



# ARDUINO en la Enseñanza de la ELECTRÓNICA, PROGRAMACIÓN, TECNOLOGÍA

## Introducción

Sesión realizada por: Ángela Diez Diez  
Dpto: Ingeniería Eléctrica y de Sistemas y Automática

UNIVERSIDAD DE LEÓN





# Objetivos

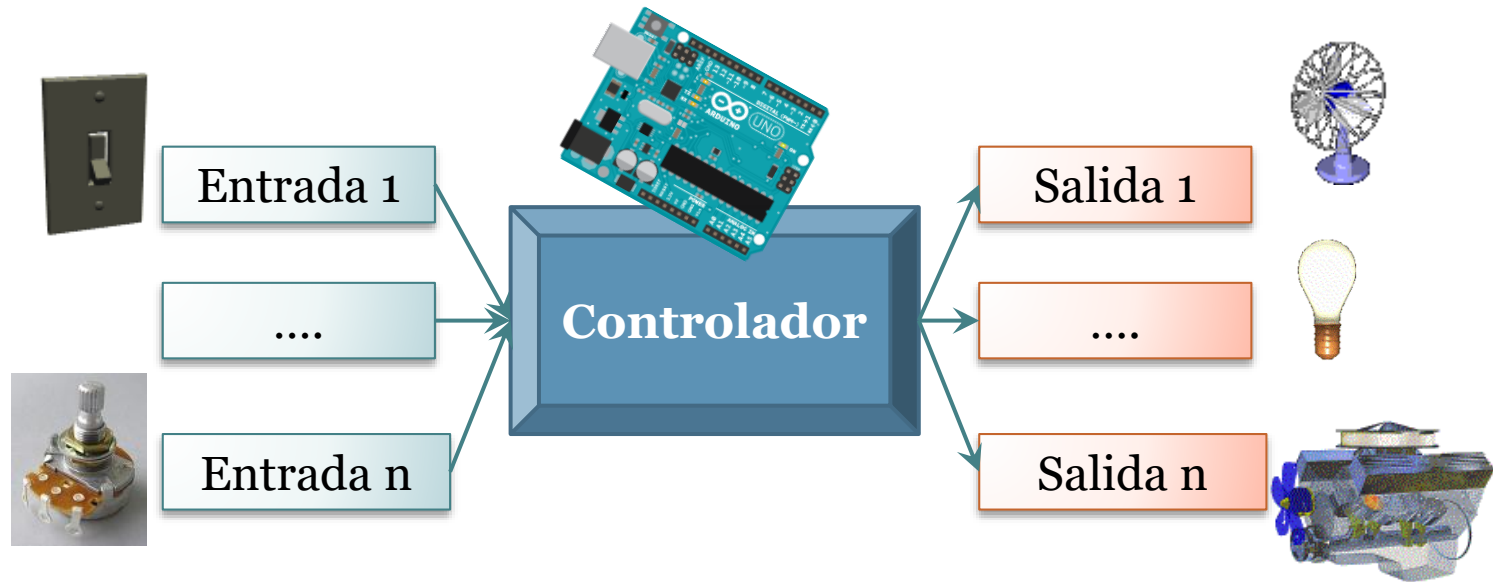
- Dar a conocer a los alumnos de Tecnología de Primaria de una herramienta Open Source.
- Aprende a :
  - **P**ensar
  - **A**nalizar,
  - **P**robar tus ideas
- Te mostramos una herramienta.
- Pero, recuerda hay otras muchas que puedes usar.



# Arduino ¿Qué es?

- Dispositivo
  - Tarjeta
  - Controlador
  - Placa
- Es un **elemento programable** que posee entradas y salidas digitales, analógicas.
  - Nos permite realizar pequeños proyectos domésticos tomando como base la electrónica, la robótica y la automática.

# Controladoras: Elementos



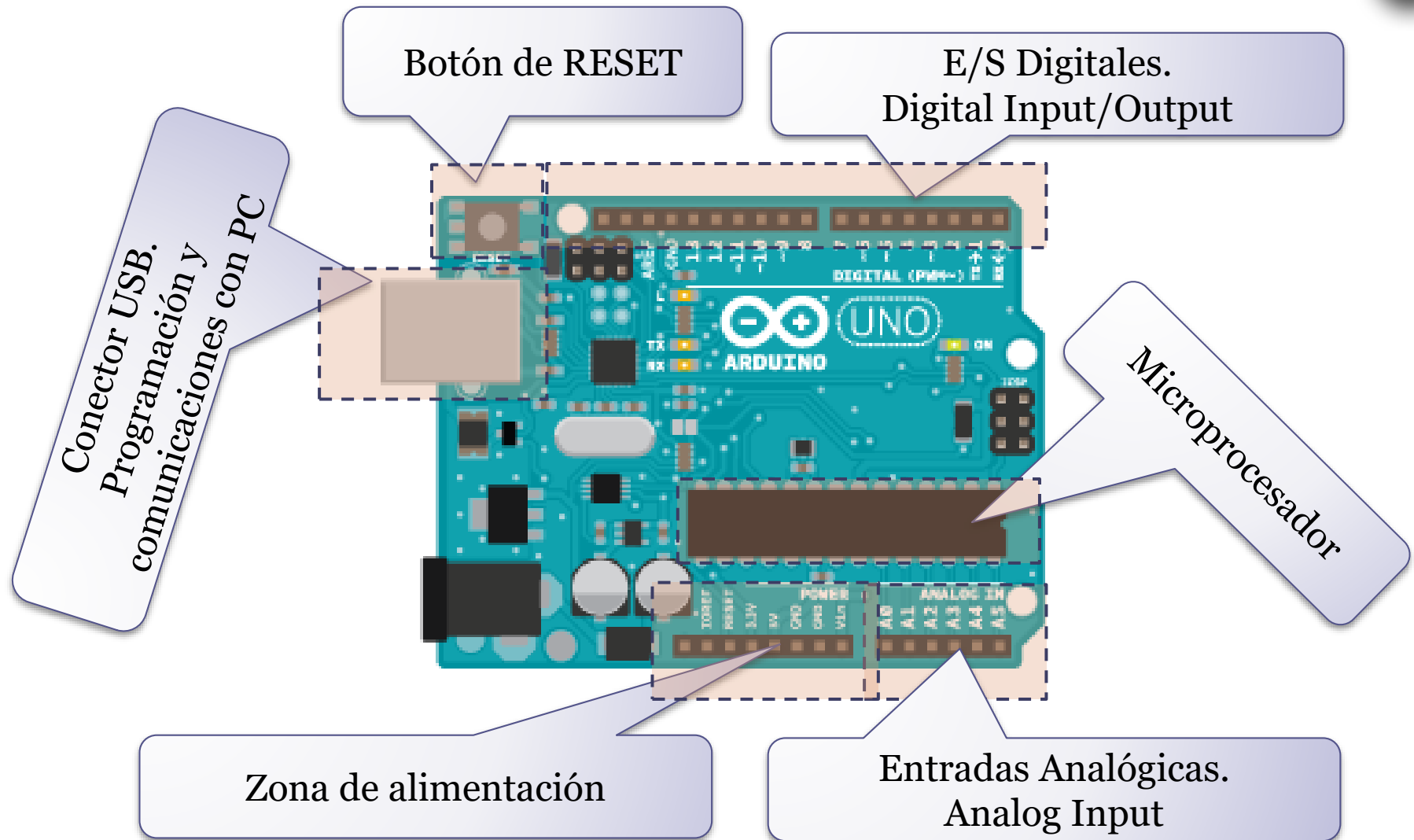
- **Señales entrada**

- Pulsadores
- Interruptores
- Sensores luz, ...

- **Señales de Salidas**

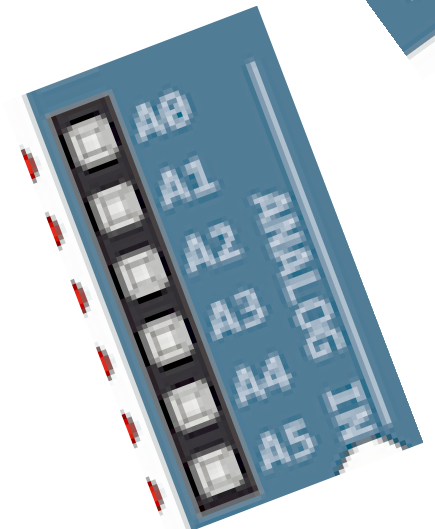
- Luces
- Sirenas
- Motores, ....

# Arduino: HARDWARE



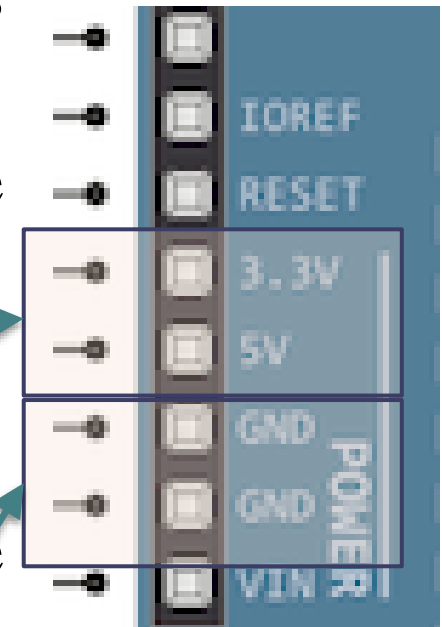
# ARDUINO: HARDWARE

- Pines digitales:
  - Se **configuran: INPUT** (Entrada), **OUTPUT** (Salida).
  - Tienen un número Identificativo: 13, 12, ,,,,2. El 0 y el 1 lo utiliza internamente para comunicarse con el PC.
- Pines analógicos
  - Son todas Entradas
  - Tienen un número que las identifica.



# ARDUINO: HARDWARE

- Arduino permite alimentar algunos dispositivos o sensores simples.
- Las tensiones eléctricas que me proporcionan son:
  - 5V
  - 3,3V
- También me da líneas de masa o de referencia (GND) para los circuitos.

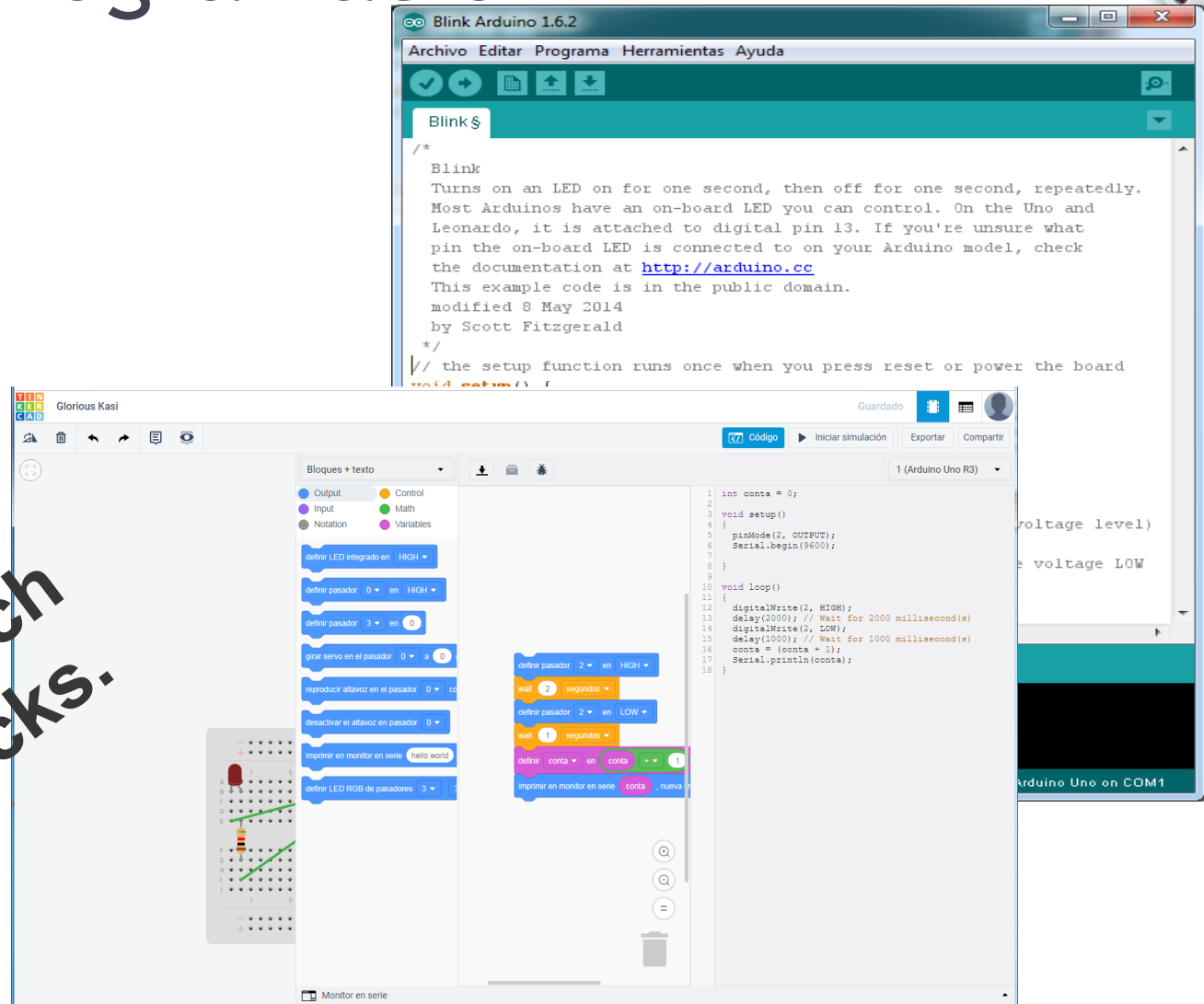




# Entornos programación

- TEXTUAL
- GRÁFICOS
- BLOQUES

Scratch  
Blocks.

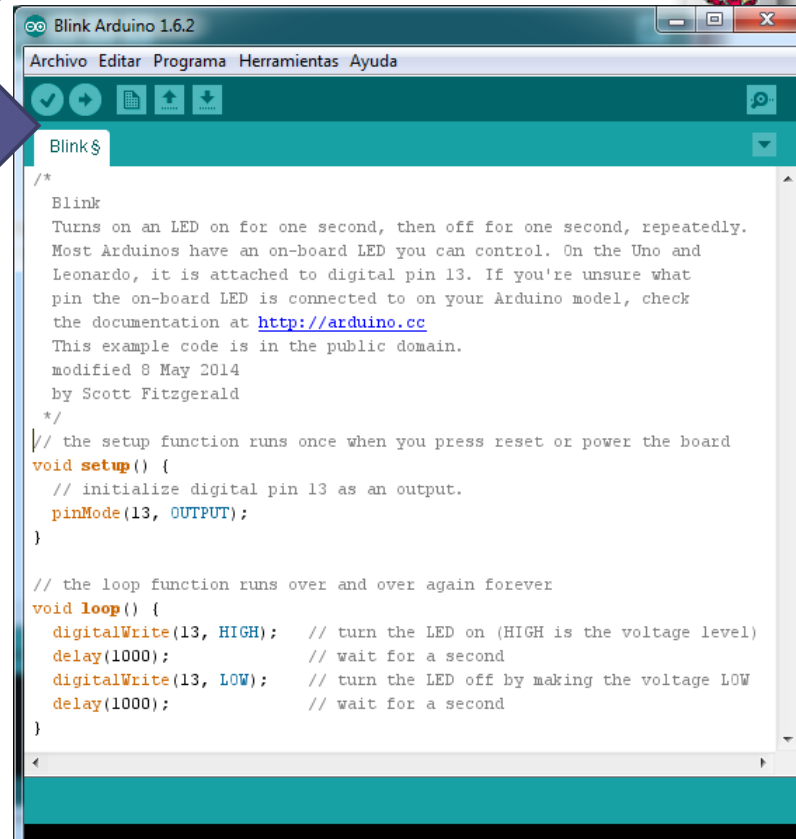
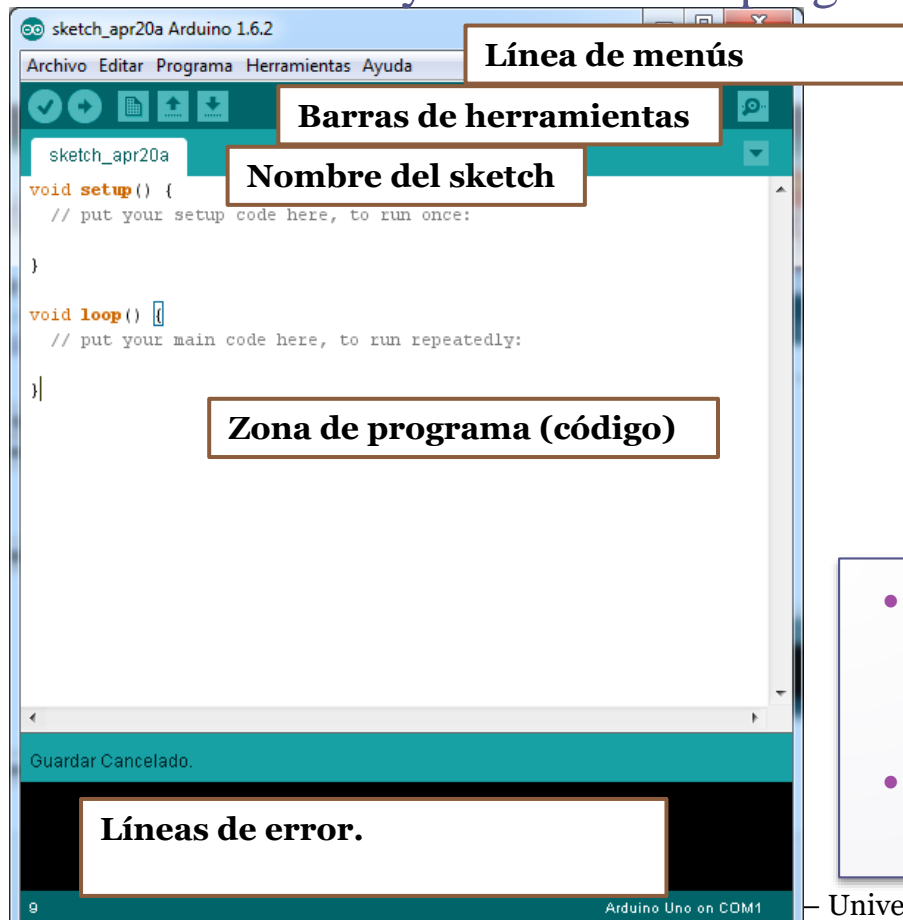




# Arduino: Entorno programación



- IDE de programación.
  - IDE=Entorno de desarrollo integrado
- Editor y el entorno de programación



- Un **sketch** (esquema) es un archivo o proyecto que se crea dentro del entorno IDE del Arduino.
- La primera vez que abrimos un nuevo proyecto aparecerá en blanco.



# ARDUINO: Software: barra de herramientas



New. Crea un sketch nuevo, en blanco.



VERIFICAR. Comprueba si el código es correcto.



Open: Abrir archivo



Save Crea/Salva el archivo. La extensión es .pde // .ino.



Subir: Descarga el programa en el controlador y se ejecuta.



# Nueva versión IDE v2

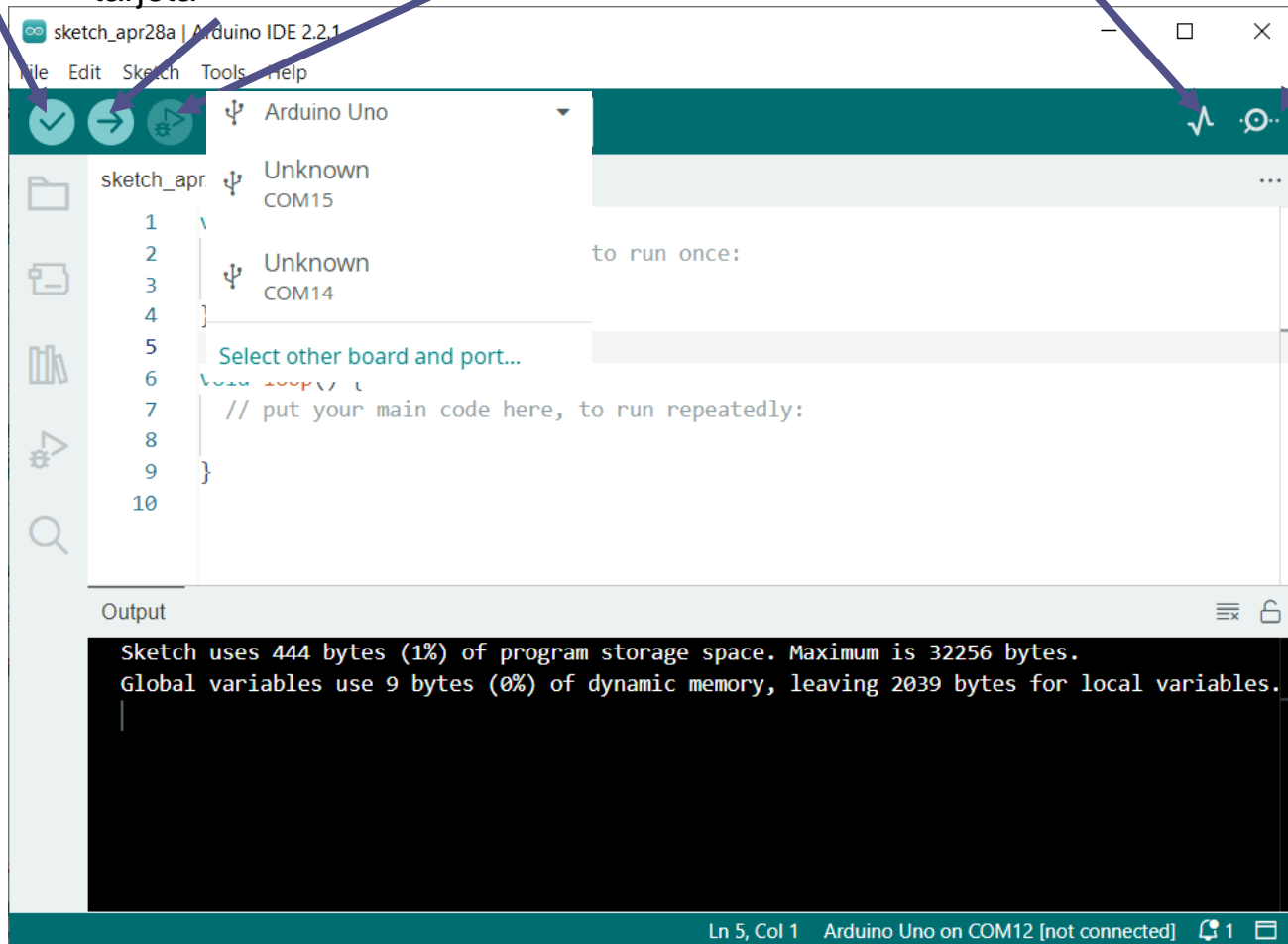
Verifica el programa

Descarga el programa en la tarjeta

Depura el programa

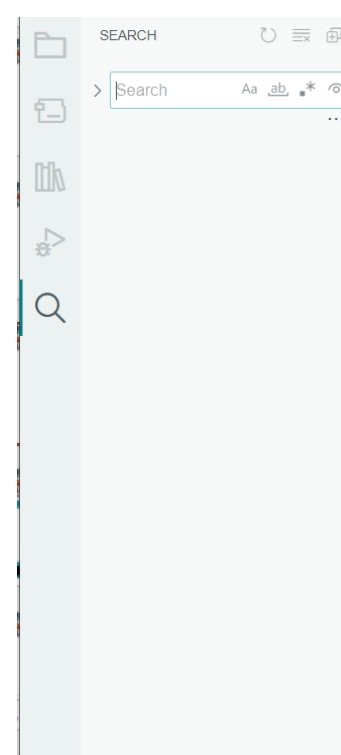
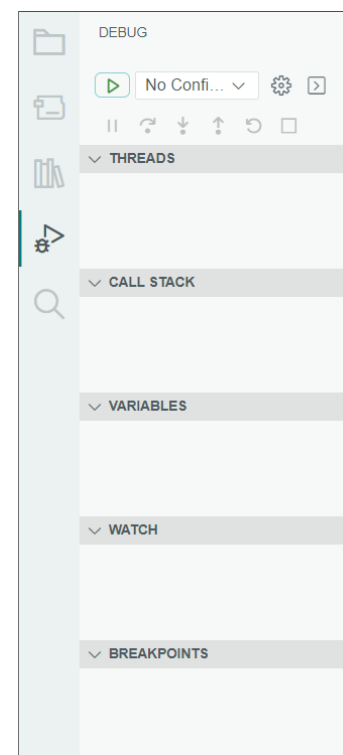
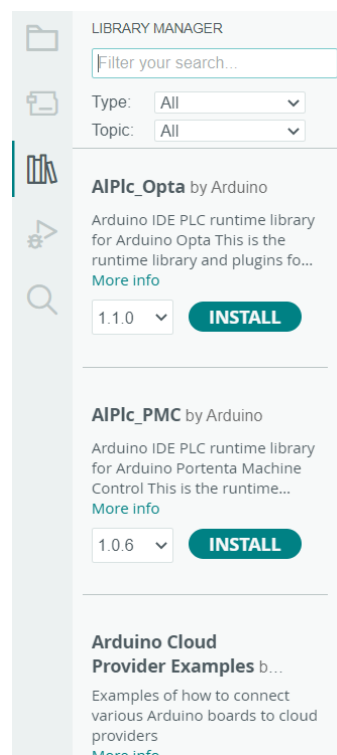
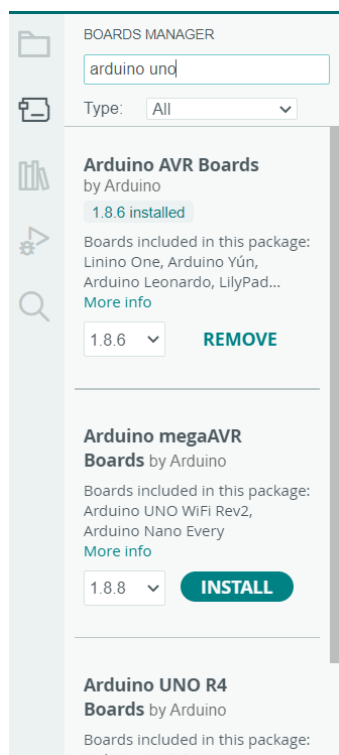
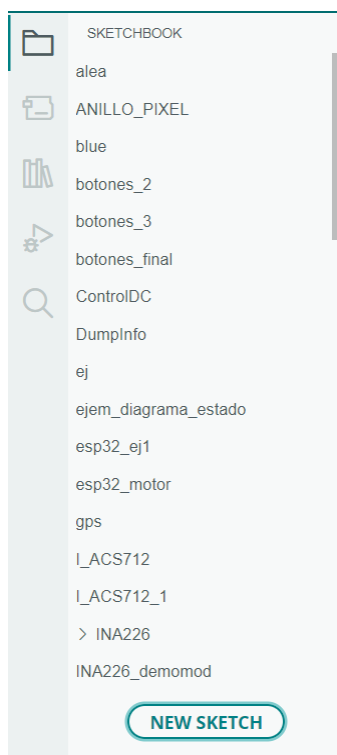
Plotter serie

Monitor serie





# Opciones del IDE v2



# ARDUINO SOFTWARE

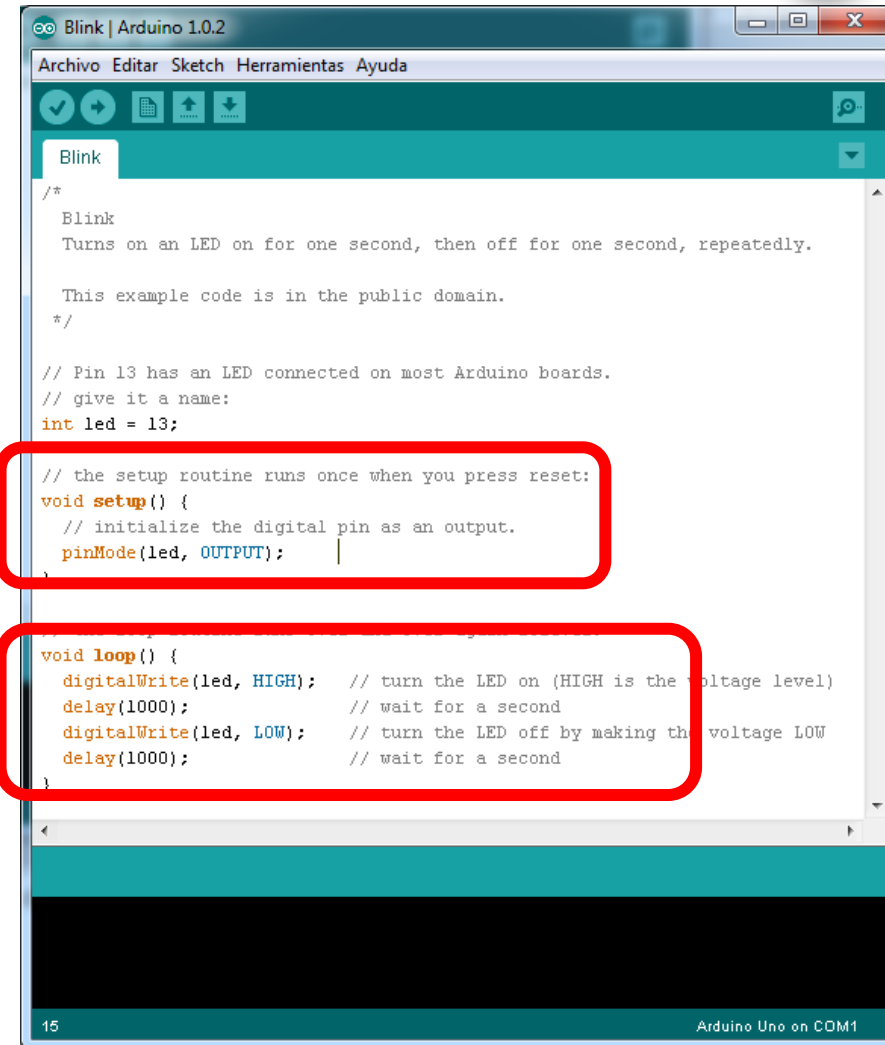
## Routines

Cada programa Arduino se suele denominar **SKETCH** y tiene dos funciones **OBLIGATORIAS**, denominadas **Rutinas**.

***void setup ( ) { }*** - Todo el código interno dentro de los corchetes se ejecuta **UNA ÚNICA VEZ** cuando se carga inicia el controlador. **Permite configurar e inicializar nuestro sistema.**

***void loop ( ) { }*** - Esta función se ejecuta **DESPUÉS** que la función setup ha finalizado.

El código dentro de los corchetes se **ejecutan de forma continua una y otra vez** hasta que se elimine la alimentación de la tarjeta.



```
Arduino IDE - Blink | Arduino 1.0.2
Archivo  Editar  Sketch  Herramientas  Ayuda

Blink

/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
  */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}

15 Arduino Uno on COM1
```



# BLOQUES

- Lenguajes están basados en una lógica de bloques donde el color identifica el tipo de bloque, y la forma permite que encajen entre sí.
- Permiten completar una secuencia de operaciones.
- Orientado a programación secuencial.

La imagen muestra una colección de bloques de programación visual, organizados por color y función. Los bloques están diseñados para encajar entre sí, formando una secuencia de operaciones.

- Bloques de Salida (Output):** Representados por bloques azules. Ejemplos: "definir LED integrado en HIGH", "definir pasador 0 en HIGH", "definir pasador 3 en 0", "girar servo en el pasador 0 a 0", "reproducir altavoz en el pasador 0", "desactivar el altavoz en pasador 0", "Imprimir en monitor en serie hello world", "definir LED RGB de pasadores 3".
- Bloques de Entrada (Input):** Representados por bloques púrpuras. Ejemplos: "leer pasador digital 0", "leer pasador analógico A0", "leer grados de servo en el pasador 0", "número de caracteres de serie disponibles", "leer de la serie", "leer el sensor de distancia ultrasónico en el pasador 0", "leer el sensor de temperatura en el pasador 0".
- Bloques de Control:** Representados por bloques naranjos. Ejemplos: "wait 1 segundos", "repeat 10 times", "repeat while", "if then", "if then else", "count up by 1 for 1 from".
- Bloques de Matemáticas (Math):** Representados por bloques verdes. Ejemplos: "1 + 1", "1 < 1", "seleccionar de forma aleatoria entre 1", "and", "no", "abs de 0", "asignar 0 al rango entre 0 y 18", "restringir 0 al rango entre 0 y 2", "HIGH".
- Bloques de Notación (Notation):** Representados por bloques grises. Ejemplos: "comentario de bloque de título describe su", "comentario útil comentario de una sola línea".
- Bloques de Variables (Variables):** Representados por bloques magenta. Ejemplos: "Create variable...", "conta", "definir conta en 0", "cambiar conta por 0".



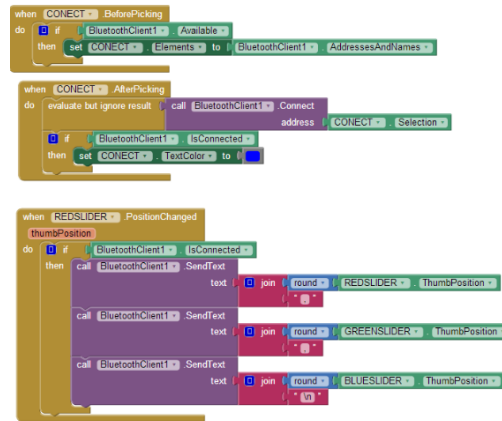
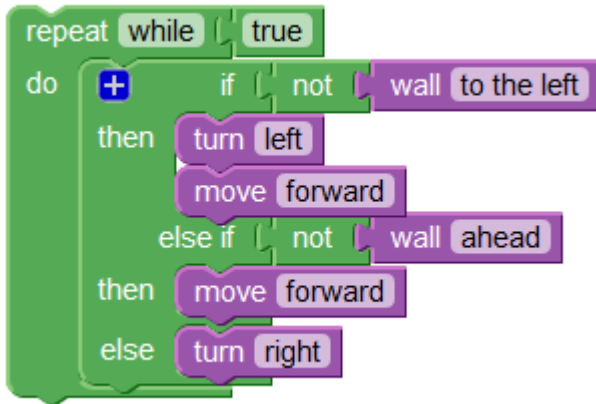
# Secuencia gráfica

- Tipo scratch

App Inventor

MIT  
App Inventor

Tinkercad



App Inventor Blocks Editor



# ARDUINO

MI PRIMER EJEMPLO





# ¿Qué QUIERO HACER?

- Necesitamos saber describir lo que queremos hacer.
- No te preocupes. Es un ejercicio de redacción muy simple.

## EJEMPLO:

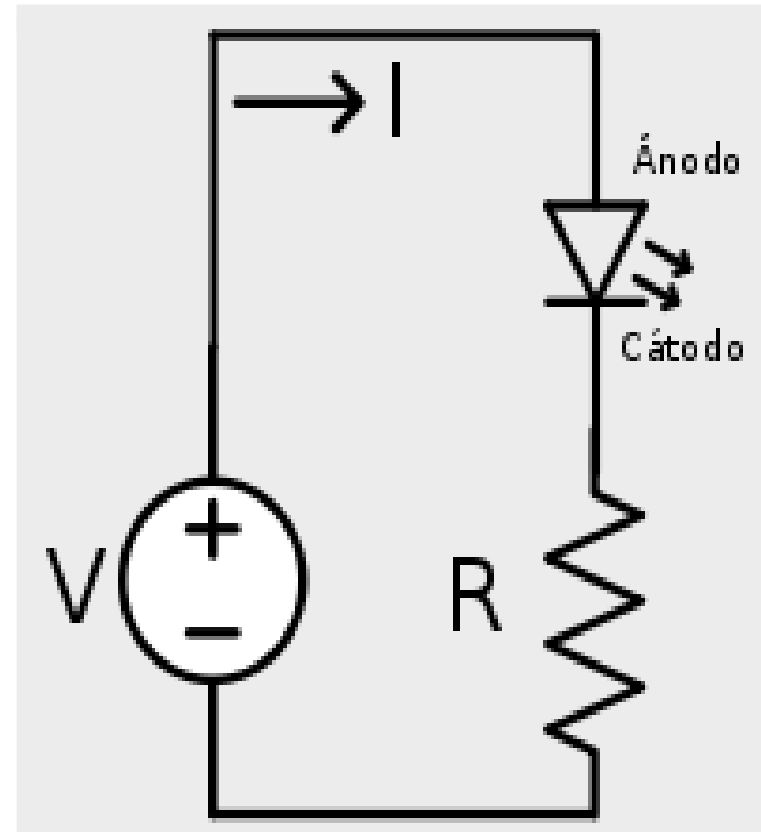
Necesitamos hacer **parpadear** una pequeña **luz** durante un determinado **tiempo**.

# ¿Qué DISPOSITIVOS o elementos NECESITO?

- Podemos usar un LED, como fuente de luz.
- Necesitamos algo que cuente o gestione tiempo (segundos).

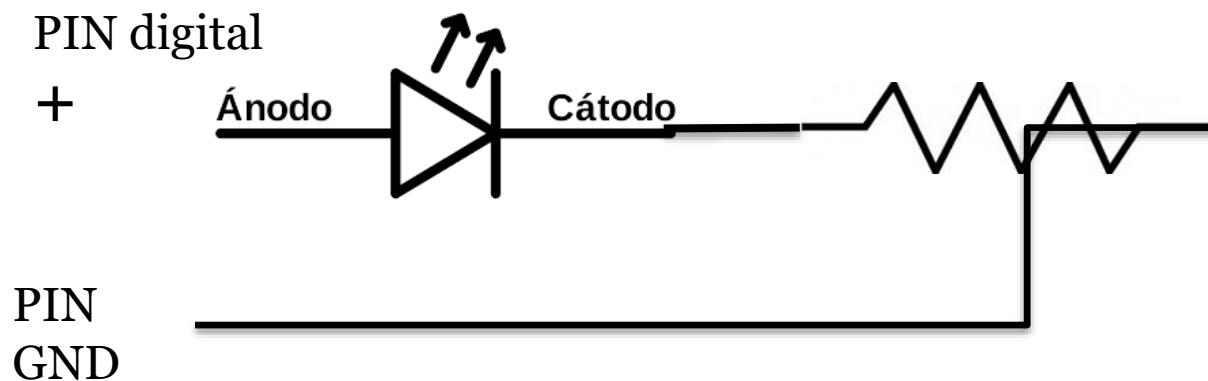
# ¿Cómo funciona LED?

- Para hacer esto tengo que documentarme sobre el dispositivo LED.
  - Podemos usar los libros de clase
  - Podemos usar la red de Internet. (!!no te creas todo lo que pone!!).  
Selecciona páginas del **entorno educativo**.

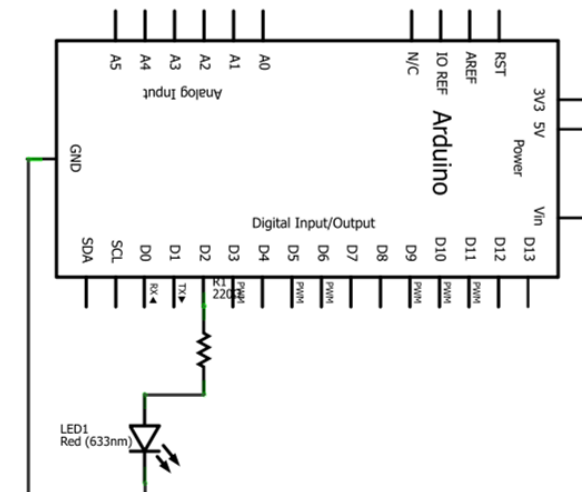
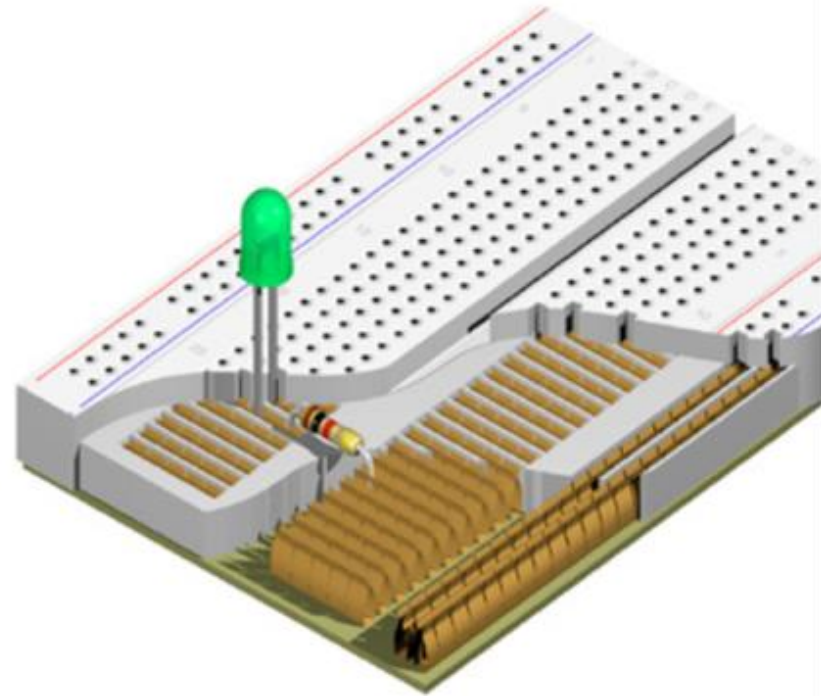
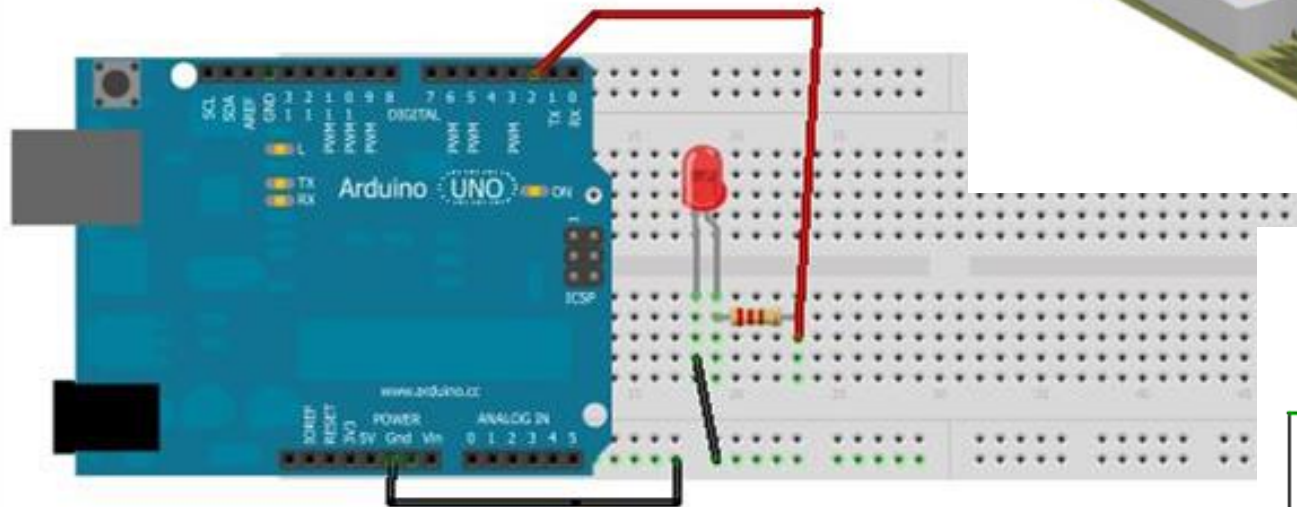


# Conexión LED a Arduino

- Usaremos un pin que podamos manejar desde el micro.
- Los pines digitales los podemos poner a 5V, que se denomina HIGH, o a 0V LOW. Decimos que lo ponemos a 1 o a 0.
- Sustituimos la pila y el pulsador por la entrada digital.



# LED: conexión a Arduino



# ¿Cómo defino la secuencia de acciones que tiene que realizar el MICROCONTROLADOR?



- Para definir la secuencia de acciones, primero pensamos en los distintos pasos que tenemos que dar.
- Por ejemplo pueden ser del tipo:
  - Inicializar comunicaciones
  - Configurar pin digital como entrada,..... O como salida
  - Poner a 1 o en Alto el pin ....
  - Poner a 0 o en Bajo el pin .....
  - Leer la entrada analógica n°....
  - Leer el valor del sensor de luz de la entrada analógica ....
  - Esperar un tiempo: segundos, milisegundos...
  - Leer puerto serie ...
  - Escribir puerto serie...
  - Activar motor
  - Parar motor
  - Girar a derecha
  - Girar a izquierda
  - .....



# Creamos el programa

- A partir del análisis anterior, generamos las órdenes de nuestro microcontrolador.
- Para ello tenemos que conocer las instrucciones que tenemos y su **Sintaxis**.

## ANÁLISIS

### INICIALIZAR

Configuramos el pin 2 como salida digital

### FIN\_INICIALIZAR

### Bucle

Poner pin digital 2 en ALTO

Esperar 1seg

Poner pin digital 2 en Bajo

Esperar 1seg

Volver a Bucle

## PROGRAMA

```
void setup() {  
    pinMode(2, OUTPUT);  
}
```

```
void loop() {  
    digitalWrite(2, HIGH);  
    delay(1000);  
    digitalWrite(2, LOW);  
    delay(1000);  
}
```



# Documentamos el programa.

- Ponemos comentarios en el programa para indicar las acciones que realiza.
- Acciones:
  - Guardar el programa
  - Verificar programa
  - Si todo correcto entonces lo cargamos en el controlador.

```
LED_1 | Arduino 1.0.3
Archivo Editar Sketch Herramientas Ayuda

LED_1 $
/*
  Parpadeo
  Conmuta de forma repetitiva el led de encendido a apagado
  con una duración de 1 segundo
  */
// Usamos el Pin 2 para conectar un LED

void setup() {
  // Inicializamos el pin 2 como pin de salida.
  pinMode(2, OUTPUT);
}

// La rutina de bucle loop se ejecutará una y en un bucle infinito
void loop() {
  digitalWrite(2, HIGH); // El LED pasa a estado Alto, (voltaje alto).
  delay(1000);           // Espera 1 segundo
  digitalWrite(2, LOW);  // El LED pasa a estado Bajo, (voltaje bajo).
  delay(1000);           // Espera 1 segundo
}
```



# Sintaxis lenguaje de programación

Nombre de la instrucción (parámetros, ....) ;

- Recuerda que todas las instrucciones terminan en ;
  - **Excepto:**
    - **setup(){} ,**
    - **loop(){} ,**
    - **y las que se llaman estructuras de control. if(){}...., for(){}...**

**pinMode(2, OUTPUT);**

Instrucción

Parámetros. Separados por coma



- loop()

- HIGH | LOW

- INPUT | OUTPUT

- pinMode()
- digitalWrite()
- digitalRead()

# pinMode()

## Description

Configures the specified pin to behave either as an input or an output. See the description of [digital pins](#) for details on the functionality of the pins.

As of Arduino 1.0.1, it is possible to enable the internal pullup resistors with the mode `INPUT_PULLUP`. Additionally, the `INPUT` mode explicitly disables the internal pullups.

## Syntax

`pinMode(pin, mode)`

## Parameters

pin: the number of the pin whose mode you wish to set

mode: `INPUT`, `OUTPUT`, or `INPUT_PULLUP`. (see the [digital pins](#) page for a more complete description of the functionality.)

## Returns

None

## Example

```
int ledPin = 13;           // LED connected to digital pin 13
void setup()
```

- = (assignment operator)
- + (addition)
- - (subtraction)

## Conversion

- char()

## Analog I/O

- analogReference()
- analogRead()
- analogWrite() - PWM

## Advanced I/O

- tone()
- noTone()
- shiftOut()
- shiftIn()
- pulseIn()

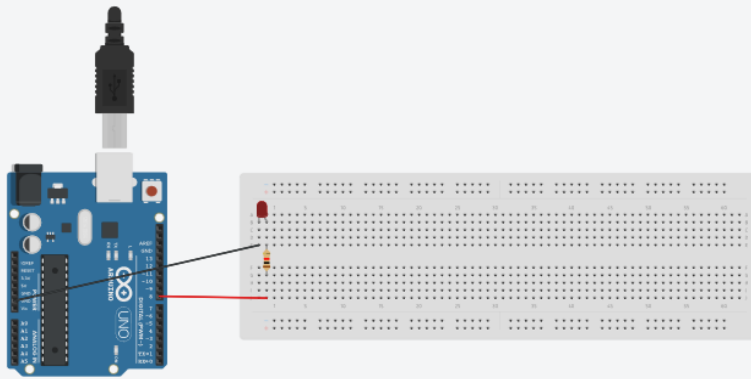
## Time

- millis()
- micros()
- delay()
- delayMicroseconds()

## Math

- min()
- max()
- abs()
- constrain()

# Bloques



Interfaz de programación de bloques para Arduino Uno R3.

**Botones de acción:** Código, Iniciar simulación, Exportar, Compartir.

**Bloques disponibles:**

- Output (Azul)
- Input (Púrpura)
- Notation (Gris)
- Control (Naranja)
- Math (Verde)
- Variables (Púrpura)

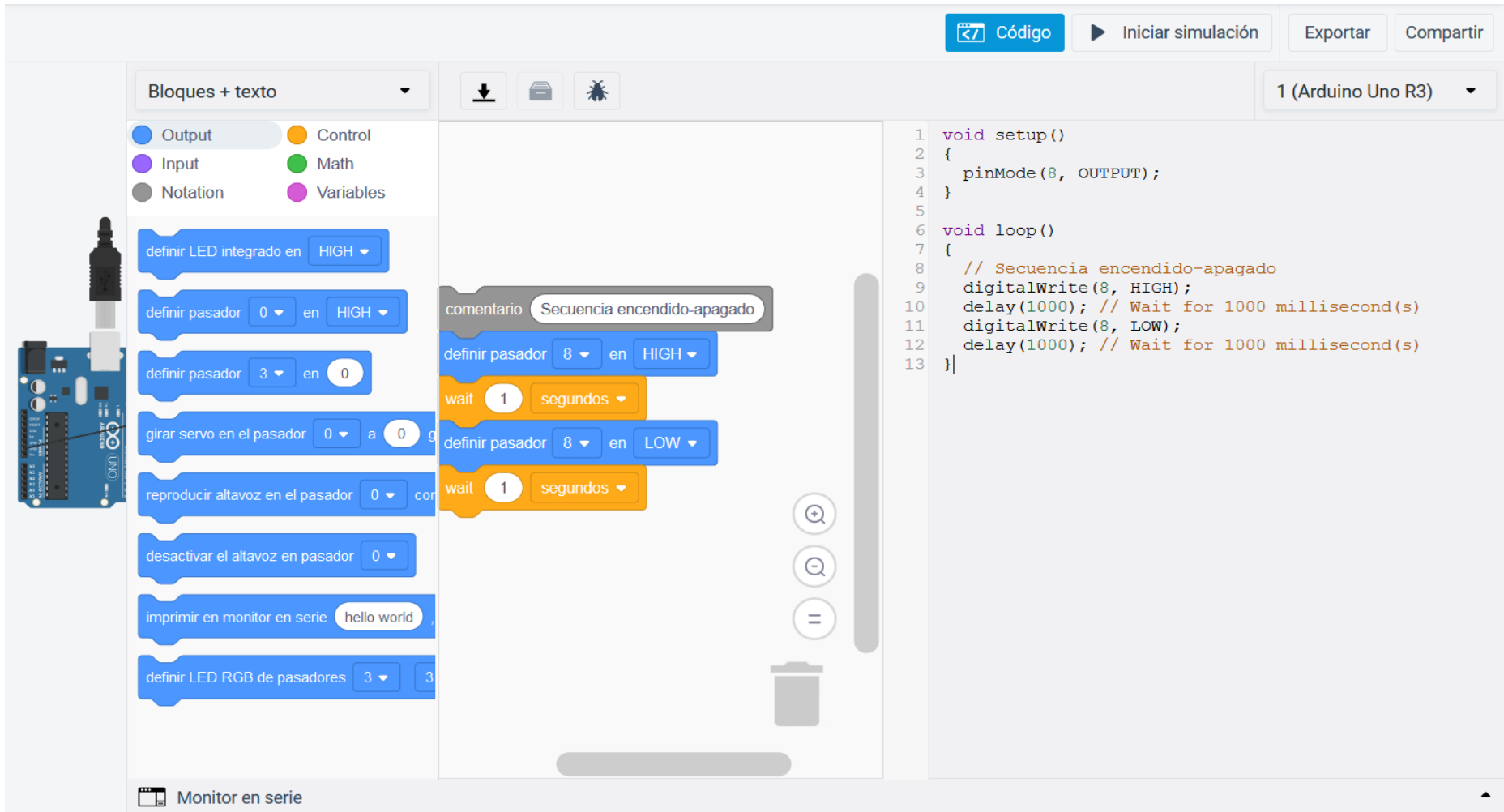
**Lista de bloques en el editor:**

- definir LED integrado en HIGH
- definir pasador 0 en HIGH
- definir pasador 3 en 0
- girar servo en el pasador 0 a 0
- reproducir altavoz en el pasador 0 con
- desactivar el altavoz en pasador 0
- imprimir en monitor en serie hello world
- definir LED RGB de pasadores 3 3

**Sequencia encendido-apagado:**

- definir pasador 8 en HIGH
- wait 1 segundos
- definir pasador 8 en LOW
- wait 1 segundos

**Monitor en serie:**





# Referencias

- Web oficial de Arduino. <http://www.arduino.cc/> [On line]. Consultado [Abril 2015]
- Referencia Software. Web oficial Arduino <  
<http://www.arduino.cc/en/Reference/HomePage>> Consultado [Abril 2015]
- Referencia Hardware Arduino UNO. Web oficial Arduino <  
<http://www.arduino.cc/en/Main/ArduinoBoardUno>> Consultado [Abril 2015]
- Imagen con la estructura de arduino UNO. <http://i.stack.imgur.com/wKz2l.png>  
Consultado [Abril 2015]
- AUTODESK: TINKERCAD- Circuits. Entorno de desarrollo online de simulación de Arduino/Electronica/3D. (Software gratuito) Web que requiere inscripción. <https://www.tinkercad.com>, [Diciembre 2018]
- Blockly: Google for education. A JavaScript library for building visual programming editors. <https://developers.google.com/blockly/> [Diciembre 2018]