
Introducción al lenguaje de programación de Arduino

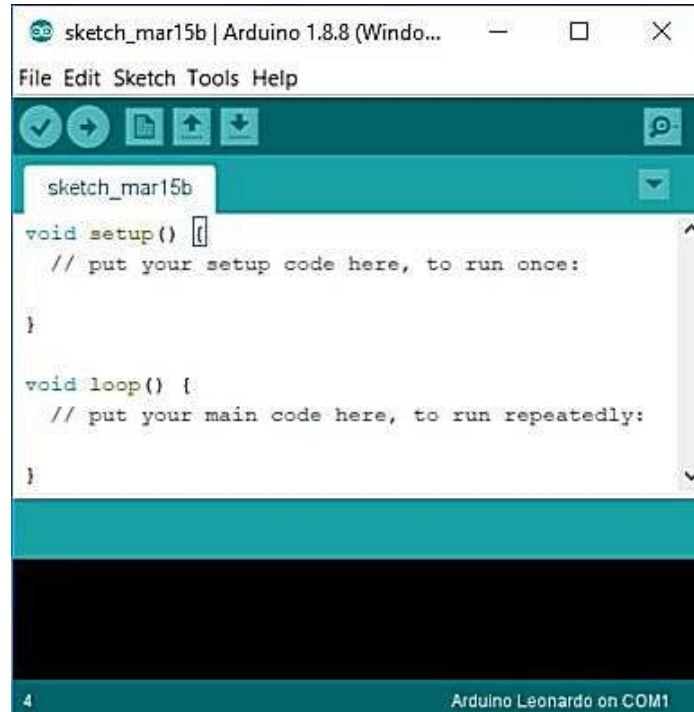
— Erick Andrés Obregón Fonseca —

Agenda

- Editores de código
- Estructura del archivo
- Tipos de datos
- Operadores aritméticos
- Operadores lógicos
- Estructuras de control o condicionales
- Funciones útiles
- Ejemplos

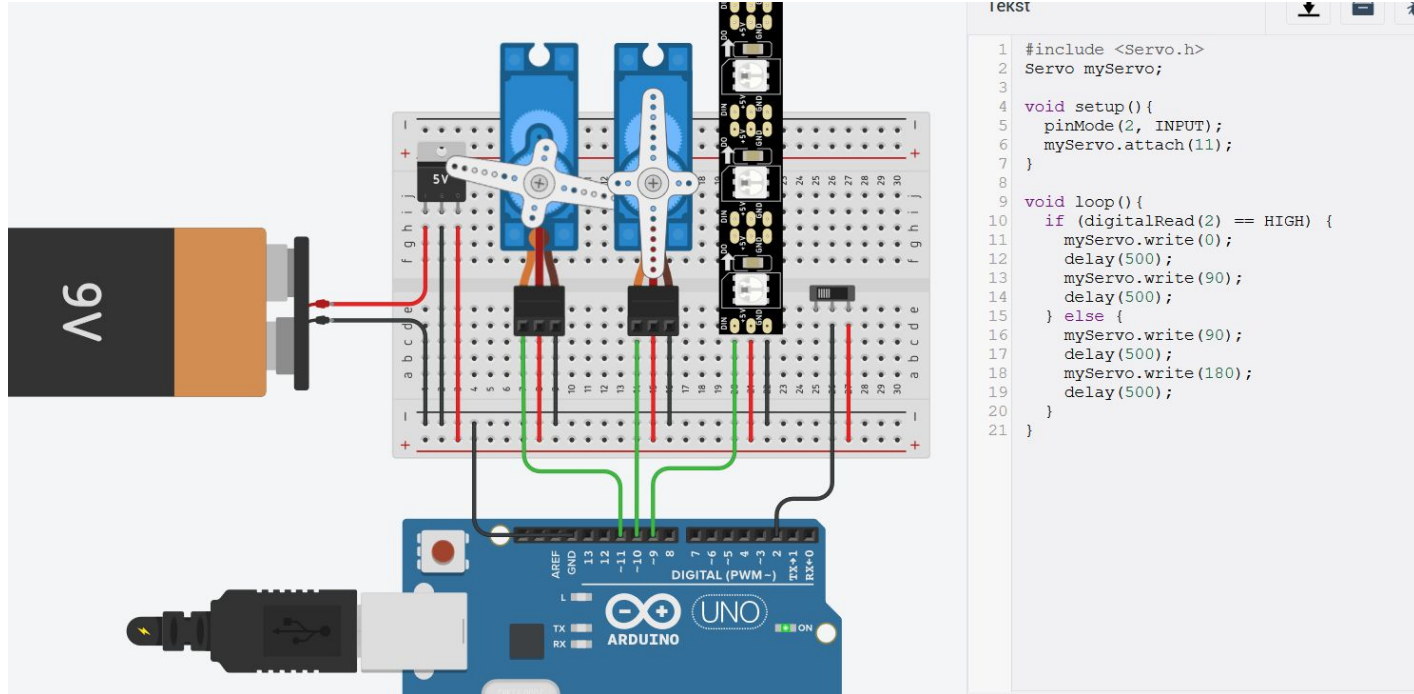
Editores de código

Arduino IDE



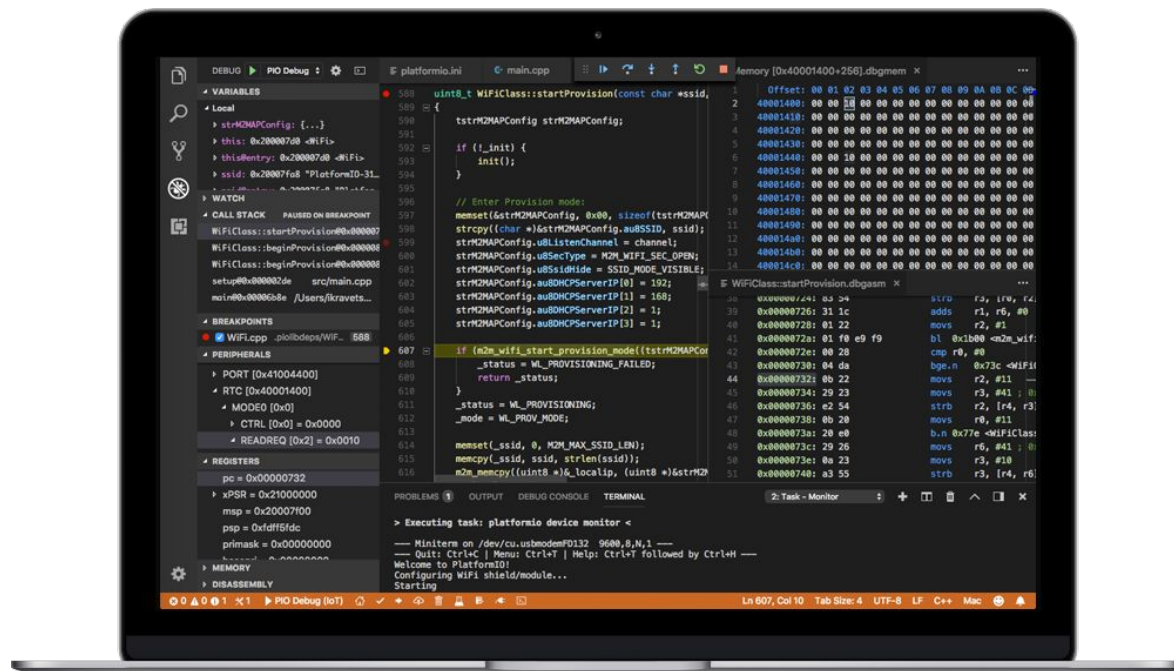
<https://www.best-microcontroller-projects.com/arduino-ide.html>

AUTODESK TINKERCAD

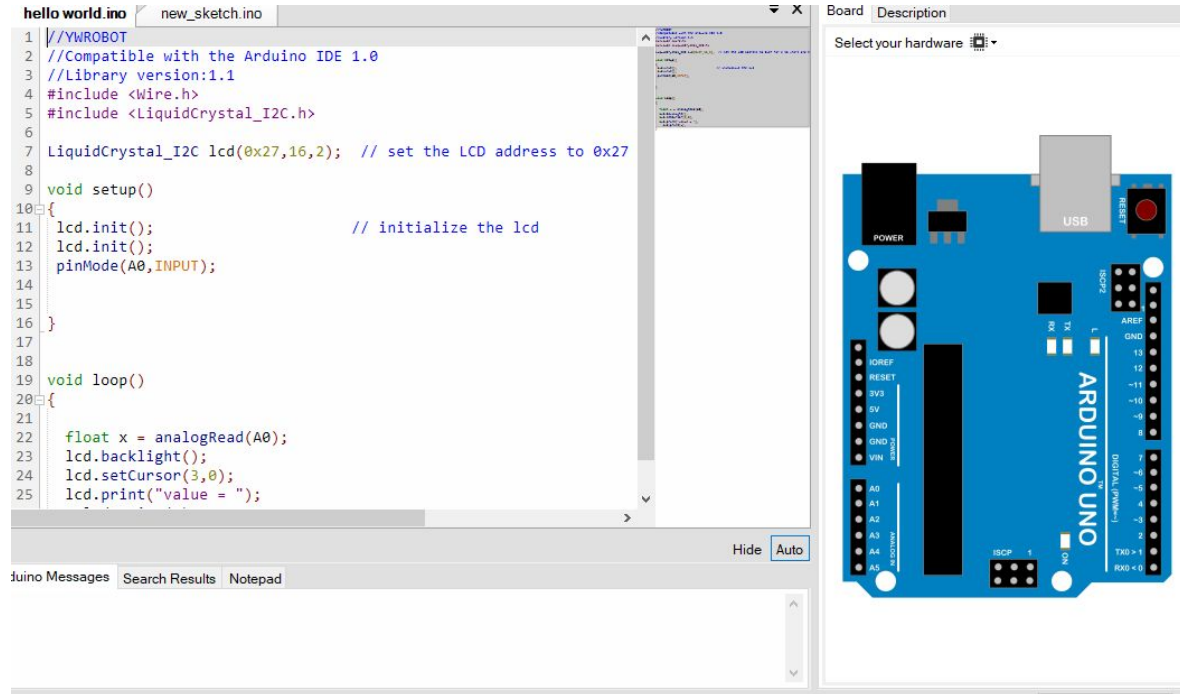


<https://stackoverflow.com/questions/58043225/switch-connected-to-an-arduino-not-working-in-tinkercad-circuits>

PlatformIO + Visual Studio Code

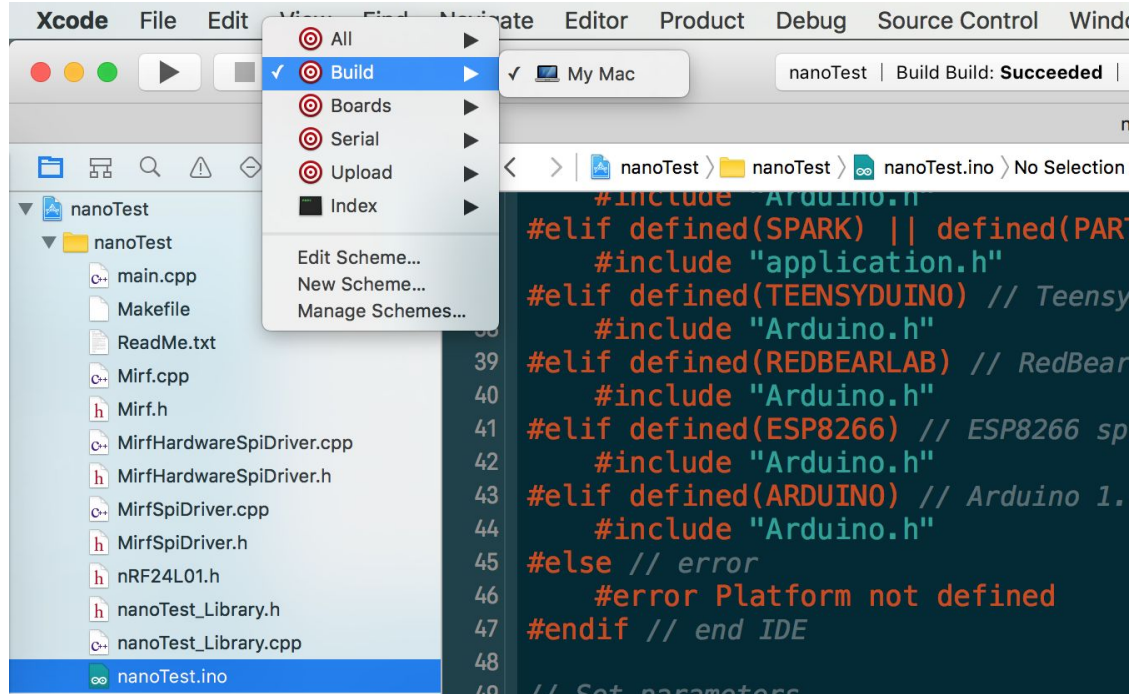


<https://platformio.org/>



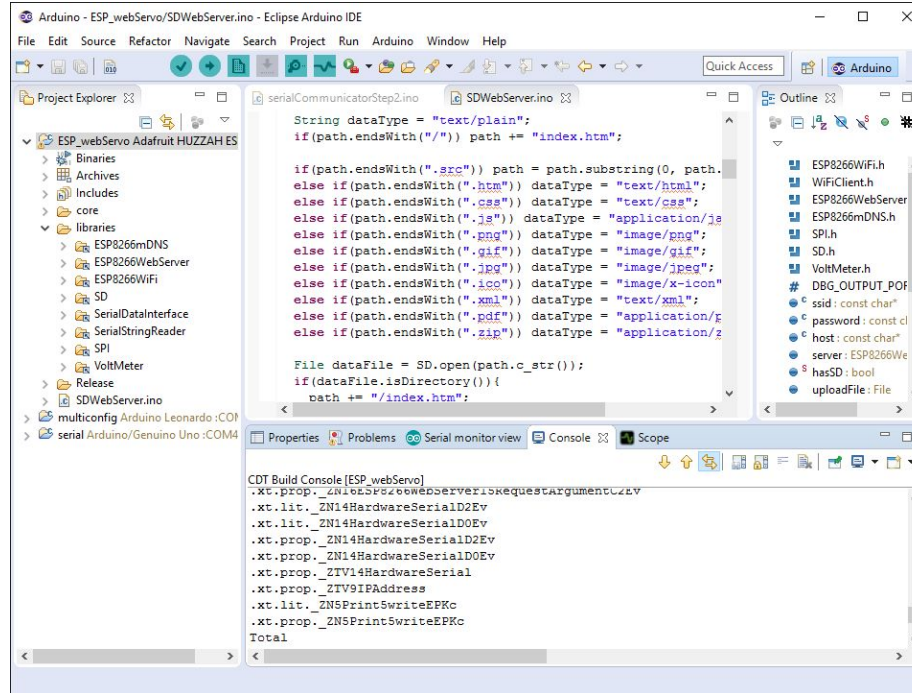
<https://theelectromentor.com/how-to-install-arduino-software-get-started-with-arduino-ide/>

embedXCode



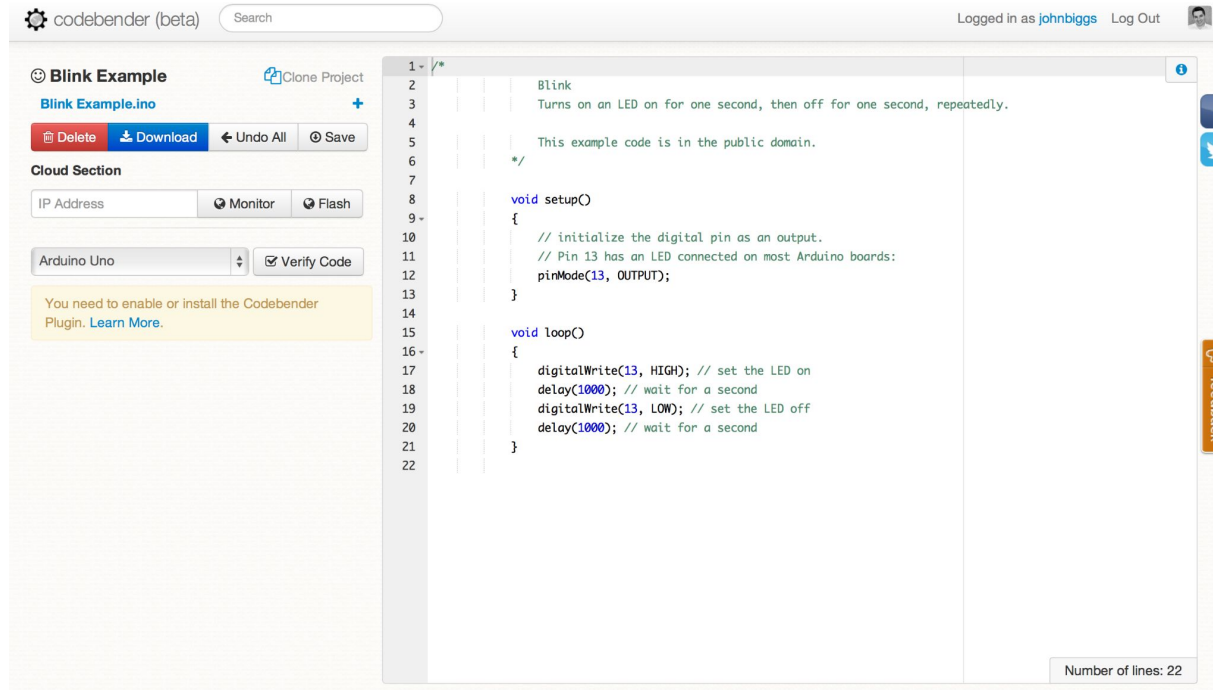
<https://www.logcg.com/en/archives/1761.html>

Eclipse Arduino Plugin



<https://marketplace.eclipse.org/content/arduino-eclipse-ide-named-sloeber-product>

Codebender



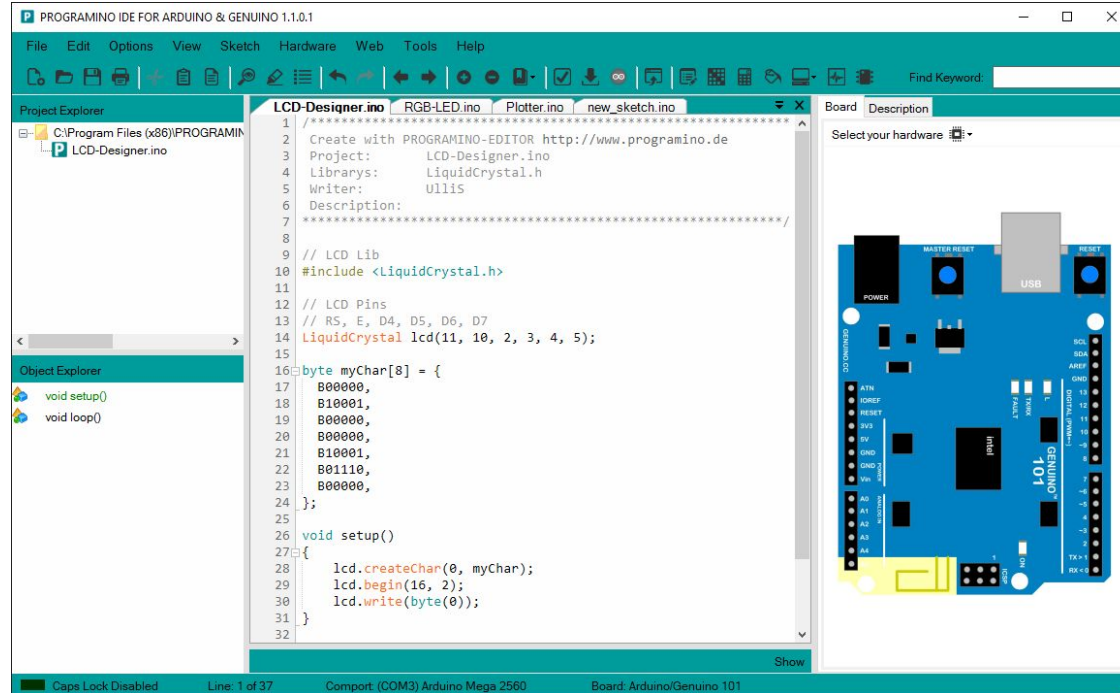
<https://techcrunch.com/2013/09/12/codebender-cc-makes-it-crazy-easy-to-program-your-arduino-board-from-your-browser/>

Visualino



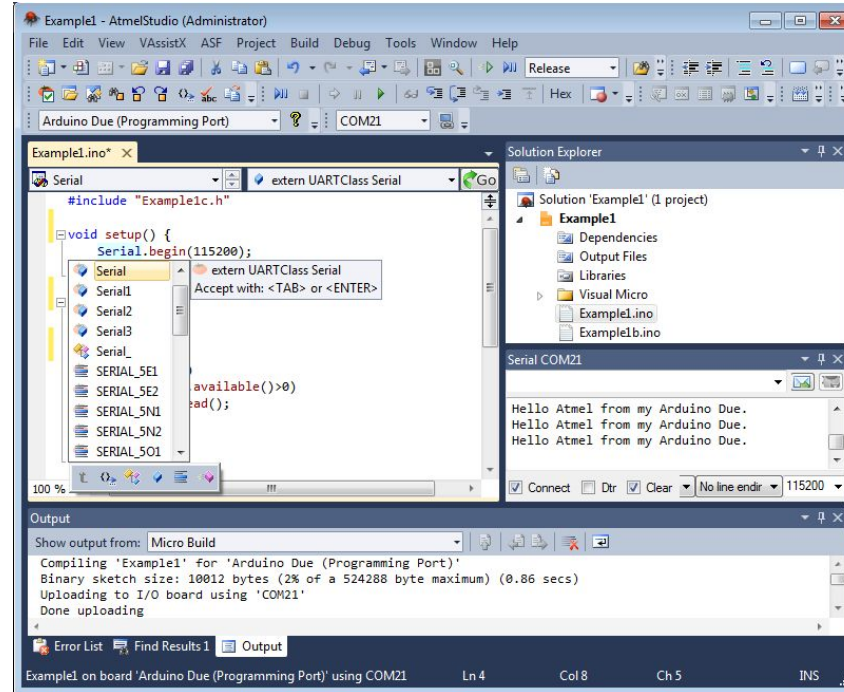
<https://aprendiendoarduino.wordpress.com/category/visualino/>

PROGRAMINO IDE



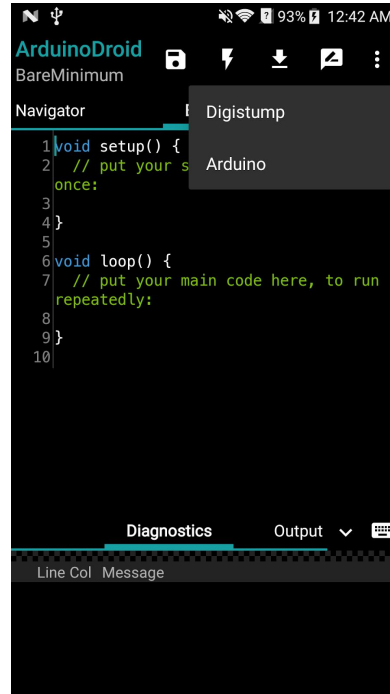
<https://www.programino.com/>

Atmel Studio



<https://forum.arduino.cc/t/free-arduino-plugin-for-atmel-studio/163816>

ArduinoDroid



<https://www.pinterest.com/pin/127015651975962783/>

Lenguaje de programación

Estructura del archivo

- *setup()*:
 - Esta función se llama una única vez cuando se enciende o se reinicia la tarjeta de Arduino
 - Se usa para inicializar variable, los modos de los pines, bibliotecas, ect
- *loop()*:
 - Se ejecuta después de la función *setup()*
 - Como su nombre lo indica, se ejecuta constantemente en un ciclo
 - Se usa para controlar la tarjeta de Arduino

Estructura del archivo

sketch_may07a Arduino 1.8.13 (Windows Store 1.8.42.0)

Archivo Editar Programa Herramientas Ayuda



```
void setup() {  
  // put your setup code here, to run once:  
  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
  
}
```

Tipos de datos en Arduino

- *void*:
 - Se usa en la declaración de funciones y significa que no retorna información
- *boolean*:
 - Almacena un valor de *true* o *false*
 - Ocupa un tamaño en memoria de 8-bit (1-byte)
- *byte*:
 - Almacena un número entero sin signo de 0 a 255
- *short*:
 - Almacena un número entero de -32 768 a 32 767
 - Ocupa un tamaño en memoria de 16-bit (2-byte)

Tipos de datos en Arduino

- *word*:
 - Almacena un número entero sin signo de 0 a 65 535
 - Ocupa un tamaño en memoria de 16-bit (2-byte)
- *int*:
 - Almacena un número entero
 - El valor máximo y mínimo depende de la cantidad de bit de la tarjeta
 - 16-bit (2-bytes) de -32 768 a 32 767
 - 32-bit (4-bytes) de -2 147 483 648 a 2 147 483 647
- *unsigned int*:
 - Almacena un número entero sin signo
 - 16-bit (2-bytes) de 0 a 65 535
 - 32-bit (4-bytes) de 0 a 4 294 967 295

Tipos de datos en Arduino

- *long*:
 - Almacena un número entero con tamaño de memoria extendido
 - 32-bit (4-bytes) de -2 147 483 648 a 2 147 483 647
- *unsigned long*:
 - Almacena un número entero sin signo con tamaño de memoria extendido
 - 32-bit (4-bytes) de 0 a 4 294 967 295
- *float*:
 - Almacena números de punto flotante (con decimales)
 - 32-bit (4-byte) de -3.4028235e38 a 3.4028235e38

Tipos de datos en Arduino

- *double*:
 - Almacena un número de punto flotante de doble precisión
 - 64-bit (8-byte)
- *char*:
 - Se usa para almacenar un caracter como 'A', 'e' o 'c'
 - Los caracteres se deben escribir entre comillas simples
 - Ocupan 8-bit (1-byte) de memoria
 - Los caracteres se almacenan como un número según la tabla ASCII

Operadores aritméticos

Operador	Nombre	Ejemplo
=	asignación	$a = b$
+	suma	$a = b + c$
-	resta	$a = b - c$
*	multiplicación	$a = b * c$
/	división	$a = b / c$
%	módulo / residuo	$a = b \% c$

Operadores de comparación

Operador	Nombre	Ejemplo
==	Igual a	a == b
!=	Diferente de	a != b
<	Menor que	a < b
<=	Menor igual que	a <= b
>	Mayor que	a > b
>=	Mayor que	a >= b

Operadores lógicos o booleanos

Operador	Nombre	Ejemplo
&&	AND lógico	A && B
	OR lógico	A B
!	NOT lógico	!A

Operadores a nivel de bits

Operador	Nombre	Ejemplo
&	AND	a & b
	OR	a b
^	XOR	a ^ b
~	NOT	!a
<<	SHIFT LEFT	a << b
>>	SHIFT RIGHT	a >> b

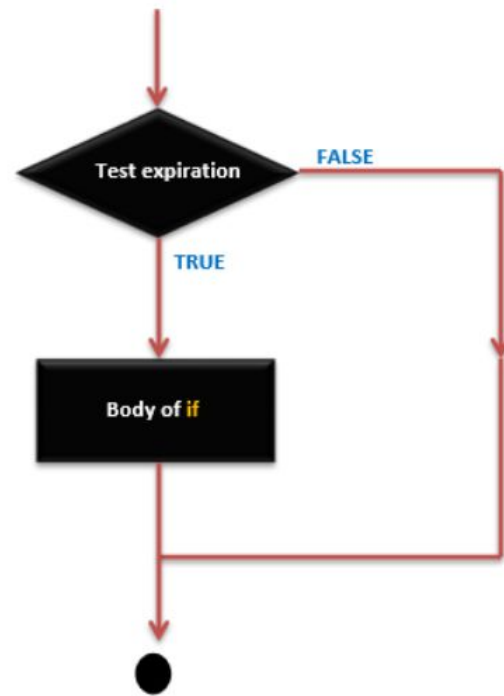
Estructuras de control o condicionales

Sentencia *if*

- Se usa para tomar decisiones
- Evalúa una expresión entre paréntesis
- Así ejecuta un bloque de código

```
if (expression)  
    statement;
```

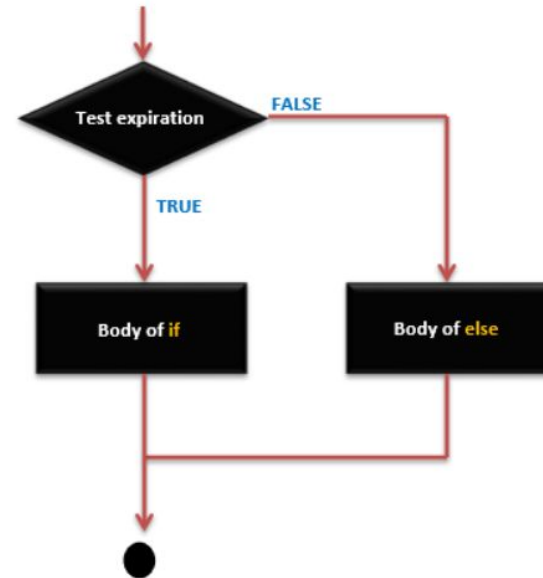
```
if (expression) {  
    Block of statements;  
}
```



Sentencia *if...else*

- El *else* se ejecuta cuando la condición no se cumplió

```
if (expression) {  
    Block of statements;  
}  
else {  
    Block of statements;  
}
```

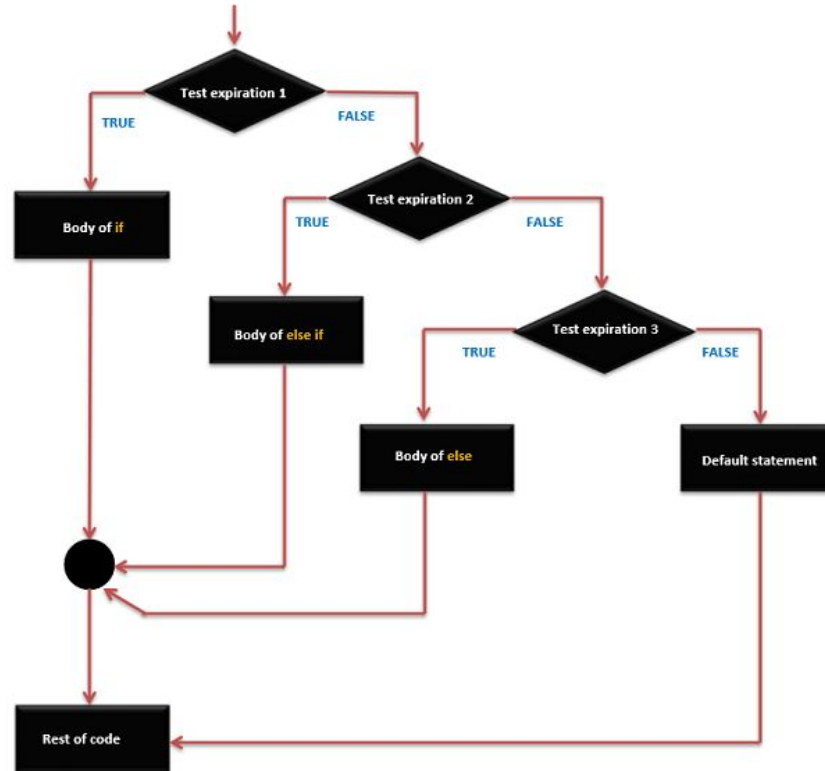


Sentencia *if...else if...else*

- Sentencia *else if* sirve para validar más condiciones
- Sentencia *else* se ejecuta cuando ninguna de las condiciones anteriores se cumplieron

```
if (expression_1) {  
    Block of statements;  
}  
  
else if(expression_2) {  
    Block of statements;  
}  
.  
.  
.  
  
else {  
    Block of statements;  
}
```

Sentencia *if...else if...else*



Funciones útiles

Función pinMode

- *pinMode(pin, mode)*
 - **Pin:** el número de pin a configurar
 - **Mode:** el modo en el que se comportará el pin (INPUT, OUTPUT, INPUT_PULLUP)
- Configura el comportamiento de un pin en específico como entrada o salida.

Comunicación Serial


- Se usa para comunicación entre la placa Arduino y una computadora u otro dispositivo
- Algunas funciones útiles básicas:
 - *Serial.begin(speed)*: Establece la velocidad de datos en bits por segundo para la transmisión de datos en serie
 - *Serial.print()*: Escribe en el puerto serial
 - *Serial.println()*: Escribe en el puerto serial con un caracter de retorno al final
 - *Serial.read()*: Lee los datos del puerto serial
 - *Serial.write()*: Escribe datos en binario en el puerto serial

Funciones digitalRead, digitalWrite, analogRead, analogWrite

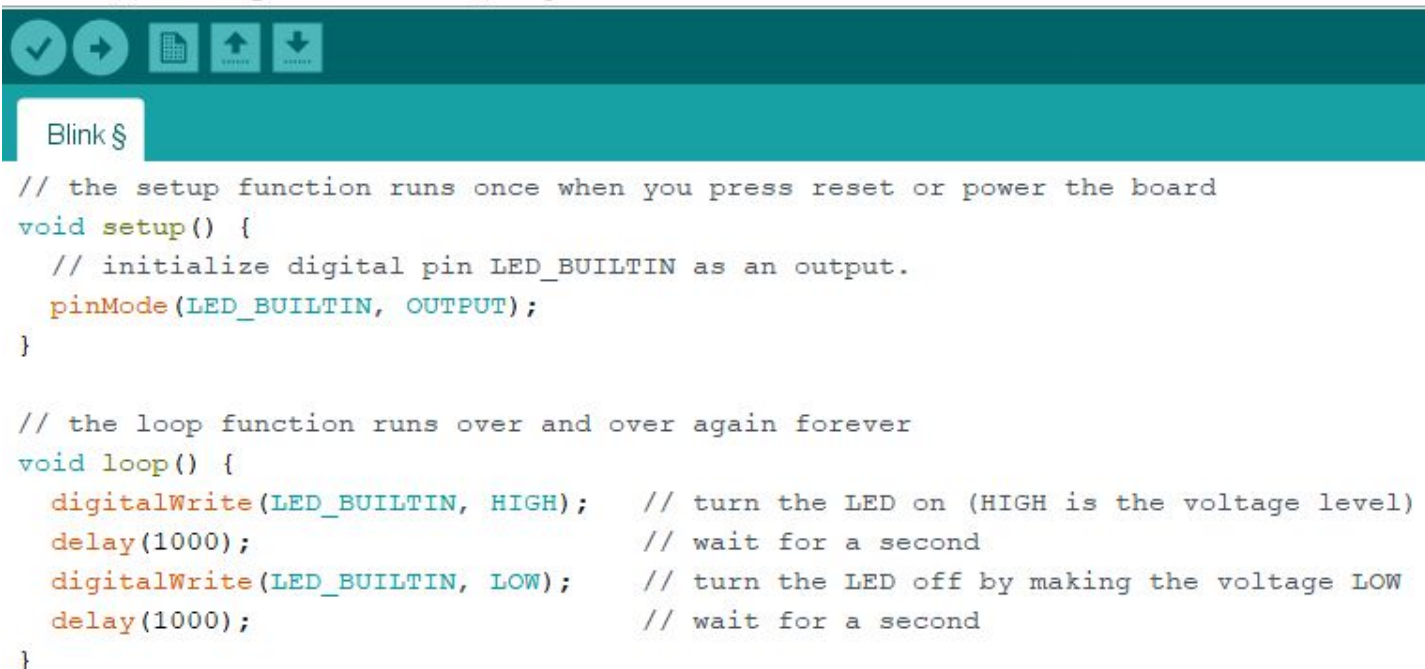
- *digitalRead(pin):*
 - Leer el valor lógico (HIGH o LOW) de un pin digital
- *digitalWrite(pin, value):*
 - Escribe un valor lógico (HIGH or LOW) en un pin digital
- *analogRead(pin):*
 - Leer el valor de un pin analógico
 - Mapea un pin que funciona con 3.3V o 5V en enteros de 0 a 1023
- *analogWrite(pin, value):*
 - Escribe un valor en un pin analógico de 0 a 1023

Ejemplos prácticos

Hola Mundo en Arduino (Blink)

 Blink Arduino 1.8.13 (Windows Store 1.8.42.0)

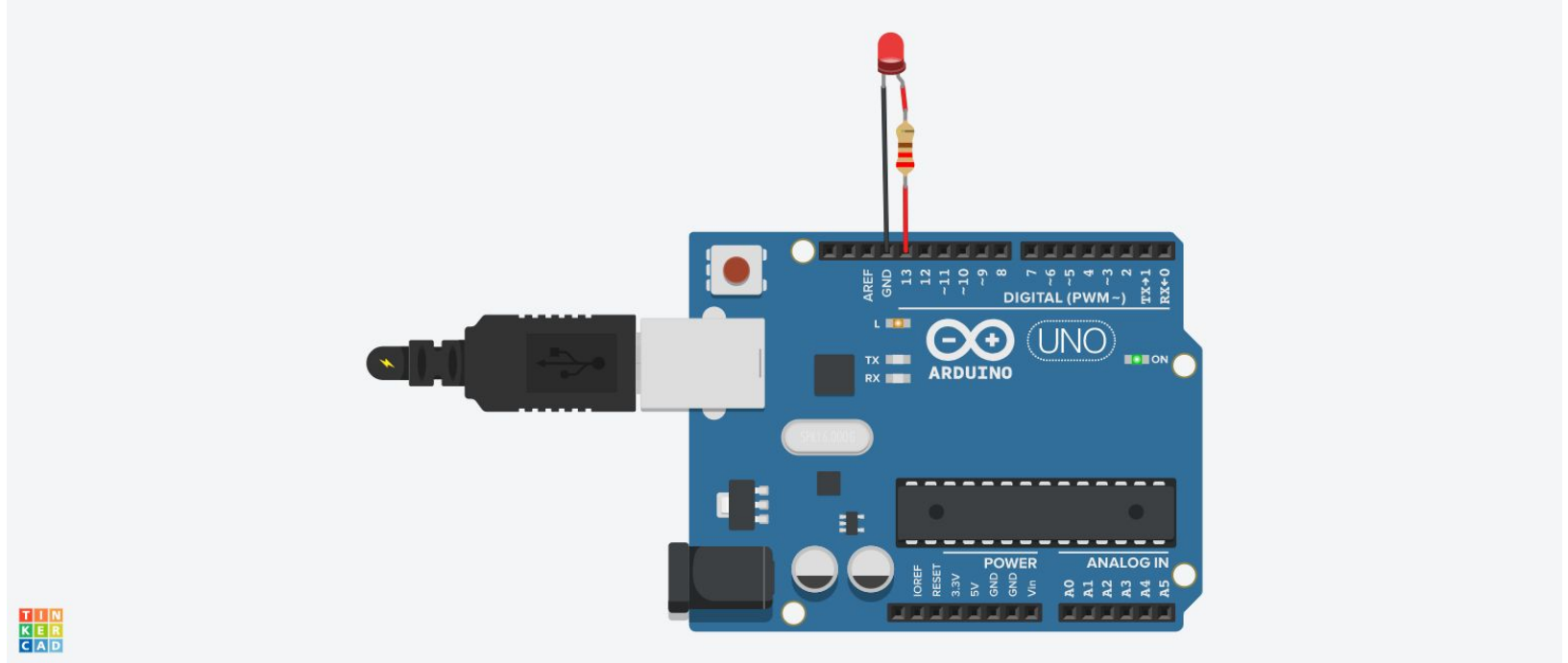
Archivo Editar Programa Herramientas Ayuda



```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```

“Hola Mundo” en Arduino (Blink)



<https://www.tinkercad.com/things/6Sd1Xr3mxwg>

Digital Read Serial

📄 DigitalReadSerial Arduino 1.8.13 (Windows Store 1.8.42.0)

Archivo Editar Programa Herramientas Ayuda

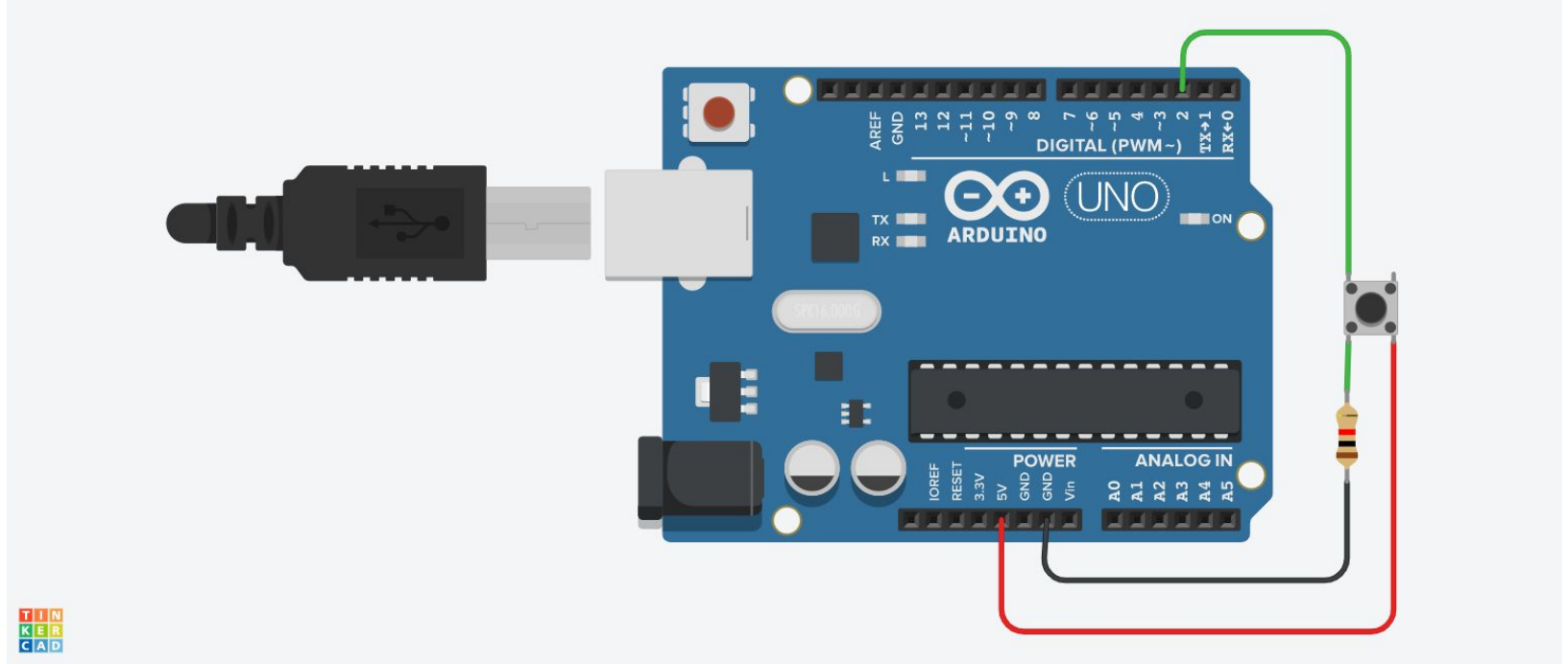


```
// digital pin 2 has a pushbutton attached to it. Give it a name:
int pushButton = 2;

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);      // delay in between reads for stability
}
```

Digital Read Serial



<https://www.tinkercad.com/things/3EJoC9bZbQm>

Muchos más ejemplos

arduino.cc

- arduino.cc/education
- <https://create.arduino.cc/projecthub>
- <https://create.arduino.cc/projecthub/projects/tags/arduino>

Erick Andrés Obregón Fonseca

Contacto

- Correo: erickobregonf@gmail.com
 - GitHub: [ErickOF](https://github.com/ErickOF)
 - LinkedIn: [erickobregonf](https://www.linkedin.com/in/erickobregonf)
 - Telegram: [ErickOF](https://t.me/ErickOF)
 - WhatsApp: [+50687134265](https://wa.me/50687134265)
-

Referencias

- [1]"setup() - Arduino Reference", Arduino.cc. [Online]. Available: <https://www.arduino.cc/reference/en/language/structure/sketch/setup/>. [Accessed: 07- May- 2021].
- [2]"loop() - Arduino Reference", Arduino.cc. [Online]. Available: <https://www.arduino.cc/reference/en/language/structure/sketch/loop/>. [Accessed: 07- May- 2021].
- [3]"pinMode() - Arduino Reference", Arduino.cc. [Online]. Available: <https://www.arduino.cc/reference/en/language/functions/digital-io/pinmode/>. [Accessed: 07- May- 2021].
- [4]"Serial - Arduino Reference", Arduino.cc. [Online]. Available: <https://www.arduino.cc/reference/en/language/functions/communication/serial/>. [Accessed: 07- May- 2021].

Referencias

- [5]"digitalRead() - Arduino Reference", Arduino.cc. [Online]. Available: <https://www.arduino.cc/reference/en/language/functions/digital-io/digitalread/>. [Accessed: 07- May- 2021].
- [6]"digitalWrite() - Arduino Reference", Arduino.cc. [Online]. Available: <https://www.arduino.cc/reference/en/language/functions/digital-io/digitalwrite/>. [Accessed: 07- May- 2021].
- [7]"analogRead() - Arduino Reference", Arduino.cc. [Online]. Available: <https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/>. [Accessed: 07- May- 2021].
- [8]"analogWrite() - Arduino Reference", Arduino.cc. [Online]. Available: <https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/>. [Accessed: 07- May- 2021].

Referencias

- [9]"Arduino - Data Types - Tutorialspoint", Tutorialspoint.com. [Online]. Available: https://www.tutorialspoint.com/arduino/arduino_data_types.htm. [Accessed: 07-May- 2021].
- [10]"Arduino - Control Statements", Tutorialspoint.com. [Online]. Available: https://www.tutorialspoint.com/arduino/arduino_control_statements.htm. [Accessed: 07- May- 2021].