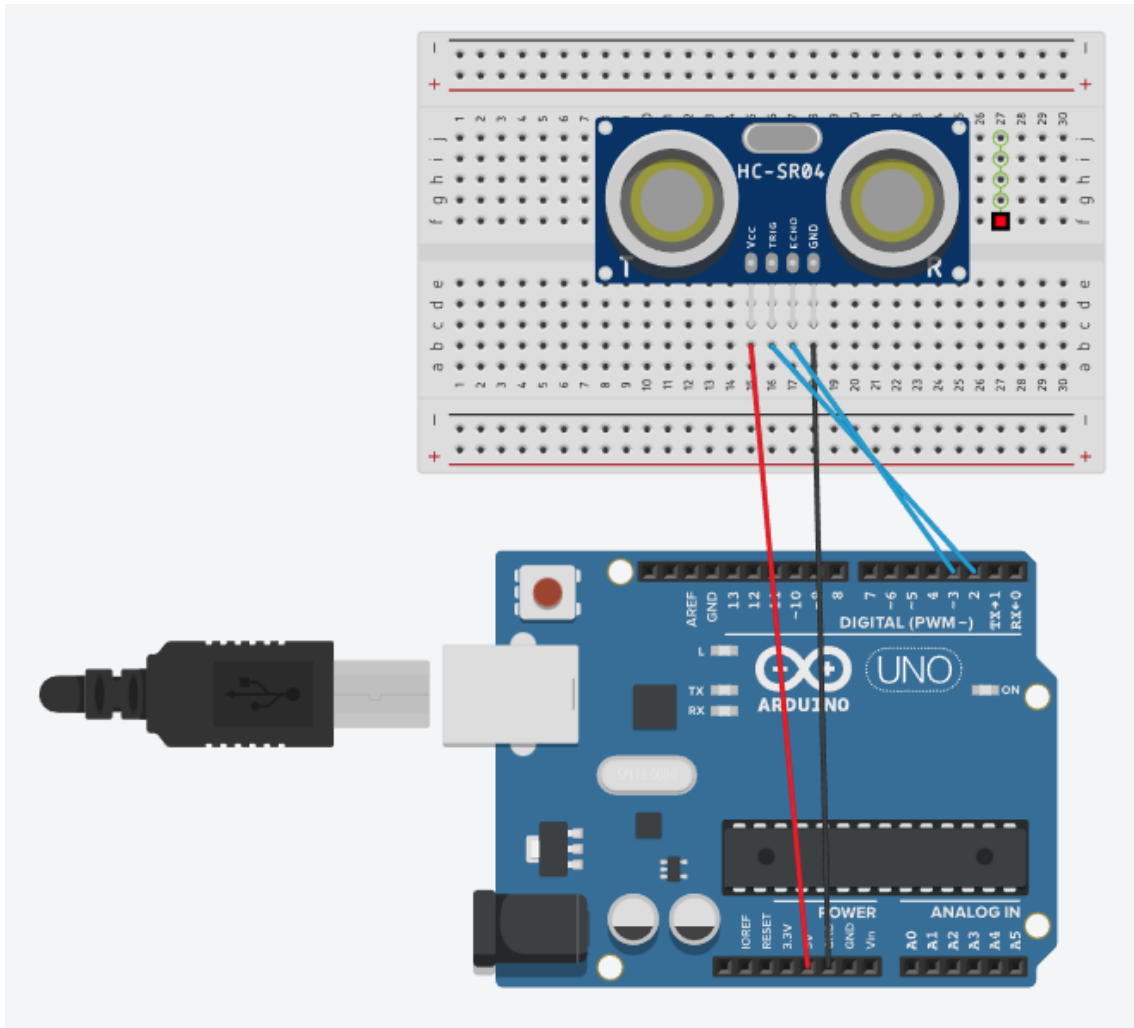
1. Debemos alimentar con corriente el Sensor a través de los pines de 5V (Indicado con Vcc en el sensor) y de Gnd.
2. Tenemos que conectar el pin Trigger del sensor al pin 2 de Arduino y el pin Echo del sensor al pin 3 de Arduino. Los conectaremos a estos pines porque por defecto son los que se suele traer el programa para utilizarlos, aunque los podríamos configurar en otros pines digitales del Arduino.

Tipo de Sensores de Ultrasonidos

Para nuestros circuitos utilizaremos el modelo HC-SR04 que es uno de los más empleados y de fácil acceso.

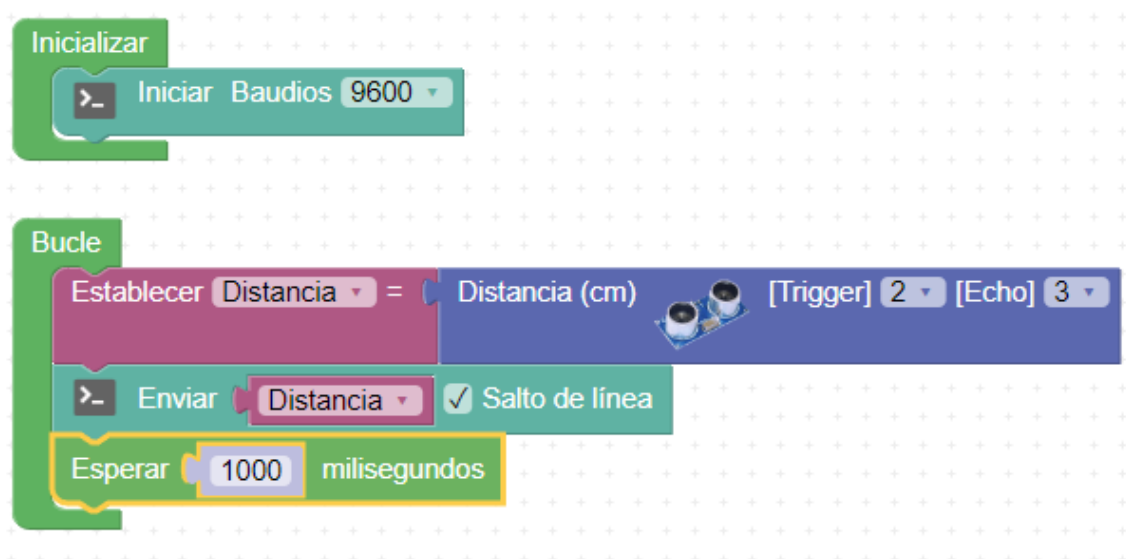


CIRCUITO ELECTRÓNICO



PROGRAMACIÓN

Programación por bloques:



Programación por código:

```

1 double Distancia;
2
3 double fnc_ultrasonic_distance(int _t, int _e){
4     unsigned long dur=0;
5     digitalWrite(_t, LOW);
6     delayMicroseconds(5);
7     digitalWrite(_t, HIGH);
8     delayMicroseconds(10);
9     digitalWrite(_t, LOW);
10    dur = pulseIn(_e, HIGH, 18000);
11    if(dur==0)return 999.0;
12    return (dur/57);
13 }
14
15 void setup()
16 {
17     pinMode(2, OUTPUT);
18     pinMode(3, INPUT);
19
20     Serial.begin(9600);
21     Serial.flush();
22     while(Serial.available()>0)Serial.read();
23 }
24
25
26
27 void loop()
28 {
29
30     Distancia = fnc_ultrasonic_distance(2,3);
31     Serial.println(Distancia);
32     delay(1000);
33 }
34 }

```

PRUEBAS

Una vez subido el programa al Arduino, accedemos a la consola y conectamos.



Ya nos está mostrando la distancia que detecta el sensor.

15.- Sensor Temperatura y Humedad. SENSORES

CONCEPTOS

¿Qué es un sensor de Temperatura y Humedad?

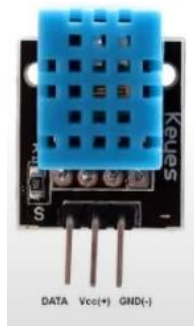
Sensor que nos permite obtener una medida de temperatura y humedad en el ambiente con un único dispositivo. Normalmente su precisión en los modelos que vamos a utilizar no es muy alta.

¿Cómo Funciona?

Los sensores de la familia DHT nos proporcionan de forma digital la **temperatura y la humedad**, con diferente precisión según el modelo.

CONEXIONADO

El pin GND y el VCC del sensor se conectan en sus correspondientes pines en Arduino (GND y +5V). El pin "data" se conecta en un pin digital para leerlo (si no es un módulo hay que añadir la resistencia de 10K).



Tipo de Sensores de Temperatura y Humedad

PROGRAMACIÓN

Por bloques:

```
graph TD
    subgraph Inicializar
        A[Iniciar Baudios 9600]
    end
    subgraph Bucle
        B[Establecer temperatura = DHT-11 Temperatura °C Pin 9]
        C[Establecer humedad = DHT-11 Humedad % Pin 9]
        D[Enviar "Humedad: " + humedad + Salto de línea]
        E[Enviar "Temperatura: " + temperatura + Salto de línea]
    end
    A --> B
    A --> C
    A --> D
    A --> E
```


Esperar 5000 milisegundos

Por código:

```
1 #include "ABlocks_DHT.h"
2
3 double temperatura;
4 double humedad;
5 DHT dht9(9,DHT11);
6
7 void setup()
8 {
9     pinMode(9, INPUT);
10
11     Serial.begin(9600);
12     Serial.flush();
13     while(Serial.available()>0)Serial.read();
14
15     dht9.begin();
16 }
17
18
19
20 void loop()
21 {
22
23     temperatura = dht9.readTemperature();
24     humedad = dht9.readHumidity();
25     Serial.println(String("Humedad: ")+String(humedad));
26     Serial.println(String("Temperatura: ")+String(temperatura));
27     delay(5000);
28 }
29 }
```

PRUEBAS

Este será el resultado:



ArduinoBlocks :: Consola serie

Baudrate: 9600 Conectar Desconectar Limpiar

Enviar

Humedad: 95.00
Temperatura: 28.00
Humedad: 95.00
Temperatura: 28.00
Humedad: 95.00
Temperatura: 28.00

16.- Servo Motor. MOTORES

CONCEPTOS

¿Qué es un ServoMotor?

Un ServoMotor también conocido como Servo, es un dispositivo eléctrico/mecánico que funciona de manera similar a un MOTOR.

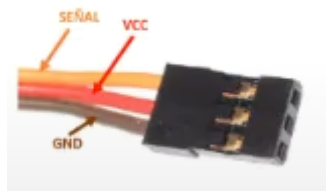
Sin embargo presenta una gran diferencia con respecto al motor clásico y es que podemos **UBICARLO** en cualquier posición dentro de un rango de funcionamiento, este rango es de 180°. En el modelo que vamos a utilizar nosotros.

¿Cómo Funciona un ServoMotor?

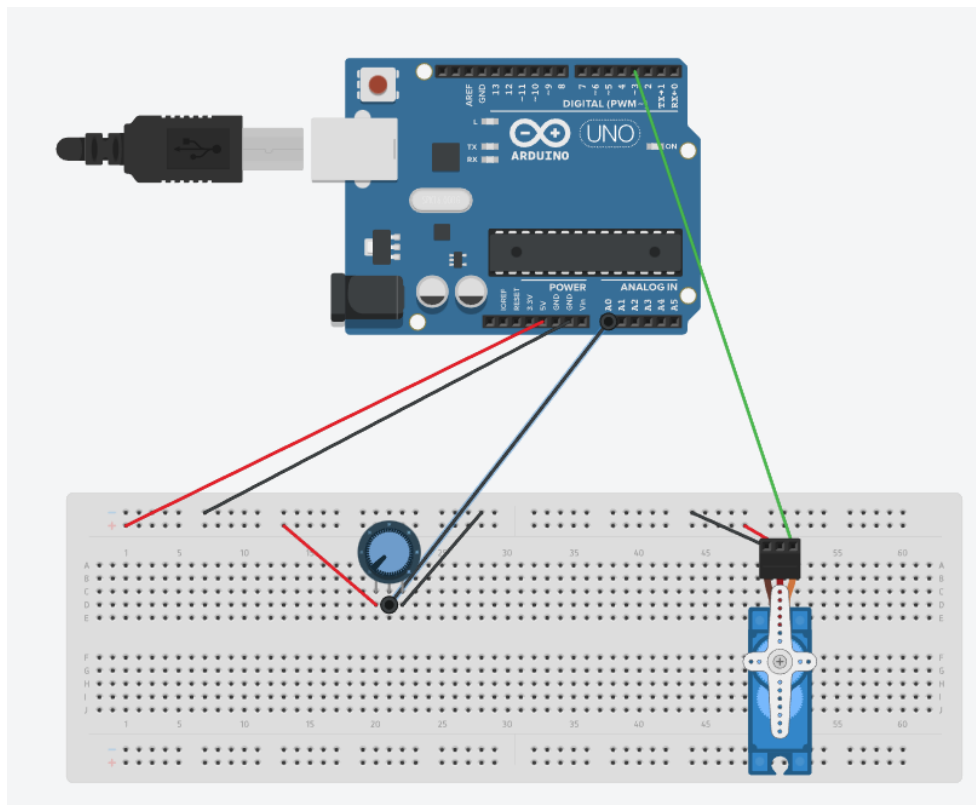
Los ServoMotores funcionan en Arduino a través de un pin PWM para controlar la posición del Servo.

Las conexiones del Servo a nuestro Arduino serían a través de 3 pines de la siguiente forma:

1. Dos pines son de alimentación VCC (rojo) y GND (marrón), que irán a los pines del Arduino 5V y GND respectivamente.
2. El tercer pin será el PIN DE SEÑAL (naranja), tendremos que conectarlo en un PIN ENTRADA DIGITAL PWM de Arduino.

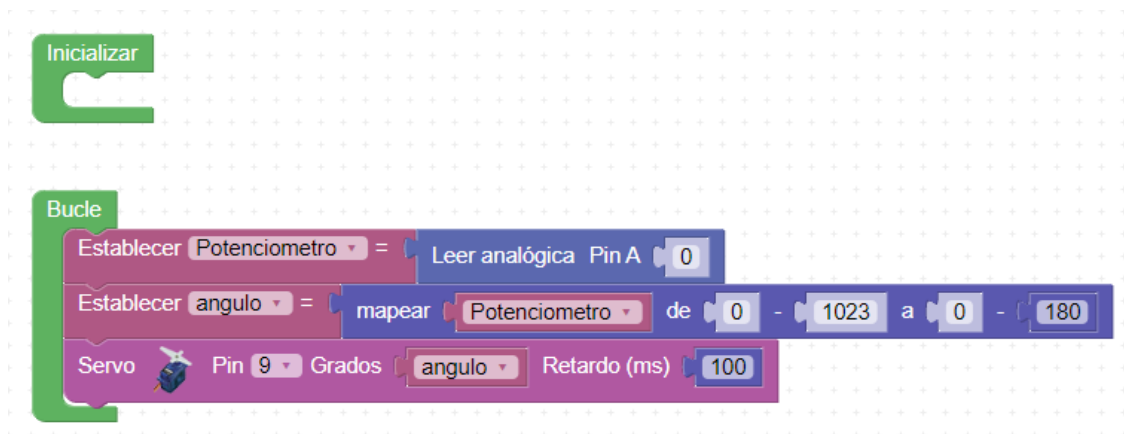


CIRCUITO ELÉCTRICO



PROGRAMACIÓN

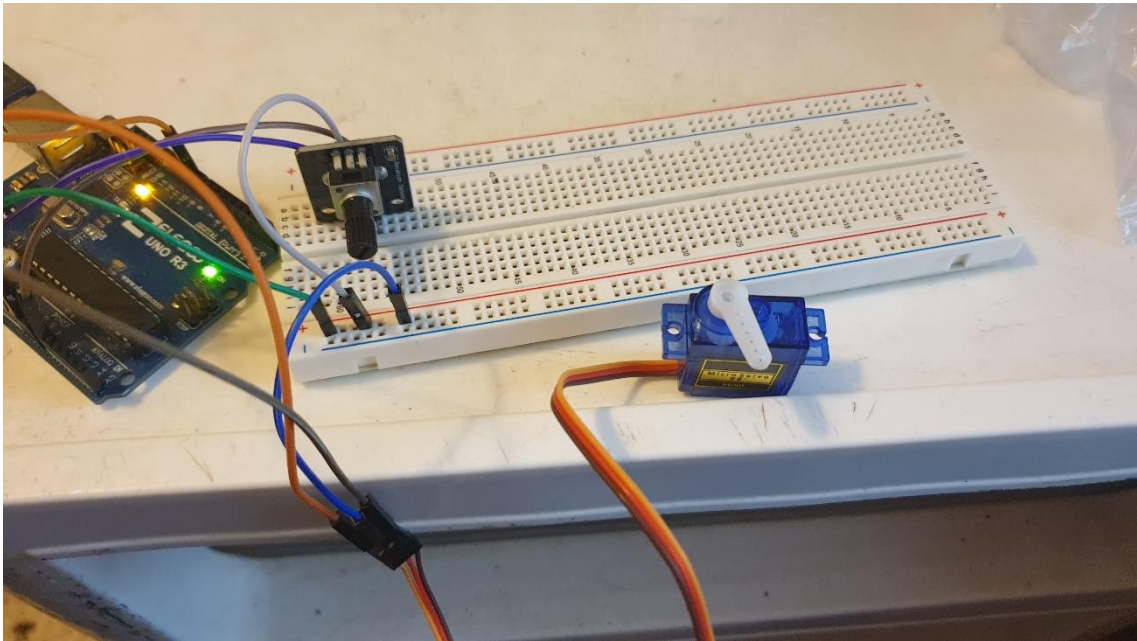
Programación por bloques:



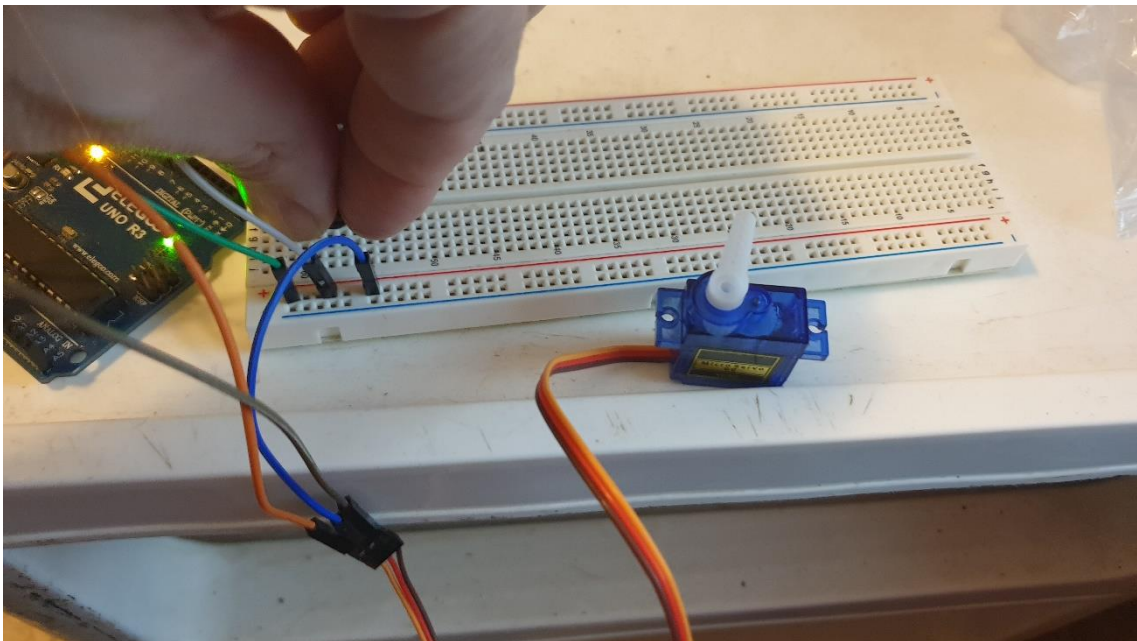
Programación por código:

```
1 #include <Servo.h>
2
3 double Potenciómetro;
4 double angulo;
5 Servo servo_9;
6
7 int fnc_dynamic_analogRead(int _pin){
8     pinMode(_pin, INPUT);
9     return analogRead(_pin);
10 }
11
12 void setup()
13 {
14     servo_9.attach(9);
15 }
16
17
18
19 void loop()
20 {
21
22     Potenciómetro = fnc_dynamic_analogRead(0);
23     angulo = map(Potenciómetro, 0, 1023, 0, 180);
24     servo_9.write(angulo);
25     delay(100);
26 }
27 }
```

PRUEBAS



Moviendo el potenciómetro movemos el servo.



17.- Pantalla LCD. PANTALLAS

CONCEPTOS

¿Qué es una Pantalla LCD?

Las pantallas LCD (Liquid Cristal Display), también denominadas DISPLAY LCD, son una de las formas más sencillas u económicas de dotar de un display a un autómeta.

Con esta pantalla podremos dotar a nuestros proyectos de Arduino de un dispositivo para poder enviar información desde el Arduino al usuario de manera autónoma e independiente del ordenador (consola puerto serie)

Tipo de Pantallas

Podemos encontrar pantallas LCD de diferentes tamaños (en función de número de caracteres máximos que permita la pantalla mostrar a la vez) nosotros vamos a trabajar con el modelo más extendido por su relación calidad/precio que es el que suele venir incluido en los kits de Arduino y que es el modelo 16x2 (2 filas x 16 columnas).

IMPORTANTE. Un dato a tener en cuenta es que este tipo de pantalla LCD nos lo podemos encontrar en diferentes formatos, depende de dónde lo compremos, las opciones puede ser:

- **DISPLAY LC CON PINES SIN SOLDAR.** En algunos kits ofrecen este dispositivo con los pines sin soldar por lo que tenemos que tener cuidado porque soldarlos además de requerir un soldador no es nada sencillo. Por lo tanto no recomendamos esta opción.



- **DISPLAY LCD CON PINES SOLDADOS.** En este caso los pines ya vienen soldados al LCD. Esta es nuestra opción recomendada relación calidad/precio. Aunque el conexionado al Arduino será un poco complejo con la cantidad de cables necesarios para conectarlo, pero si perfectamente asumible.



- **DISPLAY LCD CON CONTROLADOR I2C INTEGRADO.** En este caso el Display viene con el controlador de LCD I2C que es un dispositivo que nos permite controlar una pantalla a través del bus I2C, usando únicamente dos cables. Esta opción es más cara y no suele venir en los kits de Arduino.



¿Cómo Funciona una Pantalla LCD?

El funcionamiento y la programación de la pantalla es muy sencillo, sin embargo la dificultad radica en el número considerable de conexiones que tenemos que realizar. A continuación se detalla las mismas indicando las conexiones a realizar de los pines del display LCD – a los pines de Arduino.

ALIMENTACIÓN LCD

VSS – GND
 VDD – 5v
 RW – GND
 A – 5v
 K – GND

POTENCIÓMETRO PARA CONTROL BRILLO LCD

V0 – Salida del potenciómetro. Colocar un potenciómetro alimentando el mismo con 5v y GND y poniendo el pin de salida al pin V0 del Display LCD.

PINES DE SEÑAL

Estos pines se pueden poner a los pines digitales que queramos de Arduino, luego habrá que configurarlo tal cual en las instrucciones del programa.

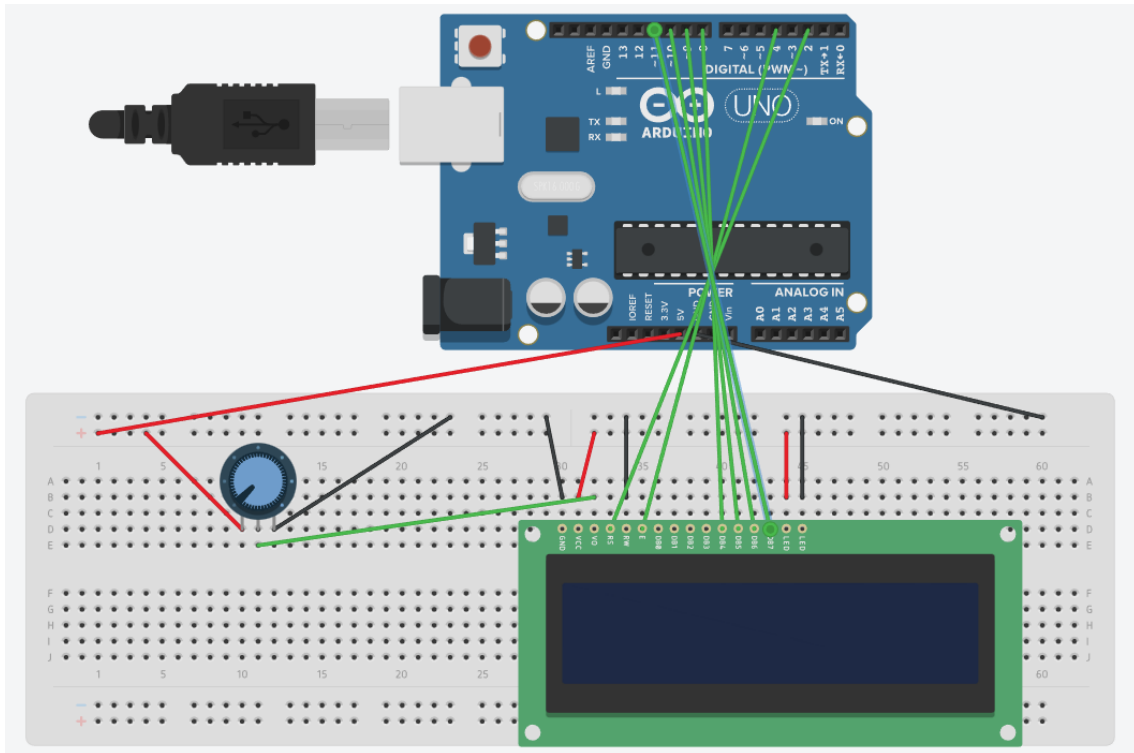
RS – 2
 E – 4 (en el lenguaje de programación viene indicado como “EN” en vez de “E”).
 D4 – 8
 D5 – 9
 D6 – 10
 D7 – 11

RESUMEN COMPLETO CONEXIONES.

LCD	ARDUINO
VSS	GND
VDD	5v
V0	Potenciómetros
RS	2
RW	GND
E	4
D4	8
D5	9

D6	10
D7	11
A	5v
K	GND

CIRCUITO ELECTRÓNICO



PROGRAMACIÓN

Programación por bloques:



Programación por código:

```

1 #include <LiquidCrystal.h>
2
3 LiquidCrystal lcd(2,4,8,9,10,11);
4
5 void setup()
6 {
7

```

```

8
9 lcd.begin(16, 2);
10
11     lcd.clear();
12
13 }
14
15
16 void loop()
17 {
18
19     lcd.setCursor(0, 0);
20     lcd.print(String("Hola Mundo"));
21
22 }

```

PRUEBAS



Con esta programación conseguimos que nuestro texto se desplace de derecha a izquierda.



18.- Mando Infrarrojos IR.MANDOS

CONCEPTOS

¿Qué es Mando/Sensor de InfraRojos?

Un mando a distancia es un dispositivo de control que emplea un **LED infrarrojo** para enviar una señal al receptor. La señal puede ser para controlar un autómatas o procesador como Arduino.

Todos estamos acostumbrados a emplear mandos a distancia infrarrojos, para la televisión, Dvd, equipos de música... De hecho, es posible emular cualquiera de estos mandos, lo que permite controlar estos dispositivos desde Arduino.



Un mando a distancia emplea un emisor de luz en el infrarrojo cercano, invisible para el ojo humano, pero que puede ser captado con facilidad por un receptor infrarrojo.

El alcance es limitado, típicamente inferior a 3m. **La distancia depende fuertemente del ángulo de emisión**, disminuyendo rápidamente a medida que nos desviamos de la dirección frontal.

Podemos emplear un mando a distancia como control remoto para controlar Arduino. Podemos, por ejemplo, encender y apagar un sistema de luces o sonido, o controlar un robot o un vehículo.

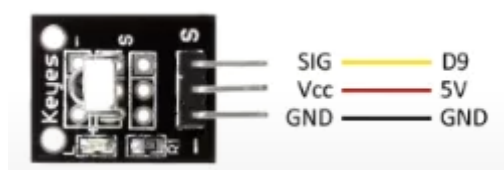
¿Cómo Funciona un Mando/Sensor IR?

Un mando a distancia transmite un cierto mensaje al receptor empleando luz infrarroja como sistema de transmisión.

Si usamos un módulo con el receptor integrado la conexión es sencilla. Alimentamos el módulo conectando Vcc y Gnd, respectivamente a 5V y Vcc de Arduino.

Por otro lado, conectamos la salida del módulo Sig a una entrada digital cualquiera de Arduino. En el ejemplo usaremos la D9, pero podéis usar cualquier entrada digital.

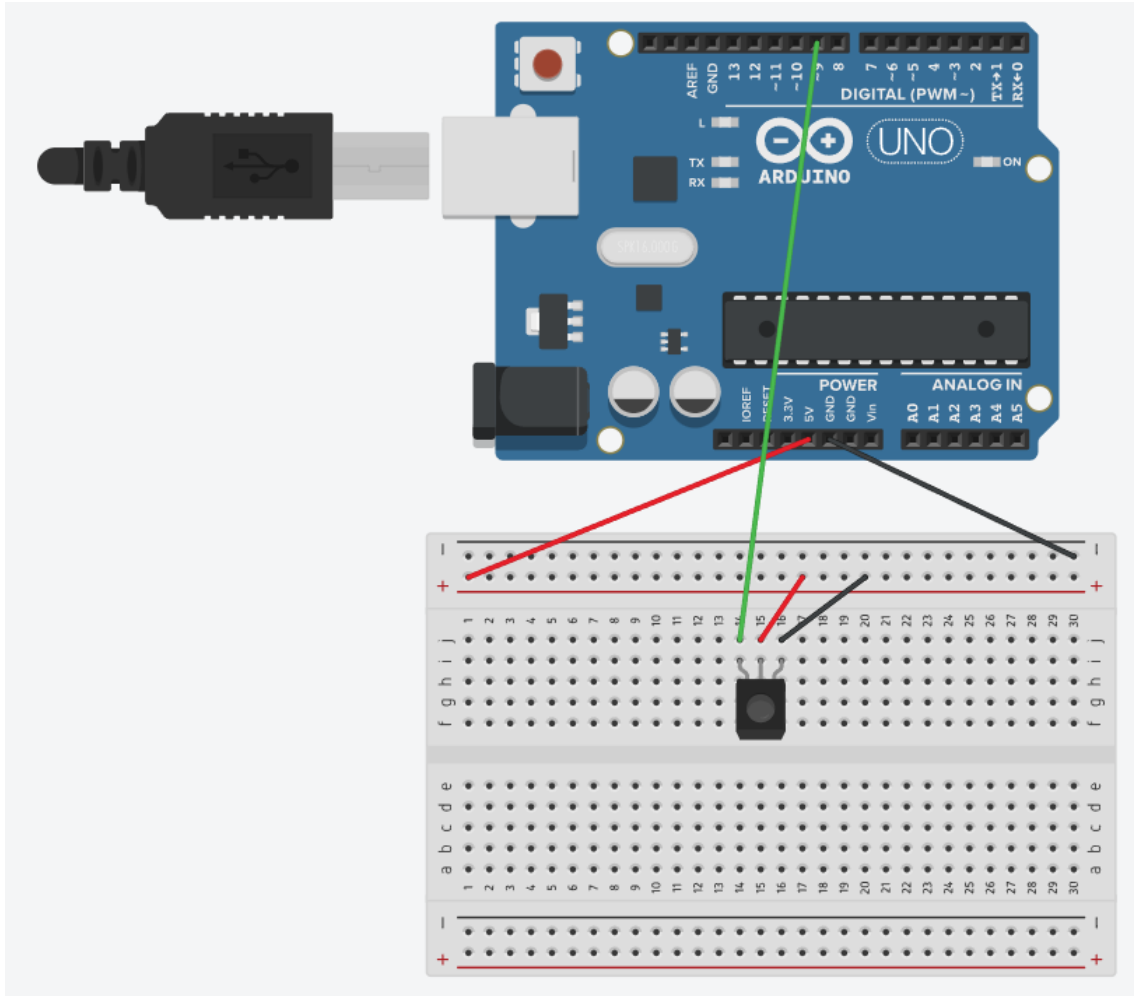
Por tanto, la conexión queda como se muestra en el siguiente esquema:



ArduinoBlocks realiza un proceso de decodificación automáticamente, pero debemos saber que no todos los mandos de control remoto de IR utilizan el mismo tipo de códigos. **Si no se recibe ningún código válido el bloque devuelve valor "0"**.

Normalmente el valor recibido por el bloque IR lo almacenaremos en una variable. El valor recibido es un valor de 32 bits sin signo, pero lo que lo recomendable a la hora de tratar el valor es ajustarlo con el bloque "Número entero sin signo".

CIRCUITO ELECTRÓNICO



PROGRAMACIÓN

Esta práctica consiste en saber el valor hexadecimal que tiene cada botón del mando a distancia, una vez los sabemos queremos que al presionar el 0 apague el led del pin 13 y al presionar el 1 encienda el led del pin 13.

Programación por bloques.





Programación por código:

```
#include "IRremote.h"
```

```
String s_mando;
IRrecv ir_rx(9);
decode_results ir_rx_results;
```

```
String fnc_ir_rx_decode_txt()
{
    char buff[16];
    buff[0]=0;
    if(ir_rx.decode(&ir_rx_results))
    {
        sprintf(buff,"%08IX", (unsigned long)ir_rx_results.value);
        ir_rx.resume();
    }
    return String(buff);
}
```

```
void fnc_dynamic_digitalWrite(int _pin, int _e){
    pinMode(_pin,OUTPUT);
    digitalWrite(_pin,_e);
}
```

```
void setup()
{

    Serial.begin(9600);
    Serial.flush();
    while(Serial.available(>0)Serial.read());
```

```

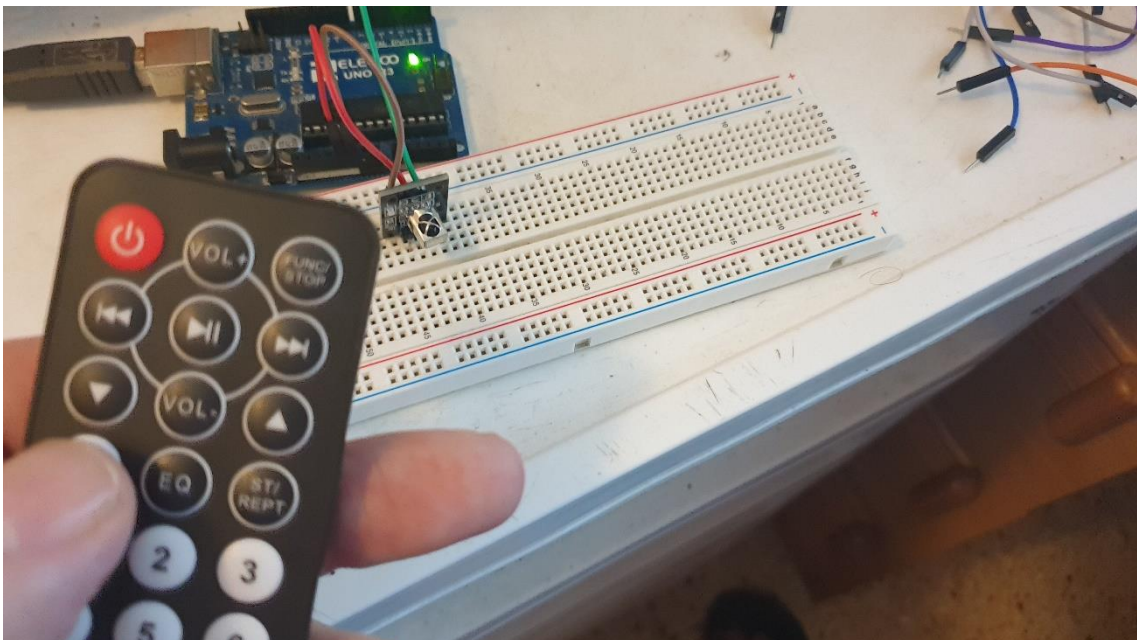
    ir_rx.enableIRIn();
}

void loop()
{
    s_mando = fnc_ir_rx_decode_txt();
    if ((!String(s_mando).equals(String("")))) {
        Serial.println(s_mando);
    }

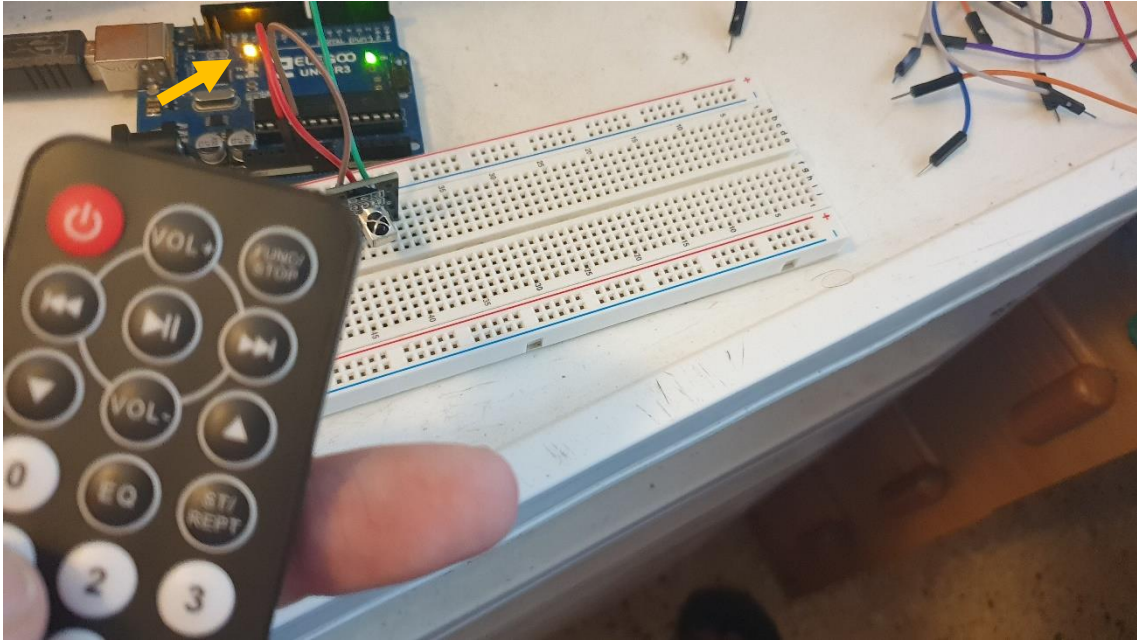
    if (String(s_mando).equals(String("00FF6897"))) {
        fnc_dynamic_digitalWrite(13, LOW);
    }
    else if (String(s_mando).equals(String("00FF30CF"))) {
        fnc_dynamic_digitalWrite(13, HIGH);
    }
}

```

PRUEBAS



Cuando seleccionamos el botón 0 apagamos el led del pin 13.



Cuando seleccionamos el 1, encendemos el led del pin 13.

19.- Buzzer Pasivo Altavoz. Sonido

CONCEPTOS

¿Qué es un Buzzer Pasivo/Altavoz?

Un buzzer pasivo o un altavoz son dispositivos que permiten convertir una señal eléctrica en una onda de sonido. Estos dispositivos no disponen de electrónica interna, por lo que tenemos que proporcionar una señal eléctrica para conseguir el sonido deseado.

¿Cómo Funciona un Buzzer Pasivo/Altavoz?

Un zumbador pasivo permite reproducir tonos con diferentes frecuencias (el sistema utilizado por ArduinoBlocks es similar al PWM pero por software). De esta forma podemos generar señales acústicas con el todo deseado y combinando duración, frecuencia y pausas podremos reproducir sencilla melodías.

Para ello deberemos conectar:

1. El PIN en el Zumbador como + (la marca la encontrareis en la zona superior del zumbador o en la parte inferior), a una entrada Digital PWM de Arduino.
2. El otro Pin del zumbador (el marcado como -) a GND.



Tipos de Buzzers

Hay 2 tipos básicos de zumbadores en los kits de Arduino, por lo que deberemos prestar atención para no confundirnos:

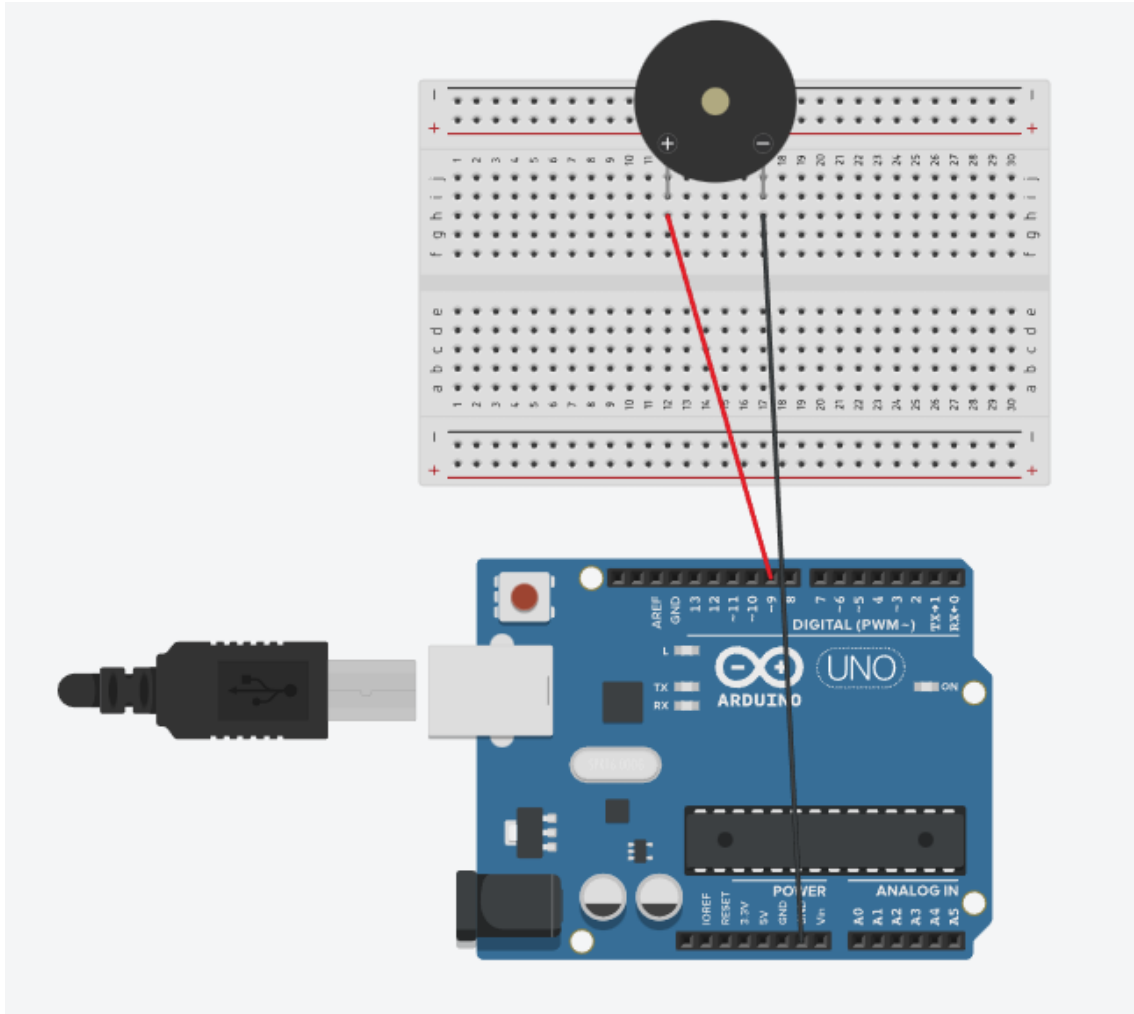
- **PASDIVOS o Altavoces.** Son los que vamos a utilizar en esta práctica, pueden emitir diferentes tonos y lo distinguiremos del otro modelo porque la dos patas tienen la misma longitud.



- **ACTIVOS o Zumbador.** Este modelo No puede emitir diferentes tonos, simplemente emite un único sonido. Lo distinguiremos del modelo anterior por que tiene una pata más larga que la otra, además suele venir con una pegatina de papel en la parte superior.



CIRCUITO ELECTRÓNICO



PROGRAMACIÓN

Programación por bloques:





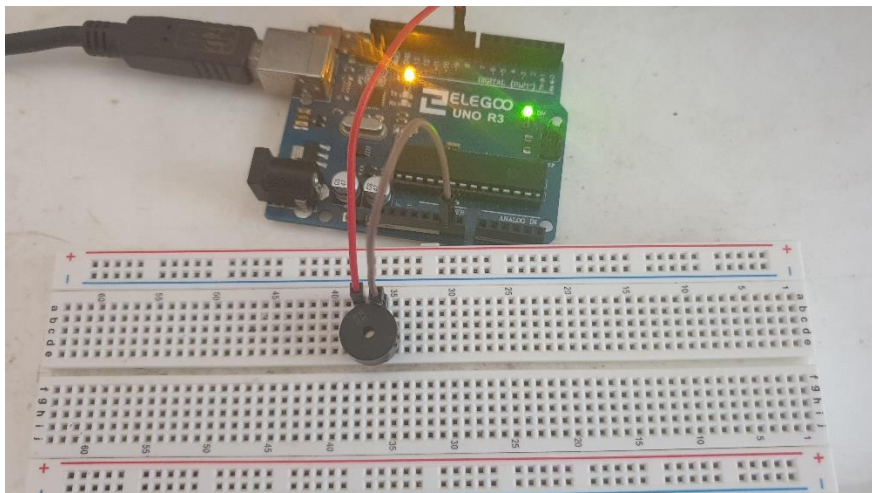
Programación por código:

```

1 #include <ABlocks_TimerFreeTone.h>
2
3 void setup()
4 {
5     pinMode(9, OUTPUT);
6 }
7
8
9
10 void loop()
11 {
12
13     TimerFreeTone(9,523.26,500);
14     TimerFreeTone(9,587.33,500);
15     TimerFreeTone(9,659.26,500);
16     TimerFreeTone(9,783.99,500);
17     TimerFreeTone(9,880.0,500);
18     TimerFreeTone(9,987.77,500);
19 }
20 }

```

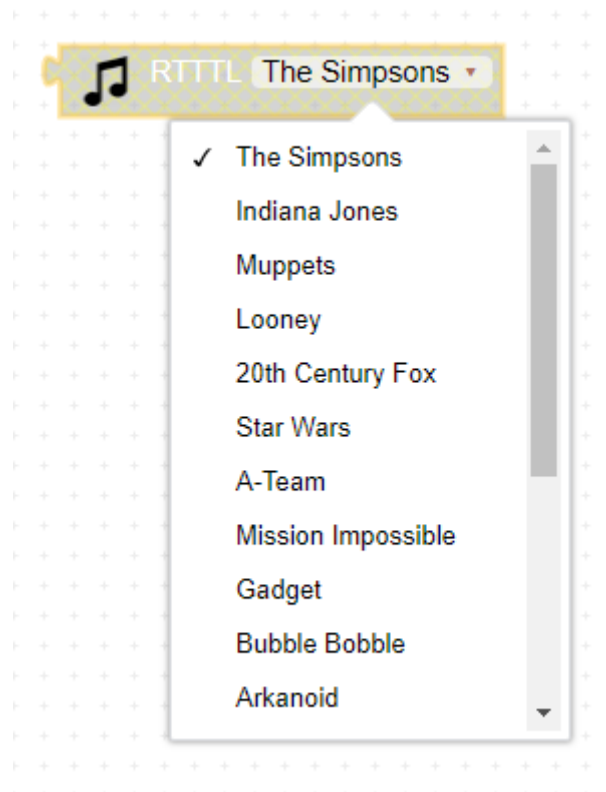
PREUBAS



Además tiene un opción muy interesante para poder escuchar canciones.



Tienes muchas para elegir.



20.- Joystick POSICIÓN

CONCEPTOS

¿Qué es un Joystick?

Un joystick analógico es un sencillo controlador que podemos añadir a nuestros proyectos de electrónica y Arduino, que tiene la ventaja de proporcionar una cantidad de información superior a la que podríamos obtener simplemente con pulsadores.

Estos joystick son similares a las palancas analógicas que tienen muchos mandos de videoconsolas, que se emplean cuando necesita ser más suave y preciso del que es posible con los mandos digitales.

¿Cómo Funciona un joystick?

Internamente los joystick están formados por un sistema de balancín con dos ejes acoplados a dos potenciómetros. Estos potenciómetros realizan la medición de la posición de la palanca en ambos ejes:

- Eje X (Horizontal)
- Eje Y (Vertical)

Por otro lado, uno de los ejes está apoyado en un microrruptor, lo que permite detectar la pulsación de la palanca.

CONEXIONADO

- **ALIMENTACIÓN.** Daremos corriente al pulsador a través de los pines +5v y GND que irán conectados a los PINES 5V y GND respectivamente del Arduino.
- **PIN VRx.** Nos dar una lectura analógica del valor del Eje X, que se traducirá en un número entre 0-1023 en función de la posición del Joystick en el Eje Horizontal. Por lo tanto lo conectaremos a una entrada analógica de Arduino.
- **PIN VRy.** No da una lectura Analógica del valor del Eje Y, que se traducirá en un número entre 0-1023 en función de la posición del Joystick en el Eje Vertical. Por lo tanto lo conectaremos a otra entrada Analógica de Arduino.
- **PIN SW.** Nos indica la pulsación del Joystick hacia abajo (como si fuera un pulsador). Podemos leer su valor de varias formas, nosotros lo conectaremos a otra entrada analógica del Arduino que tomará el valor analógico 0 (OV) cuando es pulsado. También podríamos conectarlo a una entrada digital de Arduino como si fuera un pulsador (pero poner una resistencia Pull-Up o Pull-Down para hacer la lectura).

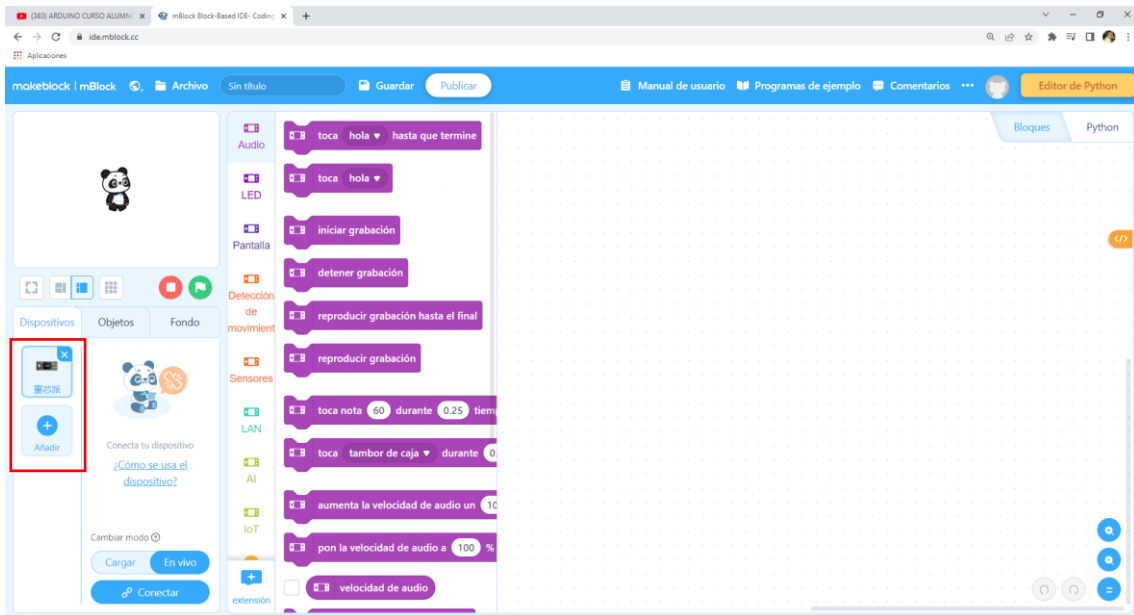
Práctica a desarrollar

Vamos a realizar una práctica con la que en función del movimiento del Joystick en el Eje X vamos a mover un personaje por la pantalla. Para esto realizaremos esta práctica en el SW de programación mBlock que nos permitirá realizar la programación simultánea y coordinada del personaje y de Arduino.

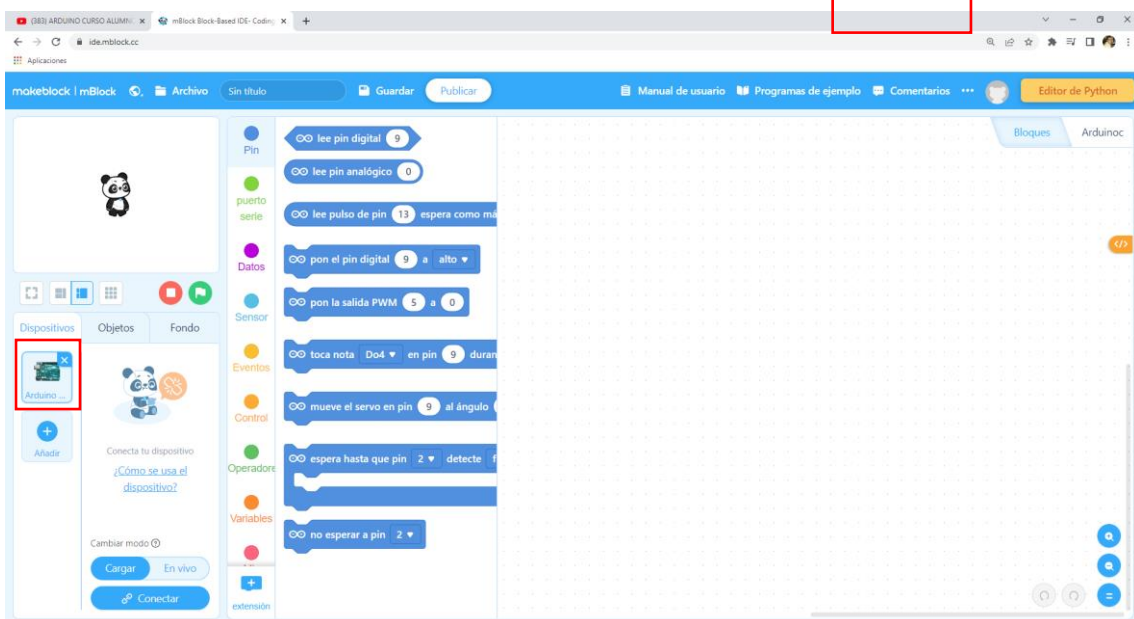
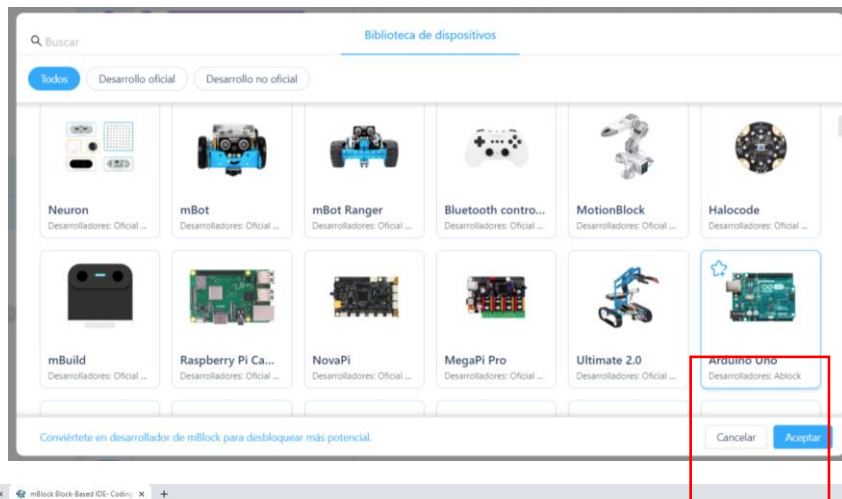
PROGRAMACIÓN con mBlock

Entramos en mBlock desde el siguiente enlace:

<https://ide.mblock.cc/>



Borramos el dispositivo y agregamos la placa de Arduino.



Ahora seleccionamos al oso panda, es decir tenemos acceso a la placa Arduino y al Oso panda.

Desde la placa de Arduino seleccionaremos en cargar en modo.



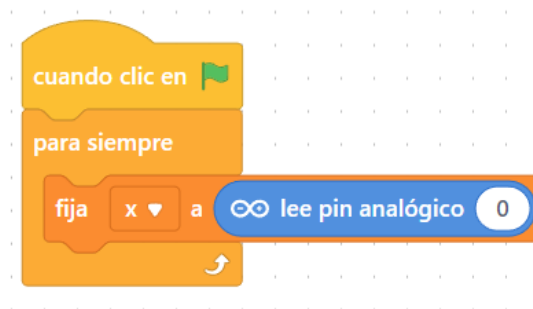
Nos vamos al grupo de bloques eventos para seleccionar.



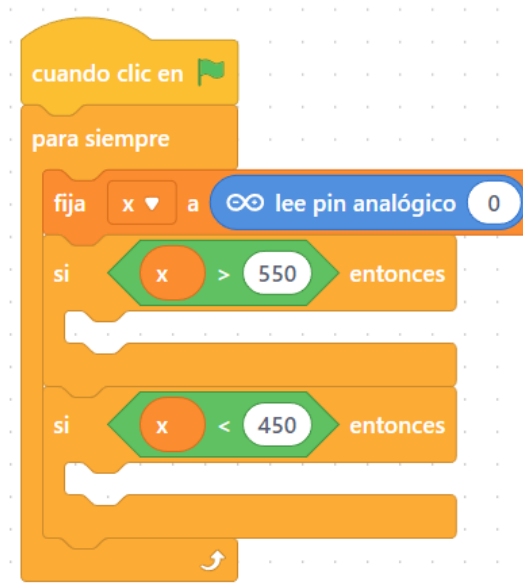
En el grupo control seleccionaremos para siempre.



Creamos una variable llamada X que leerá el pin analógico 0 donde hemos conectado el pin VRx.



Ahora queremos controlar la posición de Joystick.



Para que Arduino se pueda comunicar con el Oso panda lo haremos por mediación de mensajes.



Ahora seleccionaremos al Oso Panda.



Cuando recibe mensaje derecha avanza 10 pasos y cuando recibe mensaje izquierda retrocede 10 pasos.

Ejecuta la aplicación mLink desde el siguiente enlace lo podrás descargar.

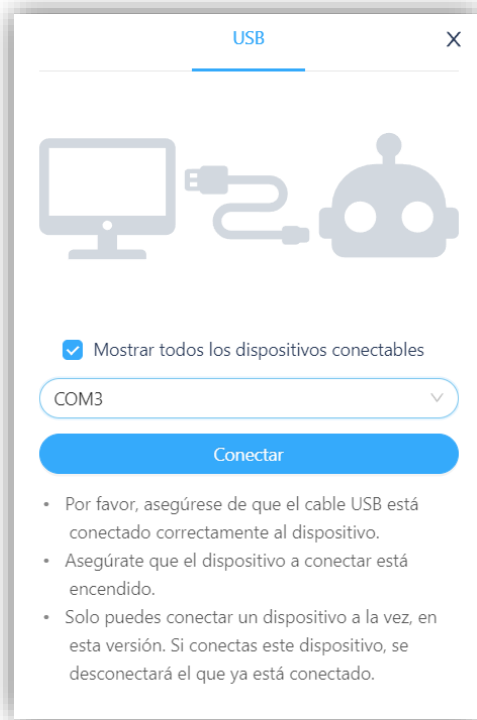
<https://mblock.makeblock.com/en-us/download/>

Tiene que estar abierto mientras realizamos la conexión.

Ahora seleccionaremos la pestaña dispositivos para realizar la conexión.

Si hay pendiente una actualización es conveniente actualizar.

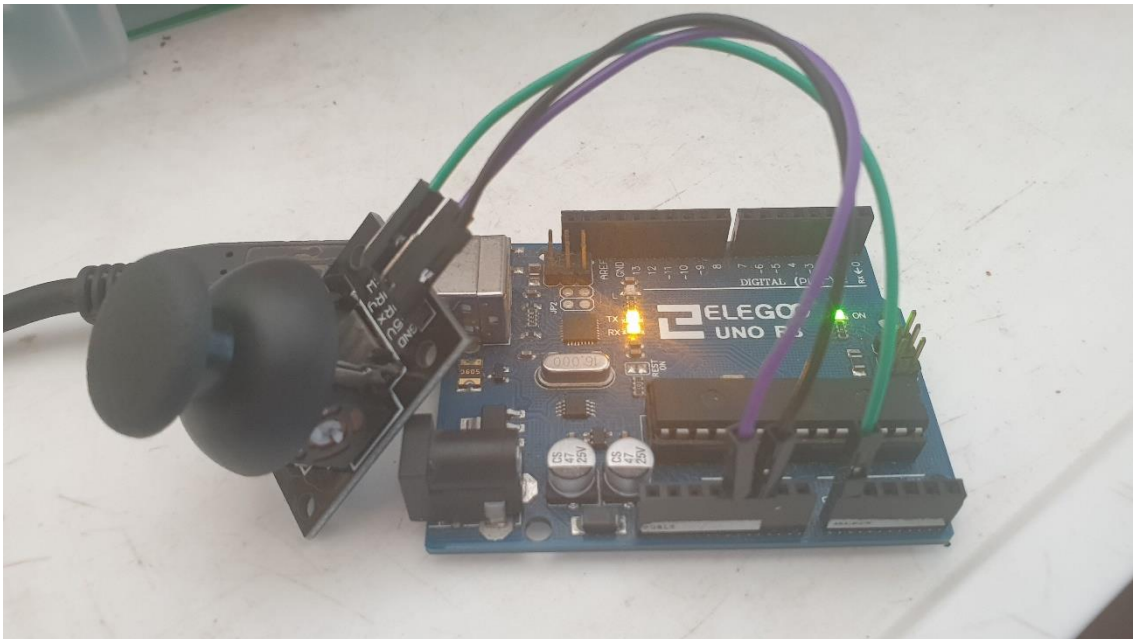
 Conectar



Seleccionamos el puerto, seguido de conectar.

Ya puedes mover tu joystick a izquierda y derecha.

PRUEBAS



El código final del Arduino.

```

cuando clic en [bandera]
para siempre
  fija x a lee pin analógico 0
  fija y a lee pin analógico 1
  fija a a lee pin analógico 2
  si x > 550 entonces
    envia derecha
  si x < 450 entonces
    envia izquierda
  si y > 550 entonces
    envia Arriba
  si y < 450 entonces
    envia Abajo
  si a = 0 entonces
    envia Saludo

```

Código del Oso panda.

```

al recibir derecha
  mueve 10 pasos
  apunta en dirección -90
al recibir izquierda
  mueve 10 pasos
  apunta en dirección 90
al recibir Abajo
  mueve 10 pasos
  apunta en dirección 0
al recibir Arriba
  mueve 10 pasos
  apunta en dirección 180
al recibir Saludo
  di ¡Hola! durante 2 segundos

```

21.- Bluetooth. COMUNICACIÓN

CONCEPTOS

¿Qué es Bluetooth?

Mucha gente puede tener la impresión de que el Bluetooth es una tecnología anticuada, que se usaba para transmitir datos entre dispositivos, y que actualmente está en desuso. Nada más lejos de la realidad.



Bluetooth tiene la **enorme ventaja de estar integrado de fábrica en la mayoría de dispositivos**. Portátiles, Tablets y Smartphones llevan integrado Bluetooth. Además, su uso es independiente del sistema operativo (Windows, Linux, Mac o Android).

Esto convierte a la tecnología Bluetooth en **uno de los mejores medios para comunicarnos de forma inalámbrica con Arduino**. Por ejemplo, podemos emplearlo para controlar un robot desde el móvil o Tablet, o recibir mediciones en un ordenador para registrarlas en un servidor web.

¿Cómo Funciona Bluetooth?

La comunicación Bluetooth es similar al uso del puerto serie normal, por tanto, resulta muy versátil y muy sencillo de usar. **La diferencia principal es que, en lugar de conectar un cable, tendremos que emparejar el módulo con nuestro dispositivo.**

EMPAREJAMIENTO MÓDULO BLUETOOTH

Normalmente utilizaremos el módulo Bluetooth del Arduino para conectarse con el Bluetooth del móvil a través de una App móvil. **ESTOS MÓDULOS SOLO PODRÁN CONECTARSE CON DISPOSITIVOS MÓVILES ANDROID.**

Para esto deberemos con anterioridad emparejar dicho módulo a nuestro dispositivo móvil, este proceso lo podremos hacer desde la App en concreto que vamos a utilizar o bien desde las opciones de configuración de Bluetooth de nuestro móvil, teniendo en cuenta la siguiente información:

1. Los módulos con los que vamos a trabajar vienen con el nombre HC-06 o nHC-06 dependiendo del modelo con el que trabajemos.
2. Debemos de localizar ese nombre del módulo en el móvil y emparejarlo.
3. Si nos solicita un PIN poner 1234.

CONEXIONADO CON ARDUINO

El conexionado será igual independiente del modelo de módulo que tengamos: HC-05 o HC-06, aunque uno tenga 4 y el otro 6 pines, porque en el de 6 solo utilizaremos los 4 pines centrales (descartando los 2 de los extremos).



- **PINES ALIMENTACIÓN.** Pines 5V y GND a los pines correspondientes de Arduino.
- **PINES DE DATOS. TXD Y RXD.** Estos pines irán a dos pines digitales de Arduino cualesquiera. Luego tendremos que indicar cada uno según corresponda en la programación de ArduinoBlocks.

Tipos de módulos Bluetooth

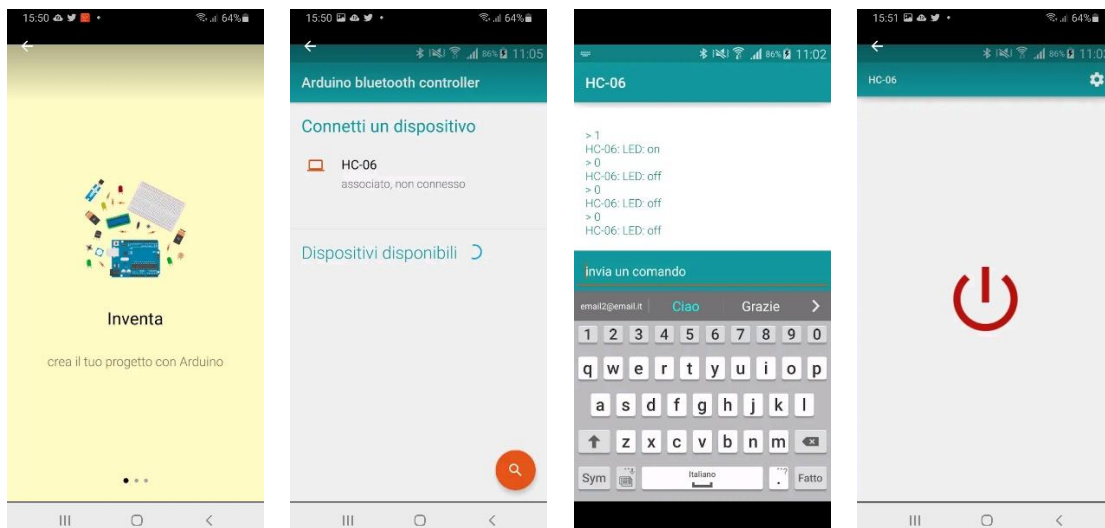
Los dos módulos, HC-05 y Hc-06, nos permiten conectar de forma sencilla un Arduino por Bluetooth. La diferencia entre ambos módulos es que el HC-06 sólo permite recibir comunicaciones (slave) mientras que el HC-05 puede recibirlas e iniciarlas (master and server).



Por tanto el módulo HC-05 es superior en características técnicas, aunque el HC-06 también nos servirá para nuestra práctica. Aunque como la diferencia de precio entre ambos es muy reducida si podemos comprar el modelo superior, pues mejor, por futuros usos.

Apps para conectar móvil a Bluetooth Arduino

Normalmente utilizaremos el módulo Bluetooth del Arduino para conectarse con el Bluetooth del móvil a través de una App móvil. Hay muchas Apps para poder hacer esto nosotros para nuestro curso utilizaremos la App Arduino Bluetooth Controller para Android.

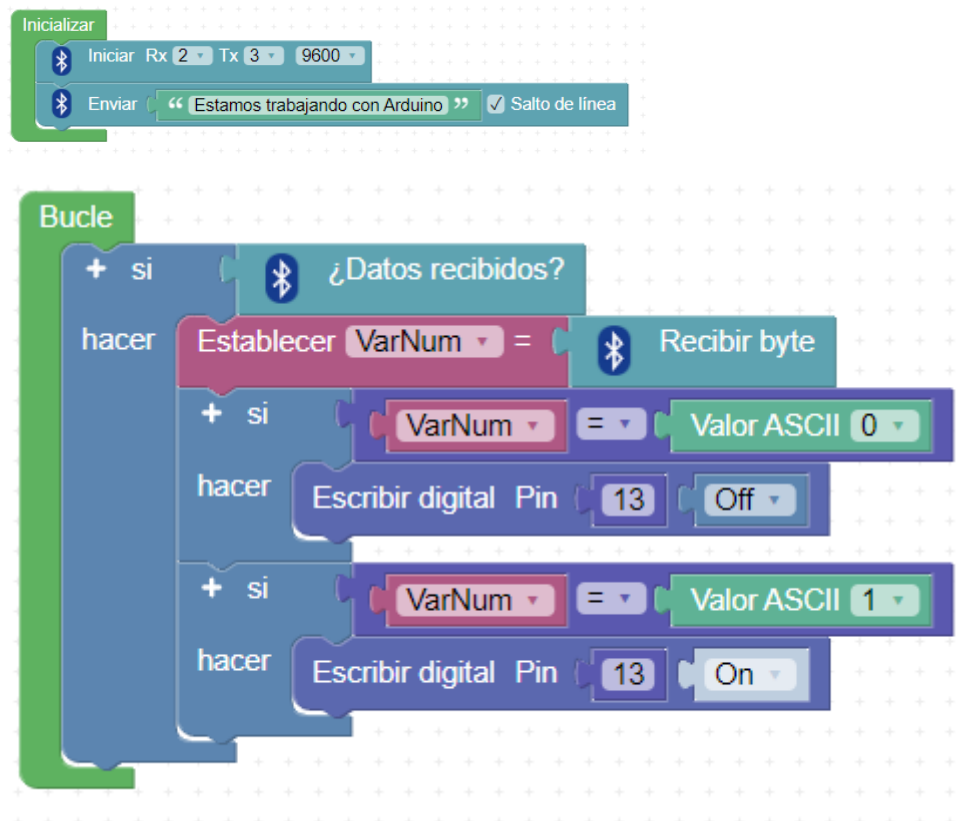


Con esta Aplicación podremos:

- **Modo CONTROLLER.** Para poder enviar el texto que queramos en función de los botones pulsados.
- **Modo SWITCH.** Para tener un único botón con dos posiciones.
- **Modo DIMMER.** Para enviar un número entre 0 y 255.
- **Modo TERMINAL.** Para enviar y recibir los caracteres que queramos.

PROGRAMACIÓN

Programación por bloques:



Programación por código:

```
#include <SoftwareSerial.h>
```

```
double VarNum;
```

```
SoftwareSerial bt_serial(3,2);
```

```
void fnc_dynamic_digitalWrite(int _pin, int _e){
    pinMode(_pin,OUTPUT);
    digitalWrite(_pin,_e);
}
```

```
void setup()
{
```

```

bt_serial.begin(9600);

    bt_serial.println(String("Estamos trabajando con Arduino"));

}

void loop()
{

    if ((bt_serial.available()>0)) {
        VarNum = bt_serial.read();
        if ((VarNum == ('0')) {
            fnc_dynamic_digitalWrite(13, LOW);
        }

        if ((VarNum == ('1')) {
            fnc_dynamic_digitalWrite(13, HIGH);
        }

    }

}

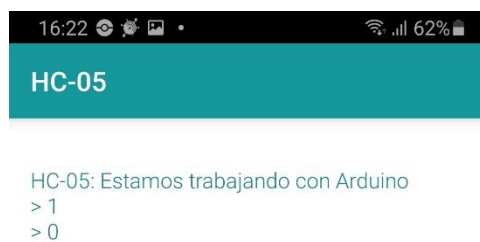
```

PRUEBAS

Una vez cargada la aplicación a nuestro Arduino ejecutaremos la aplicación que comentamos con anterioridad.

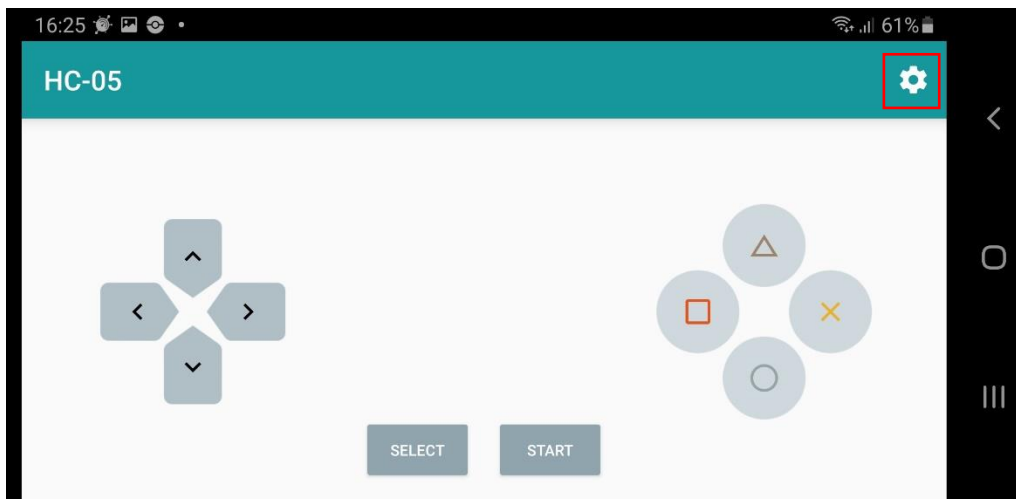
Seleccionamos el Bluetooth que ya tenemos emparejado.

Seleccionaremos "Terminal mode".



El Arduino nos ha enviado un mensaje, nosotros al enviarle un 1 enciende el led del pin 13 y al enviar un 0 apaga el led del pin 13.

Ahora vamos a seleccionar la opción "Controler mode"



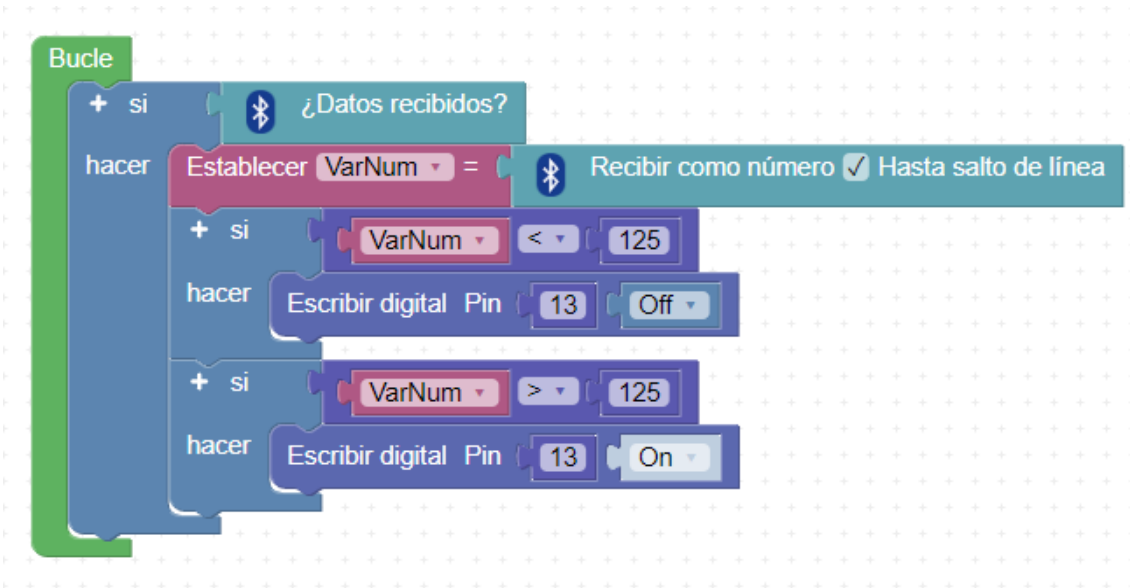
Vamos a configurar que con la x mandamos un 0 y con el círculo mandamos un 1.

Con la opción "Switch mode" un solo botón.

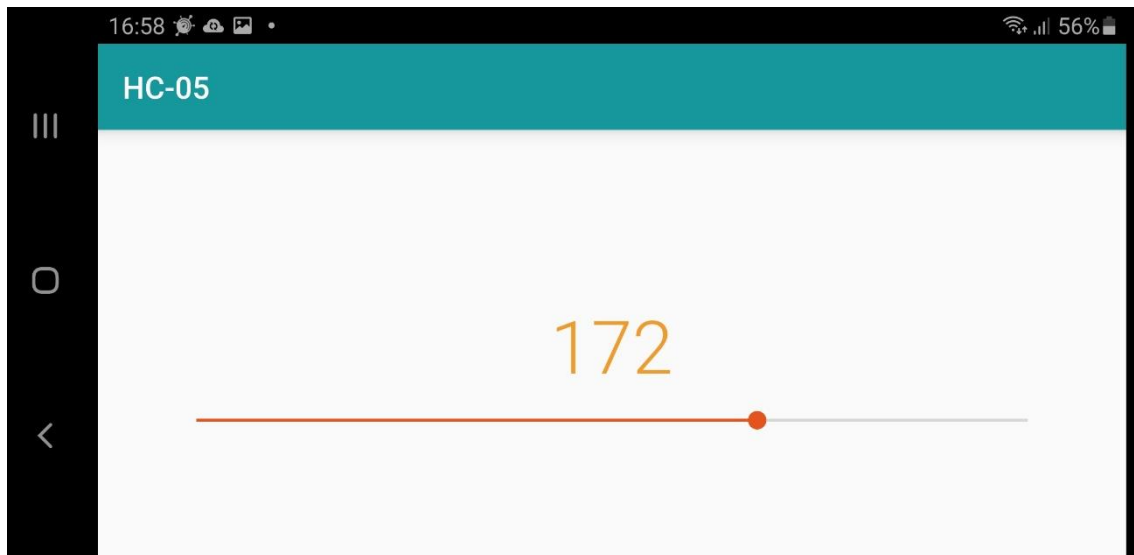


Cuando encendemos y apagamos le decimos en configuración que es lo que queremos enviar.

Para el siguiente ejemplo vamos a cambiar a recibir un número.



Ahora seleccionaremos la opción “Dimmer mode”.



Según donde este la barra su valor el led estará encendido o no.

Para acceder a los videotutoriales del canal de YouTube Aprendemanía.



Contenido

1.- ¿Cómo usar ARDUINO?	1
SW para programar Arduino	1
ArduinoBlocks	2
Registro nuevo Usuario	3
Empezar un nuevo Proyecto	3
ArduinoBlocks Conector	3
Nuestro primer proyecto	5
2.- ¿Qué es la Robótica?	10
¿Qué disciplinas intervienen en la Robótica?	10
¿Cómo funciona un Robot?	11
Objetivo de la Robótica	11
Futuro / Peligros Robótica	12
3.- ¿Qué es Arduino?	14
Elementos que componen la Plataforma Arduino	14
HARDWARE	14
SOFTWARE	15
COMUNIDAD	16
Modelos Arduino	16
Placas Oficiales VS No Oficiales	17
4.- ¿Qué kit de Arduino Comprar?	18

Orientaciones: la decisión es vuestra.....	18
Compatibles VS Original.....	18
¿Dónde Comprar?	19
Dispositivos Recomendados.....	19
5.- Introducción Electricidad	24
¿Qué es la corriente Eléctrica?.....	24
Magnitudes Eléctricas	24
Corriente Continua Vs Alterna	25
Circuito Eléctrico	25
Ley de Ohm	28
Primer circuito Eléctrico con Arduino	28
6.- Placa Arduino Uno.....	32
PINES	32
Otros Componentes de la Placa	33
Características Técnicas	34
7.- Encender un Led. SALIDAS DIGITALES.....	35
CONCEPTOS.....	35
¿Qué es un Led?	35
¿Cómo Funciona un Led?	35
Tipos de Leds	35
CIRCUJITO ELECTRÓNICO	36
PROGRAMACIÓN.....	36
PRUEBAS.....	37
8.- Led en Serie VS Paralelo.....	38
CONCEPTOS.....	38
Conectar varios Leds a una Salida Digital.....	38
Opciones: Series VS Paralelo	38
CIRCUITO ELECTRÓNICO	38
PRUEBAS.....	39
9.- Led intensidades. SALIDAS ANALÓGICAS PWM	40
CONCEPTOS.....	40
¿Salida Analógica?.....	40
¿Qué es PWM?	40
CIRCUITO ELECTRÓNICO	40
PROGRAMACIÓN.....	41
PRUEBAS.....	41

10.- Led RGB. SALIDAS DIGITALES Y ANALÓGICAS	42
CONCEPTOS.....	42
¿Qué es un Led RGB?	42
¿Cómo funciona en Led RGB?	42
¿Cómo obtener diferentes colores?.....	42
Tipos de Leds RGB	43
CIRCUITO ELECTRÓNICO.....	44
PROGRAMACIÓN.....	44
PRUEBAS.....	46
11.- Pulsador interruptor. ENTRADAS DIGITALES	47
CONCEPTOS.....	47
¿Qué es un Pulsador?.....	47
¿Cómo Funciona un Pulsador?.....	47
Entradas Digitales: ejemplo con un Pulsador.....	48
Montaje Pull-Up on Pull-Down	48
Tipos de Pulsadores.....	49
CIRCUITO ELECTRÓNICO	50
PROGRAMACIÓN.....	50
PRUEBAS.....	51
12.- Potenciómetro. ENTRADAS ANALÓGICAS.....	52
CONCEPTOS.....	52
¿Qué es un Potenciómetro?.....	52
¿Cómo Funciona un Potenciómetro?.....	52
Entradas Analógicas	52
Tipos de Potenciómetros	52
CIRCUITO ELECTRÓNICO	53
PROGRAMACIÓN.....	53
PRUEBAS	54
13.- Comunicación Serie Consola	55
CONCEPTOS.....	55
¿Qué es la Comunicación Serie?	55
¿Qué es la Consola de Arduino?.....	55
¿Cómo usar la Consola de Arduino?	55
Utilizar la consola para encender/apagar Led.....	56
PROGRAMACIÓN.....	56
PRUEBAS.....	57

14.- Sensor Ultrasonidos. SENSORES	58
CONCEPTOS.....	58
¿Qué es un Sensor de distancia por Ultrasonidos?.....	58
¿Cómo Funciona un Sensor de Ultrasonidos?.....	58
Tipo de Sensores de Ultrasonidos.....	58
CIRCUITO ELECTRÓNICO	59
PROGRAMACIÓN.....	59
PRUEBAS.....	60
15.- Sensor Temperatura y Humedad. SENSORES	61
CONCEPTOS.....	61
¿Qué es un sensor de Temperatura y Humedad?.....	61
¿Cómo Funciona?	61
Tipo de Sensores de Temperatura y Humedad.....	61
PROGRAMACIÓN.....	61
PRUEBAS.....	62
16.- Servo Motor. MOTORES.....	63
CONCEPTOS.....	63
¿Qué es un ServoMotor?	63
¿Cómo Funciona un ServoMotor?	63
CIRCUITO ELÉCTRICO.....	63
PROGRAMACIÓN.....	64
PRUEBAS.....	65
17.- Pantalla LCD. PANTALLAS.....	66
CONCEPTOS.....	66
¿Qué es una Pantalla LCD?.....	66
Tipo de Pantallas	66
¿Cómo Funciona una Pantalla LCD?	67
CIRCUITO ELECTRÓNICO	68
PROGRAMACIÓN.....	68
PRUEBAS.....	69
18.- Mando Infrarrojos IR.MANDOS.....	70
CONCEPTOS.....	70
¿Qué es Mando/Sensor de InfraRojos?	70
¿Cómo Funciona un Mando/Sensor IR?.....	70
CIRCUITO ELECTRÓNICO	71
PROGRAMACIÓN.....	71

PRUEBAS.....	73
19.- Buzzer Pasivo Altavoz. Sonido.....	75
CONCEPTOS.....	75
¿Qué es un Buzzer Pasivo/Altavoz?	75
¿Cómo Funciona un Buzzer Pasivo/Altavoz?	75
Tipos de Buzzers.....	75
CIRCUITO ELECTRÓNICO	76
PROGRAMACIÓN.....	76
PREUBAS.....	77
20.- Joystick POSICIÓN	79
CONCEPTOS.....	79
¿Qué es un Joystick?	79
¿Cómo Funciona un joystick?.....	79
Práctica a desarrollar.....	79
PROGRAMACIÓN con mBlock	79
PRUEBAS.....	83
21.- Bluetooth. COMUNICACIÓN	85
CONCEPTOS.....	85
¿Qué es Bluetooth?.....	85
¿Cómo Funciona Bluetooth?.....	85
Tipos de módulos Bluetooth	86
Apps para conectar móvil a Bluetooth Arduino	86
PROGRAMACIÓN.....	87
PRUEBAS.....	88