

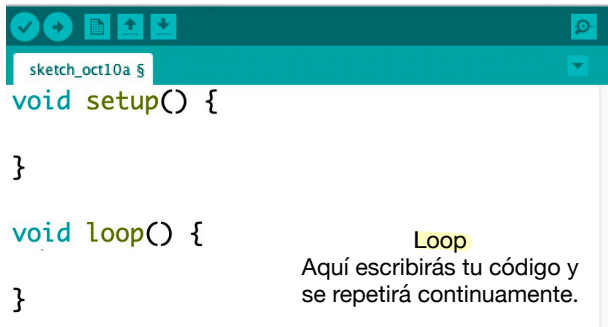
## Práctica 1 ( Múltiples LEDS y mi Arduino)

### Objetivo

El alumno será capaz de conocer nuevos componentes en su Arduino con la ayuda de prácticas sencillas que lo llevarán a tener un mejor entendimiento de que es y cómo funciona esta herramienta.

### Ambiente de Desarrollo

El Desarrollo de Arduino se compone de dos bloques principales:



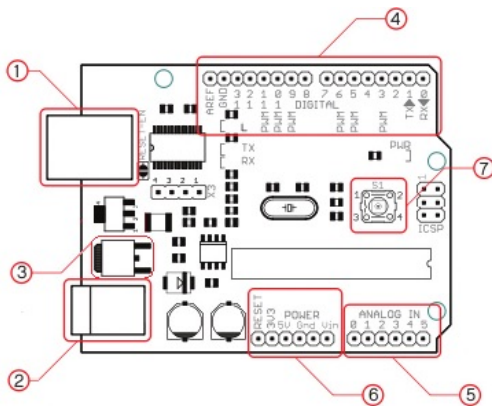
#### Setup code

Este bloque de código solo se ejecuta una sola ocasión.

#### Loop

Aquí escribirás tu código y se repetirá continuamente.



Es importante diferenciar cada uno de los pines de nuestro Arduino, para poder configurarlos de manera adecuada:



1. **Puerto USB.**
2. **Fuente de alimentación.**
3. **Regulador de voltaje.**
4. **Pines Digitales.**
5. **Pines Analógicos.**
6. **Pines auxiliares de alimentación.**
7. **Botón de Reset.**

### Configurando los pines

Existen diferentes tipos de Pines en nuestro Arduino:

-  **Pines Salida / Entrada Digital.**
-  **Pines Analógicos**

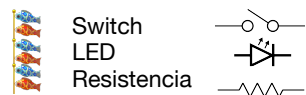
En esta práctica utilizaremos la configuración Pines Digitales ya que es la que utilizaremos como salida para encender nuestro LED.

**Nota:** En prácticas posteriores iremos utilizando cada una de las configuraciones, al final de estas 8 prácticas ustedes serán capaces de comprender como, cuando y donde pueden utilizar cada una de ellas.

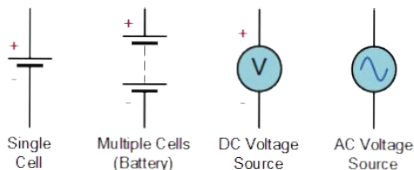
# Material

## 1 Arduino UNO

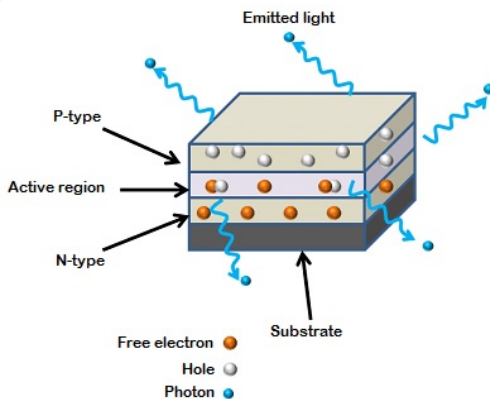
### Símbolos Básicos



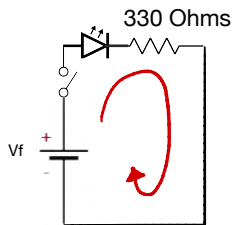
### Fuentes de alimentación



### ¿ Como funciona un LED ?



### Circuito



$$V_f - V_d - V_r = 0$$

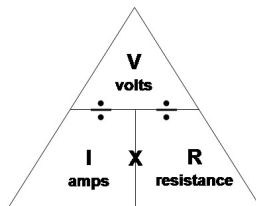
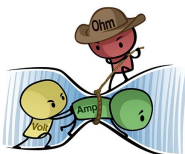
$$5V - 1.2V - V_r = 0$$

$$V_r = 3.8V$$

$$I_r = V_r / R = 3.8V / 330 \\ = .011 \text{ Amp}$$

$$I_r = V_r / R = 3.8V / 220 = \\ .017 \text{ Amp}$$

### Ley de Ohm



## Ejercicio 1

Con ayuda de tu profesor:

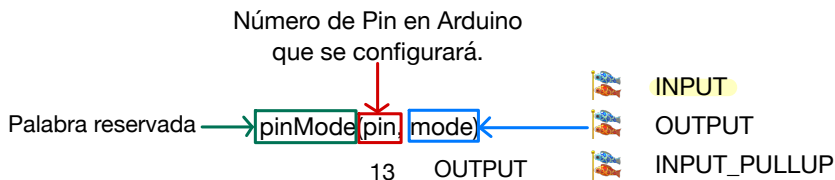
1. Configuren el Pin número 13 en modo digital .
2. Mediante código enciéndalo.

### Objetivo

Al terminar este pequeño código el alumno será capaz de verificar si su código es correcto, conectar su Arduino en su ordenador, subir el código en su tarjeta y observar visualmente que todo está funcionando de manera adecuada.

### pinMode()

Configura el comportamiento de un pin en específico, ya sea como salida o entrada digital.



### digitalWrite()

Escribe en HIGH o LOW en un Pin digital.

Si el Pin es configurado como OUTPUT, el voltaje es configurado de la siguiente manera:

**HIGH** - 5V (3.3V para tarjetas 3.3V) [1 lógico]

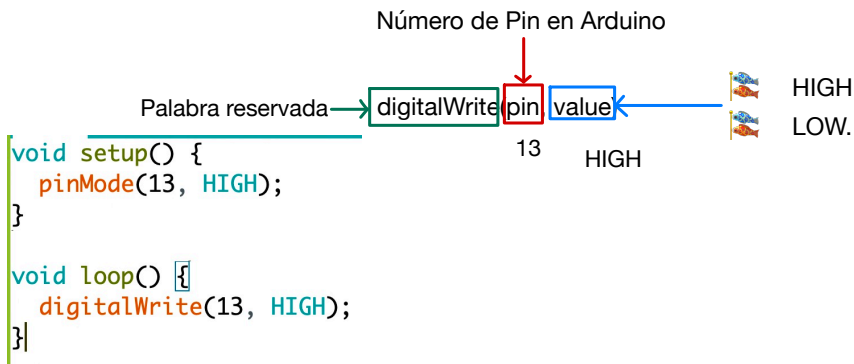
**LOW** - 0V (ground) [0 lógico]

Si el Pin es configurado como INPUT,

**HIGH** - Activa el Pull-up Interno en el pin.

**LOW** - Desactiva el Pull-up interno en el pin.

Aunque si se desea activar o desactivar las resistencias de Pull-up es recomendable utilizar el `pinMode()` en `INPUT_PULLUP` para activar el internal pull-up resistor.



**Nota:** Es importante configurar el `pinMode()` antes de utilizar este método ya que puede generar un efecto dimming en nuestra salida.

## Ejercicio 2

Nota: Existen 1000 milisegundos en un segundo.

Con ayuda de tu profesor:

1. Configuren el Pin número 13 en modo digital.
2. Mediante código enciéndalo y apágalo.

Palabra reservada → delay(ms)

delay()

Pause el programa por un periodo de tiempo (milisegundos).

Numero de milisegundos a  
parar( tipo de datos: unsigned long)

```
void setup() {  
  pinMode(13, HIGH);  
}  
  
void loop() {  
  digitalWrite(13, HIGH);  
  delay(1000);  
  digitalWrite(13, LOW);  
  delay(1000);  
}
```

### Ejercicio 3

Con ayuda de tu profesor:


1. Configuren el Pin número 13 en modo digital.
2. Configura una variable, la cual será la que defina el estado de encendido y apagado del led.
3. Mediante código enciéndalo y apágalo.


### Variable

Una variable es una forma de nombrar y almacenar un valor, para ser utilizado posteriormente en el programa.

### Tipos de datos en Arduino


 **Boolean** (8 bit) - Lógico simple verdadero/falso.


 **Byte** (8 bit) - Número sin signo entre 0 y 255.

 **Char / Unsignedchar** (8 bit) - Número con signo, entre -128 y 127.

**Nota:** Es lo mismo que 'byte'; si es que eso es lo que necesitas, deberías usar 'byte', para que el código sea más claro.

**Nota:** En algunos casos el compilador intentará interpretar este tipo de dato como un caracter, lo que puede generar resultados inesperados.


 **Word / Unsignedint** (16 bit) - Número sin signo entre, 0 y 65535.


 **Int** (16 bit) - Número con signo, entre -32768 y 32767.

**Nota:** Este tipo es el más usado para variables de propósito general en Arduino, en los códigos de ejemplo que vienen con el IDE.

 **Unsignedlong** (32 bit) - Número sin signo entre 0 y 4294967295.

**Nota:** Este tipo se usa comúnmente para almacenar el resultado de la función millis(), la cual retorna el tiempo que el código actual ha estado corriendo, en milisegundos.

 **Long** (32 bit)- Número con signo, entre -2,147,483,648 y 2,147,483,647.

 **Float** (32 bit)- Número con signo, entre 3.4028235E38 y 3.4028235E38.

**Nota:** El Punto Flotante no es un tipo nativo en Arduino; el compilador debe realizar varios saltos para poder hacerlo funcionar. Evítalo siempre que te sea posible.

Tipo de datos → `int inputVariable1;`  
`int inputVariable2 = 0;` ← Valor asignado en la variable.

```
void setup() {  
  pinMode(13, HIGH);  
}  
  
void loop() {  
  int variable = 1;  
  int variable1 = 13;  
  digitalWrite(variable1, variable);  
  delay(1000);  
  digitalWrite(13, LOW);  
  delay(1000);  
}
```

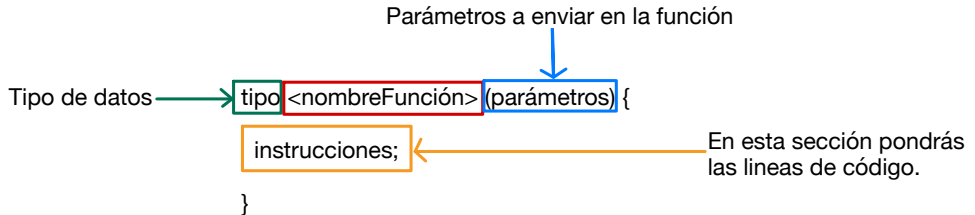
#### Ejercicio 4

Con ayuda de tu profesor:

1. Configuren el Pin número 13 en modo digital.
2. Crea una función en donde una variable, definirá el estado de encendido y apagado del led.

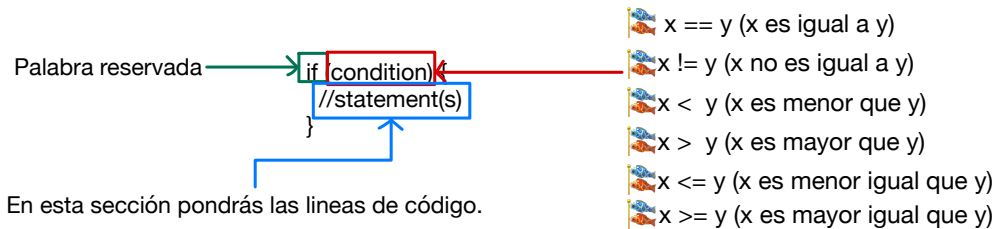
#### Función

Una función es un bloque de código que tiene un nombre y un conjunto de instrucciones que son ejecutadas cuando se llama a la función



#### if

Checa si la condicional es cumplida y ejecuta una o varias líneas de código.



#### !(Logical NOT)

Resulta en verdadero si el operando es falso, y viceversa.

$x = !y;$

```
void setup() {  
    pinMode(13, HIGH);  
}  
  
int funcion(int variable){  
    return !variable;  
}  
  
void loop() {  
    int variable = 1;  
    int variable1 = 13;  
  
    digitalWrite(variable1, variable);  
    delay(1000);  
    digitalWrite(13, funcion(variable));  
    delay(1000);  
}
```