# Computación Ubicua

**Introducción a Arduino**

# Arduino

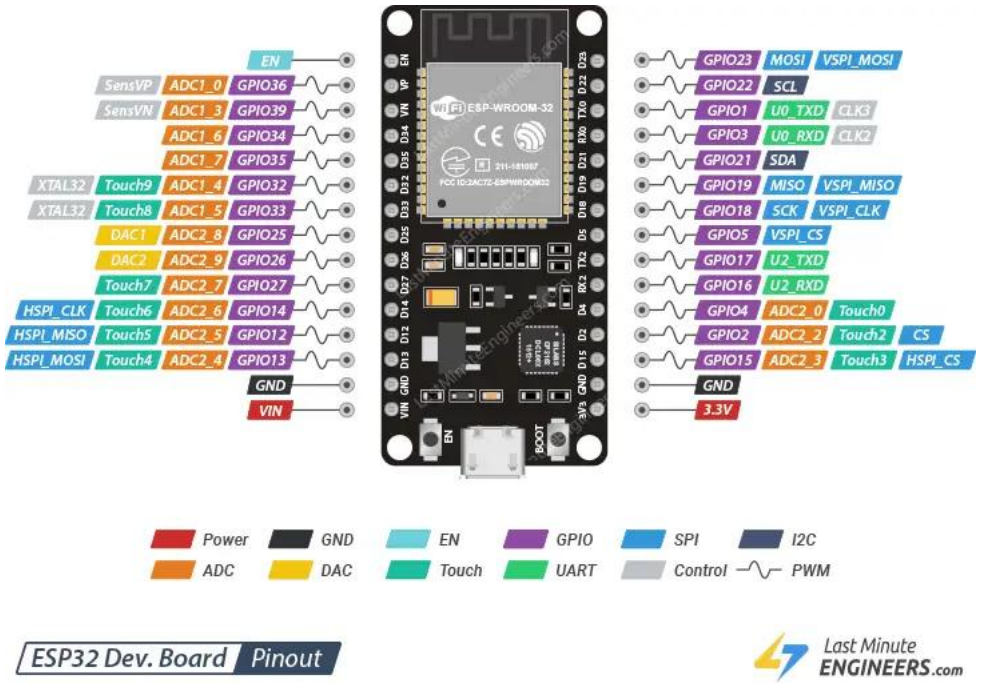
- **Arduino** es una plataforma de prototipado electrónico de código abierto basada en hardware y software fáciles de usar. Está pensada para artistas, diseñadores, aficionados y cualquier persona interesada en crear objetos o entornos interactivos.

# Tipos de placas más comunes

| Placa | Descripción breve | Ideal para... |
|---|---|---|
| **Arduino Uno** | La más popular. 14 pines digitales, 6 analógicos. | Principiantes y proyectos básicos |
| **Nano** | Más pequeña, similar a la Uno. | Proyectos compactos |
| **Mega** | Más memoria y más pines. | Proyectos grandes (robótica, domótica) |
| **Leonardo** | Puede simular teclado/ratón. | Interfaces humanas (HID) |
| **Due** | Más potente (procesador ARM). | Proyectos más exigentes |
| **ESP8266 / ESP32** | Incluyen Wi-Fi y/o Bluetooth. | IoT y conectividad inalámbrica |


# Pinout

- El esquema de patillaje es de los documentos de referencia más útiles a la hora de realizar un montaje. El caso de Arduino no es una excepción, más aún teniendo en cuenta la gran cantidad de pines y modelos de placas disponibles.



# Montaje

- Existen 2 tipos de entradas:
  - Digitales
    - Una señal digital es una variación de voltaje entre -Vcc a +Vcc sin pasar por los valores intermedios
    - Al valor inferior -Vcc le asociamos un valor lógico LOW o 0 lógico
    - Al valor superior +Vcc le asociamos HIGH o 1 lógico.
  - Analógicas
    - Una señal analógica es una magnitud que puede tomar cualquier valor dentro de un intervalo –Vcc y + Vcc.
    - las entradas analógicas proporcionan una medición codificada en forma de un número digital (por ejemplo de 0 a 255).

# Hojas de características

- Es un documento crucial que detalla todas las especificaciones y propiedades relevantes de un sensor específico.
- Información clave que se encuentra en una hoja de características:
  - Rango de medición: Define los límites superior e inferior de los valores que el sensor puede medir con precisión.
  - Precisión: Indica la cercanía de las mediciones del sensor a los valores reales.
  - Resolución: Determina la menor variación que el sensor puede detectar.
  - Repetibilidad: Mide la consistencia de las mediciones al repetir la misma medición varias veces.
  - Linealidad: Indica si la relación entre la entrada y la salida del sensor es lineal o no.
  - Sensibilidad: Mide la respuesta del sensor a cambios en la variable medida.
  - Tiempo de respuesta: Define la velocidad con la que el sensor reacciona a cambios en la variable medida.
  - Condiciones ambientales: Especifica la temperatura, humedad y otras condiciones ambientales que el sensor puede soportar.
  - Conexiones eléctricas: Describe los pines o conexiones necesarias para alimentar el sensor y obtener sus datos.
  - Software y protocolos: Incluye información sobre el software o protocolos de comunicación necesarios para interactuar con el sensor.

# Arduino IDE

# Descarga del IDE

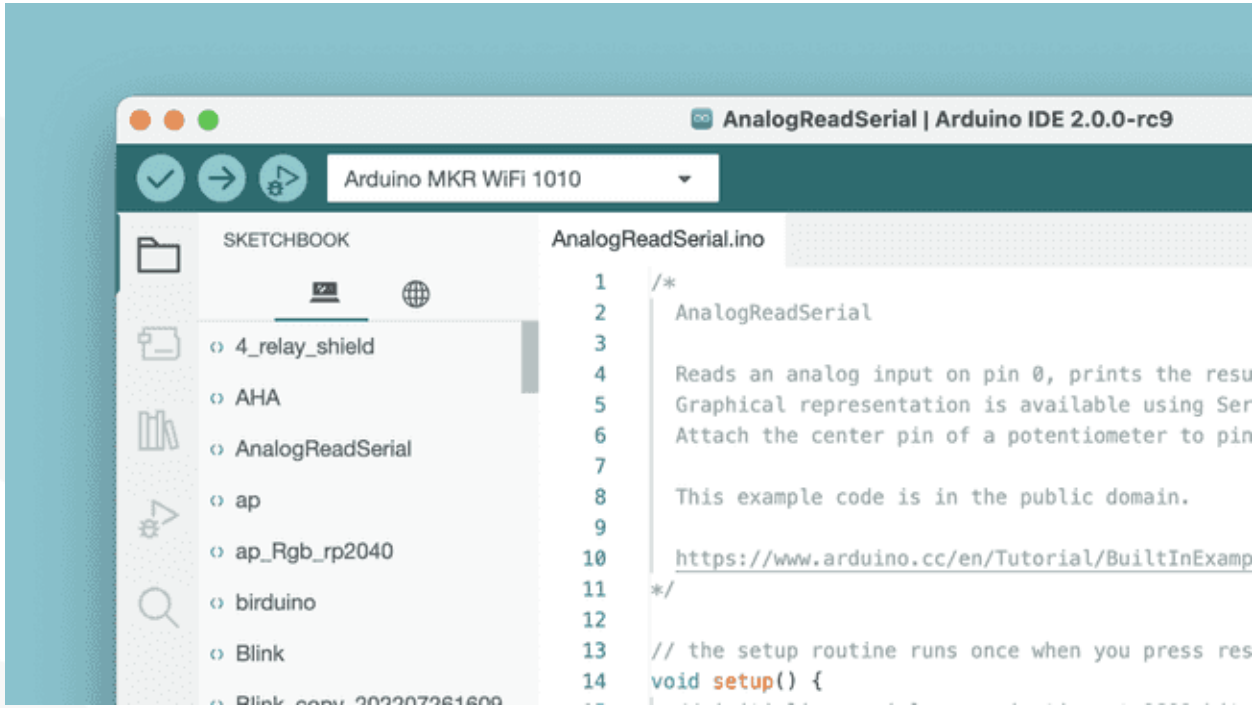
- https://www.arduino.cc/en/software/

# Arduino IDE


VERIFY/UPLOAD
SELECT BOARD & PORT
OPEN SERIAL MONITOR
AnalogReadSerial | Arduino IDE 2.0.0-rc9
Arduino MKR WiFi 1010
SKETCHBOOK
OPEN SERIAL PLOTTER
BOARD MANAGER
LIBRARY MANAGER
DEBUGGER
SEARCH
/*
AnalogReadSerial
Reads an analog input on pin 0, prints the result to the Serial Monitor.
Graphical representation is available using Serial Plotter (Tools > Serial Plotter menu).
in of a potentiometer to pin A0, and the outside pins to +5V and ground.
This example code is in the public domain.
https://www.arduino.cc/en/Tutorial/BuiltInExamples/AnalogReadSerial
*/
// the setup routine runs once when you press reset:
void setup() {
// initialize serial communication at 9600 bits per second:
Serial.begin(9600);
}
// the loop routine runs over and over again forever:
void loop() {
// read the input on analog pin 0:
int sensorValue = analogRead(A0);
// print out the value you read:

# Arduino IDE - Sketchbook

- Los cuadernos de Arduino se guardan como archivos .ino y deben almacenarse en una carpeta con el mismo nombre.
- El icono de la carpeta situado en la barra lateral permite acceder a los archivos del proyecto



# Arduino IDE – Gestor de placas

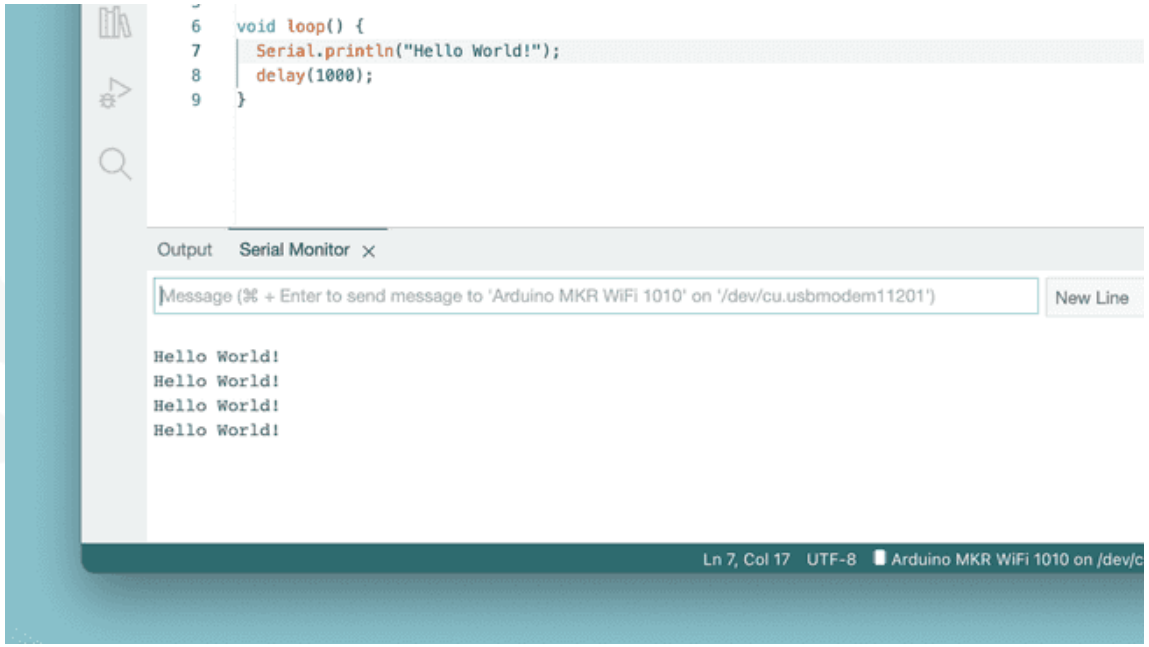
- El gestor de platas permite buscar e instalar paquetes que permiten compilar el código  y adaptarlo a las distintas placas.
- Hay varios paquetes de placas Arduino disponibles, como avr, samd, megaavr y más.

# Arduino IDE – Gestor de Bibliotecas

- El gestor de bibliotecas permite explorar e instalar distintas bibliotecas que facilitan el desarrollo de programas

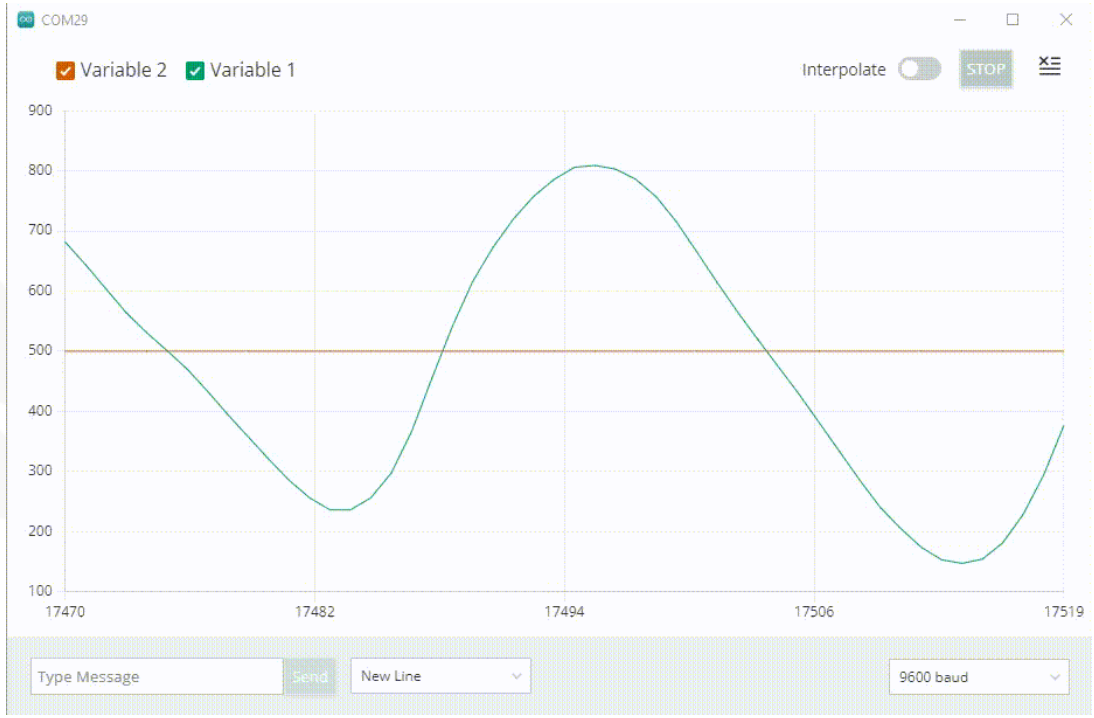
# Arduino IDE – Monitor Serie

- El monitor serie es una herramienta que permite ver los datos que se transmiten desde la placa, por ejemplo, mediante el comando Serial.print().
- Históricamente, esta herramienta se encontraba en una ventana separada, pero ahora está integrada en el editor. Esto facilita la ejecución simultánea de varias instancias en el ordenador.



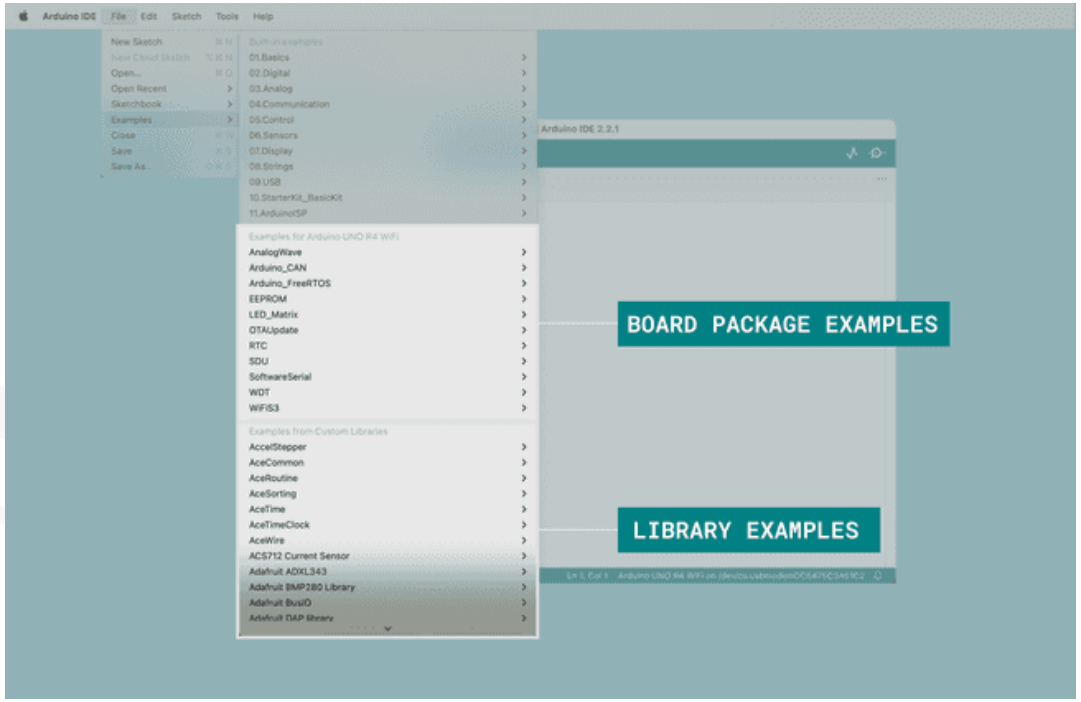
# Arduino IDE – Serial Plotter

- La herramienta Serial Plotter es ideal para visualizar datos mediante gráficos y supervisar, por ejemplo, los picos de tensión.
- Puede supervisar varias variables simultáneamente, con opciones para habilitar solo determinados tipos.



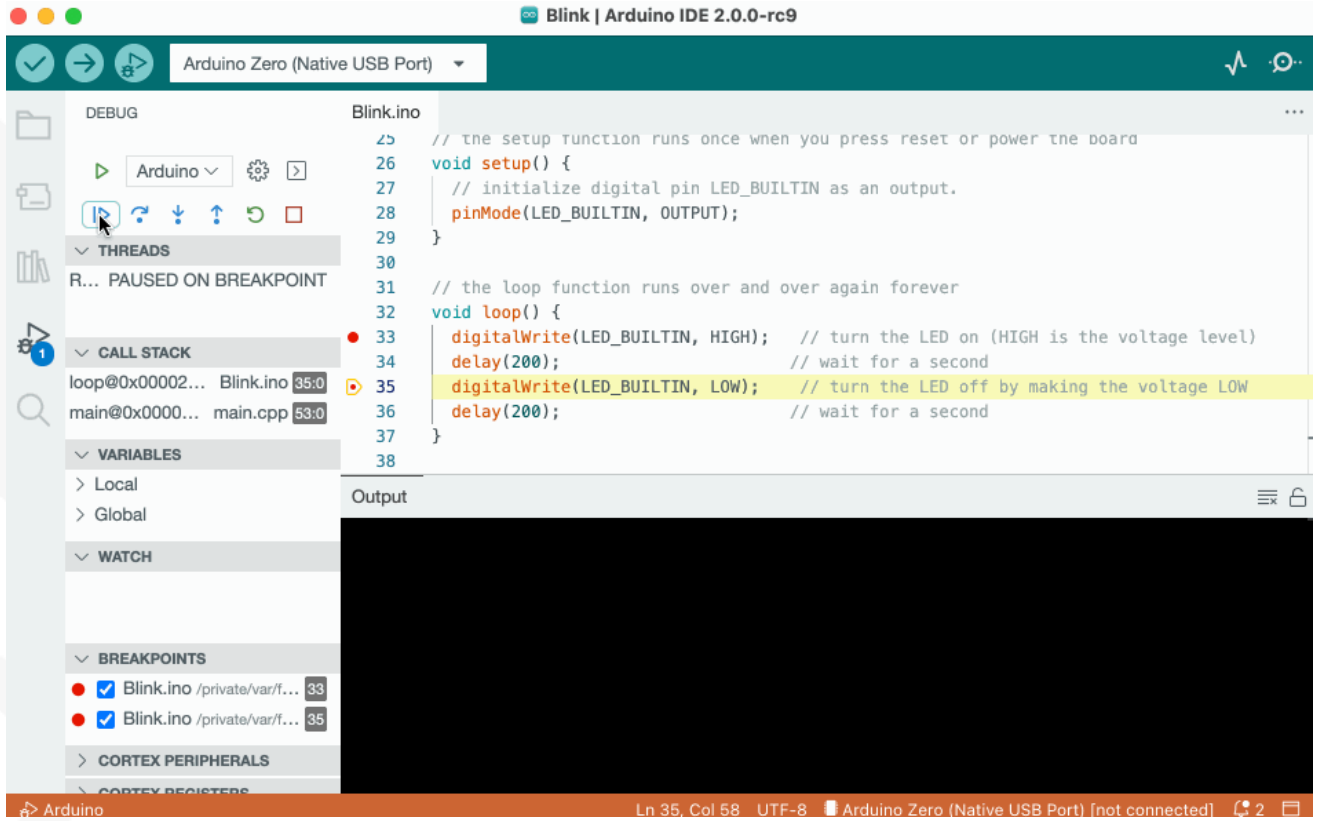
# Arduino IDE – Ejemplos

- Una parte importante de la documentación de Arduino son los ejemplos de proyectos que se incluyen con las bibliotecas. Estos muestran ejemplos de las funciones utilizadas en la práctica, ilustrando el uso previsto y las características de una biblioteca.



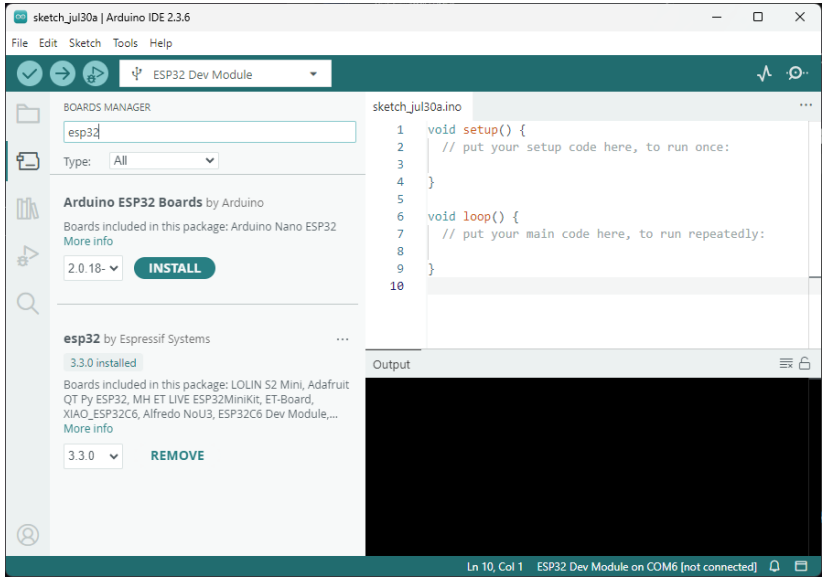
# Arduino IDE – Depuración

- La herramienta de depuración se utiliza para probar y depurar programas.
- Se puede utilizar para navegar por la ejecución de un programa de forma controlada.



# Arduino IDE – Configurar ESP32

- El ESP32 no viene incluido en el IDE de Arduino por defecto, pero es posible agregarlo utilizando el Gestor de tarjetas:
  - Menú Archivo > Preferencias. En el campo “URLs adicionales de gestor de tarjetas”, añadir la siguiente URL:
    - https://espressif.github.io/arduino-esp32/package_esp32_index.json
  - Menú Herramientas > Placa > Gestor de tarjetas.
    - En el panel que aparece, buscar “ESP32” en la barra de búsqueda, y pulsar en “Instalar” en el paquete que aparece.
- Una vez completada la instalación, ya se podrá elegir los modelos de ESP32 en el selector de placa.



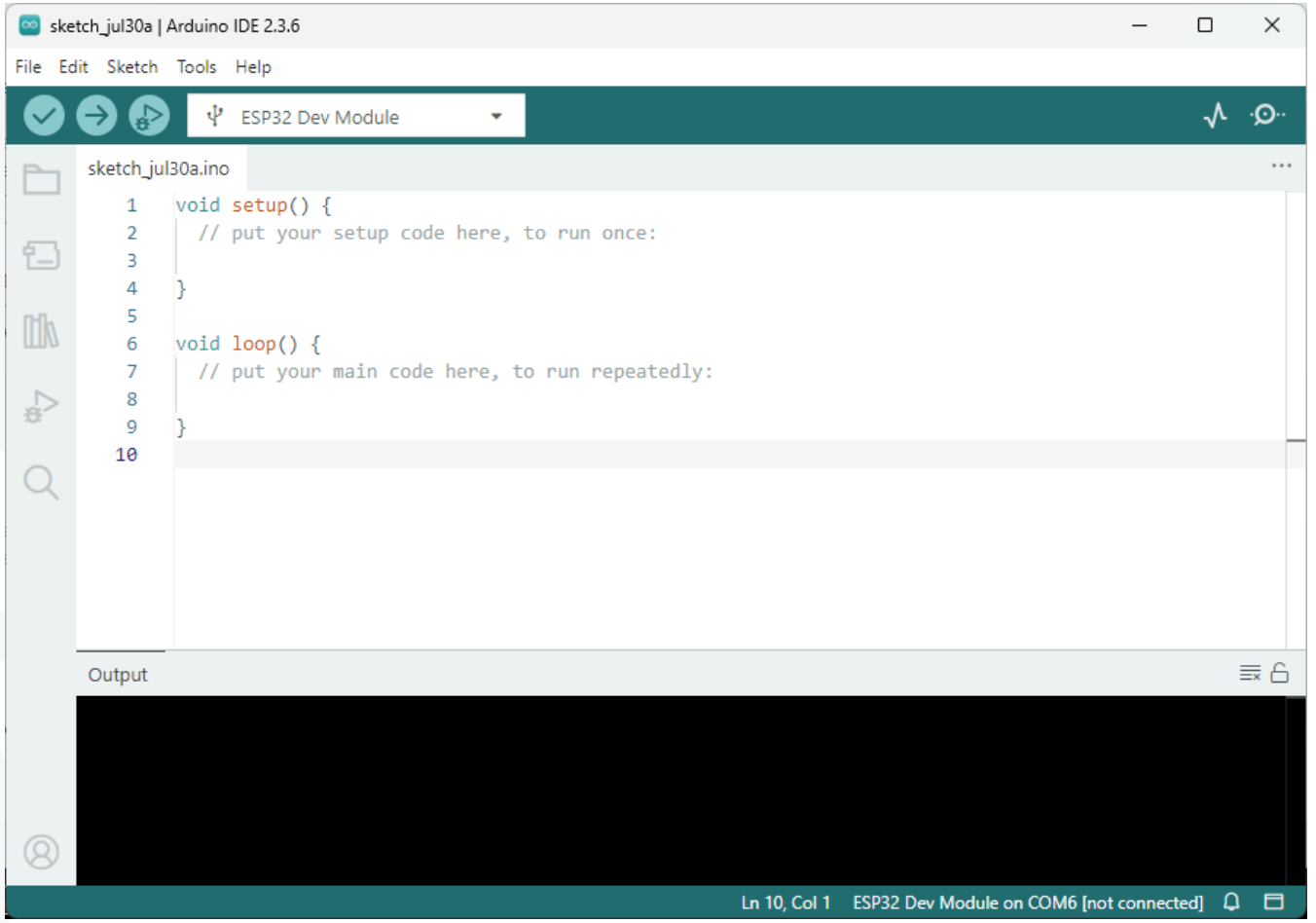
# Arduino IDE – Configurar ESP32

- Para verificar el correcto funcionamiento:
  - Conectar el ESP32 mediante un cable USB.
  - En el IDE de Arduino, seleccionar la placa a utilizar.
    - Herramientas > Placa y seleccionar “ESP32 Dev Module”.
  - Seleccionar el puerto COM al que está conectado el ESP32:
    - Herramientas > Puerto
    - Si no se conoce qué puerto es se recomienda desconectar el ESP32 y volver a esta sección para ver qué puerto desaparece y vuelve a aparecer cuando se vuelve a conectar el dispositivo.

# Programar en Arduino

# Estructura básica de un programa en Arduino

- Los programas de Arduino presentan 2 funciones principales:
  - setup: se ejecuta una vez cuando el Arduino enciende
  - loop: se ejecuta continuamente mientras está encendido



# Estructura básica de un programa

```
// -------- ADJUSTS ------- */
#define PIN_SENSOR 13
/* ------- LIBRARIES ------- */
#include <Servo.h>
#include "infoextra.cpp"
/* ------- VARIABLES/CONST ------- */
int temperature;
const float calibrationValue = 1.48;
/* ------- SETUP ------- */
void setup(){}
/* ------- LOOP ------- */
void loop(){}
/* ------- OTHER FUNCTIONS ------- */
void myFunction(){}
```

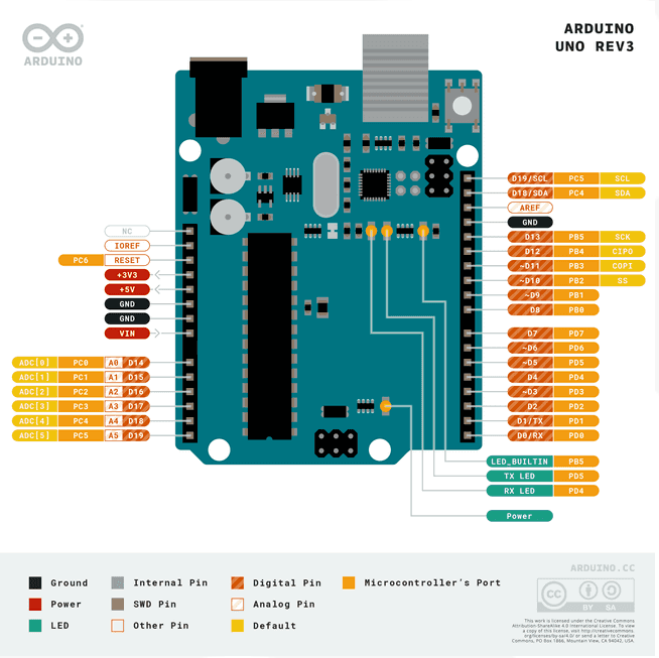
# Ejecutar un programa

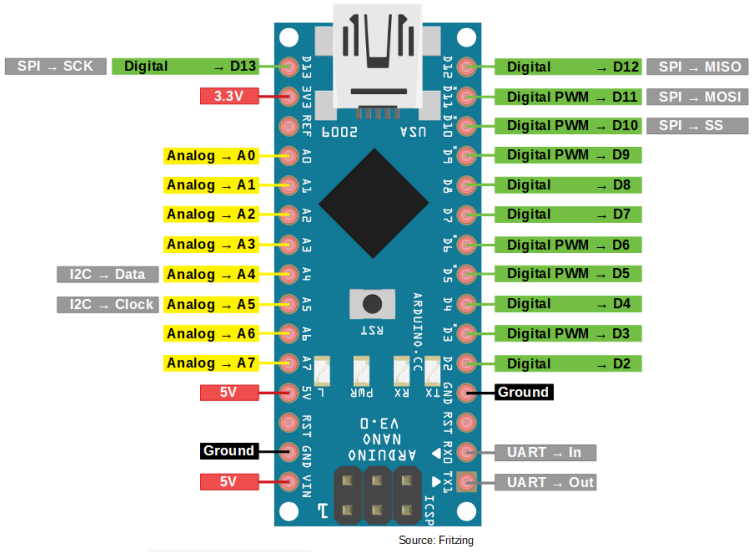
- Selección de placa y puerto:
  - Abre el IDE de Arduino.
  - Ve a "Herramientas" > "Placa" y selecciona el modelo de tu placa Arduino (por ejemplo, Arduino Uno).
  - Ve a "Herramientas" > "Puerto" y selecciona el puerto serial al que está conectada la placa.
- Carga del programa:
  - Abre el programa (sketch) que deseas ejecutar en el IDE.
  - Haz clic en el botón "Subir" (flecha hacia la derecha) para cargar el programa en la placa.
- Verificación:
  - Una vez que el programa se ha subido con éxito, la placa Arduino comenzará a ejecutarlo.
  - Si tu programa interactúa con la computadora (por ejemplo, a través de la comunicación serial), puedes abrir el Monitor Serial en el IDE para ver los datos.

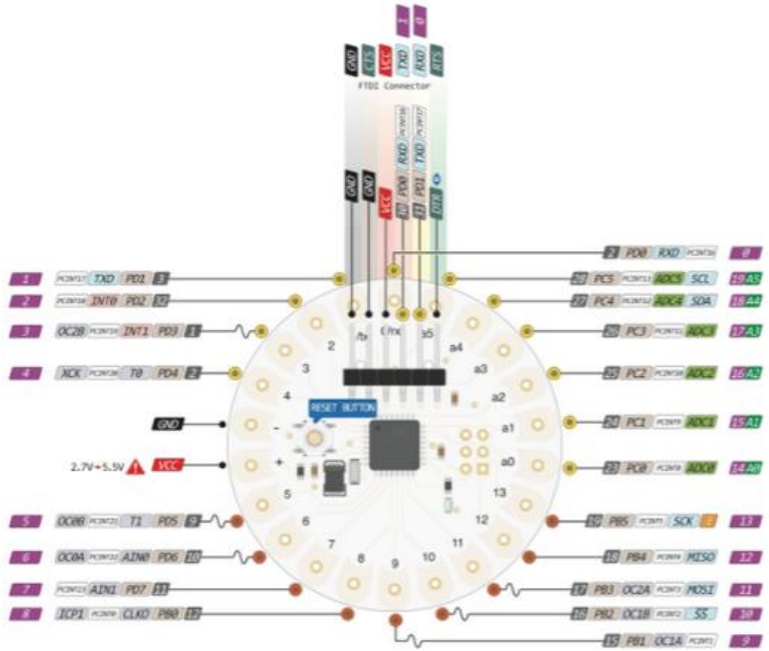
# Conexiones

# Conexiones placa

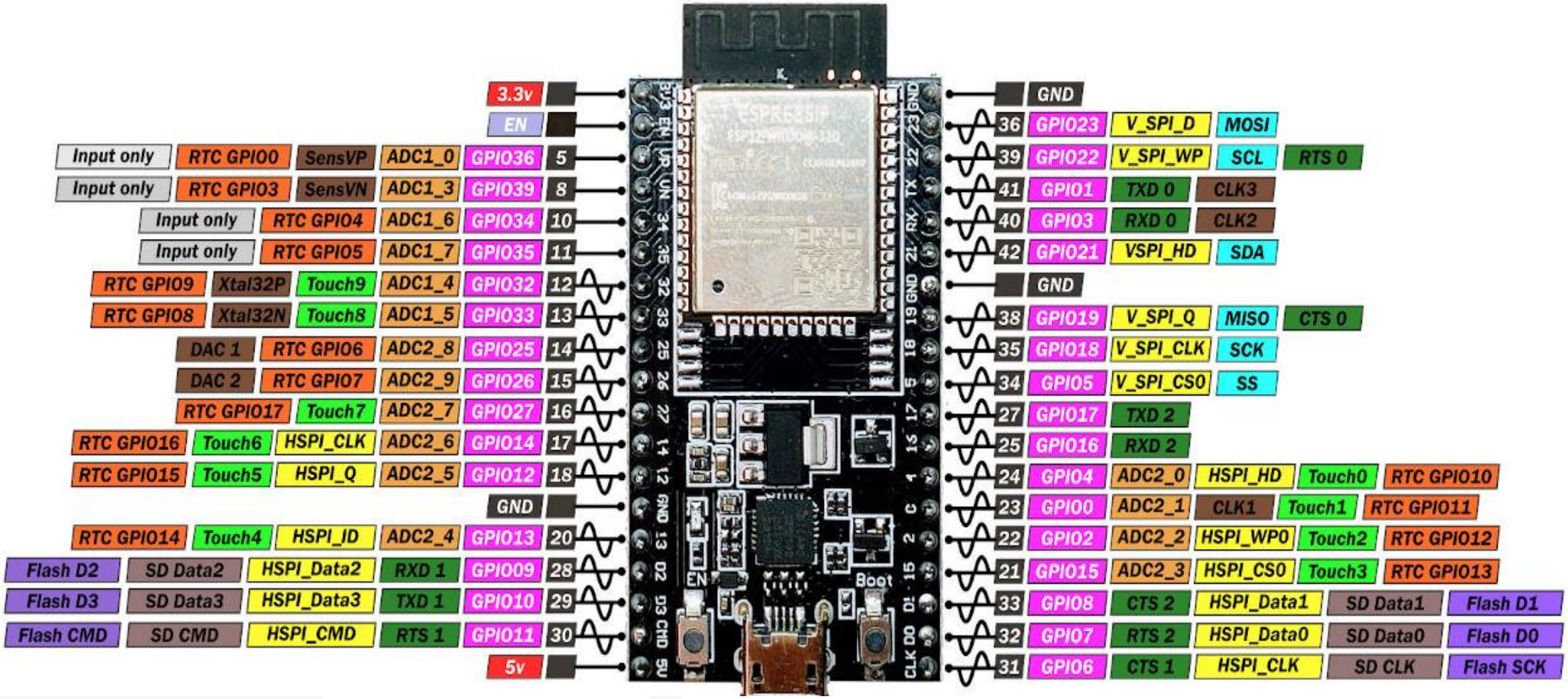
- Para conocer la funcionalidad de cada uno de los pines de la placa es necesario contar con el Pinout que será específico de cada placa



# ESP32



# Pines especiales en el ESP32

- Pines de configuración:
  - Intervienen en la configuración durante el arranque
- Pines GPIO RTC
  - Se utilizan para despertar el ESP32 del modo de bajo consumo
- GPIO de Entrada Únicamente
  - Solo pueden emplearse como entradas digitales o analógicas
- Pines táctiles
  - Cuando una carga capacitiva está cerca del pin detecta el cambio en la capacitancia
- Pin Enable (EN)
  - Controla el regulador de 3V3
- Pines de PWM
  - Pueden utilizarse para controlar motores y Leds digitales

# Pines especiales en el ESP32

- Pines ADC
  - Sirven para leer señales analógicas y convertirlas en valores digitales
- Pines DAC
  - Permite convertir señales digitales en señales analógicos
- Pines UART
  - Permite establecer comunicación serie asíncrona entre placa y otros dispositivos
- Pines I2C
  - Pines asociados al bus I2C. Por defecto incluye los siguientes pines: SDA y SCL
- Pines SPI
  - Permite la comunicación serial síncrona de alta velocidad con protocolo SPI