Maleta de la Innovación 4.0 IoT

UNIVERSITAT POLITÈCNICA DE VALÈNCIA
CAMPUS D'ALCOI

Proyectos IoT
con ESP32 STEAMakers y ArduinoBlocks

Documento resumen 6 sesiones formativas:

Repaso conceptos básicos
Conexionado modular
Conexión WiFi: STA y AP
Almacenamiento en tarjeta SD
Bluetooth y BLE HID: teclado, ratón
Protocolo MQTT: dashboards, Adafruit, ThingSpeak
Protocolo HTTP cliente/servidor
Dispositivo compatible Alexa
Integración con Telegram Bot
WiFiMesh

Formador:

Juanjo López
arduinoblocks@gmail.com

Índice

Sesión 1 - Presentación e introducción

Sesión 2 - Repaso sensores básicos y conexión WiFi

Sesión 3 - Tarjeta uSD, Bluetooth HID, MQTT

Sesión 4 - WiFi, Servidor HTTP, Cliente HTTP

Sesión 5 - Alexa , Telegram Bot

Sesión 6 - AppInventor + HTTP , WifiMesh

Bibliografía y enlaces de interés

Sesión 1

- Presentación del curso
- Entrega del material
- Introducción a la plataforma ArduinoBlocks
- Instalación de ArduinoBlocks-Connector
- Placa ESP32 STEAMakers
- Repaso del conexionado modular
- Sensores, actuadores y periféricos
- Primer proyecto ESP32 STEAMakers con ArduinoBlocks:

Material del curso:

- Kit Arduino UNO SmartHome usado en la formación anterior (para utilizar algún sensor, cables, pantalla, o cualquier dispositivo que utilizabamos con Arduino UNO ahora lo podremos usar con ESP32)
    - https://shop.innovadidactic.com/es/standard-placas-shields-y-kits/1455-keyestudio-smart-home-para-arduino-con-placa-keyestudio-plus.html

- Material extra para este curso:
    - Placa ESP32 STEAMakers
    - Pantalla OLED
    - Reloj RTC (+pila)
    - Lector RFID
    - Tarjeta uSD

¿Qué es IoT?

IoT viene del inglés *"Internet Of Things"*, es decir, *"Internet de las cosas"*.
La definición de IoT podría ser la agrupación e interconexión de dispositivos y objetos a través de una red (bien sea una red privada o bien Internet), dónde todos ellos podrían ser visibles e interaccionar.

IoT tiene grandes aplicaciones en la industria 4.0, en la domótica e inmótica, y en otras areas como agricultura, control de flotas, redes de sensores/actuadores en ciudades, etc.

Los dispositivos IoT deben tener capacidades de conectividad (normalmente inalámrbrica: WiFi, Bluetooth, LoRa, etc.), deben poder gestionar sensores y actuadores de diversos tipos y es recomendable que tengan bajo consumo energético. El microcontrolador ESP32 es un chip potente, con altas capacidades de conectividad y el caso de la ESP32 STEAMakers en formato Arduino y con conexión modular lo que nos ayudará a acercar el desarrollo IoT a ambientes educativos y prototipos funcionales que hasta hace poco estaban al alcance de pocos (por dificultad técnica tanto en el hardware necesario como en la forma de programarlo).

Placa ESP32 STEAMakers

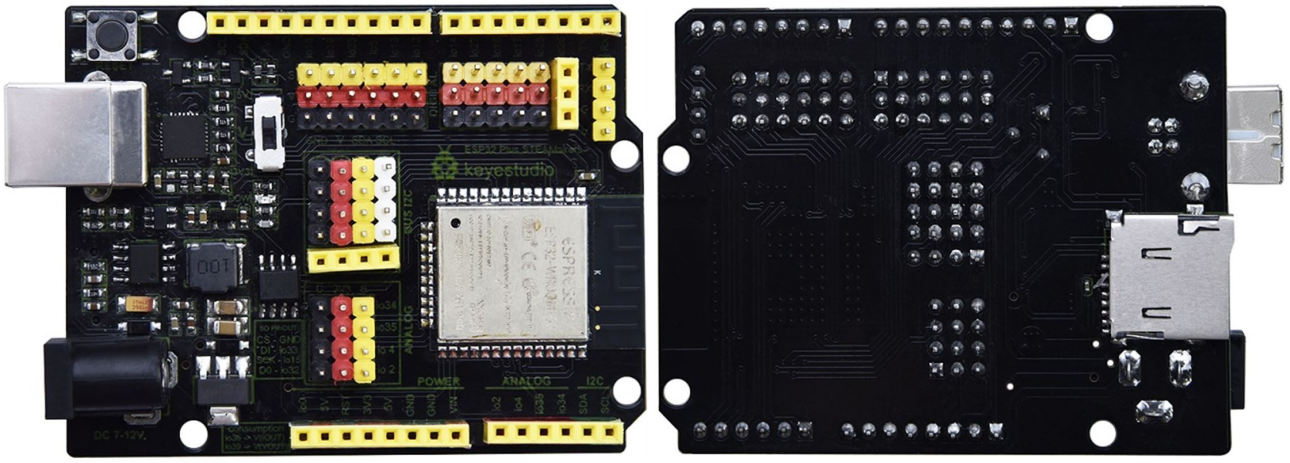
La placa ESP32 STEAMakers es una placa con formato Arduino desarrollada conjuntamente por Juanjo López (ArduinoBlocks), la empresa Innovadidàctic y el fabricante Keyestudio.
Es una placa pensada para el mundo educativo. Está basada en el potente microcontrolador ESP32. Tiene una conexión modular para evitar utilizar placas de prototipos (breadboard) y además incluye algunos sensores incorporados y una ranura para utilizar una tarjeta uSD en nuestros proyectos.

Más información y tienda oficial:

https://shop.innovadidactic.com/es/standard-placas-shields-y-kits/1567-placa-esp32-steamakers.htm

Videos demostrativos:

https://www.youtube.com/watch?list=PL1pKD-Bz2QBAgfy580m8OaQ2Z60v6DOhC&v=MQjIEI7I4ik&feature=emb_logo



Arduino

Web oficial y Arduino IDE

https://www.arduino.cc/

ArduinoBlocks

ArduinoBlocks es una plataforma de programación por bloques inicialmente pensada para placas tipo Arduino UNO que actualmente soporta más de 15 tipos de placas y tipos de proyectos, acercando al mundo educativo sistemas tan potentes como ESP8266 o ESP32 a los alumnos y docentes desde etapas tempranas.

ArduinoBlocks nación como un proyecto del profesor Juanjo López en Salesianos Juan XXIII en el año 2016 y actualmente es una herramienta utilizada en todo el mundo, traducida a 5 idiomas y en continuo desarrollo.

Además de programar las placas Arduino o ESP, ArduinoBlocks incorpora herramientas y funcionalidades específicas para el mundo docente. Documentación de los proyectos, proyectos supervisados por el profesor, posibilidad de compartir proyectos, etc.

http://www.arduinoblocks.com/

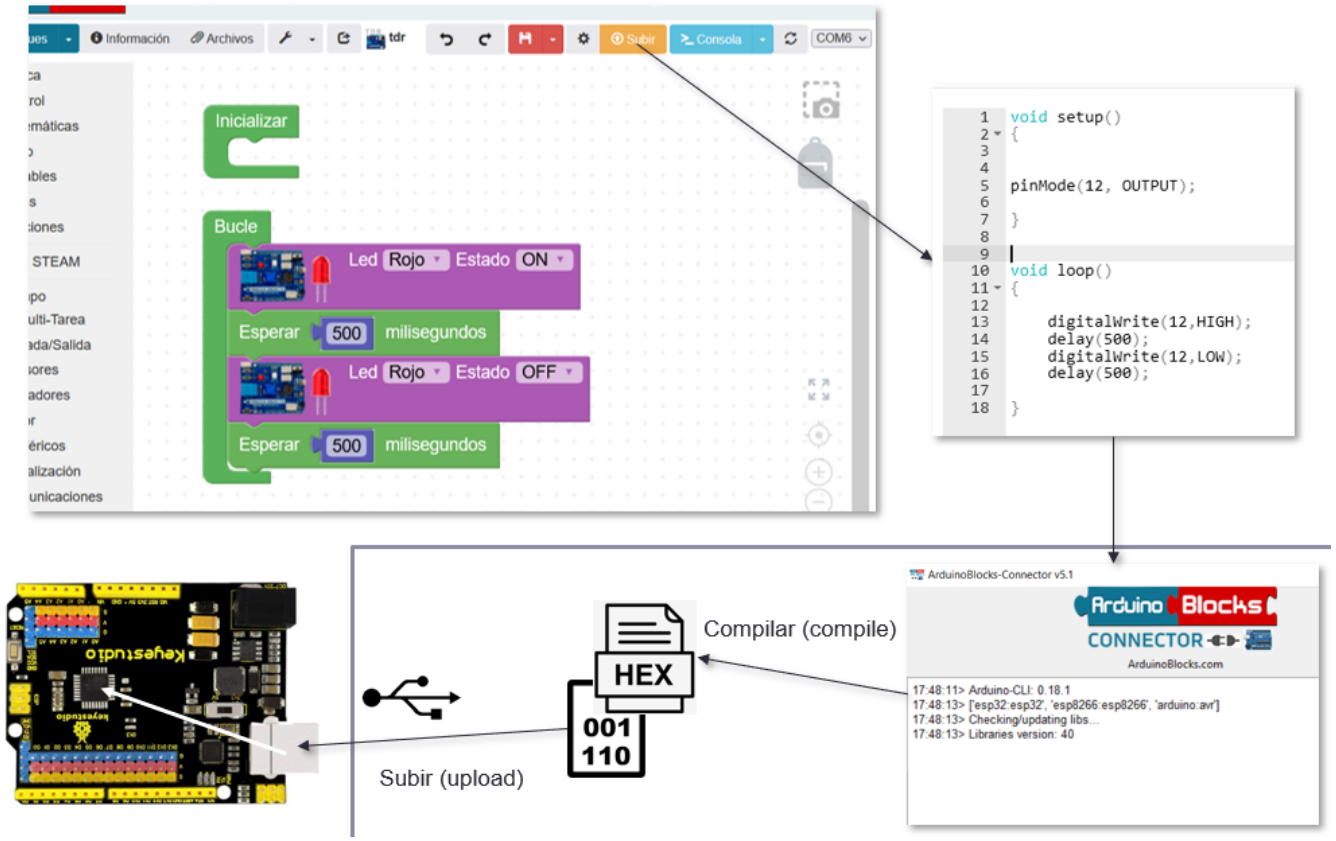
ArduinoBlocks Connector

ArduinoBlocks Connector es la aplicación que se encarga de recibir los datos de la web de ArduinoBlocks y “compilar” y “subir” el progama a la placa coenctada físicamente al PC a través del USB. Es una aplicación que utiliza internamente el framework oficial de Arduino y ESP.

Compilar: convertir el código fuente generado por ArduinoBlocks + las librerías necesarias en un archivo binario que se grabará en la memoria de la placa.

Subir (upload): Transferir el archivo binario generado en el proceso de la compilación enviándolo a través de la conexión serie vía USB.

Esquema del funcionamiento de ArudinoBlocks:



Descargar e instalar ArduinoBlocks-Connector v5

http://www.arduinoblocks.com/web/site/abconnector5

Windows: Instalar driver para ESP32: CP2102 usb-serial converter

Linux/Ubuntu: Instalar connector y "fix serial esp32"

Chromebook: Instalar connector en modo linux + "fix serial esp32"

macOS: Instalar versión portable + driver CP2102

Recopilación de documentación sobre ArduinoBlocks

http://www.arduinoblocks.com/web/site/doc

Libro online (actualizado periódicamente de forma automática):

https://docs.google.com/document/u/1/d/e/2PACX-1vQSrOKHpbLQHVbGFdAvp7DcndoftoHDI20nvwGMaxu_7bGc1bUCmi4U6DZrJWRSudc2iXBg43QMuzCT/pub

Presentación "Introducción a ArduinoBlocks"

https://drive.google.com/file/d/1C67ni-USMJ2SzTJAeAGUog4xT7t1ty8p/view?usp=sharing

Empezando con ArduinoBlocks

- Registro / Inicio de sesión
  - http://www.arduinoblocks.com/web/site/login

- Mis proyectos
  - http://www.arduinoblocks.com/web/site/projects

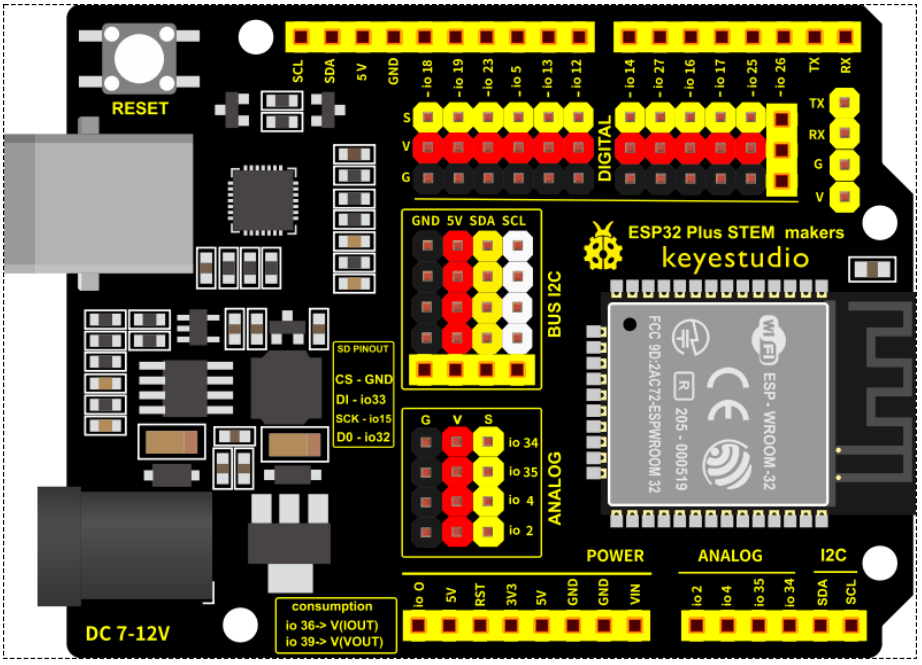
- Nuevo proyecto
  - Tipos de proyectos
  - Documentación del proyecto

- Proyectos para los alumnos
  - Proyecto para alumno nuevo (vacío)
  - Proyecto para alumno a partir de proyecto personal
  - Crear enlaces a proyectos (para moodle, classroom, …)

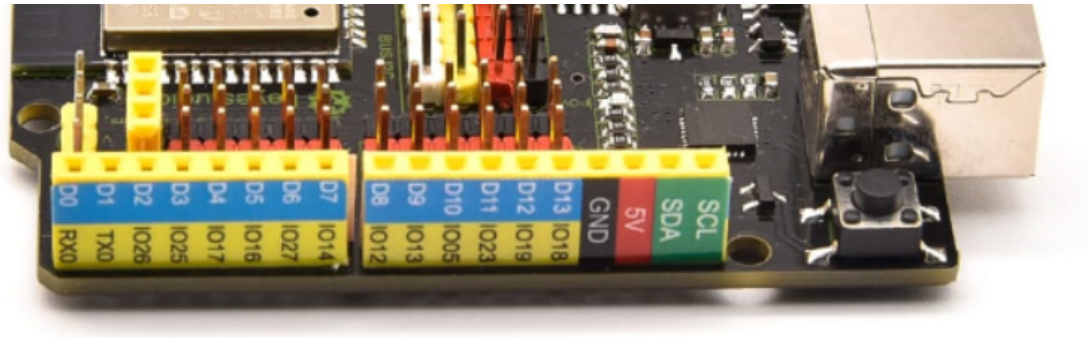
- Repaso de conexión modular en ESP32 y Arduino

- Crear un proyecto personal tipo “*ESP32 STEAMakers*”
  - Conectar un led al pin IO26 (no hace falta cableado, se puede “pinchar” directamente en los pines hembra)
  - Hacer un parpadeo cada 500ms

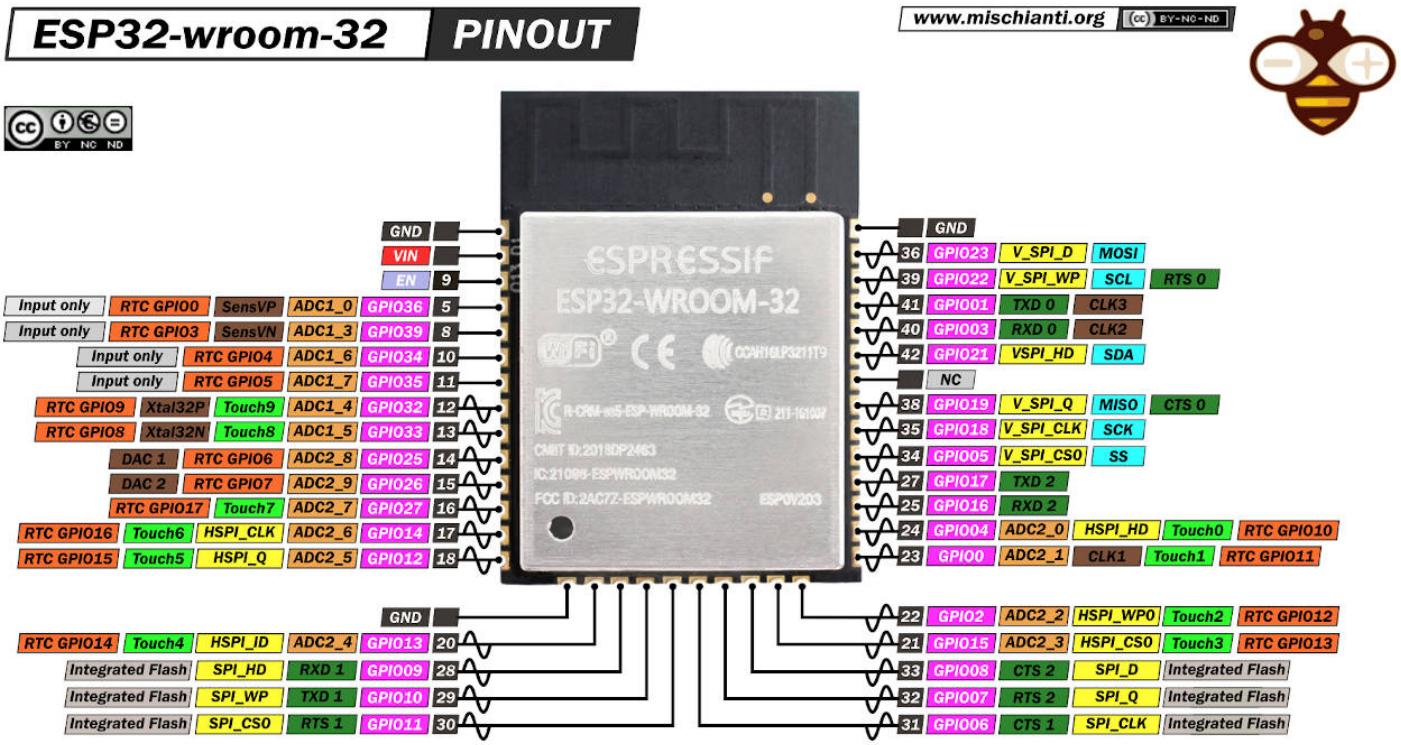
Conexión modular



Las conexiones laterales *“tipo Arduino”* vienen doblemente numeradas: numeración pines según Arduino UNO, y con el nombre interno correspondiente al chip ESP32



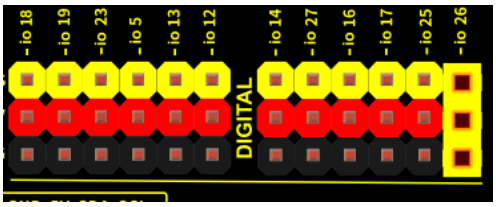
Esquema original de los pines del chip ESP32:



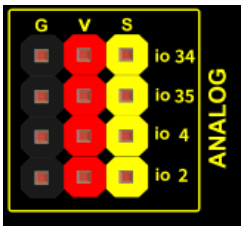
En la placa ESP32 STEAMakers, a parte de la conexiones propias del formato Arduino, tenemos los pines accesibles junto a GND y VCC de la forma standard:

AMARILLO: Pin de señal conectado al micro *(señal, S)*
ROJO: VCC, 3.3v o 5v *(positivo, +)*
NEGRO: GND, 0v *(negativo, -)*

-**Pines digitales** (algunos pueden funcionar como analógicos también)

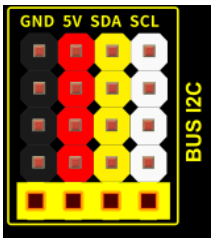


-**Pines analógicos** *(entrada ADC 12 bits: valor leído entre 0…4095)*



-**Bus I2C**

(hub incorporado para conectar hasta 5 dispositivos, podemos seguir ramificando el bus I2C)

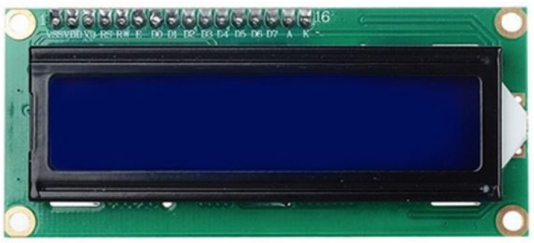


Si necesitamos expandir el bus I2C con más dispositivos podemo usar una placa de prototipos o más fácilmente mediante un hub i2c conectado a cualquiera de las conexiones:

https://shop.innovadidactic.com/es/standard-perifericos/872-keyestudio-hub-i2c.html

Además algunos dispositivos ya están pensados para continuar el bus i2c permitiendo anidar dispositivos i2c de forma sencilla.

Pantalla LCD i2c con hub i2c incorporado para “anidar” varias pantallas u otros sensores i2c:

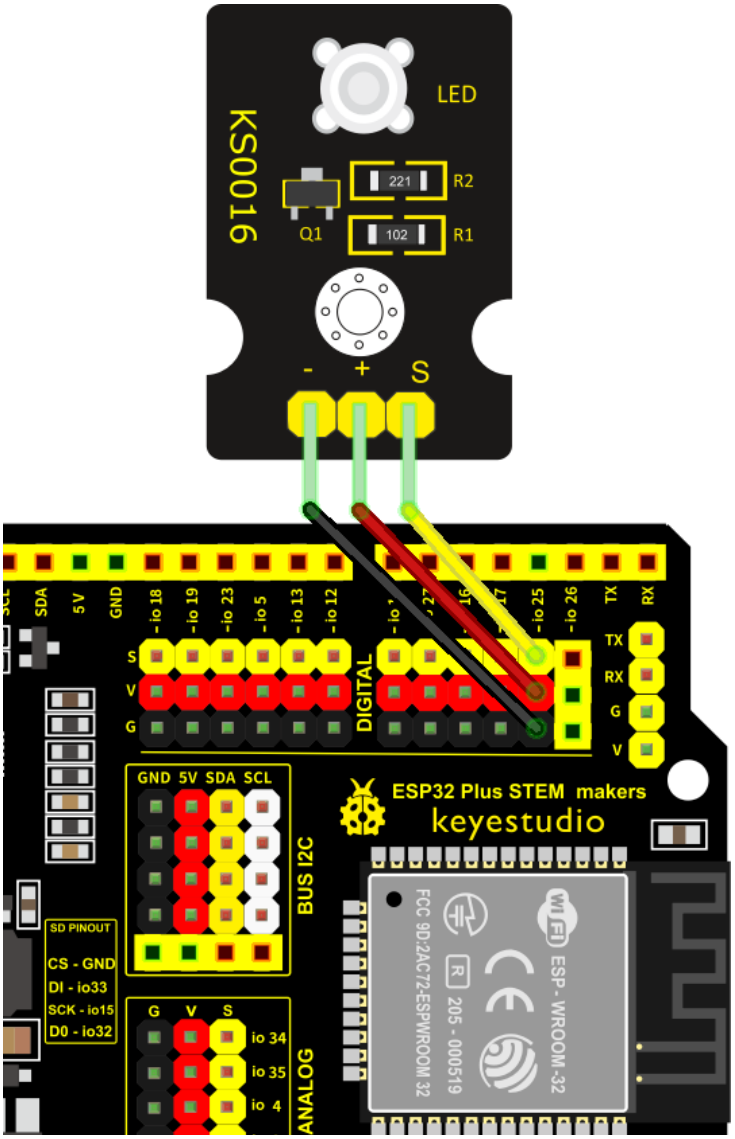


https://shop.innovadidactic.com/es/standard-perifericos/1570-keyestudio-pantalla-lcd-1602-i2c-direccionable-con-dos-conectores-i2c-macho-y-uno-hembra.html

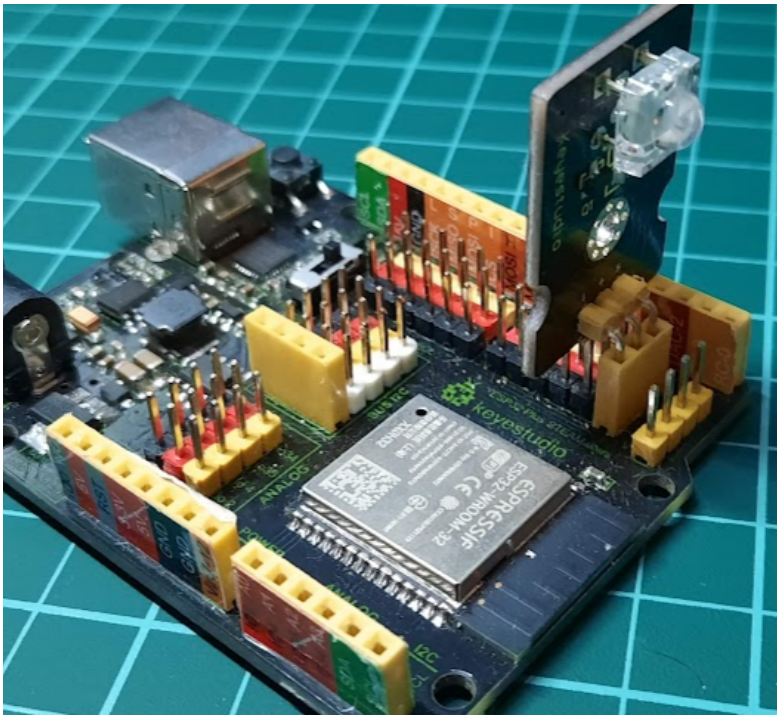
-Pines de comunicación RX/TX

Están conectados directamente a los pines RX,TX del conversor USB <-> serie, por lo que **si conectamos algo a estos pines puede interferir en el proceso de subida (upload) del programa** desde el PC

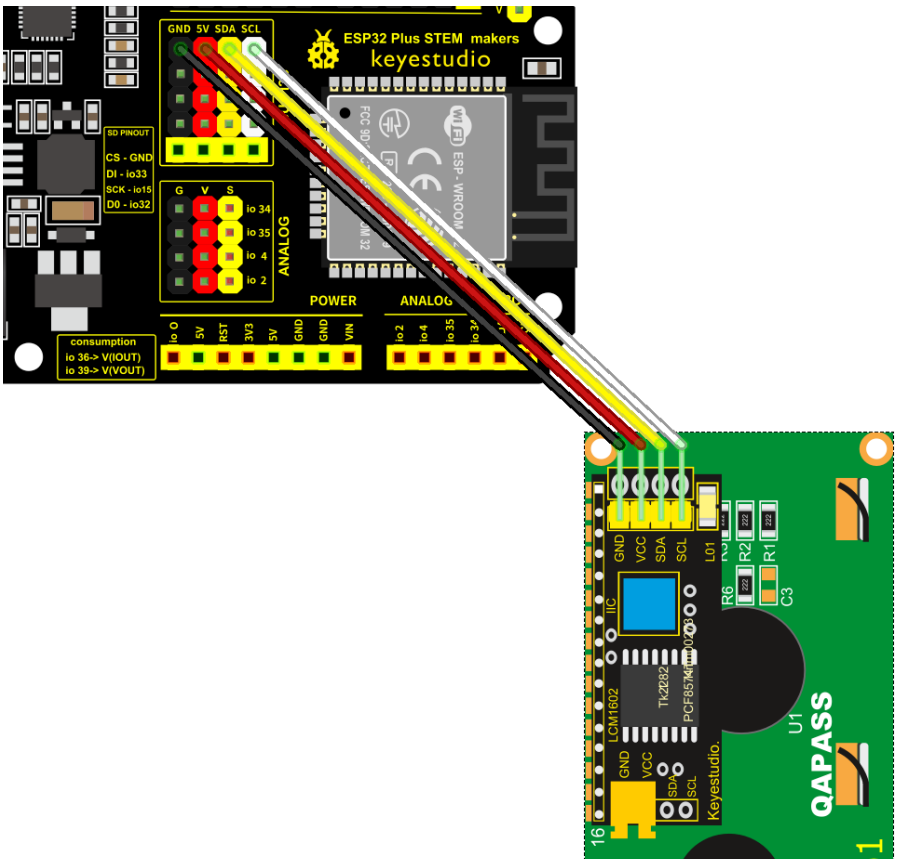
Ejemplo de conexión de un módulo Led de forma modular:

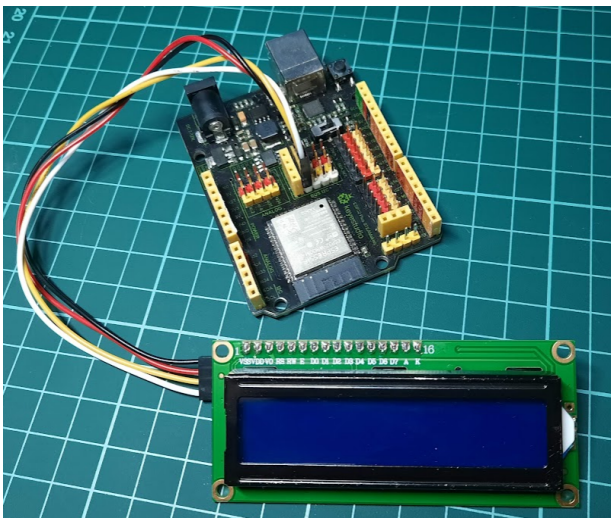


En algunos casos podemos incluso conectar el módulo a los pines hembras y así evitar conectar cables y simplificar el montaje (para pequeñas pruebas o cuando solo necesitamos un sensor)



Y el mismo proceso para dispositivos con conexión I2C:



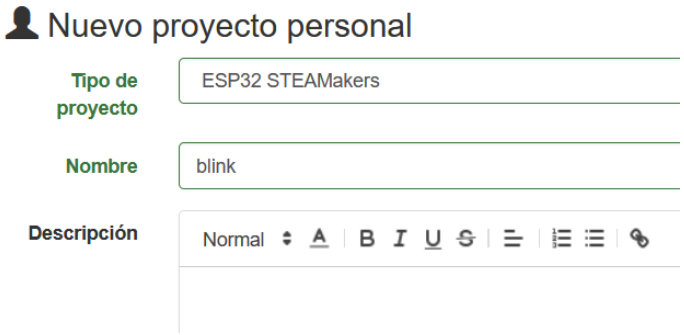


Programa de prueba (final de la sesión 1)

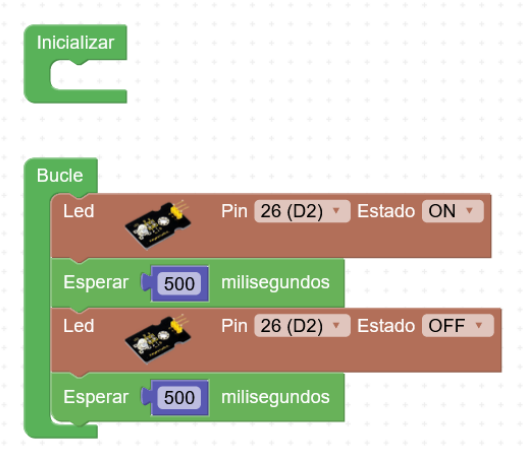
Conectando el módulo led al pin io26 (para usarlo directamente sin cables, pinchado en los conectores hembra) , hacer que el led parpadeé cada 500ms

1) Nuevo proyecto

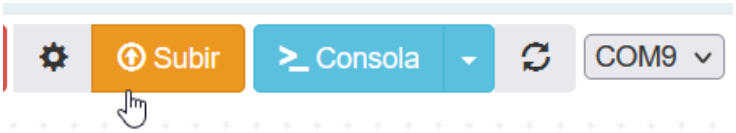
2) Seleccionar tipo “ESP32 STEAMakers” y ponerle un nombre



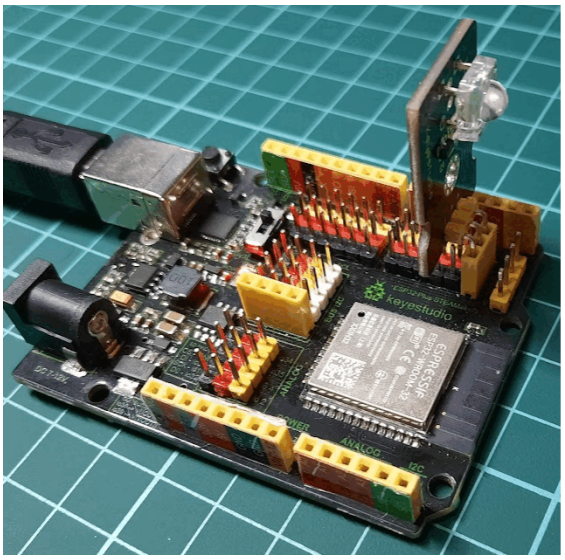
3) Crear el siguiente programa:



4) Con el connector abierto debe aparecer el listado de puertos (si hay varios debemos averiguar cual es el correcto), le daremos a subir para realizar el proceso de compilar y subir el programa al microcontrolador de la placa ESP32.



5) Una vez grabada la placa es autónoma, podemos usar una batería portátil o alimentador externo para alimentarla sin necesidad de conectarla al PC

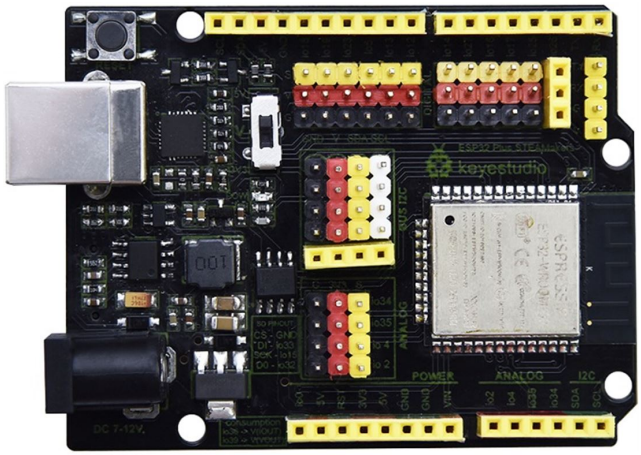


Sesión 2

- Repaso sensores y actuadores básicos (led, relé, pulsador, etc.)
    - Consola serie y SerialPlotter
- Sensores extra: Pantalla OLED, RFID, RTC
    - Textos en pantalla OLED
    - Gráficos vectoriales en pantalla OLED
    - Mapas de bits en pantalla OLED
    - Lectura de RFID
    - Control de acceso con RFID
    - Fechay hora con RTC
- Conexión a una red WiFi
- Obtener fecha y hora desde internet con NTP
    - Mostrar fechay hora desde internet en pantalla OLED

-Material extra suministrado para completar el kit de la maleta de la innovación 4.0

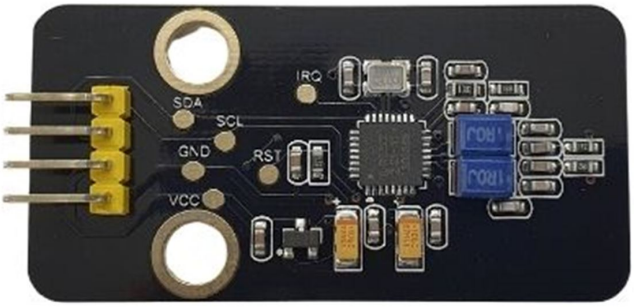
Placa ESP32 STEAMakers



Pantalla OLED i2c 128x64 píxeles

Reloj RTC i2c + Pila

Lector RFID i2c



Tarjeta uSD + Adaptador

Prácticas

2.1.-Repaso

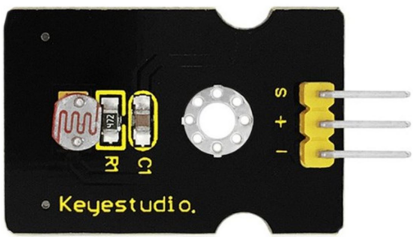
-Numeración pines

Los pines incluyen la doble numeración: numeración Arduino y numeración ESP32

Pines digitales: D0 (rx), D1 (tx) , D2 (26) , D3 (25) , D4 (17), D5 (16) , D6 (27), D7 (14) …
Pines analógicos: A0 (2), A1 (4), A2 (35) , A3 (34)
Pines i2c: SDA, SCL (conectados también en la posición de los pines A4/A5)

-Conexión modular

Para cada pin de señal (digital, analógico o i2c) tenemos junto a ese pin uno de alimentación (VCC) y otro de GND (0v)



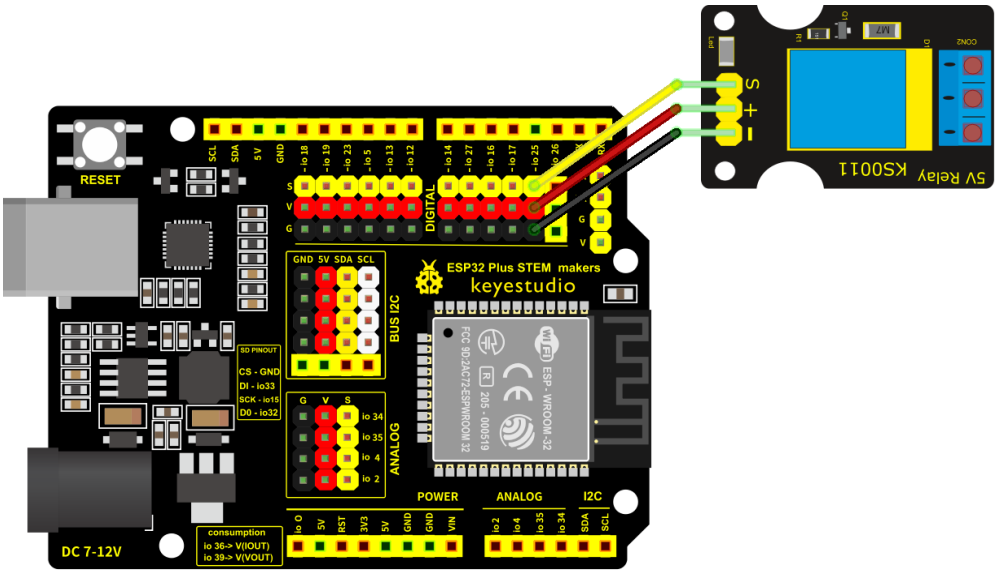
S -> Señal (amarillo)

\+ -> VCC , 5v, 3.3v (rojo)

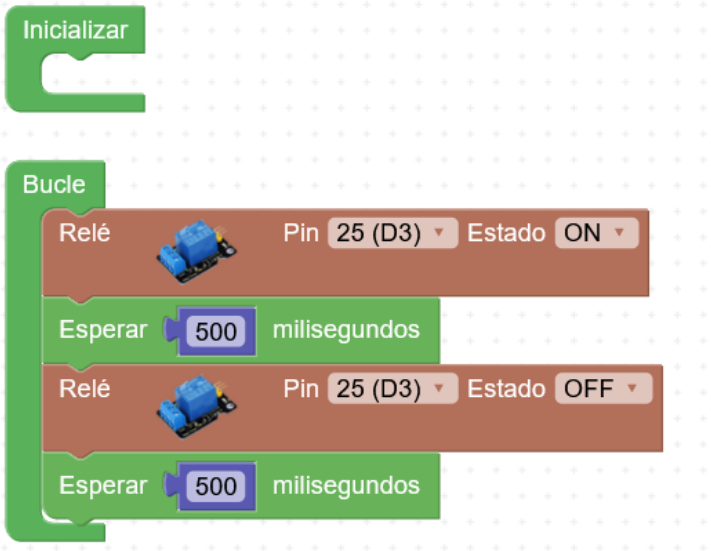
\- -> GND, 0v (negro / azul)

2.1.1.-Conexión de un relé y activación por temporización

Esquema de conexión modular

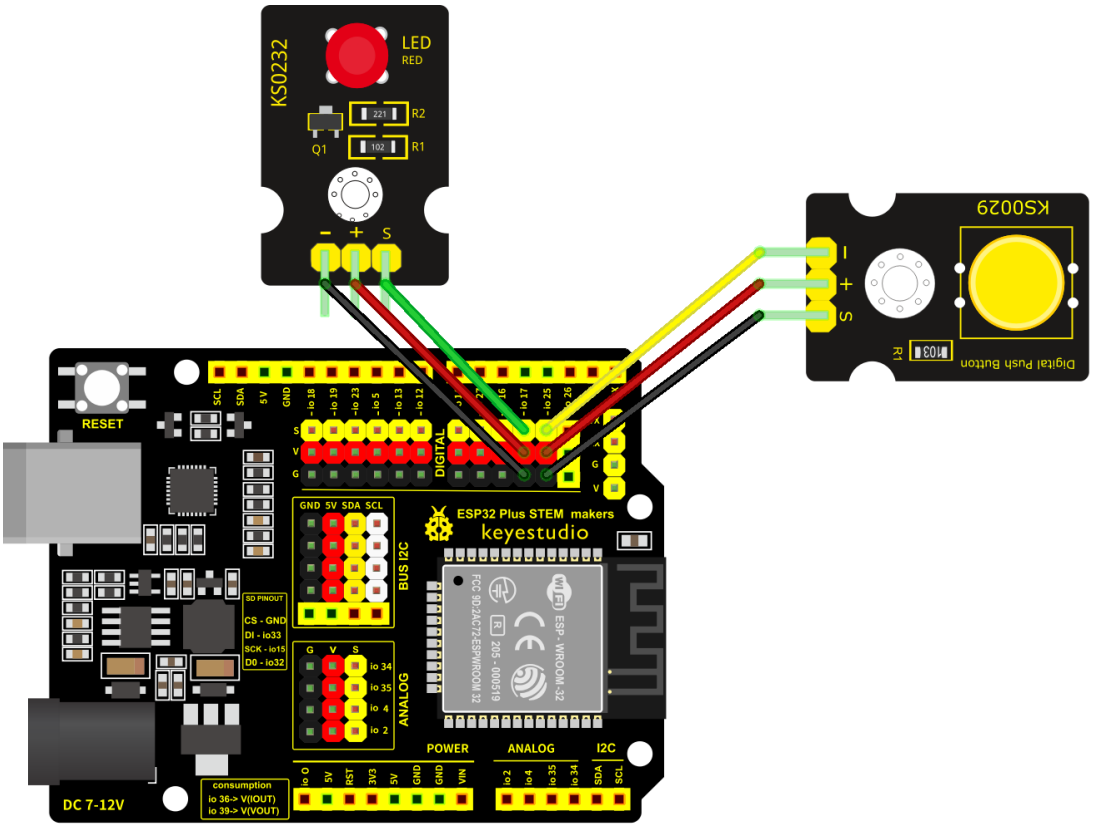


Programa de ejemplo

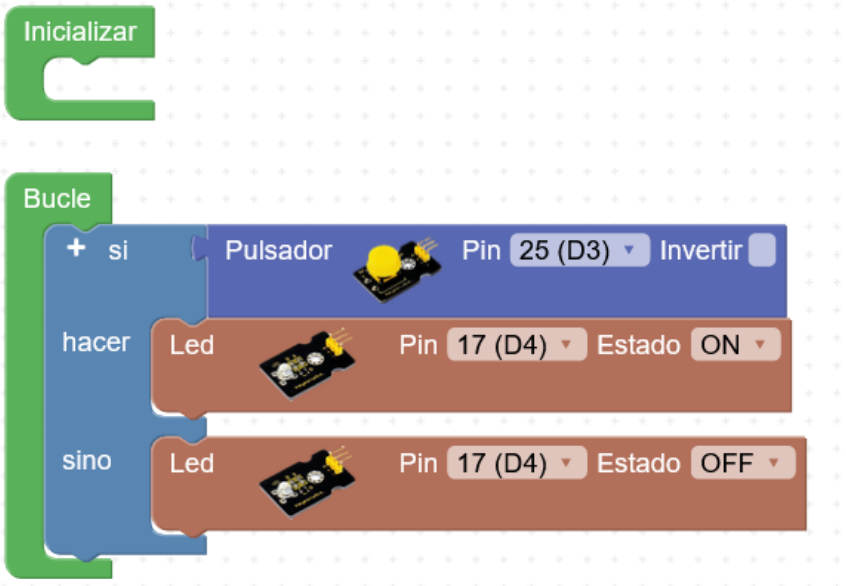


2.1.2.-Conexión de un pulsador y un led

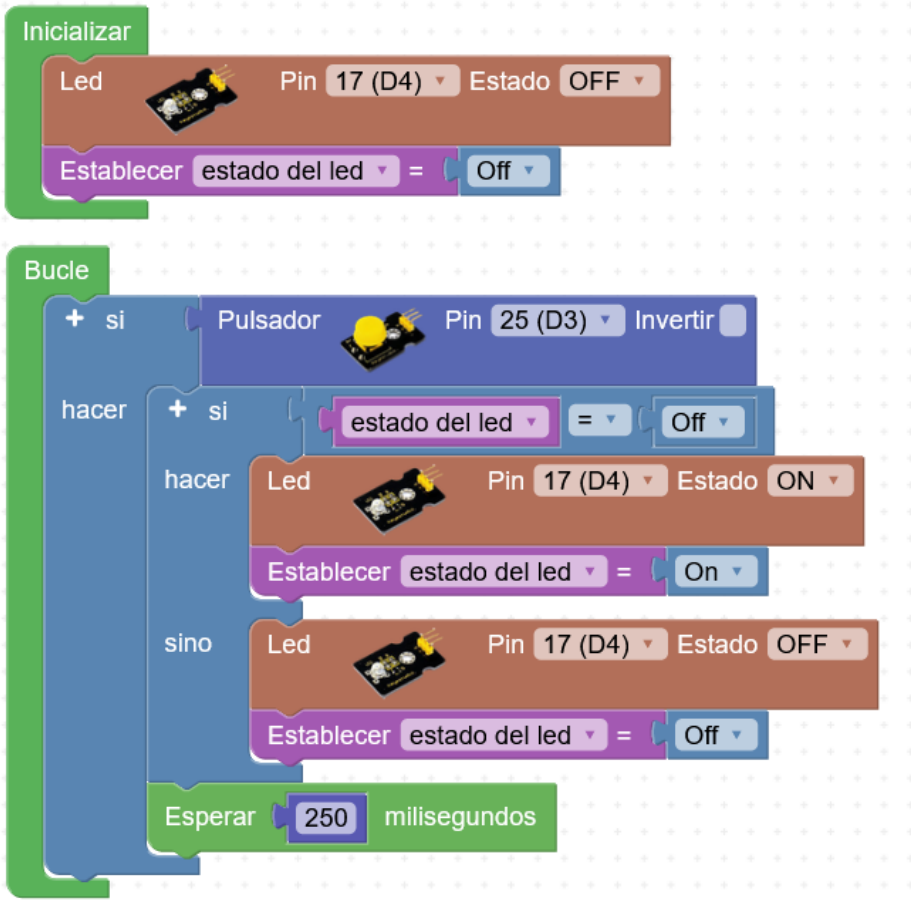
Esquema de montaje



Programa de ejemplo (encendido mientras está pulsado)

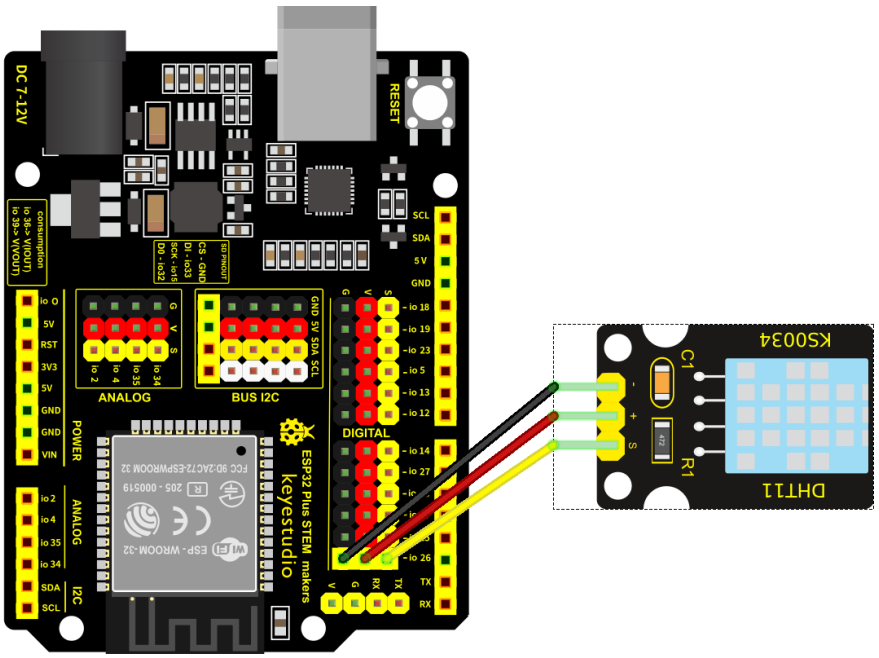


Programa de ejemplo (encendido/apagado)

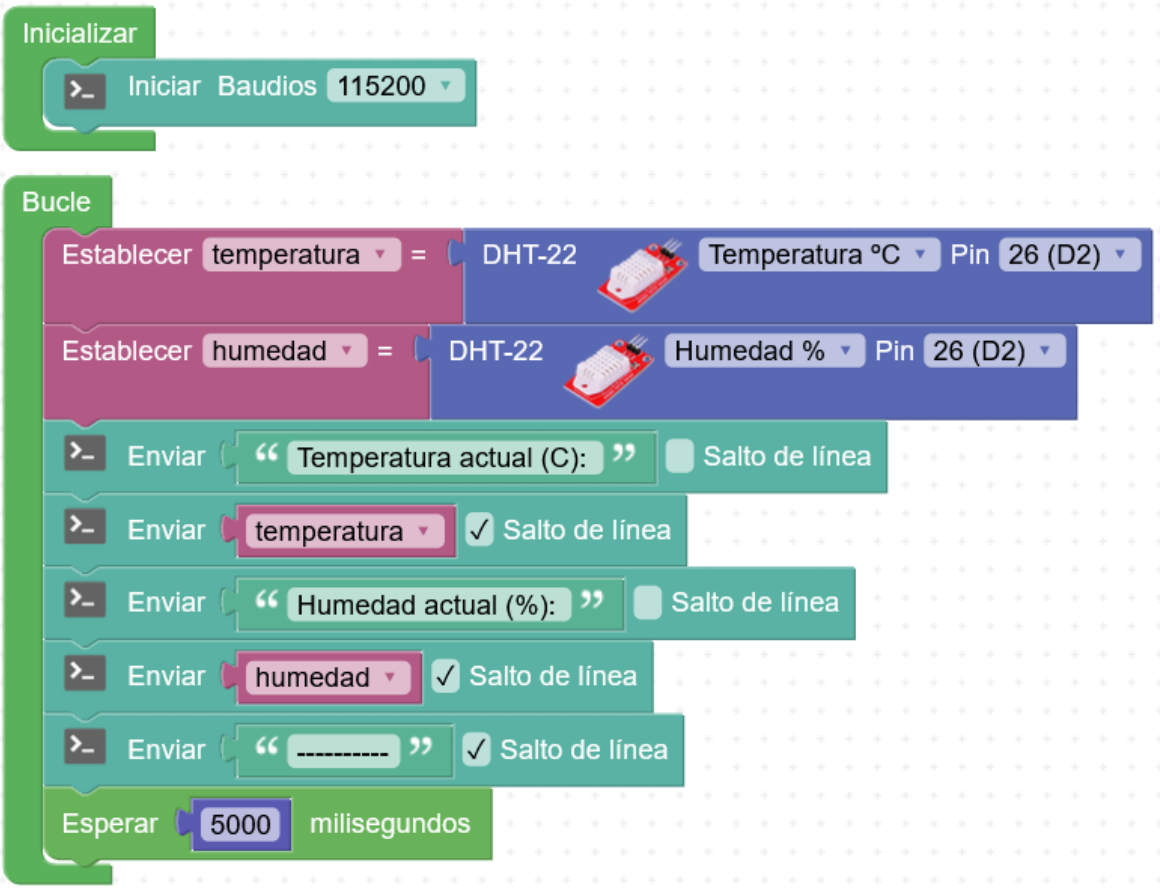


2.1.3.-Sensor DHT22 (temp/hum) y envío de datos a la consola serie / serial plotter (excel)

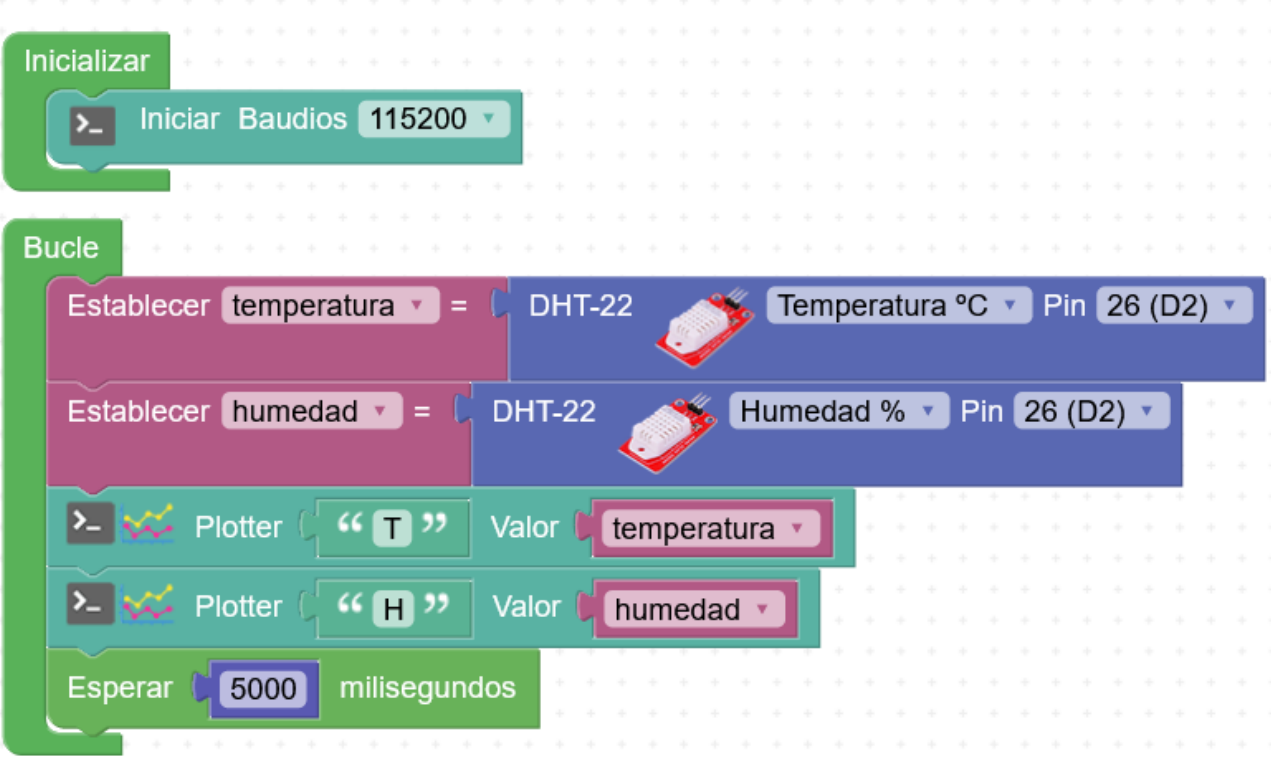
Esquema de montaje (el sensor DHT22 es el de color blanco):



Programa de ejemplo (consola serie):



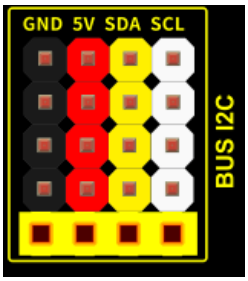
Programa de ejemplo (serial plotter)



2.2.-Pantalla OLED

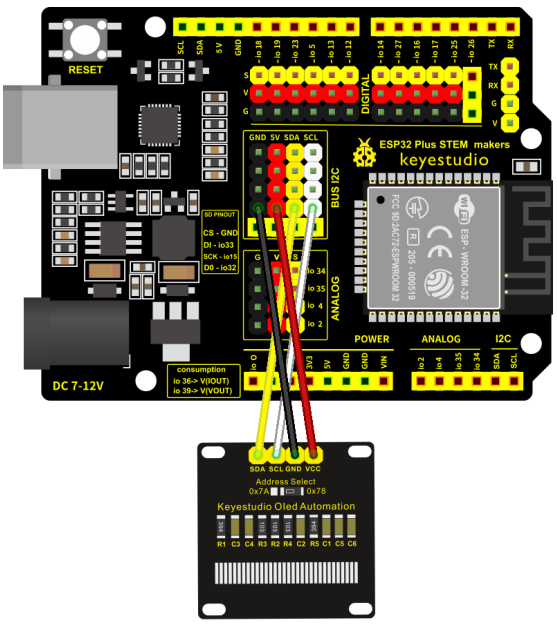
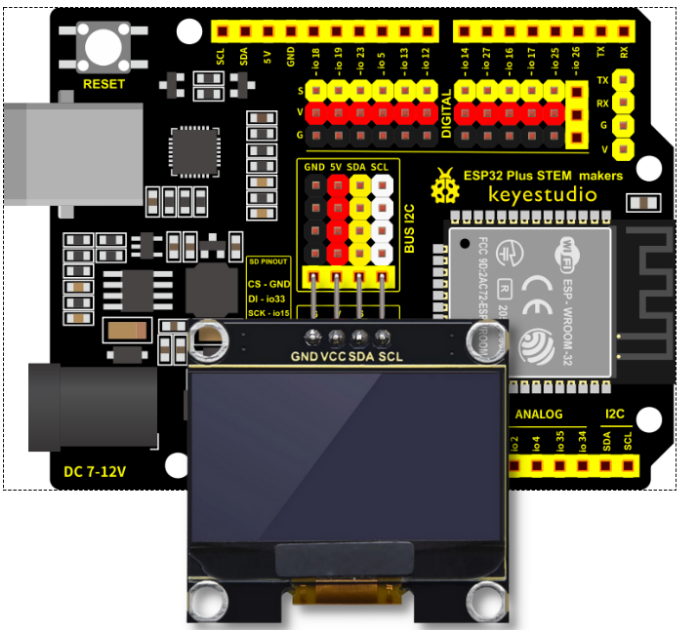
La pantalla OLED es un periférico que funciona por conexión i2c. En la placa ESP32 STEAMakers tenemos la posibilidad de conectar varios dispositivos i2c sin necesidad de un hub externo (la propia placa incluye un hub i2c donde conectar hasta 5 dispositivos). En caso de necesitar más podemos conectar un hub a cualquiera de los pines i2c y seguir ramificando el bus.

Señales I2C: GND (-), VCC (+) , SDA , SCL



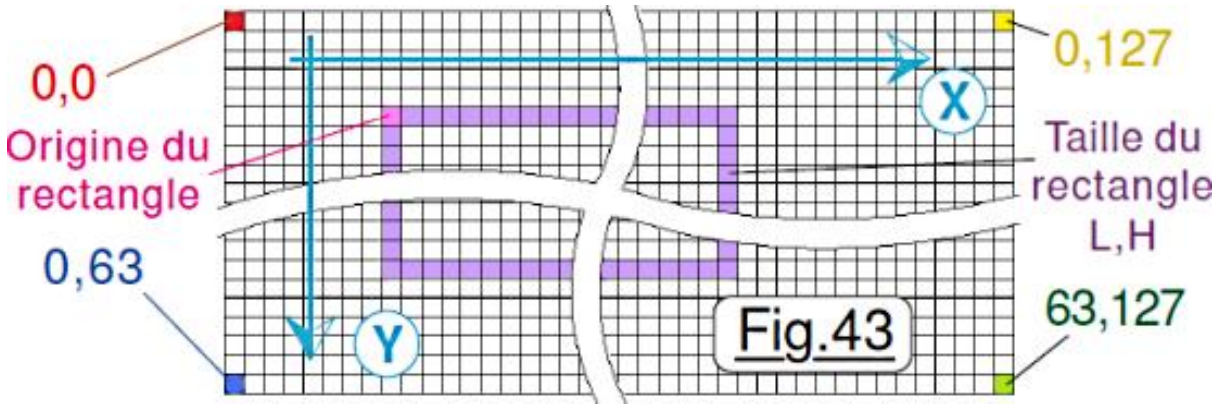
+Info sobre bus i2c:
https://www.luisllamas.es/arduino-i2c/

En el caso de la pantalla OLED podemos utilizar los pines hembra para conectar la pantalla a la placa sin necesidad de cables (asegurarse el orden de los pines en caso de utilizar pantallas o periféricos i2c de otro fabricante diferente a keyestudio)

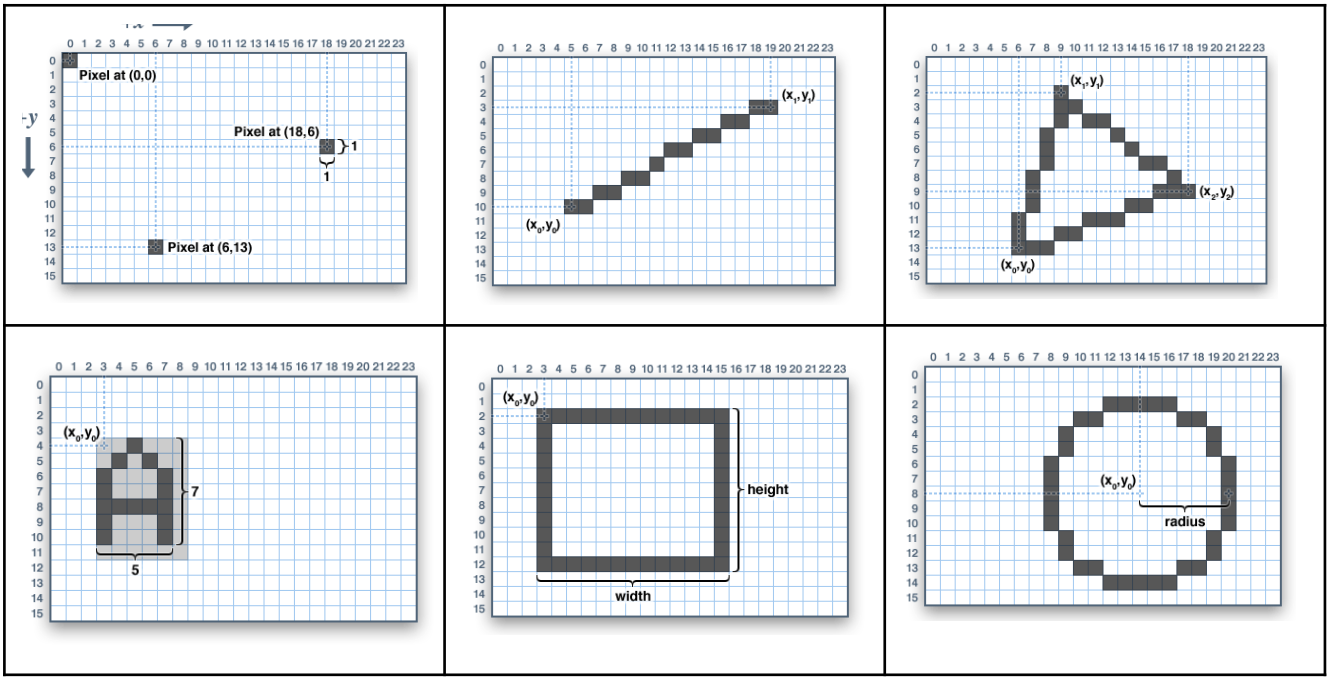
Conexión cableada	Conexión directa a pines hembra i2c
	

Sistema de coordenadas en la pantalla OLED 64x128 píxeles:

Coordenada **X**: 0 … 127
Coordenada **Y**: 0 … 63

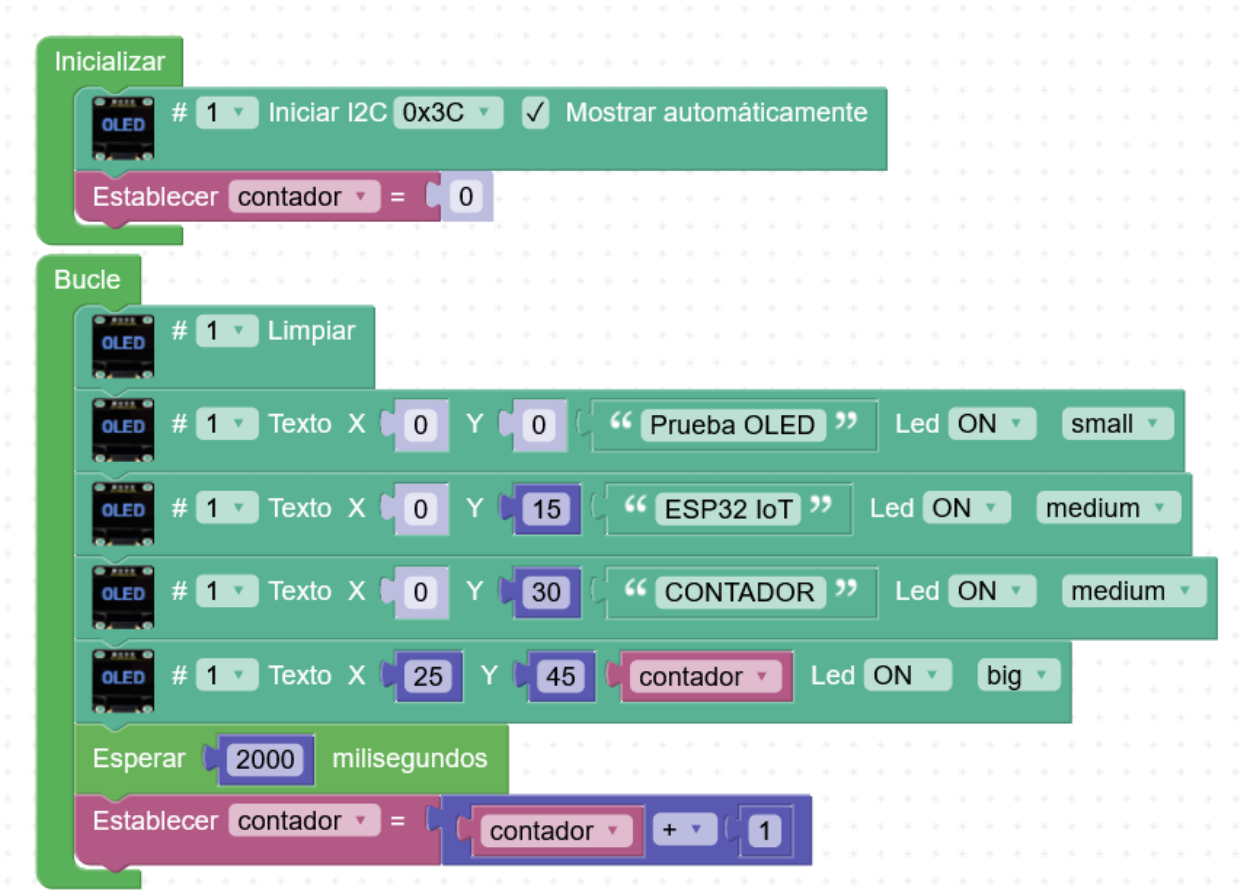


Ejemplo de gráficos vectoriales “*renderizados*” en pantalla de baja resolución:



2.2.1.-Pantalla OLED y textos

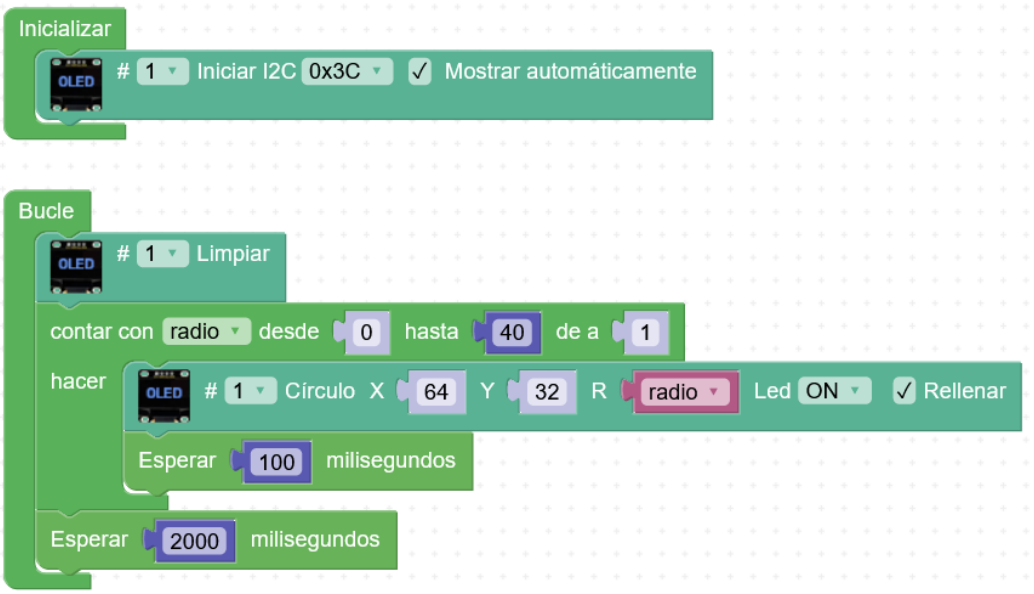
Programa de ejemplo:



2.2.2.-Pantalla OLED y gráficos vectoriales

Programa de ejemplo:

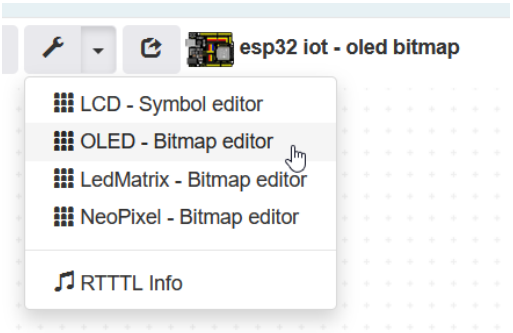
Programa de ejemplo (animación círculo):



2.2.3.-Pantalla OLED y Bitmaps

Convertir una imagen a un mapa de bits (monocromo). Cada bit codifica un píxel (on/off)

Herramienta "[OLED - Bitmap editor]"

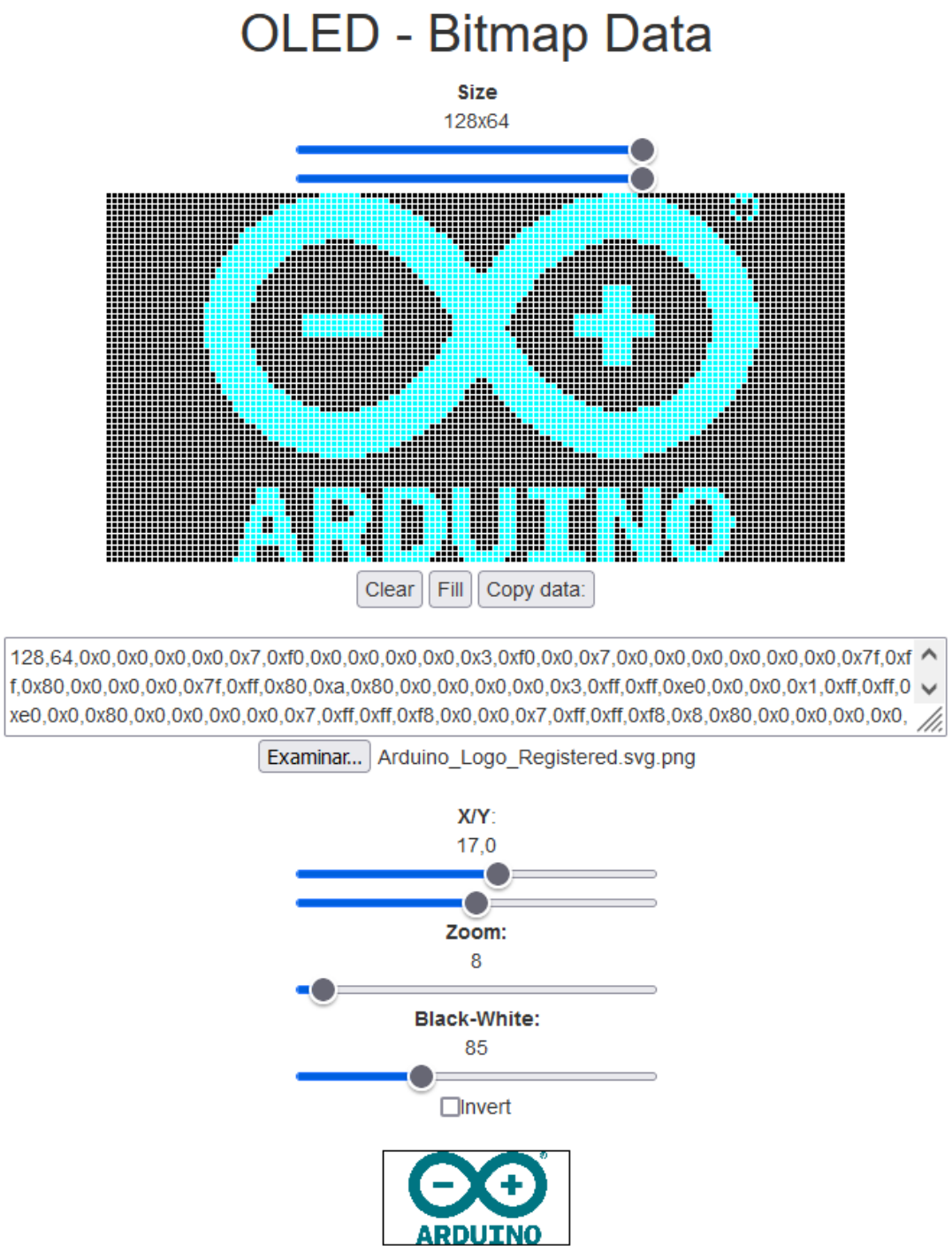


Podemos generar un bitmap del tamaño completo de la pantalla o ajustar al tamaño deseado.

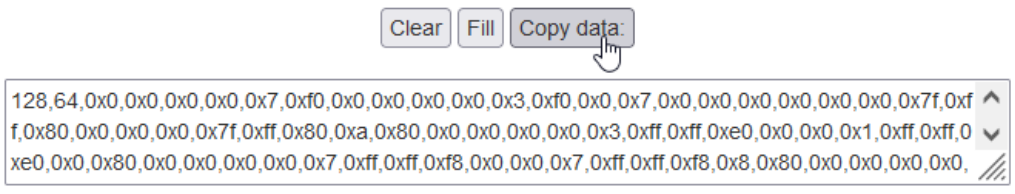
Imagen original:

https://upload.wikimedia.org/wikipedia/commons/thumb/5/5b/Arduino_Logo_Registered.svg/1200px-Arduino_Logo_Registered.svg.png

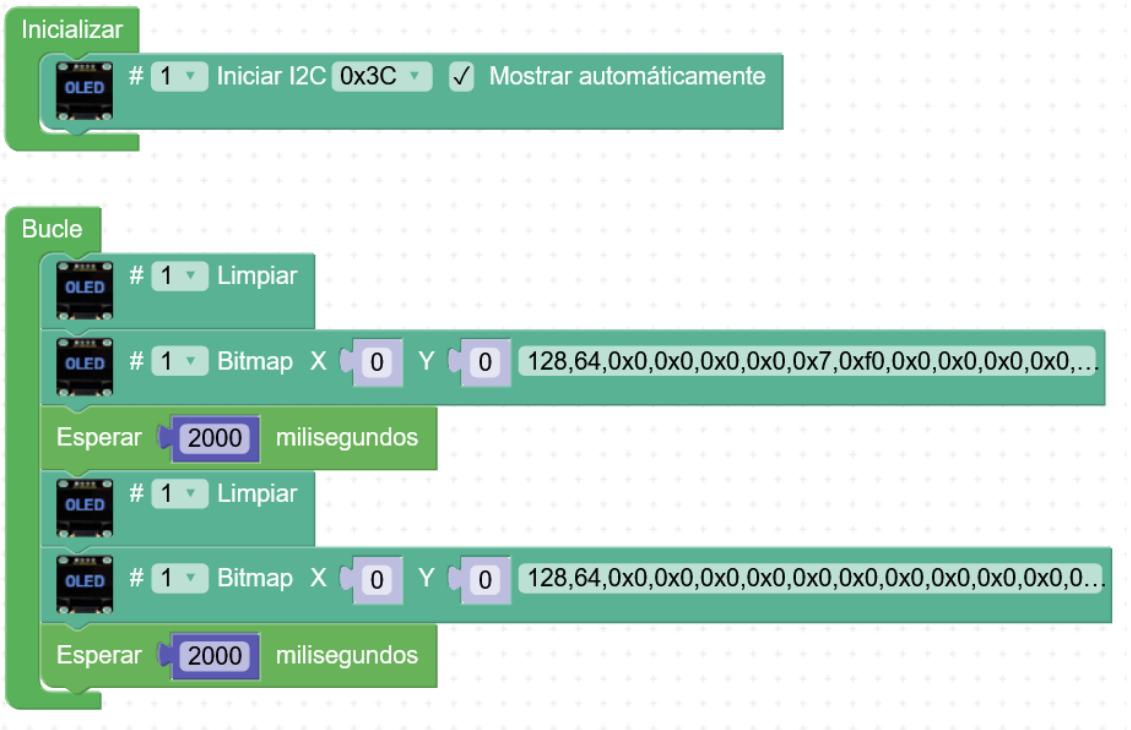
Mediante el editor ajustamos la conversión a monocromo, la posición, el zoom, etc.



Finalmente copiamos la secuencia de bits (en hexadecimal) que representarán la imagen:



Programa de ejemplo (2 bitmaps tamaño completo):



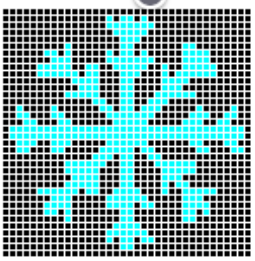
Programa de ejemplo (icono moviéndose por la pantalla)

https://img.freepik.com/vector-premium/logo-icono-copo-nieve-azul-simbolo-nieve-vector-navidad_231786-4028.jpg

OLED - Bitmap Data

Size

36x36



Clear | Fill | Copy data:

```
36,36,0x0,0x0,0x0,0x0,0x0,0x0,0x1,0x68,0x0,0x0,0x0,0x1,0xfc,0x0,0x0,0x0,0x0,0xf8,0x0,0x0,0x0,0x0
,0x70,0x0,0x0,0x0,0x40,0x60,0x30,0x0,0x3,0x41,0x68,0x3c,0x0,0x1,0xc1,0xfc,0x3c,0x0,0x7,0xcc,0xf
9,0x3e,0x0,0x7,0xec,0x71,0x7e,0x0,0x0,0x3c,0x61,0xe0,0x0,0x0,0x1c,0x61,0xc0,0x0,0x0,0x7c,0x63,
```

Examinar... copo.png

X/Y:

0,0

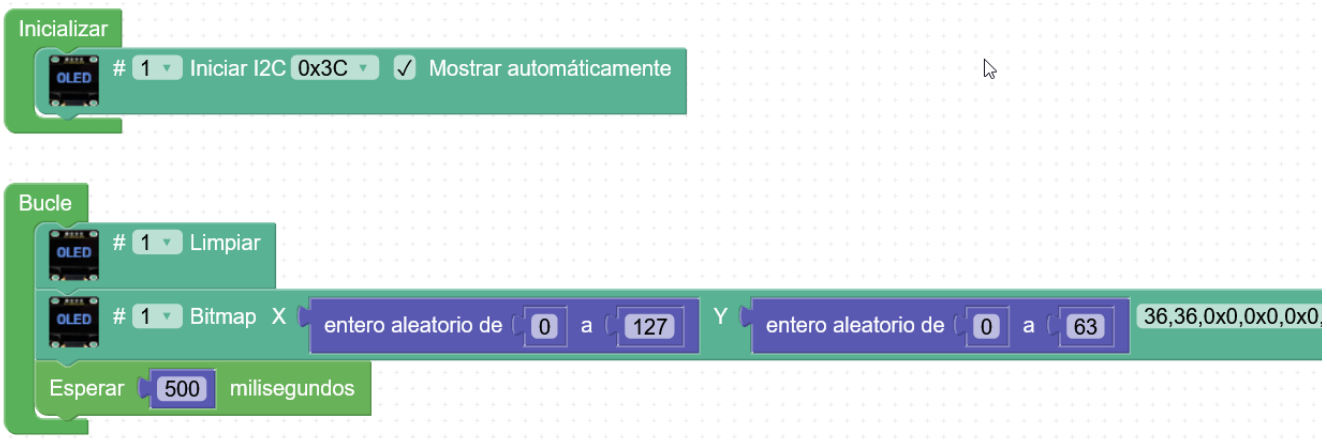
Zoom:

7.000000000000001

Black-White:

128

Invert



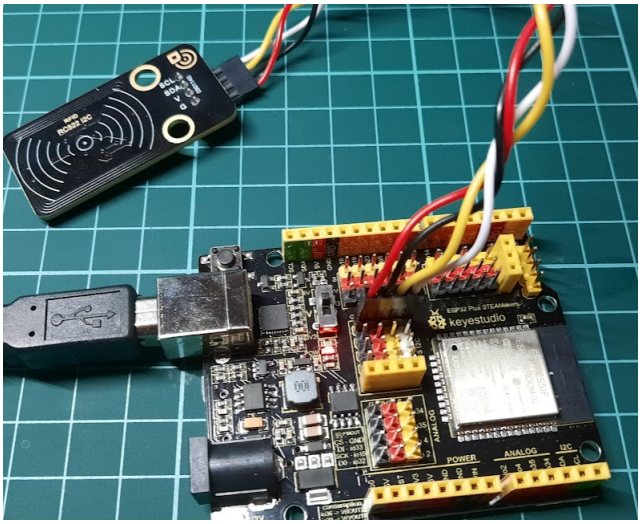
2.3.-RFID (i2c)

El módulo RFID se conecta mediante bus i2c.
Este sensor es capaz de leer tarjetas (llaveros, tarjetas, tags) RFID cuando se acerca a una distancia menor de unos 5 cm (no hace falta llegar a tocarlo, incluso puede estar debajo de una superficie como plástico, cartón, vidrio, etc.)

Cada tarjeta RFID viene grabada con un ID único que es lo que podremos leer para diferenciar unas tarjetas de otras.

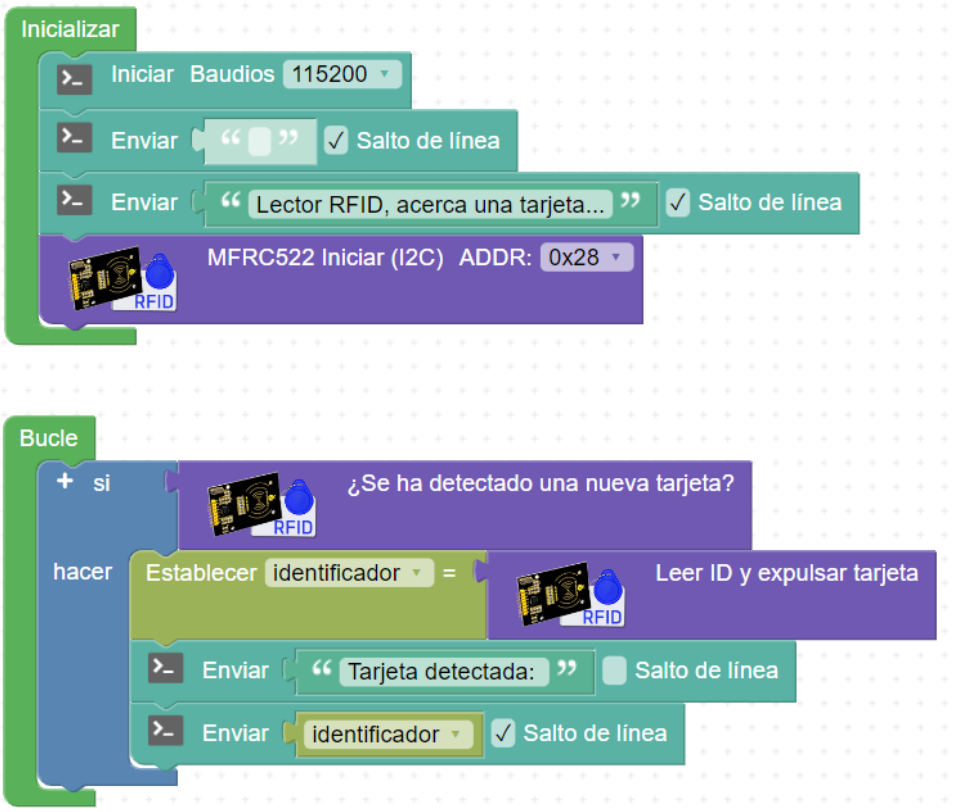
El ID de la tarjeta será un texto con el valor codificado en hexadecimal.

Conexión del módulo RFID:

Conexión directa a pines i2c hembra	Conexión con cable de 4 pines
	

2.3.1.RFID y lectura de códigos

Programa de ejemplo (muestra el ID de la tarjeta detectada por consola)



Consola:

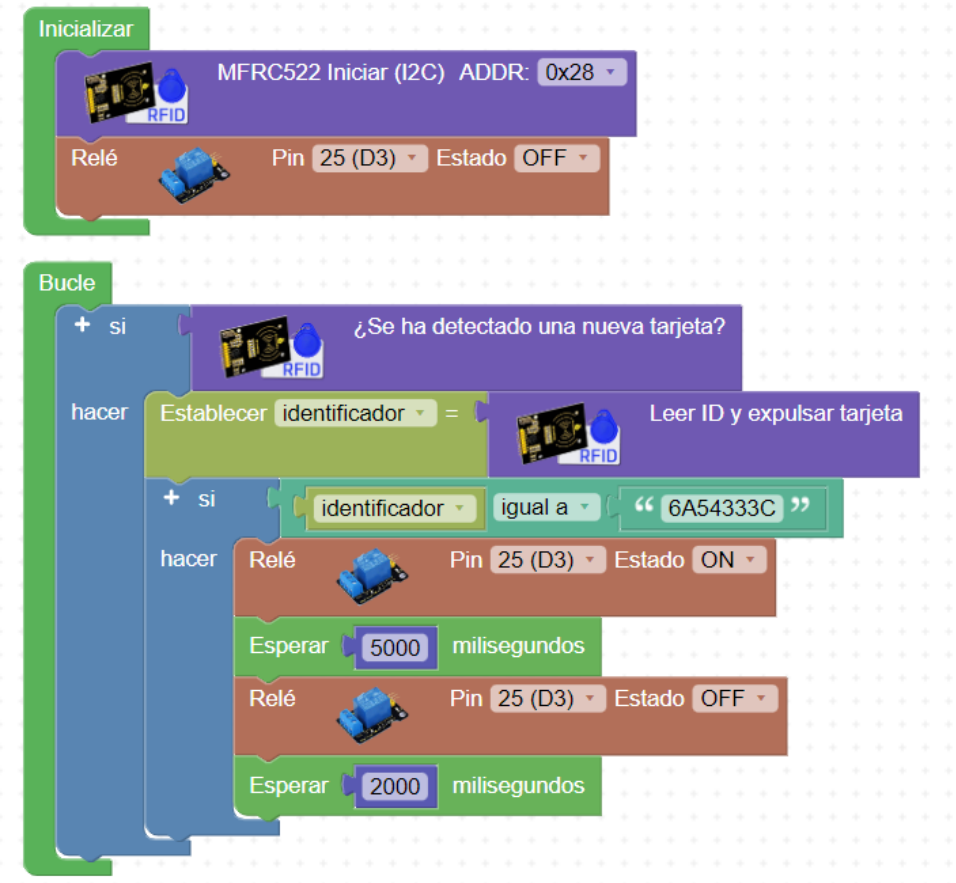
```
Lector RFID, acerca una tarjeta...
Tarjeta detectada: 6A54333C
Tarjeta detectada: 6A627F0C
```

2.3.2.-RFID y control de acceso

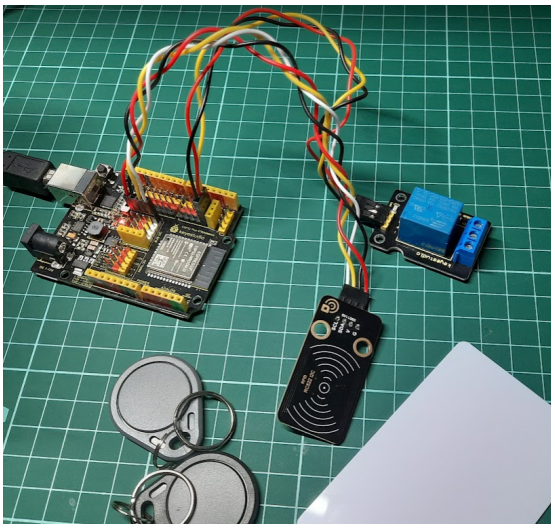
Para este ejemplo debemos obtener antes el ID de la tarjeta (o tarjetas) que queramos que el sistema acepte y en su caso abra la puerta.

Por ejemplo basándonos en el ejemplo anterior, sólo aceptaremos como válida la tarjeta o llavero:
ID: 6A54333C

Programa de ejemplo (activación de un relé para abrir una puerta)



Montaje y prueba de control de acceso:

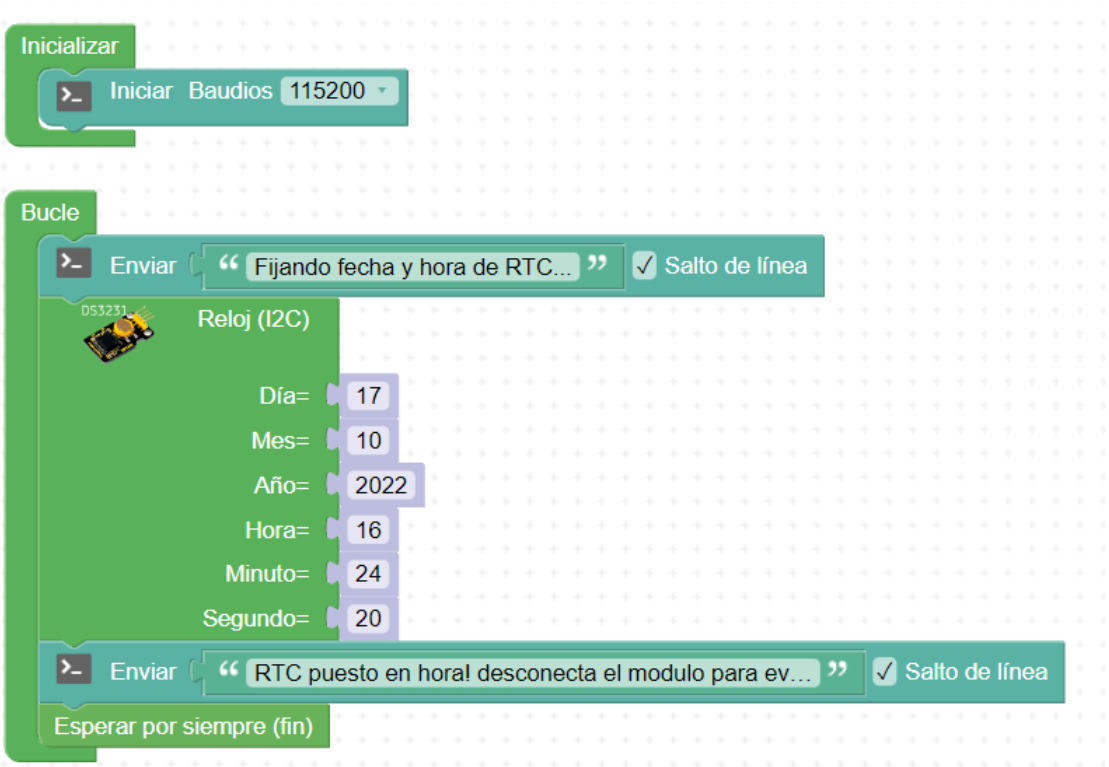


2.4.-RTC

El módulo de reloj RTC con conexión i2c permite almacenar la fecha y hora y recuperarla posteriormente. El propio módulo incorpora una pila para mantener la fecha y hora actualizada aunque el sistema al que esté conectado no tenga alimentación.

Debemos, al menos, una vez (si se cambia la pila otra vez) poner el módulo en fecha y hora y ya se quedará para sucesivos montajes

2.4.1.-Fijar manualmente fecha y hora RTC (solo lo haremos una vez)



2.4.2.-Fecha y hora RTC en pantalla OLED

2.5.-WiFi

ESP32 incorpora WiFi, y además puede actuar como:

-Cliente WiFi (nos conectamos a una red existente)

-Punto Acceso AP (creamos una red a la que se pueden conectar otros dispositivos)

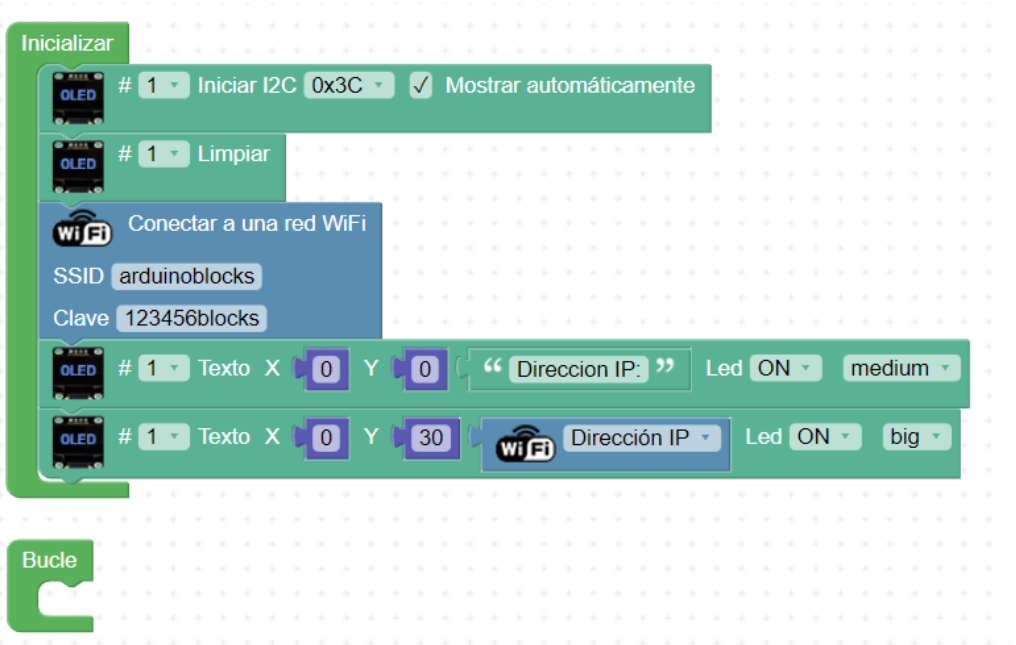
Por defecto la configuración del cliente WiFi es automática (DHCP) aunque podemos también asignar una IP fija, DNS, etc.

Una vez conectados tenemos bloques para obtener nuestra dirección IP y otros datos de interés para la conexión.

2.5.1.- Conectar y mostrar en pantalla OLED la IP de la ESP32 en nuestra red

Necesitamos el SSID (nombre de la red) y la clave (si es necesaria si no se deja en blanco)

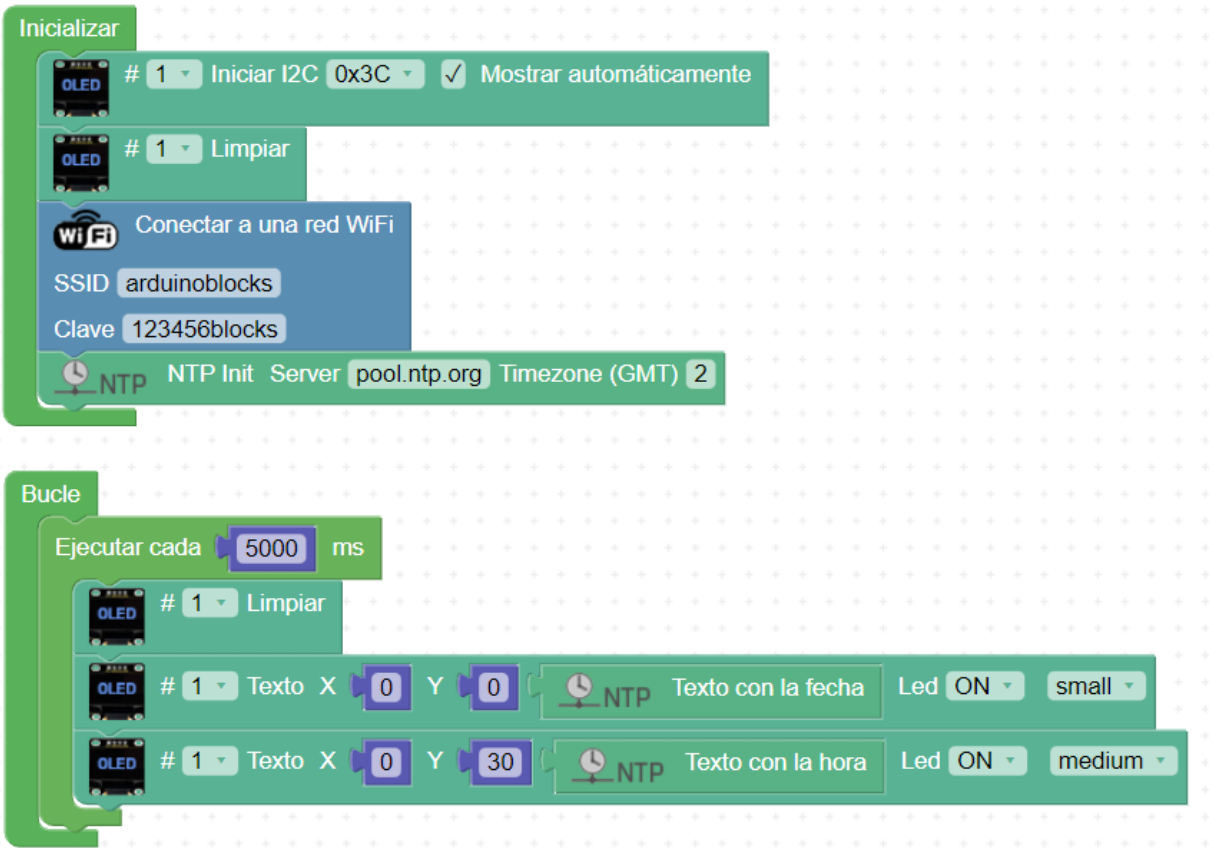
Programa de ejemplo:



2.5.2.- NTP: obtener fecha y hora de internet y mostrarla en pantalla OLED

Gracias al protocolo NTP (Network Time Protocol) podemos obtener la fecha y hora actual desde internet.

Programa de ejemplo:



Sesión 3

- Tarjeta uSD en ESP32 STEAMakers
  - Lectura, escritura y gestión de archivos
  - Registro de datos en tarjeta uSD (sensor DHT22)
- Bluetooth HID
  - Simulación Teclado BT para envío de datos
  - Ratón BT
- Protocolo MQTT
  - MQTT + broker de pruebas + App (MyMQTT / IoT MQTT Panel)
  - MQTT + Thingspeak / Adafruit.io

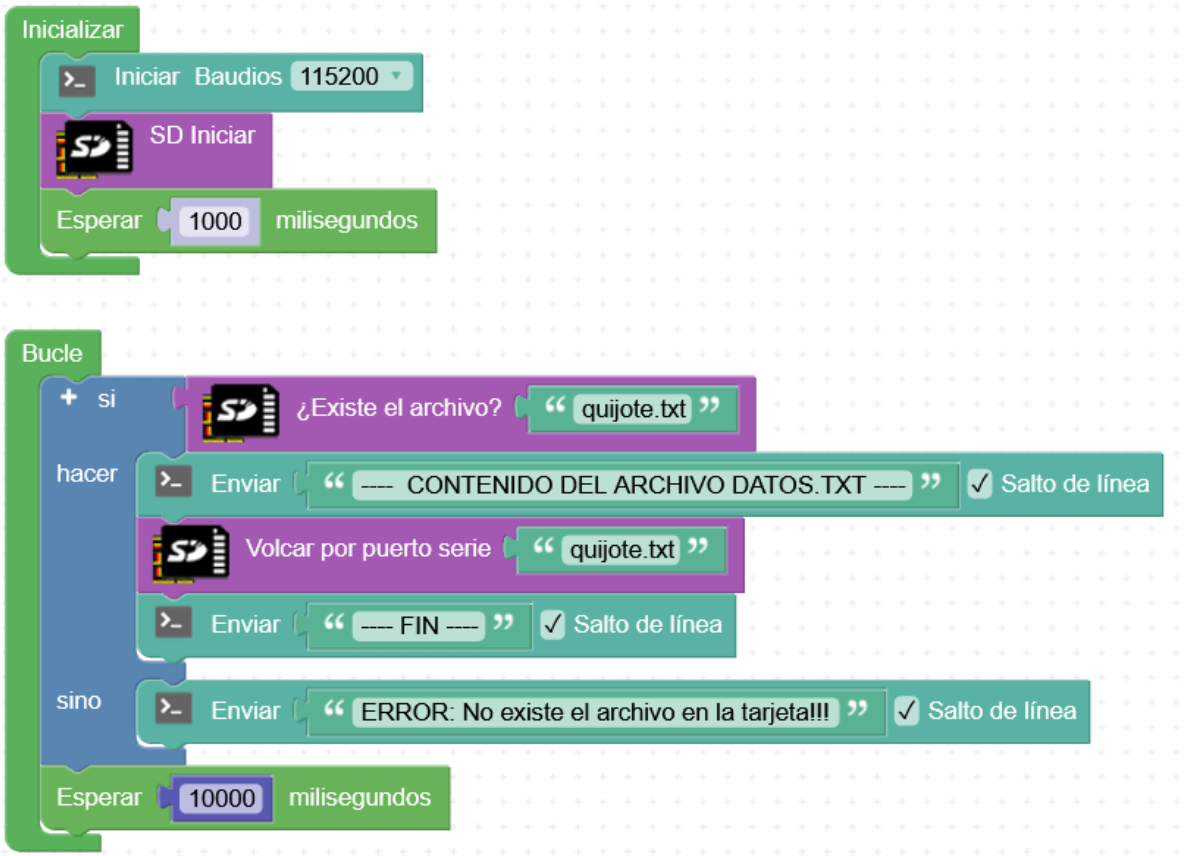
3.1.-Tarjeta uSD

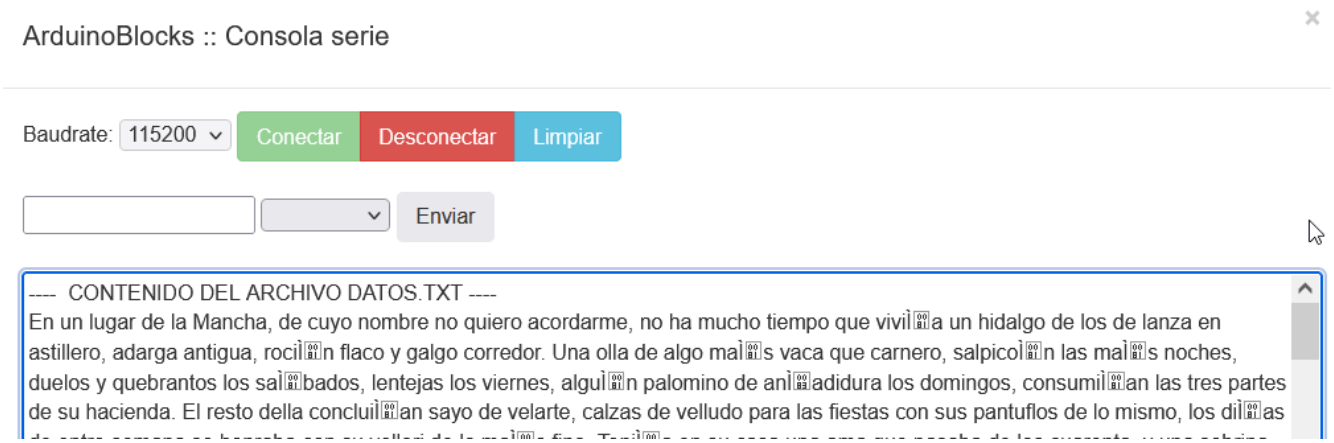
El almacenamiento en tarjeta uSD nos permitirá utilizar tarjetas de hasta 32GB en formato FAT

Mediante sencillos bloques podemos:

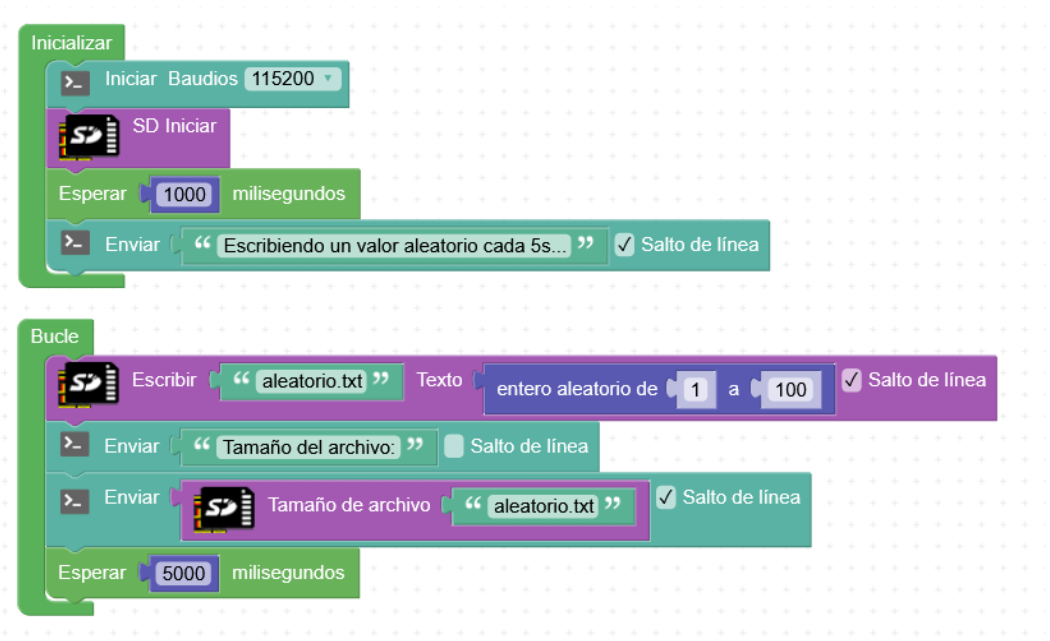
-Leer de un archivo
-Escribir de un archivo
-Eliminar un archivo

3.1.1.-Lectura de un archivo de texto dentro de la tarjeta SD y mostramos por consola el contenido.

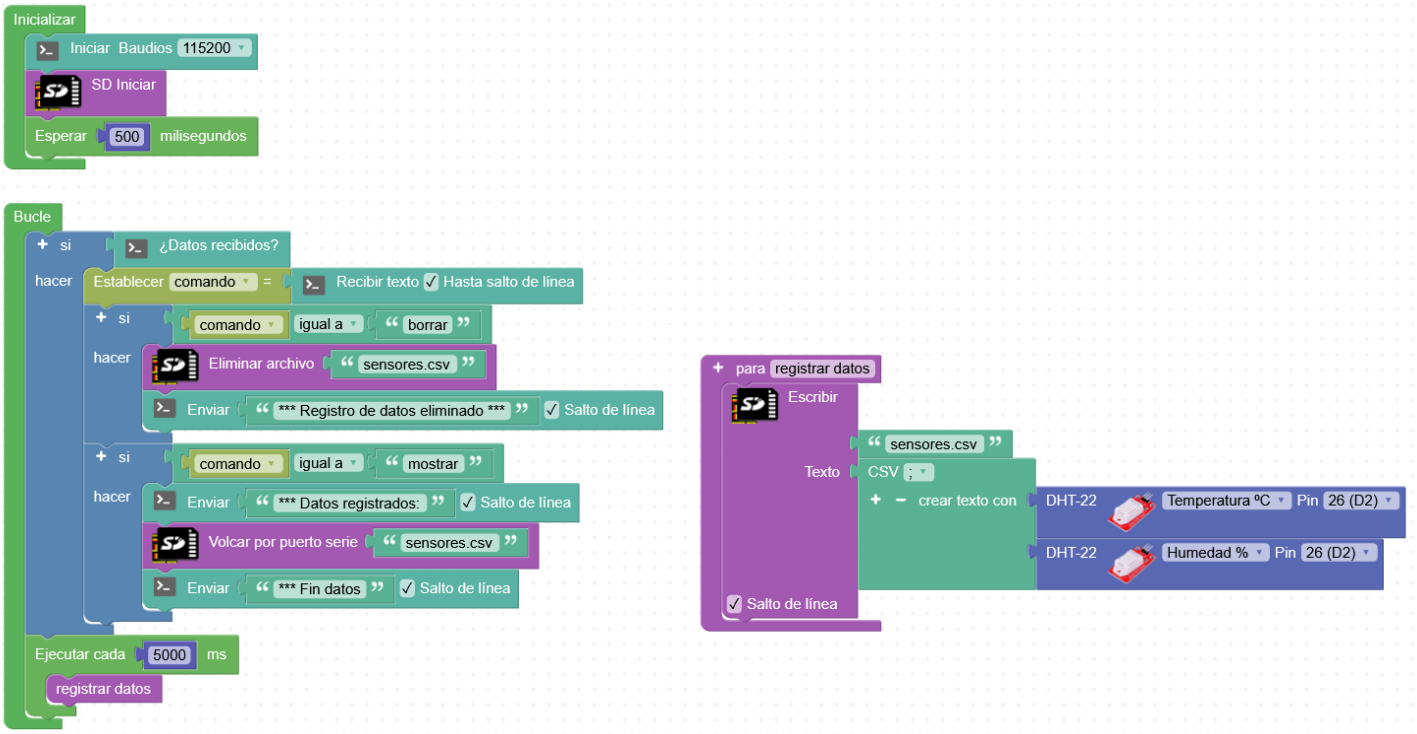




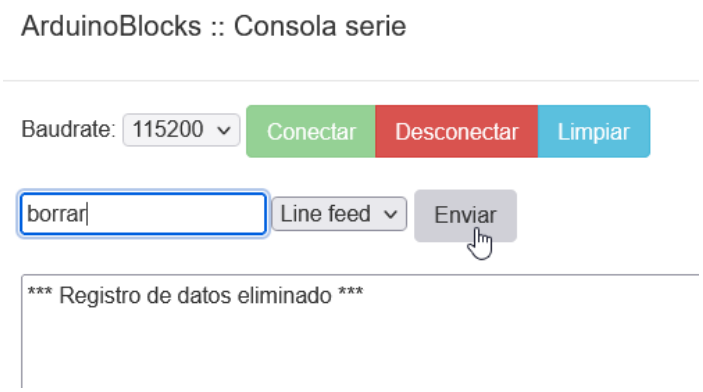
3.1.2.-Escritura en archivos de texto

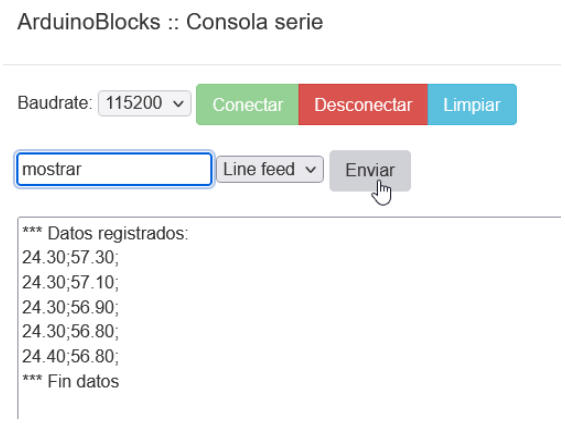


3.1.3.-Registro de datos en archivo de texto en formato CSV y comandos para borrar o volcar los datos por consola serie

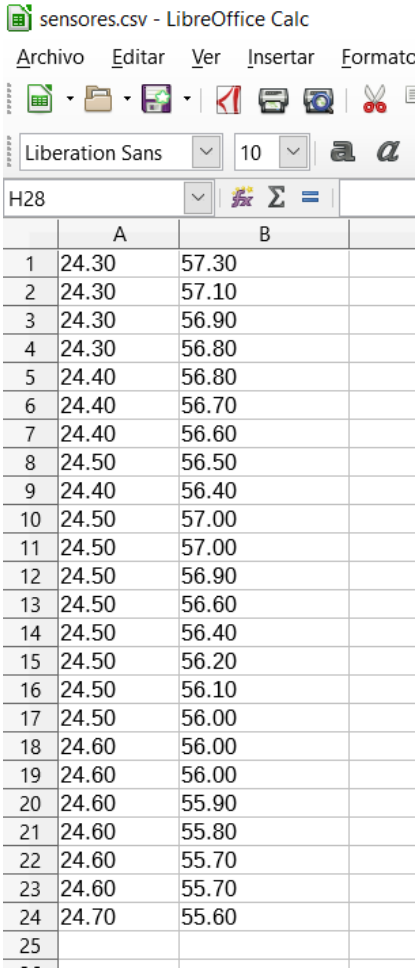


Comandos mediante consola serie:





Archivo CSV leído en el PC mediante Libre Office Calc:



	A	B
1	24.30	57.30
2	24.30	57.10
3	24.30	56.90
4	24.30	56.80
5	24.40	56.80
6	24.40	56.70
7	24.40	56.60
8	24.50	56.50
9	24.40	56.40
10	24.50	57.00
11	24.50	57.00
12	24.50	56.90
13	24.50	56.60
14	24.50	56.40
15	24.50	56.20
16	24.50	56.10
17	24.50	56.00
18	24.60	56.00
19	24.60	56.00
20	24.60	55.90
21	24.60	55.80
22	24.60	55.70
23	24.60	55.70
24	24.70	55.60
25		

3.2.-Teclado HID

La placa ESP32 STEAMakers puede simular un dispositivo HID como por ejemplo un teclado. Podemos emparejarlo con dispositivos con Bluetooth BLE compatibles con este tipo de periféricos (móvil, tablet, PC, TV, ...)

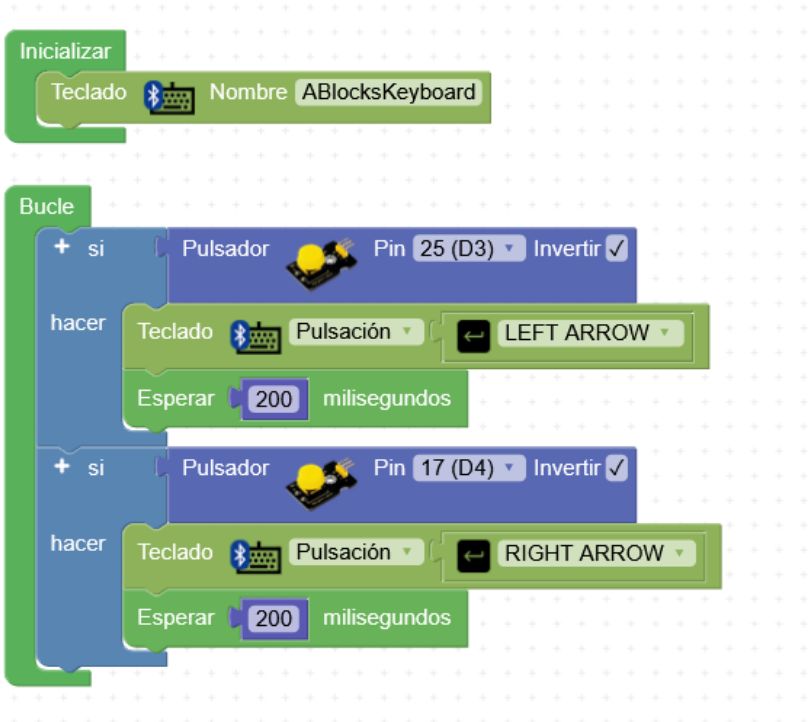
Puede ser útil para diseñar interfaces físicos personalizadas para aplicacioens, juegos, etc.

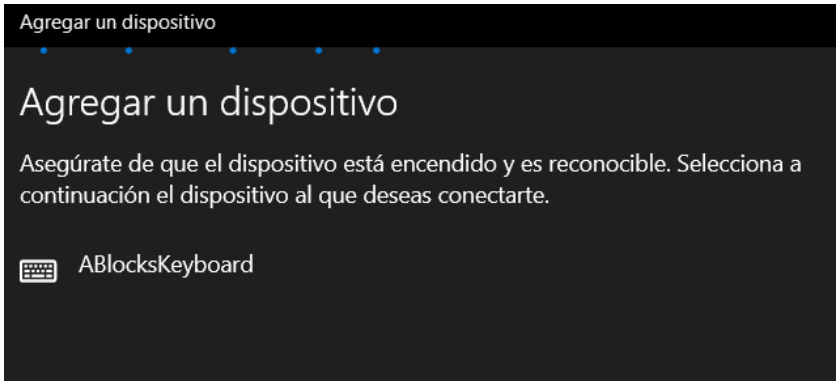
3.2.1.-Controlador personalizado (juegos , aplicaciones, etc.)

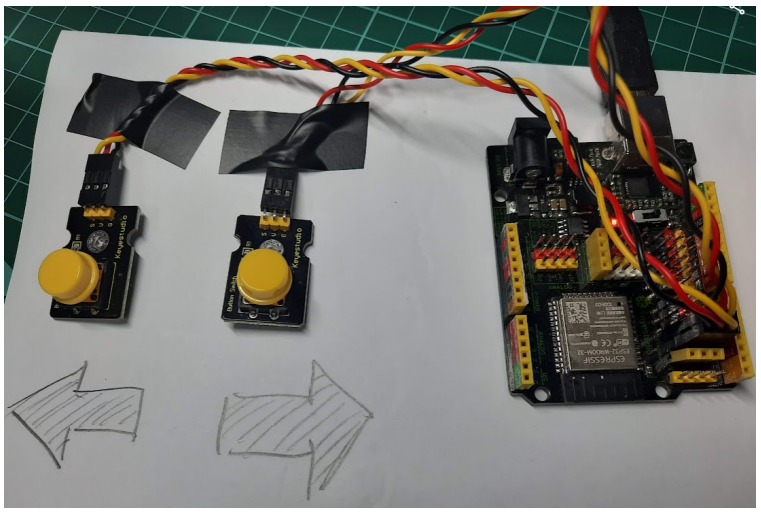
Mediante dos pulsadores simularemos la pulsación de las teclas:

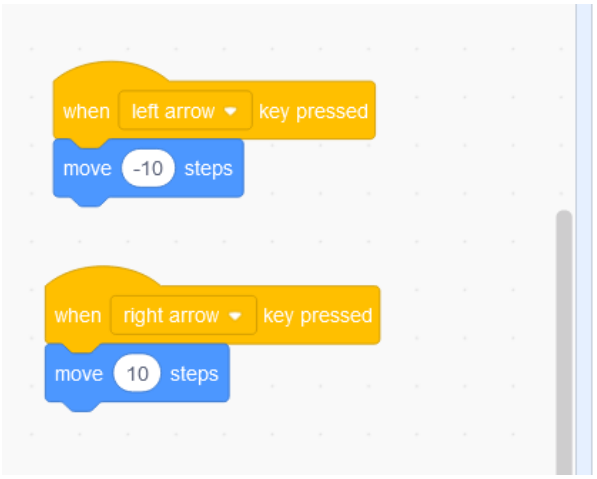
Botón 1: flecha izquierda

Botón 2: flecha derecha



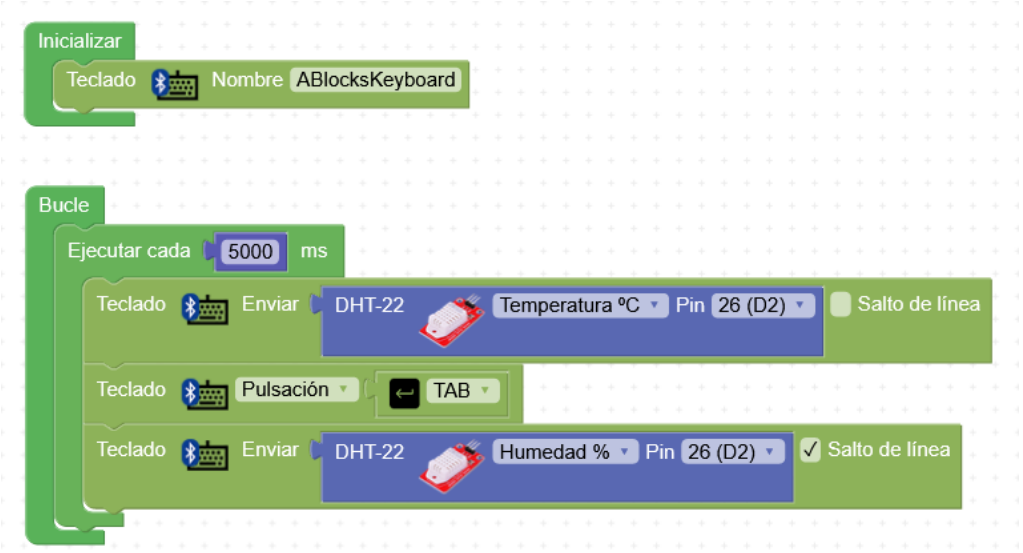




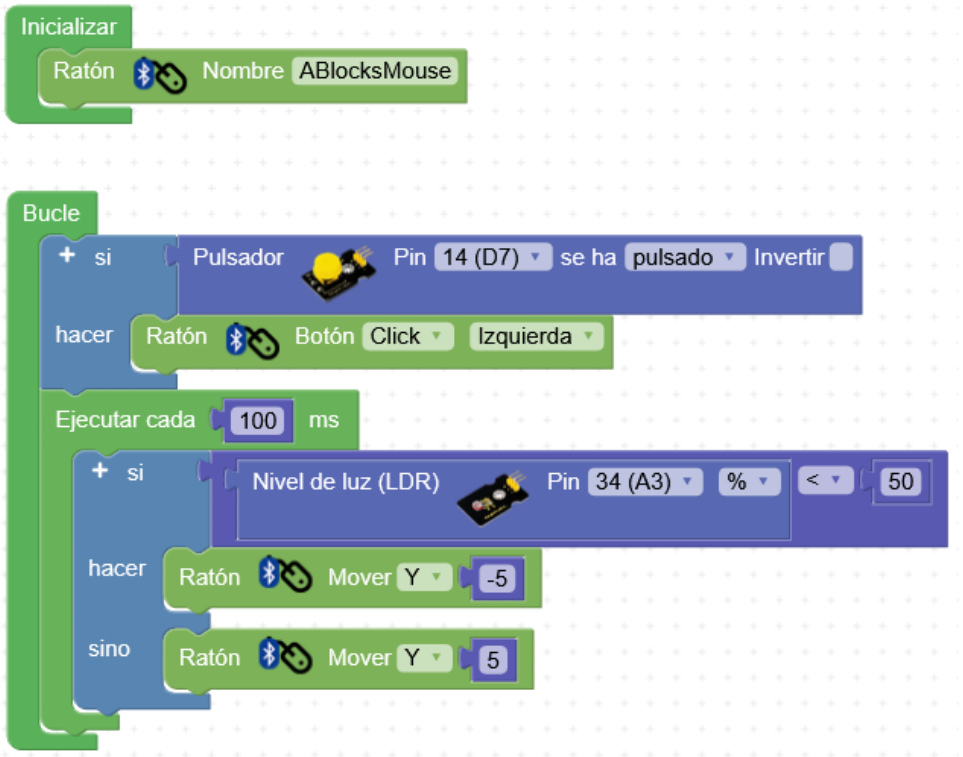


3.2.1.-Escritura de datos automática

La ESP32 STEAMakers simulará que escribe los datos de un sensor, podriamos abrir un documento de texto y se irían “tecleando” automáticamente o una hoja de cálculo.



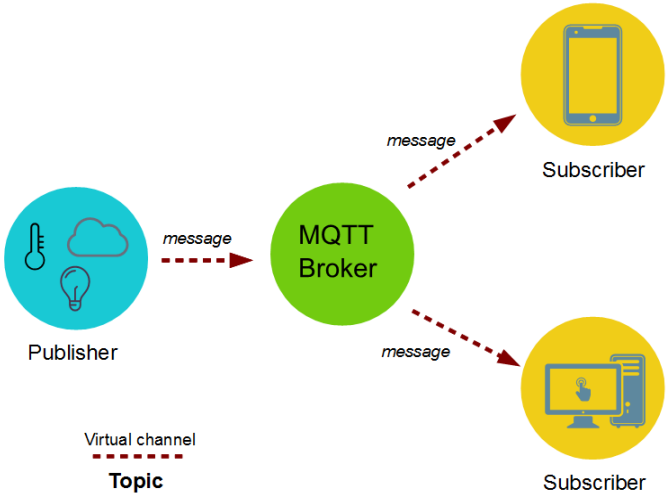
3.2.2 Simular movimiento y click del ratón



3.3.-Protocolo MQTT

MQTT es un protocolo de comunicación para redes TCP/IP muy sencillo y ligero en el que todos los dispositivos se conectan a un servidor (llamado "*broker*"). Los dispositivos pueden enviar (*publicar*) o recibir (*suscribirse*) mensajes asociándoles un "*topic*" (tema).

El "*broker*" se encarga de gestionar los mensajes y distribuirlos entre todos los dispositivos conectados.



Podemos implementar nuestro propio servidor/broker. Existen brokers MQTT de código libre como "**Mosquitto**" que podemos instalar en diferentes sistemas operativos de forma sencilla. Un ejemplo típico es configurar una Raspberry Pi como servidor MQTT en casa. Si queremos que el sistema esté abierto a internet deberemos configurar nuestra conexión adecuadamente al igual que obtener nuestra IP pública actual o contratar una IP pública fija.

https://mosquitto.org/

Por otro lado podemos utilizar brokers MQTT públicos disponibles en internet con fines experimentales o docentes y en cualquier otro caso podemos contratar servicios de brokers de pago con diferentes limitaciones de ancho de banda o número de conexiones según nuestras necesidades.

Algunos brokers MQTT para utilizar: *(los servidores gratuitous públicos no son seguros, pues cualquiera puede suscribirse a nuestros mensajes o publicar en ellos, solo se usan para pruebas)*
Broker.hivemq.com Info: http://www.mqtt-dashboard.com/
test.mosquitto.org Info: https://mosquitto.org/

La comunicación entre los nodos de un sistema MQTT se realizan enviando mensajes. Los nodos envían los mensajes al broker y éste se encarga de distribuirlos entre el resto de nodos. Cada mensaje consta de un "topic" o tema y el cuerpo del mensaje en sí. Un nodo se puede suscribir a un "*topic*" de forma que recibirá todos los mensajes que tengan ese "*topic*". Cada nodo puede publicar mensajes con el "*topic*" deseado.

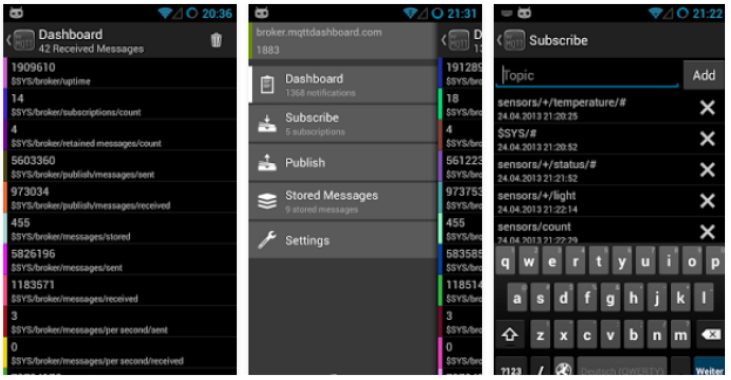
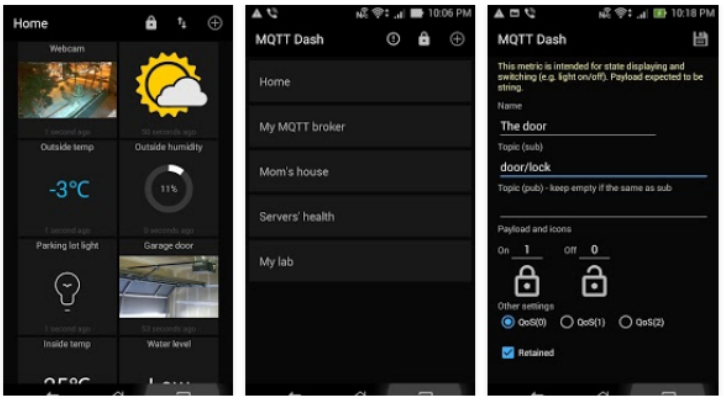
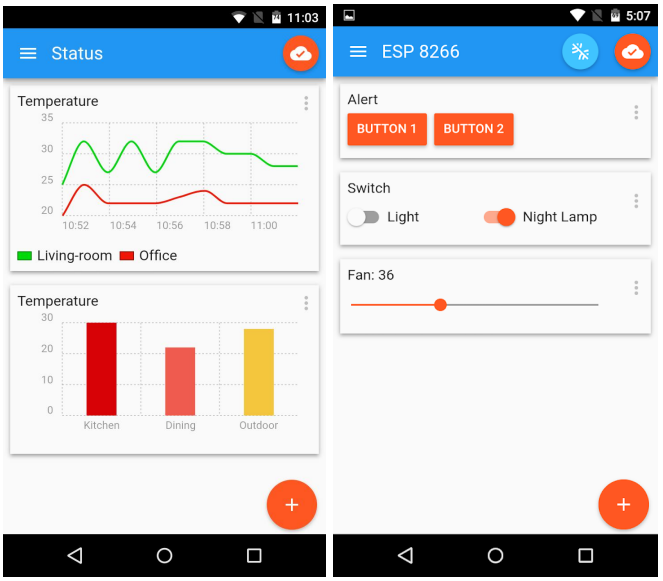
Por ejemplo podemos utilizar el topic: *"temp/comedor"* para que un nodo envíe la temperatura del comedor, por otro lado todos los nodos que deseen conocer la temperatura del comedor se suscribirán al topic : *"temp/comedor"* y recibirán automáticamente los mensajes de este tipo.

Mediante diferentes aplicaciones podemos conectarnos al servidor MQTT para enviar (publicar) o recibir (suscribirse) al mismo servidor MQTT que nuestra placa y así poder controlar remotamente o recibir información

Web onlinte para conectarnos a un servidor MQTT y publicar/suscribirse:

http://www.hivemq.com/demos/websocket-client/

Aplicaciones móviles para conectarse a un servidor MQTT y publicar/suscribirse. Algunas de ellas nos permiten realizar paneles de visualización o control.

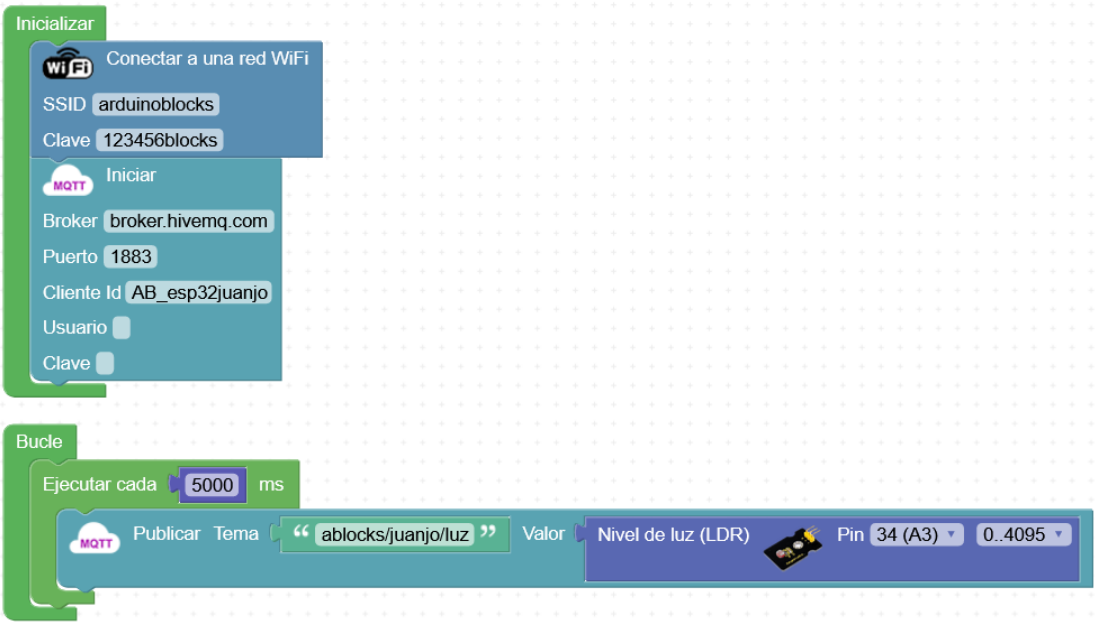
MyMQTT	
	
MQTT Dash	
	
IoT MQTT Panel	
	

3.3.1.-ESP32 Publica datos -> App se suscribe y recibe datos

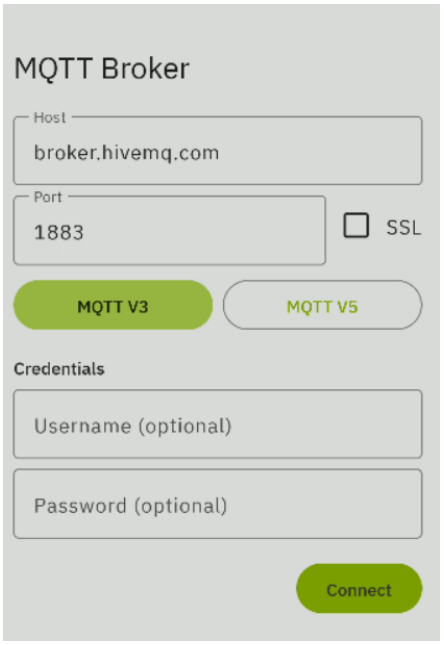
Servidor de pruebas: **broker.hivemq.com**

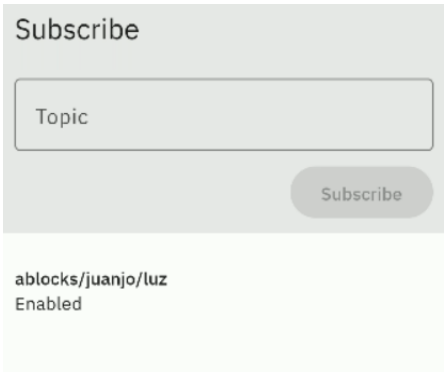
Mensaje publicado:

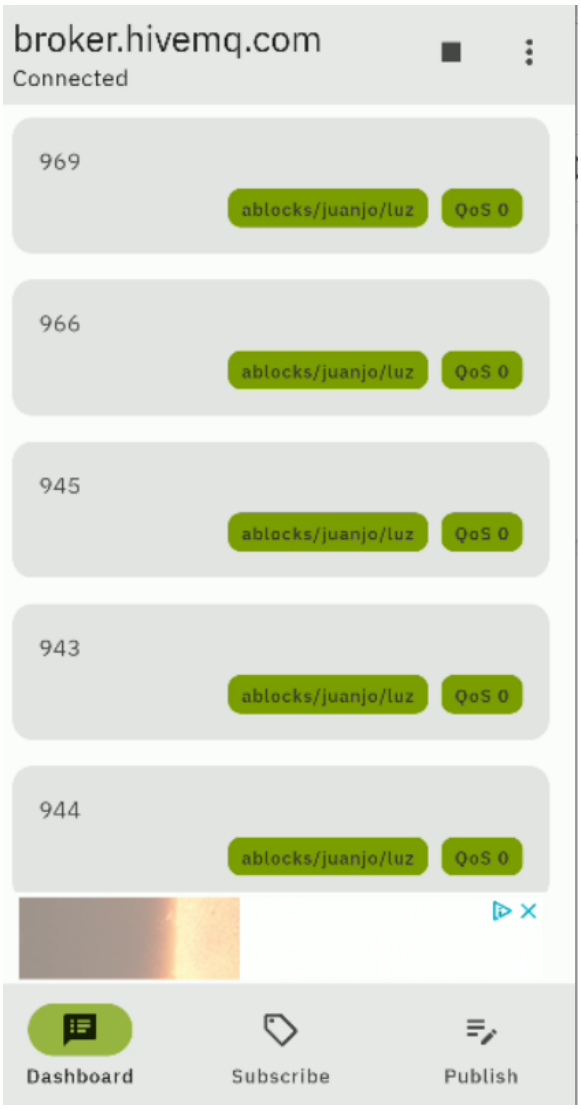
Topic:	Valor:
ablocks/juanjo/luz	Dato de nivel de luz (0…4095)



Desde la aplicación “MyMQTT” nos suscribimos a este topic en el servidor y recibiremos las actualizaciones




Subscribe
Topic
Subscribe
ablocks/juanjo/luz
Enabled


broker.hivemq.com
Connected
969
ablocks/juanjo/luz
QoS 0
966
ablocks/juanjo/luz
QoS 0
945
ablocks/juanjo/luz
QoS 0
943
ablocks/juanjo/luz
QoS 0
944
ablocks/juanjo/luz
QoS 0
Dashboard
Subscribe
Publish

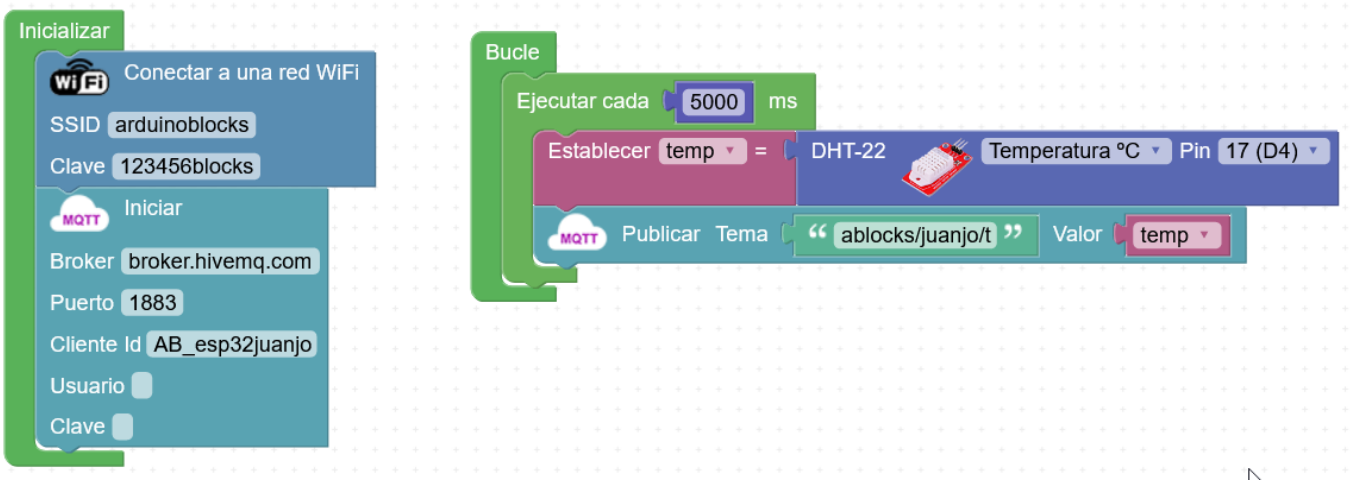
3.3.2.-ESP32 publica datos temperatura -> App se suscribe y recibe datos

Servidor de pruebas: **broker.hivemq.com**

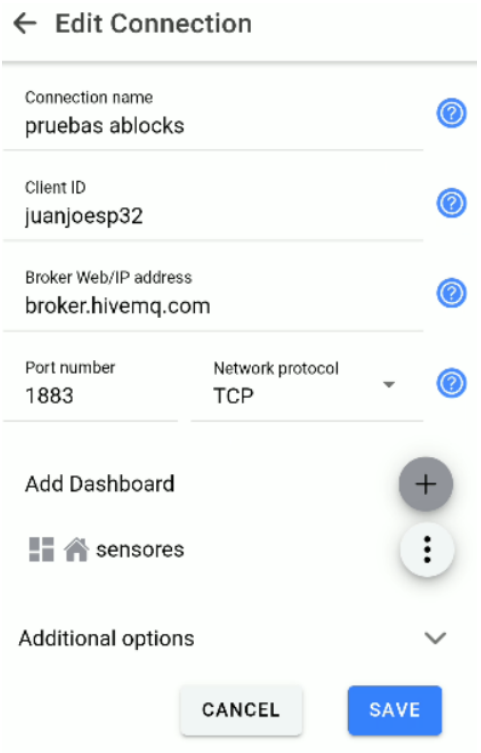
Mensaje publicado:

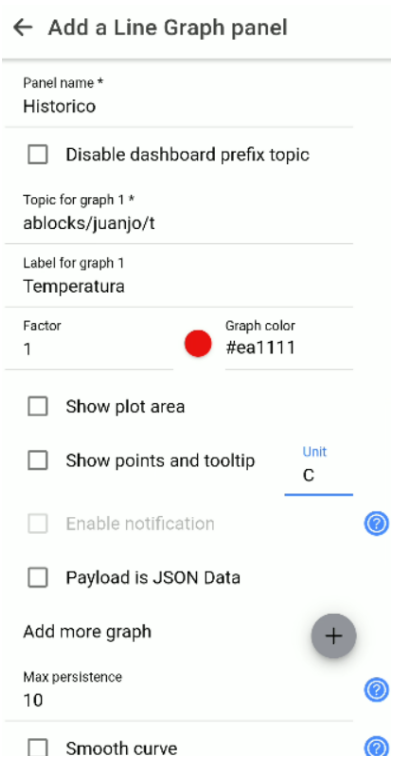
Topic:	Valor:
ablocks/juanjo/t	Nivel de temperatura del sensor DHT22
ablocks/juanjo/h	Nivel de humedad del sensor DHT22

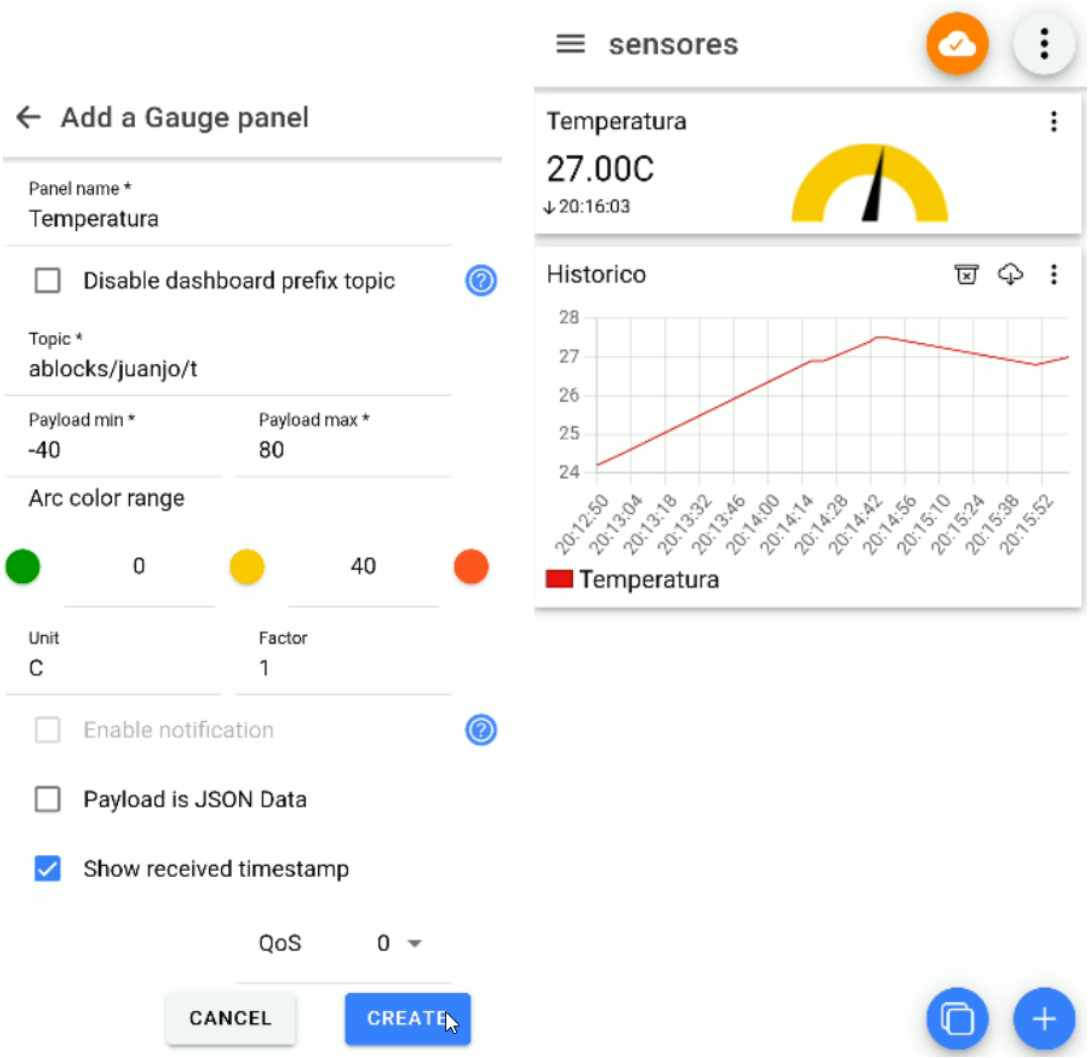
Programa en la placa ESP32 STEAMakers



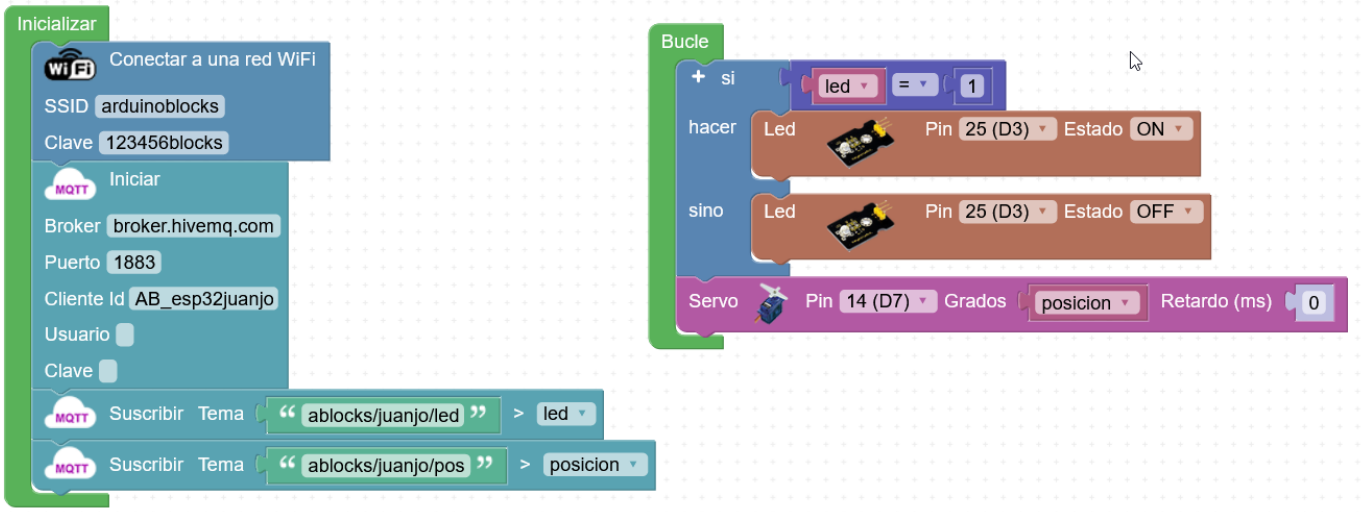
Utilizaremos la aplicación Iot Panel para crea un panel de monitorización



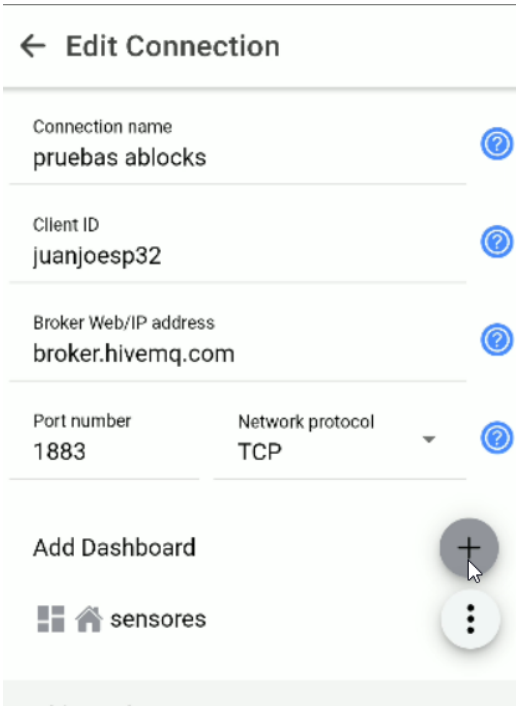


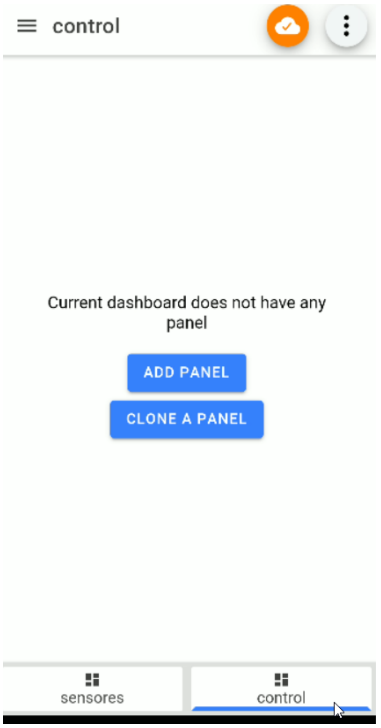


3.3.3.-Control remoto de un led y posición de servo vía MQTT

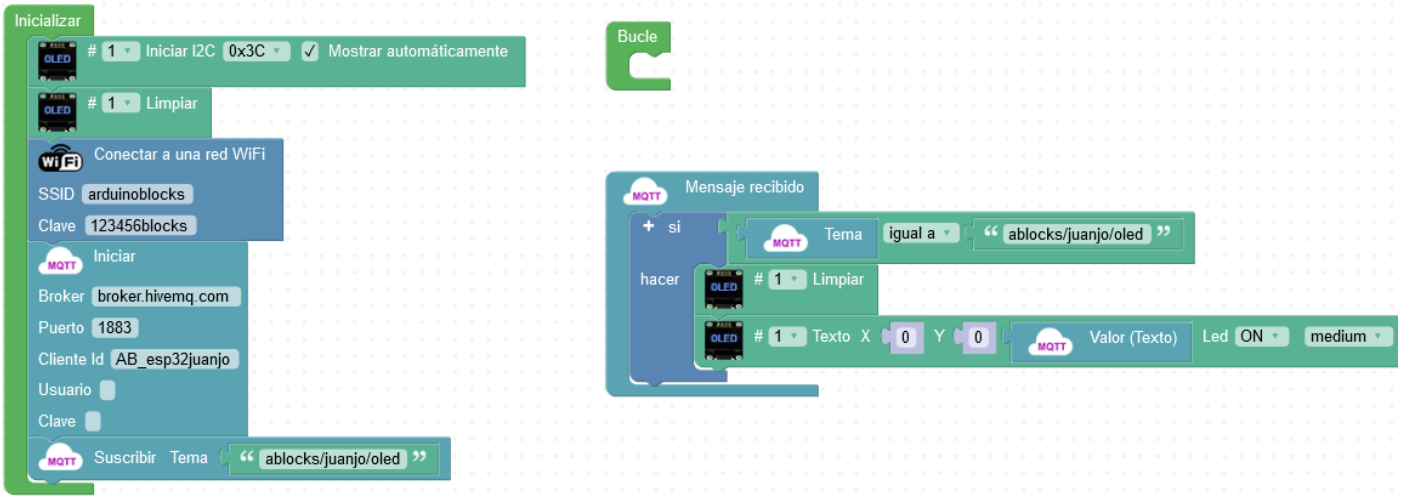


IoT MQTT Panel

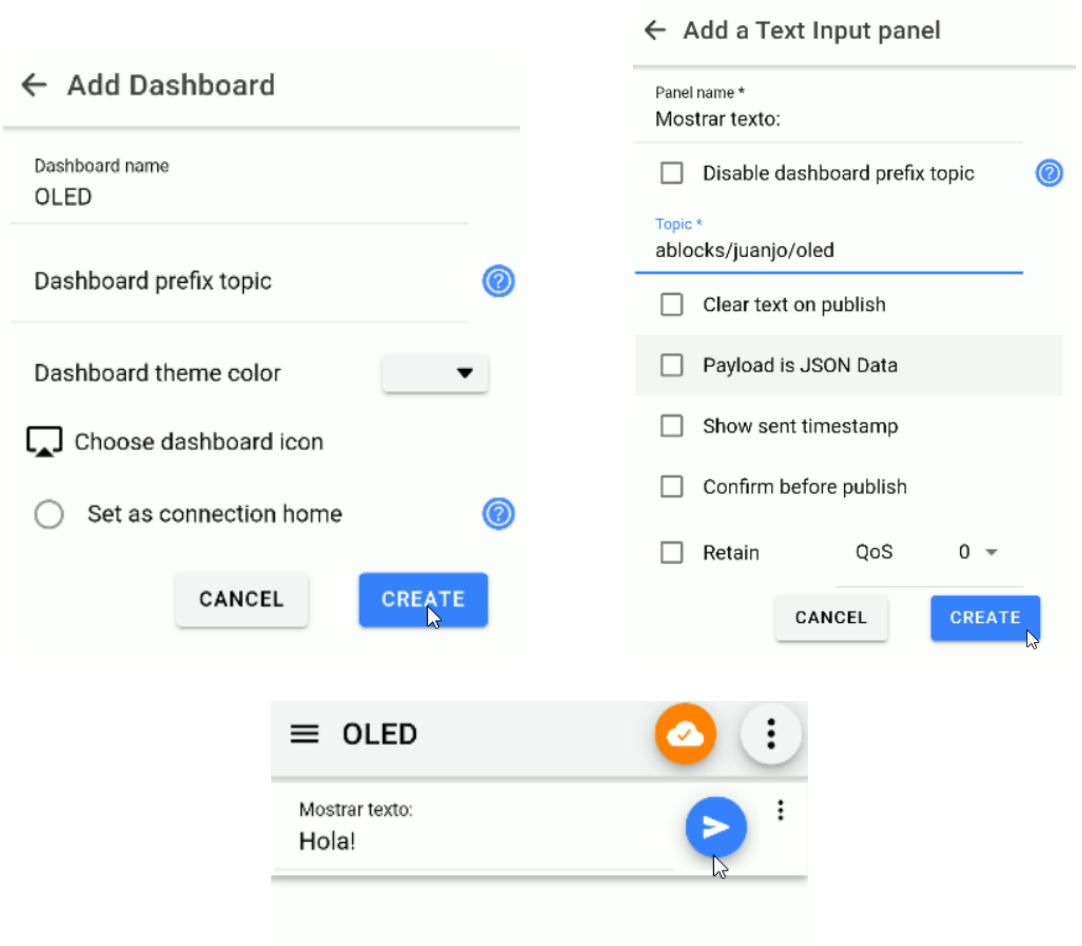




3.3.4.-Mostrar textos en LCD desde MQTT



IoT Panel para enviar el texto a mostrar en la pantalla OLED vía MQTT



3.3.5.- MQTT + ThingSpeak

ThingSpeak es un plataforma de Internet of Things (IoT) que permite recoger y almacenar datos de sensores en la nube y desarrollar aplicaciones IoT. Thinkspeak también ofrece aplicaciones que permiten analizar y visualizar tus datos en MATLAB y actuar sobre los datos. Los datos de los sensores pueden ser enviados desde Arduino, Raspberry Pi, BeagleBone Black y otro HW.

https://thingspeak.com/

*Users of the free option will be limited to sending **no more than 3 million messages each year** to the ThingSpeak service. Users of the free license will also be **limited to 4 channels**. For users of the free option, the message **update interval limit remains limited at 15 seconds**. Other limitations are described on the How to Buy pages.*

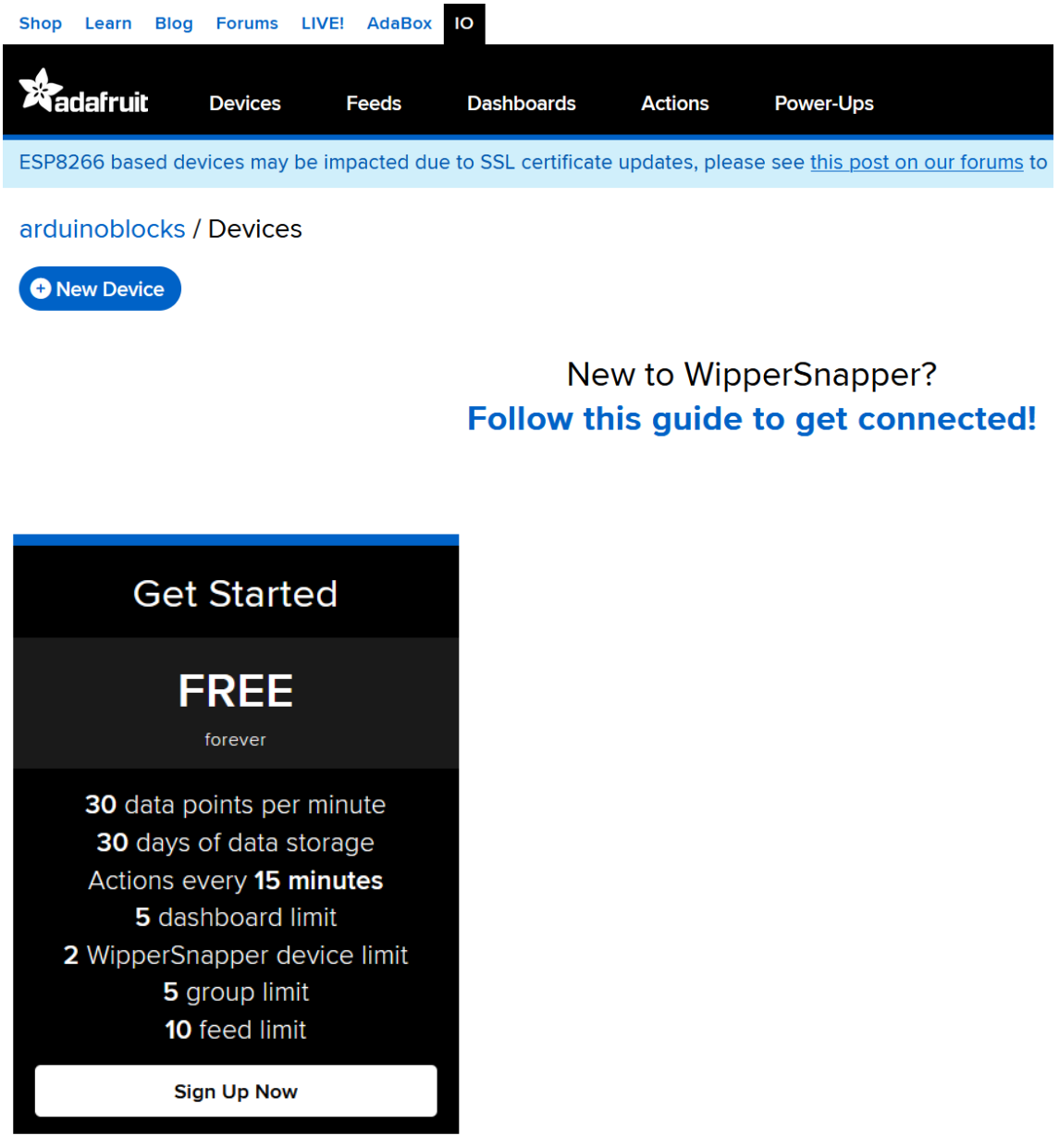
Manual paso a paso para publicar datos desde ArduinoBlocks en Thingspeak:

https://drive.google.com/file/d/17Acz8kZnqTtW0QNK40-RYPhvZ2XunYrJ/view

3.3.6.-MQTT + Adafruit

Adafruit.IO Es una solución para la construcción de aplicaciones IoT creada por Adafruit Industries, la conocida comercializadora de hardware open-source, han creado esta plataforma para el internet de las cosas basándose en plataformas conocidas como Arduino, Raspberry pi, ESP8266 , Intel Galileo, dispositivos Seriales y Wifi entre otros, La API de comunicación es basado en cliente MQTT con servidores de Adafruit.IO en pocos minutos puedes crear un dashboard de gran calidad.

https://io.adafruit.com/



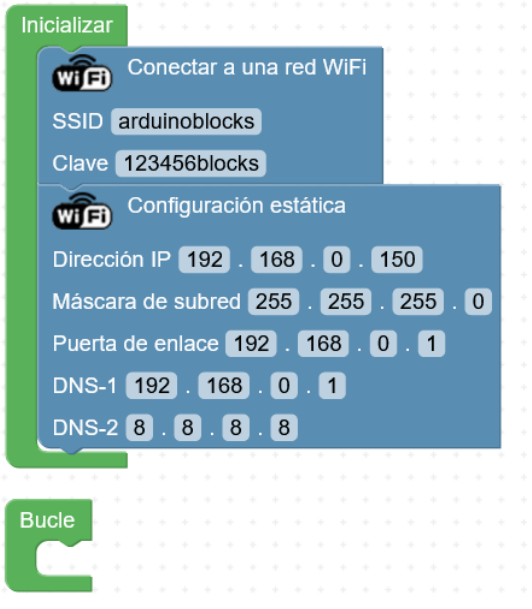
https://www.youtube.com/watch?v=gu9pKX-Tp0Y

Sesión 4

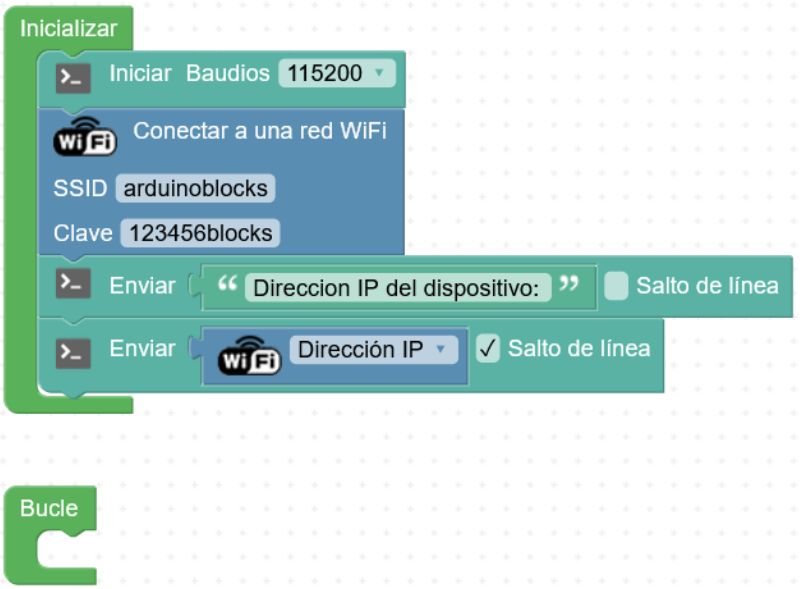
- Conexión WiFi
  - Crear un punto de acceso AP
  - Configurar IP estática
  - Obtener IP dinámica (consola / oled)
- Servidor web HTTP
  - Servidor web de Temperatura y humedad
- Cliente web HTTP
  - Registro de datos en hoja de cálculo de Google

4.1.1.-Crear punto de acceso WiFi

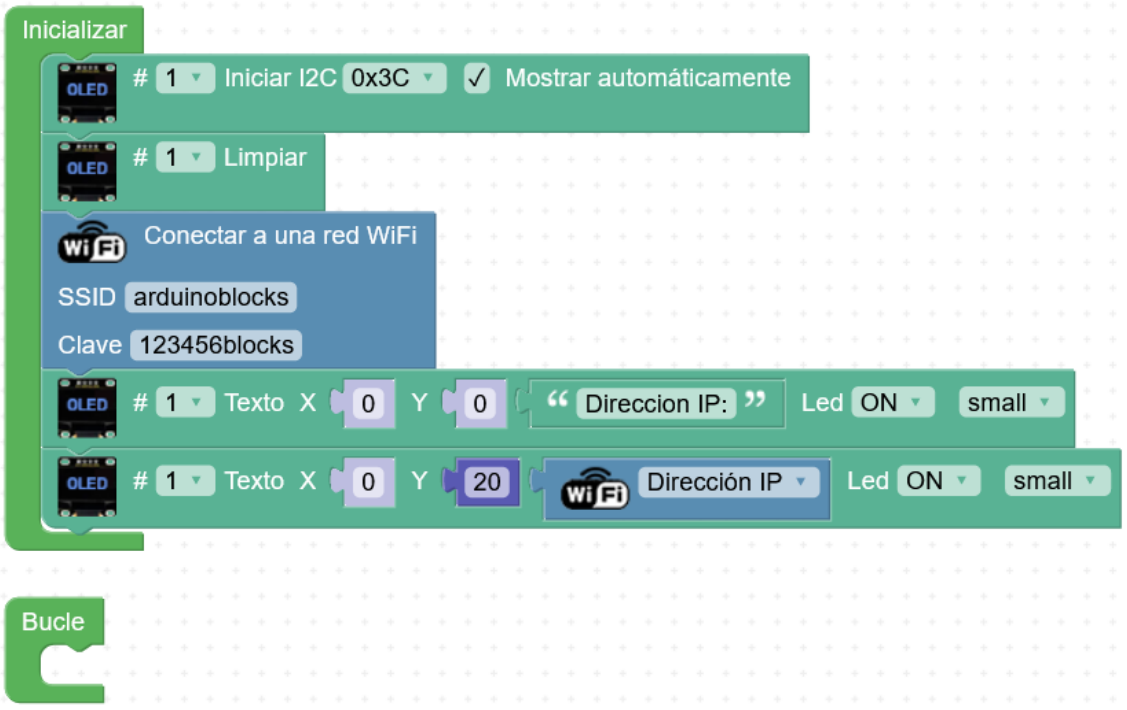
4.1.2.-Configurar IP estática



4.1.3.-Obtener la IP dinámica asignada



Opción con pantalla OLED



4.2.-Servidor / Cliente web

Un servidor web permite "*escuchar*" a clientes e intercambiar información mediante el protocolo HTTP. Por defecto un servidor web "escucha" en el puerto 80.

El cliente web por defecto es el navegador web, y en la dirección (url) indicamos el servidor al que nos conectamos.
Formato url (petición http / http request):

http://www.google.e/search?client=firefox&q=esp32+steamakers

Protocolo	*Host servidor*	*Puerto **	*Petición **	*Parámetros **
http://	www.google.es	:80	/search	**?**client=firefox**&**q=esp32+steamakers

El puerto si no se indica el navegador pone por defecto el 80

http://www.google.es:80/search?client=firefox&q=esp32+steamakers

La respuesta puede ser en cualquier formato:

Ejemplos de respuestas desde el servidor web (lo que nos devuelve el servidor a la petición):

a) Texto plano sin formato,
b) HTML : formato estándar para visualizar contenido web en un navegador
c) JPG, PNG, … : formatos de imágenes
d) CSV: formato de datos separados por comas

…

Códigos de estado en la respuesta:

- Respuestas informativas (100–199),
- Respuestas satisfactorias (**200**–299),
- Redirecciones (300–399),
- Errores de los clientes (400–499),
- y errores de los servidores (**500**–599).

Los más utilizados:

200 OK

La solicitud ha tenido éxito. El significado de un éxito varía dependiendo del método HTTP:

400 Bad Request

Esta respuesta significa que el servidor no pudo interpretar la solicitud dada una sintaxis inválida.

401 Unauthorized

Es necesario autenticar para obtener la respuesta solicitada. Esta es similar a 403, pero en este caso, la autenticación es posible.

403 Forbidden

El cliente no posee los permisos necesarios para cierto contenido, por lo que el servidor está rechazando otorgar una respuesta apropiada.

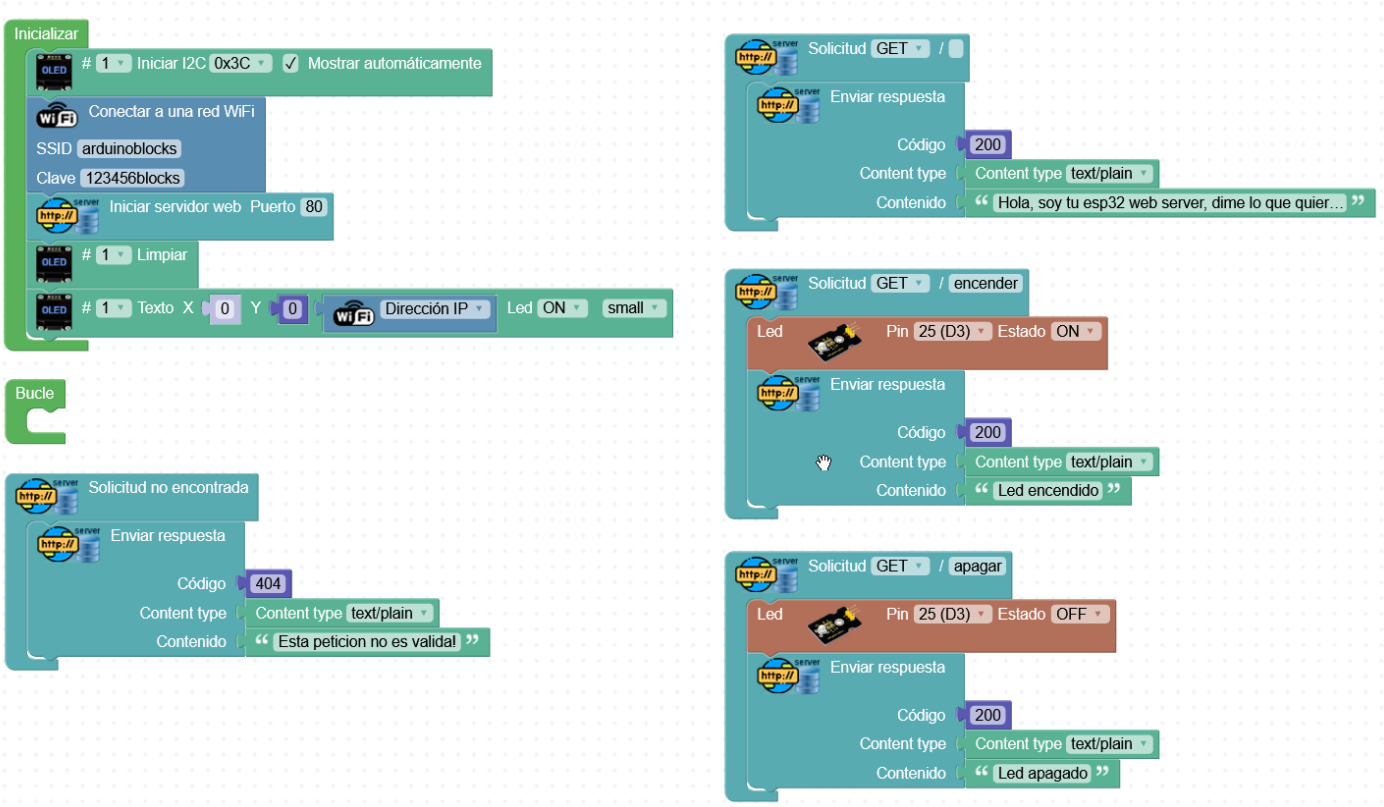
404 Not Found

El servidor no pudo encontrar el contenido solicitado. Este código de respuesta es uno de los más famosos dada su alta ocurrencia en la web.

500 Internal Server Error

El servidor ha encontrado una situación que no sabe cómo manejarla.

4.2.1.-Servidor web simple



4.2.2.-Servidor web + respuesta HTML

Mejora de la versión anterior, con respuesta en formato HTML

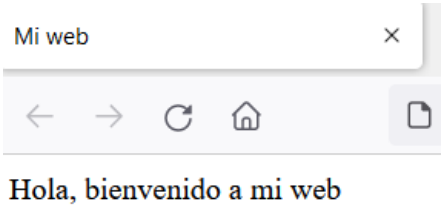
https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/HTML_basics

https://ayudawp.com/html-basico/

Estructura de un documento HTML básico

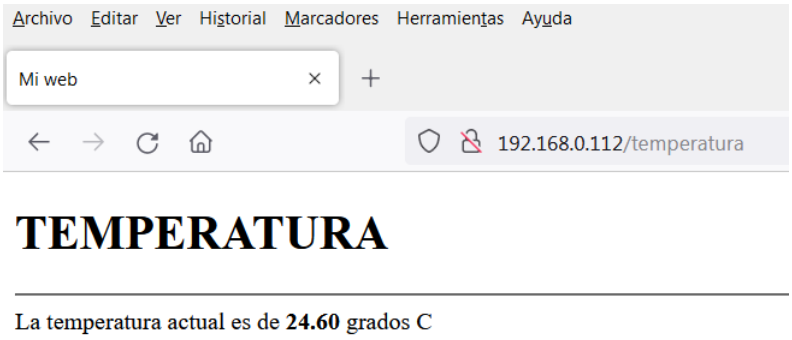
```
<HTML>
        <HEAD>
                <TITLE>Mi web</TITLE>
        </HEAD>
</HTML>
<BODY>
Hola, bienvenido a mi web
</BODY>
</HTML>
```

Esto se vería así en un navegador:



Podemos crear contenido HTML básico mediante bloques:

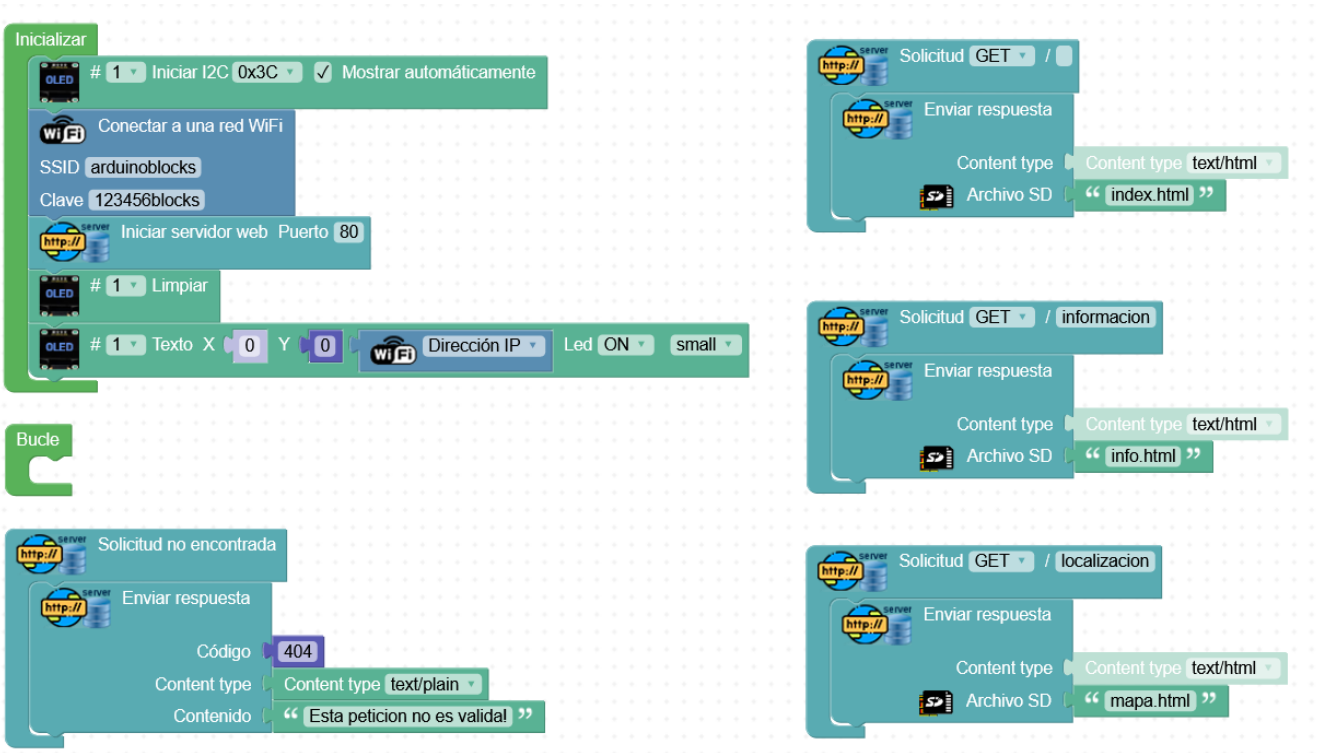
Ejemplo de servidor web que genera una web con un valor del sensor de temperatura:



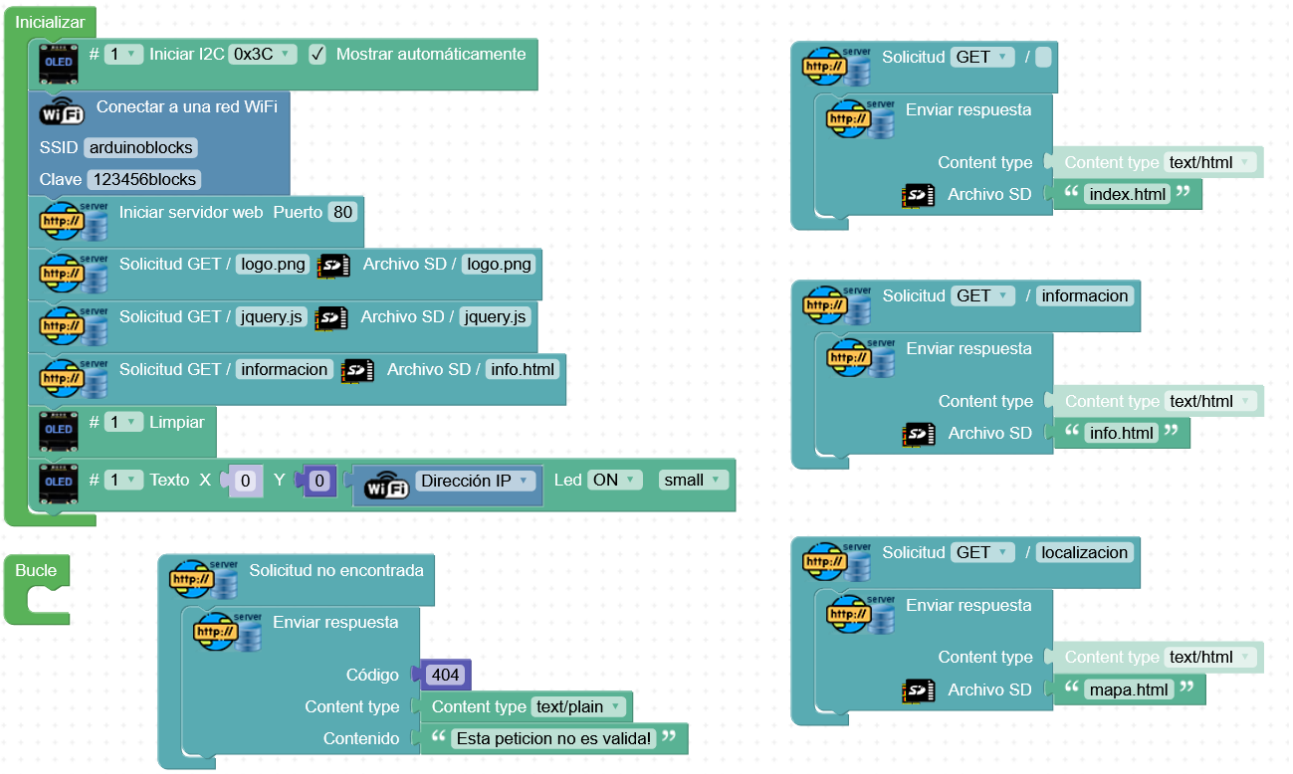
(Refrescando la página (F5) , es decir haciendo otra vez la petición, regeneramos con los valores actualizados del sensor)

4.2.3.-Servidor web + respuesta HTML + SD

Podemos utilizar la tarjeta SD para almacenar archivos a utilizar en nuestro servidor web. Los archivos de la SD los podemos servir como respuesta a una petición HTTP



O “mapearlos” a una petición web directamente en la inicialización, de forma que ante esa petición se serivirá ese archivo desde las SD (especialmnete útil para imágenes que se utilicen dentro del HTML o para archivos estáticos utilizados en nuestras webs: javascript, css, etc…)



Ejemplo:

Web sencilla con documento HTML en la tarjeta SD e imagen (logotipo) en la SD también

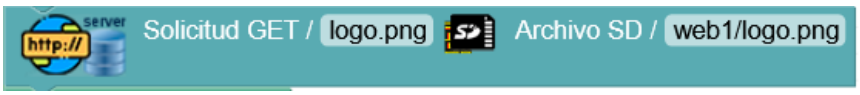
Carpeta “web1”

Código HTML en “inicio.html”

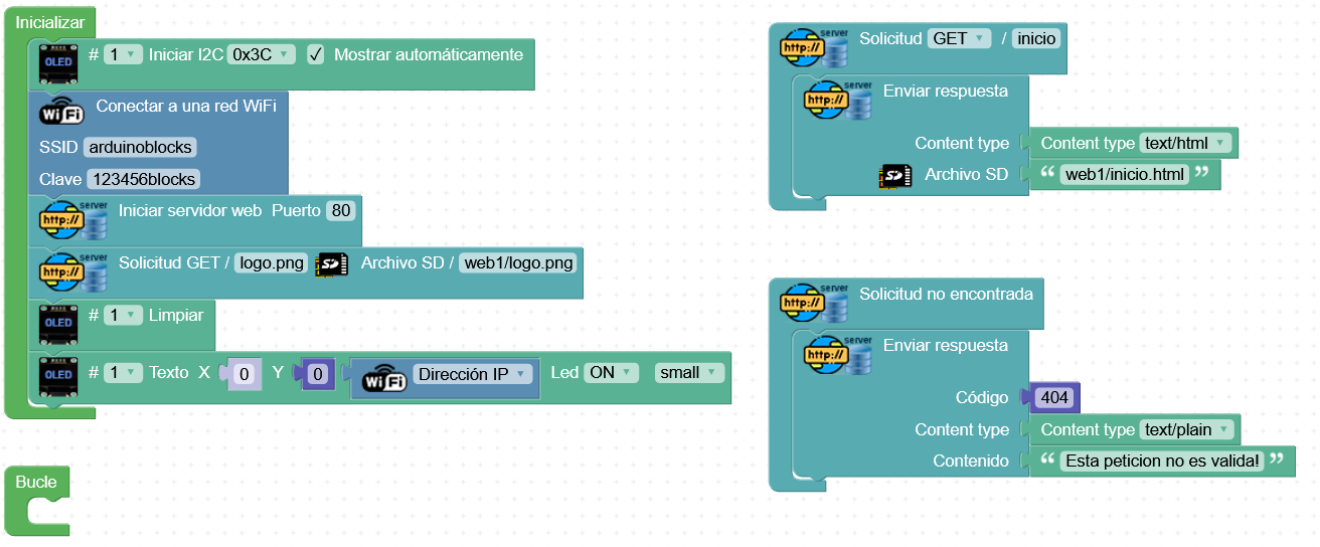
```
<HTML>
<HEAD>
  <TITLE>Web1</TITLE>
</HEAD>
<BODY>

<IMG src="logo.png">
<HR>
Mi primera pagina web
</BODY>
</HTML>
```

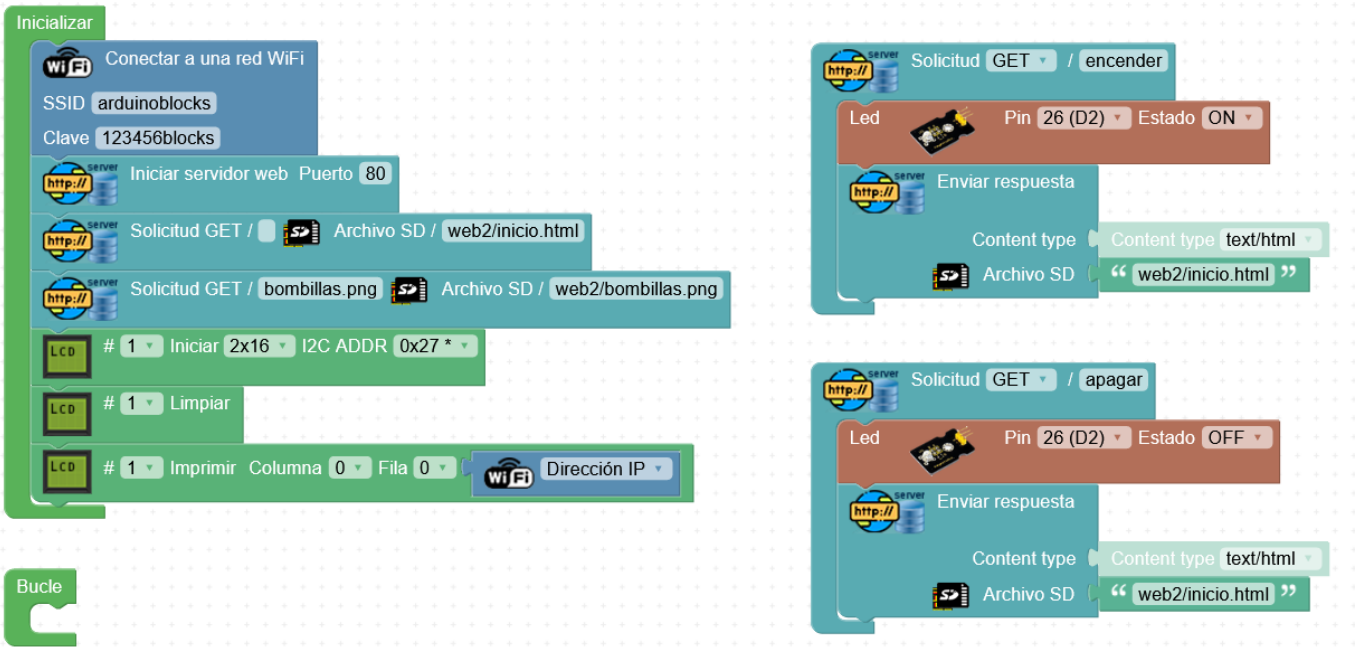
Para que la imagen sea servida por el servidor web (al cargar el archivo HTML el navegador hará una petición HTTP al servidor pidiéndole la imagen “logo.png) hacemos un mapeo:



Ejemplo completo:



Ejemplo encendido/apagado via web



Código HTML “inicio.html”

```
<HTML>
<HEAD>
  <TITLE>Web2</TITLE>
</HEAD>
<BODY>

<IMG src="bombillas.png" width="200">
<HR>
Web2 - Control encendido/apagado (io26)
<HR>
<BR>
<a href="/encender">Encender el led</a>
<BR><BR>
<a href="/apagar">Apagar el led</a>
</BODY>
</HTML>
```

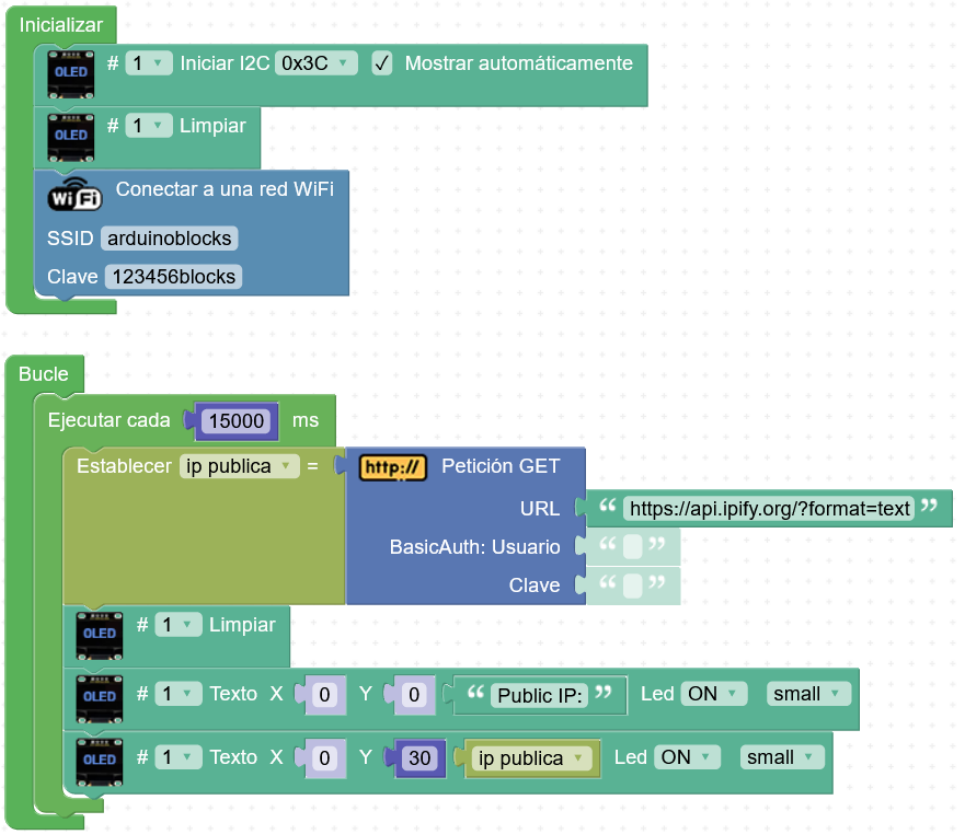
4.3.4.-Cliente web

Podemos hacer peticiones HTTP desde nuestra placa para actuar en servicios remotos HTTP (APIs) y si es necesario también leer la respuesta

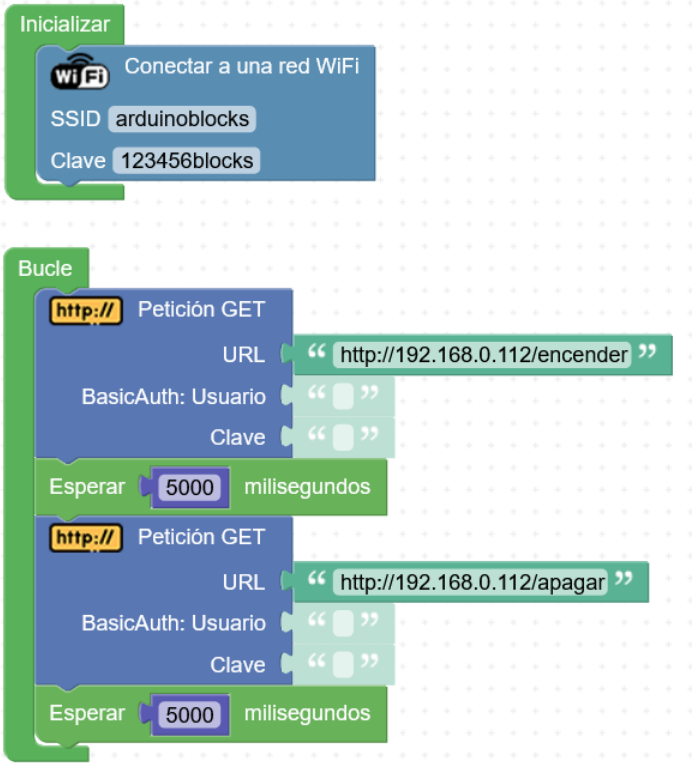
Ejemplo: esta petición nos devuelve nuestra IP pública (IP asignada por nuestro ISP)

https://api.ipify.org/?format=text

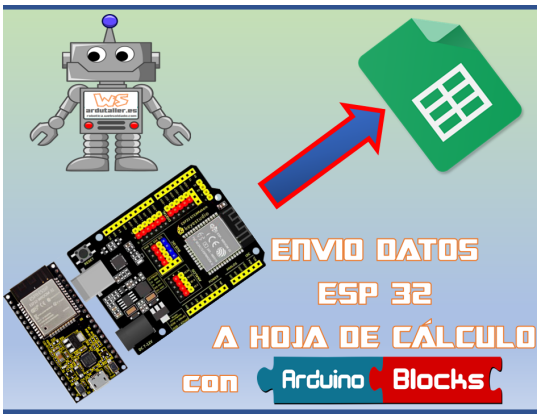
Ejemplo para mostrar en la pantalla OLED la IP pública de nuestra conexión a internet:



Desde una ESP32 podemos enviar una petición web a un servidor web de otra ESP32
Basándonos en el ejemplo **4.2.1**



4.3.5.-Cliente web + Hoja cálculo Google



https://docs.google.com/document/d/1FyLg-QltrnNgkwLuhQARi46Htmpnj3IC3s3grPHzUug/edit

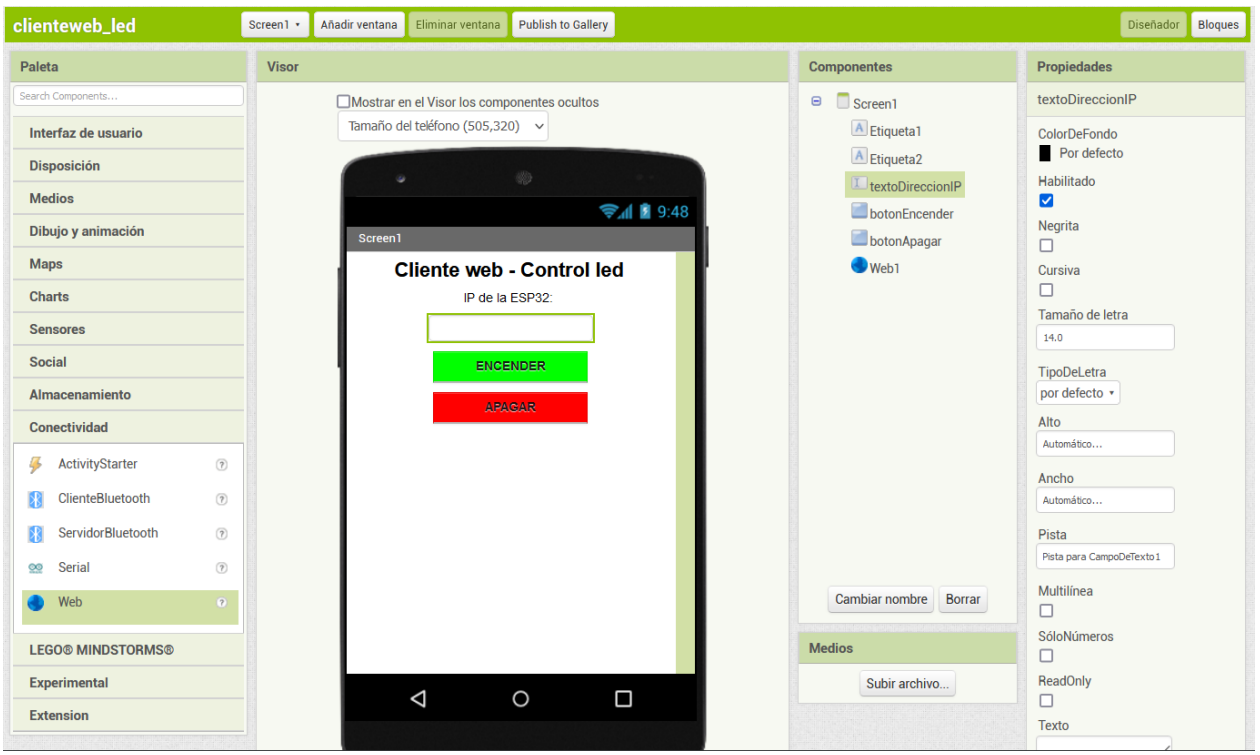
https://www.ardutaller.com.es/taller-de-rob%C3%B3tica-online/arduinoblocks/envio-datos-esp32-steamakers-a-hoja-calculo

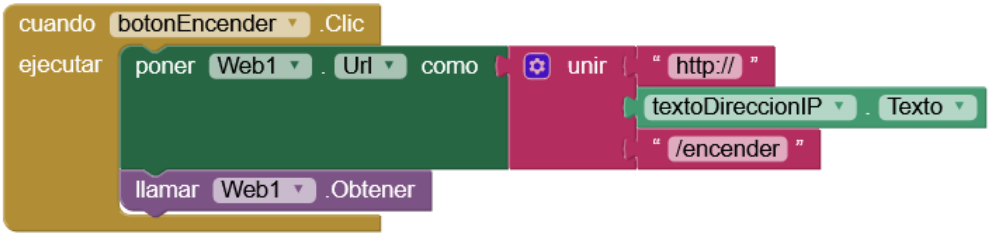
http://www.arduinoblocks.com/web/project/910431

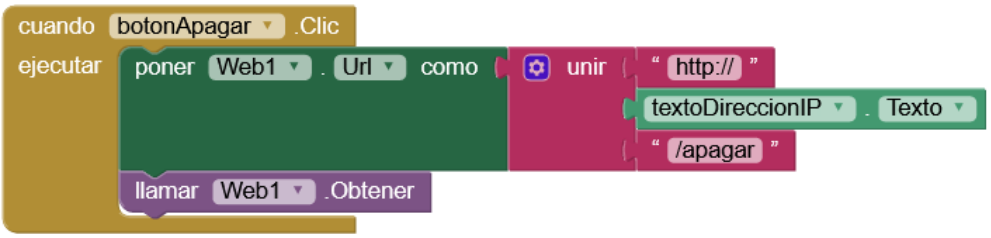
4.3.6.-Cliente web + AppInventor

Creamos una aplicación sencilla en AppInventor que conectará con el servidor web de la ESP32 y enviará una petición /encender o /apagar para controlar el encendido de un led remotamente.

La aplicación permitirá indicar la dirección IP de la ESP32, pues puede variar en cada caso.




cuando botonEncender .Clic
ejecutar
poner Web1 . Url como
unir
" http:// "
textoDireccionIP . Texto
" /encender "
llamar Web1 .Obtener


cuando botonApagar .Clic
ejecutar
poner Web1 . Url como
unir
" http:// "
textoDireccionIP . Texto
" /apagar "
llamar Web1 .Obtener

Sesión 5

- Dispositivo compatible Alexa
    - Led RGB controlado por Alexa
- Bot Telegram
    - Obtener temperatura y humedad vía Telegram
    - Control de led via Telegram

5.1 Dispositivo compatible Alexa

Mediante estos bloques podemo crear un dispositivo que será detectado por nuestro dispositivo compatible con Alexa de Amazon (echo, echo dot, app móvil, …)

El dispositivo emulará el protocolo de iluminación de las bombillas HUE de Philips, de esta forma es sencillo de implementar.

Podremos tenera hasta 8 dispositivos independientes controlados por nuestra placa ESP32.
Cada dispositivo se refiere a un nombre dentro de nuestros dispositivos Alexa al que dirigirnos al hablar con el asistente de voz.

Por ejemplo podemos hacer un sistema con una única placa ESP32 que controle dos luces, una llamada “cocina” y otra “comedor” a la vez que controle la posición de una persiana llamada “ventana” y todo controlado desde el mismo dispositivo (aunque lo más habitual será un único dispositivo Alexa por placa)

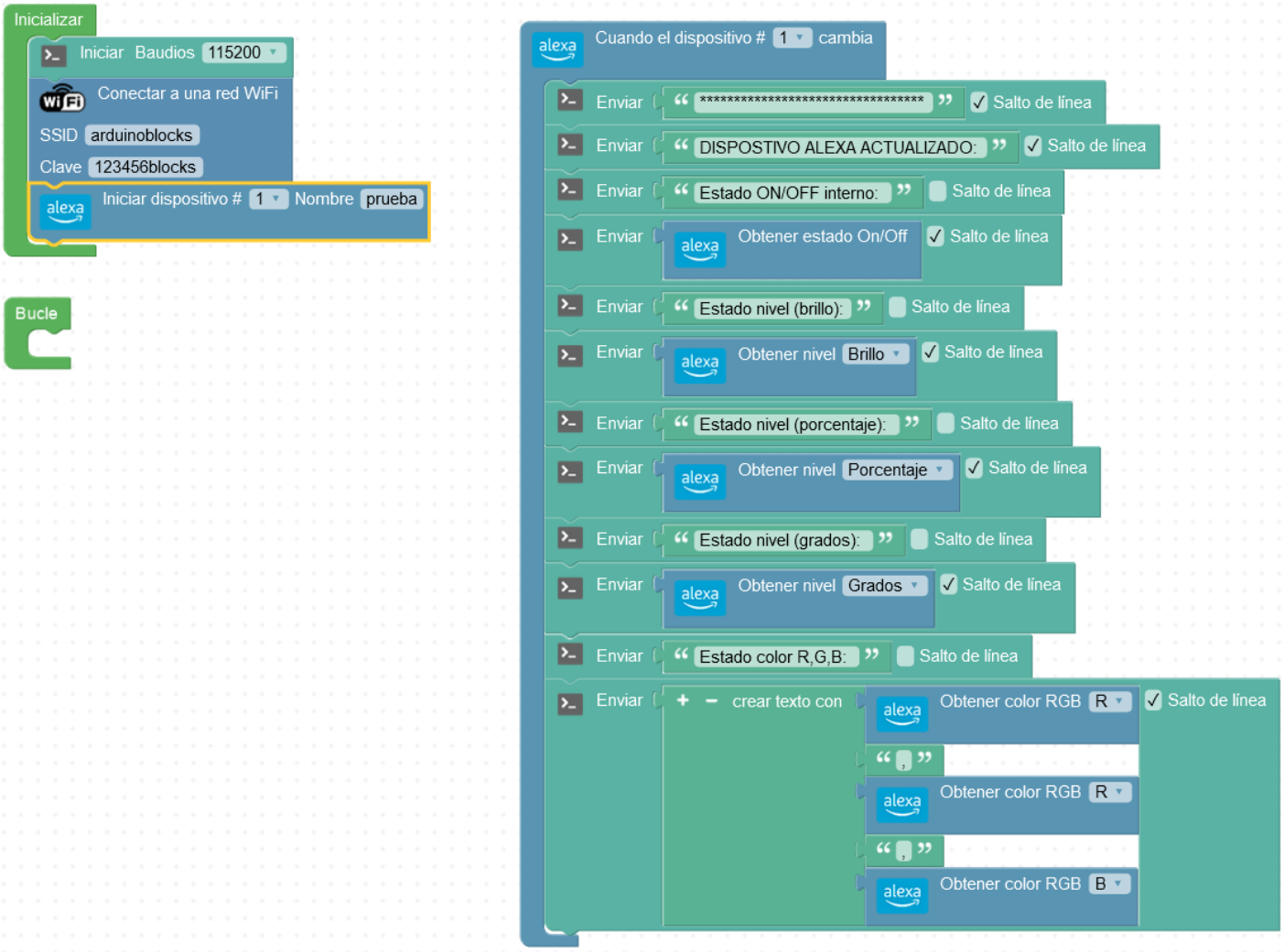
https://m.facebook.com/ArduinoBlocks/videos/arduinoblocks-esp32-steamakers-alexa/4965315083547867/

Ejemplo de programa para crear una dispositivo llamado “*prueba*” y por la consola mostramos los datos internos que podemos alterar mediante commando de voz o a través de la app de Amazon Alexa.

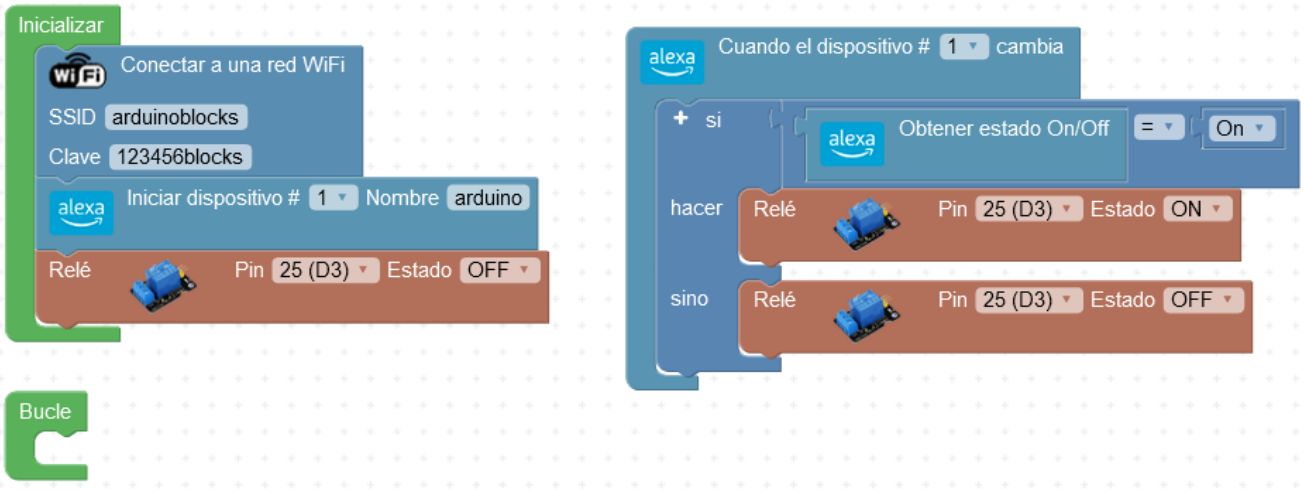
Ejemplo de comandos de voz:

1) *Alexa “buscar dispositivos”*
2) *Alexa “encender prueba”*
3) *Alexa “apagar prueba”*
4) *Alexa “subir intensidad de prueba”*
5) *Alexa “bajar intensidad de prueba”*
6) *Alexa “poner prueba en rojo”*

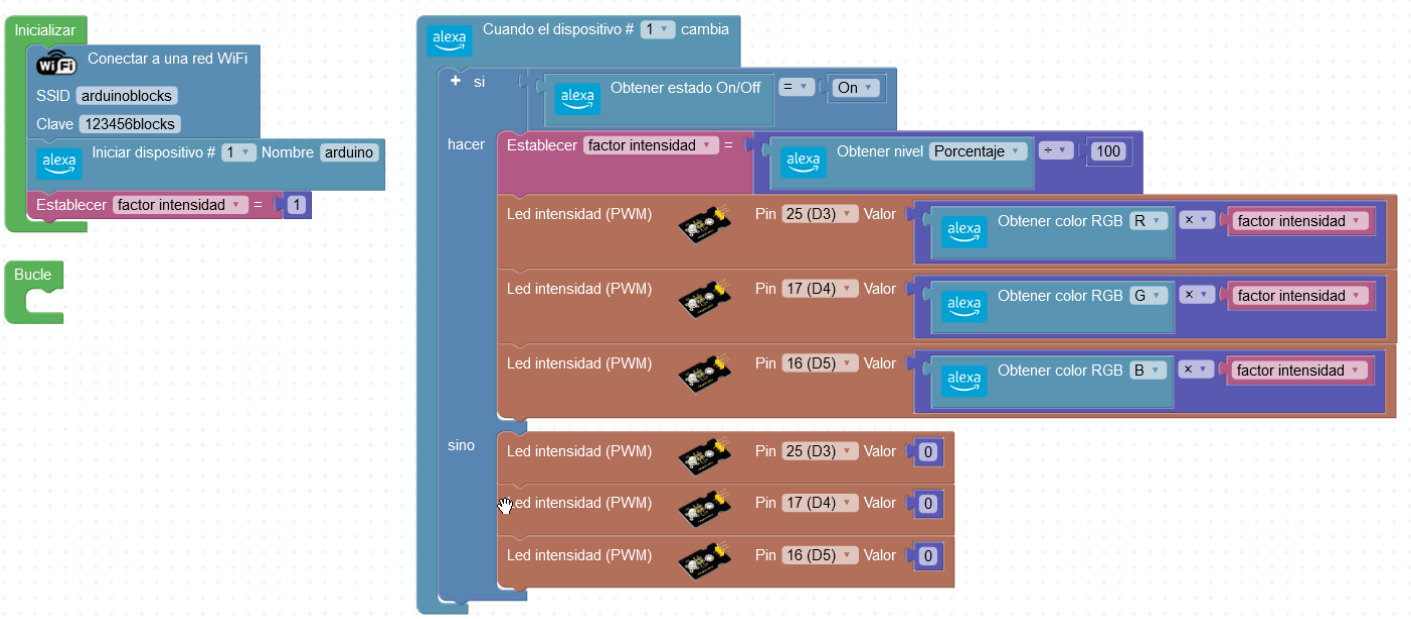
https://www.philips-hue.com/es-es/explore-hue/works-with/voice-control/amazon-alexa/alexa-commands-for-hue-lights



Ejemplo sencillo de encender/apagar:



Ejemplo de control de la intensidad de un led y del color RGB:



Otros ejemplos:

- Control de un motor
- Termostato o control de climatización
- Control de iluminación Neopíxel

5.2 Telegram Bot

Telegram es una plataforma de mensajería y VOIP, desarrollada por los hermanos Nikolái y Pável Dúrov. La aplicación está enfocada en la mensajería instantánea, el envío de varios archivos y la comunicación en masa. Telegram es hoy por hoy una de las grandes alternativas a WhatsApp, una auténtica navaja suiza que ofrece todo tipo de opciones que van más allá de las que suelen dar la mayoría de apps de este tipo.

https://telegram.org/

Telegram es una aplicación que podemos instalar en cualquier dispositivo móvil, y posee una plataforma web desde la que podrás acceder desde tu ordenador.

El envío y recepción de mensajes cuenta con un protocolo de seguridad que cifra cada texto y asegura tus mensajes.

En Telegram podemos crear grupos o canales de difusión.

Un bot de telegram es una aplicación que permite interactuar a través de la aplicación Telegram con usuarios o grupos de forma automática.

https://core.telegram.org/bots/features

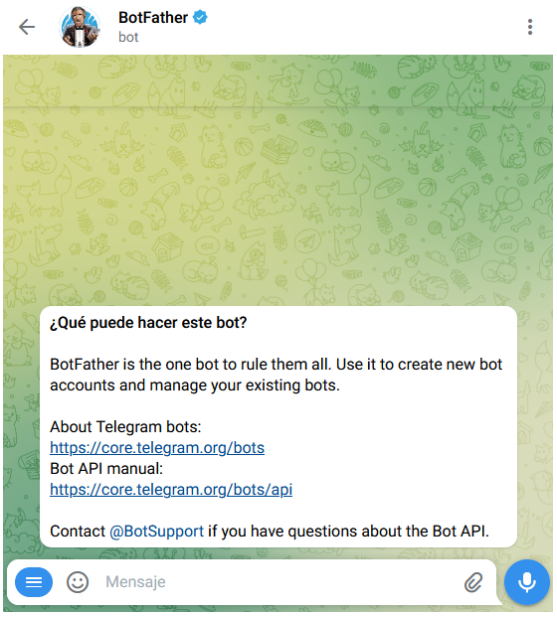
Crear un bot en Telegram

Para crear un Bot en Telegram lo hacemos desde el propio Telegram (app o versión web, como más te guste).

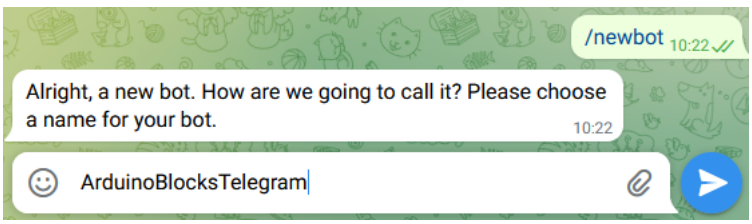
Via web: https://web.telegram.org

Debemos "*chatear*" con el bot de los bots… el **@botfather**

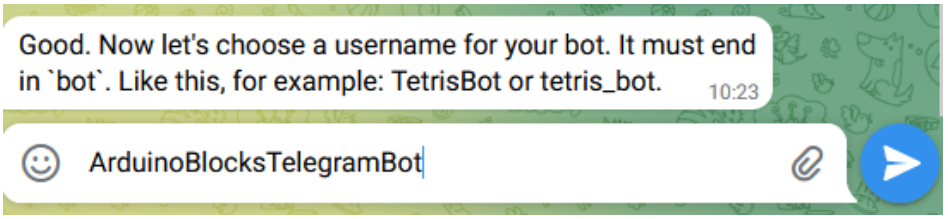
Una vez iniciada la conversación con el **@botfather** nos aparece un listado de comandos para crear el nuevo Bot y poder configurarlos (*¡muy sencillo!*)



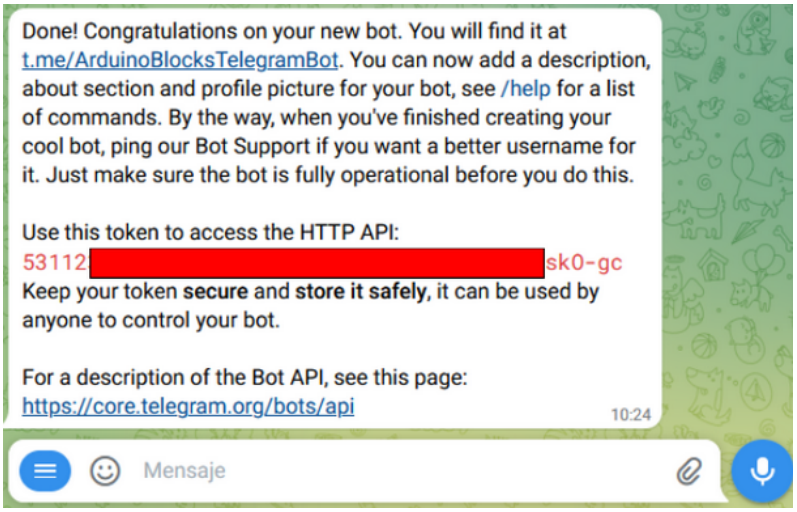
El comando para crear un nuevo Bot es **/newbot** y seguidamente nos pregunta el nombre del Bot:



Seguidamente nos pregunta por un nombre de usuario para este Bot recien creado (el nombre debe acabar en **Bot** o **_bot**):

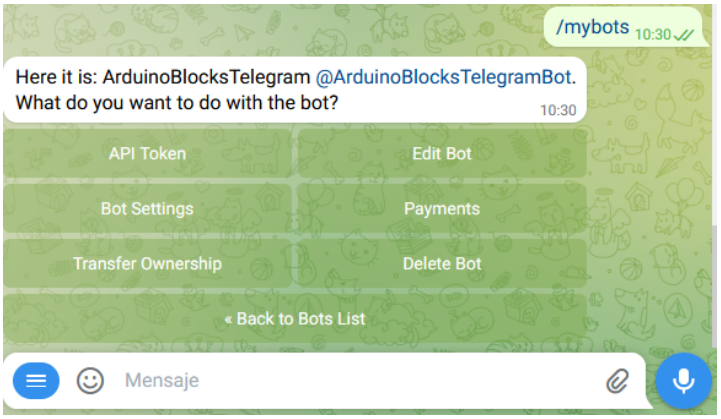


¡Ya lo tenemos! **@botfather** ya ha creado el nuevo Bot y ya nos da un **Token** para conectar con la API de Telegram (bloque Iniciar Telegram Bot en ArduinoBlocks)



<u>El Token para la API es muy importante, debes guardarlo a buen recaudo pues nos permitirá tener el control total de nuestro Bot (no lo difundas)</u>

Si queremos modificar o completar la información de nuestro Bot, lo más sencillo es mediante el comando **/mybots** seguir las opciones que nos muestra en forma de menú y botones para ir completando la información.

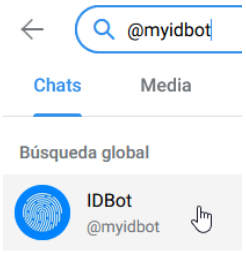


¿Algo más?

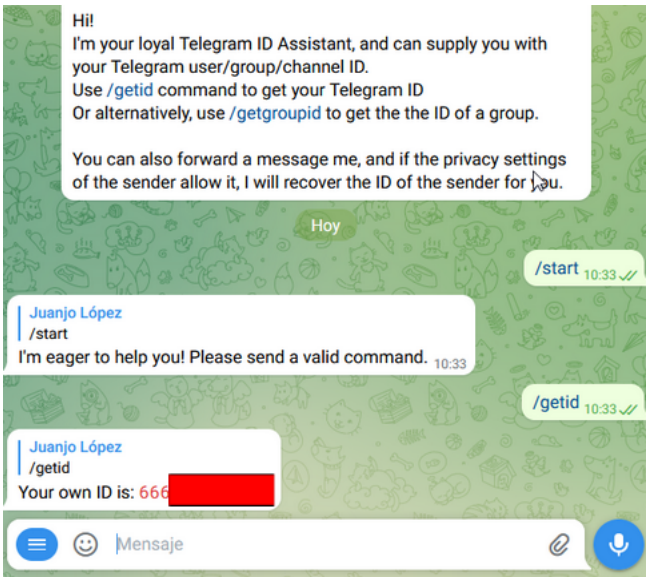
Con ésto ya podríamos empezar a “jugar”, pero si queremos que nuestro Bot pueda enviar mensajes a un chat directamente (para responder no haría falta), necesitamos el ID del Chat en cuestión. Para obtener el Chat-ID podemos preguntárselo a otros Bots.

- **Obtener el ID de mi chat privado** (para que mi Bot envíe mensajes a mi usuario de Telegram directamente)

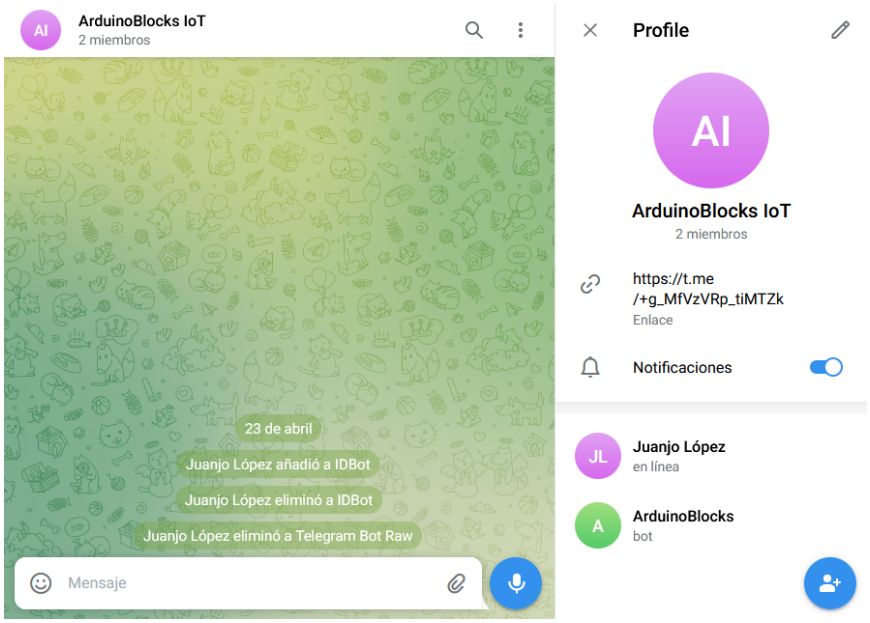
  Busco al Bot **@myidbot** en Telegram y le pregunto mi ID



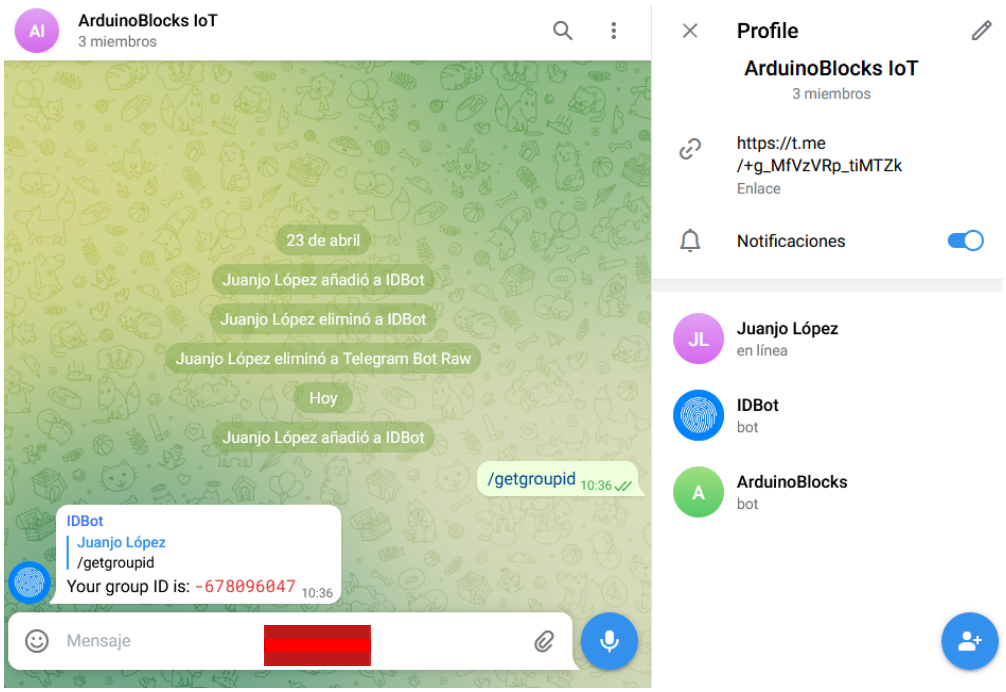
  Debemos ejecutar el comando **/getid**



- **Obtener el ID de un grupo:** Para ello debemos añadir a **@myidbot** al grupo que queremos obtener su ID



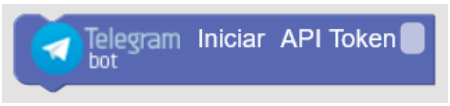
Y ejecutar el comando **/getgroupid** (saldrá un número de ID negativo)



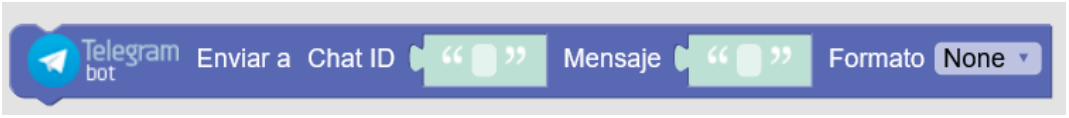
Posteriormente eliminamos al Bot **@myidbot** del grupo

Bloques Telegram-Bot en ArduinoBlocks

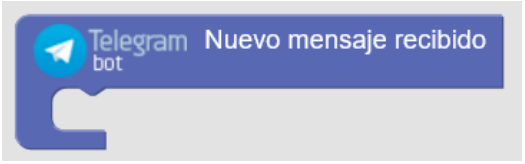
- **Iniciar Telegram Bot**: permite iniciar el sistema Telegram en la ESP32, debemos indicar el Token obtenido para la API al crear el bot (con **@botfather**)



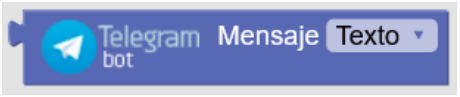
- **Enviar un mensaje:** permite enviar mensajes a cualquier chat de telegram directamente
  - **Chat-ID:** puede ser el ID de una conversación privada a un usuario en concreto, de un grupo o de un canal).
  - **Texto del mensaje**: el texto con el mensaje que se quiere enviar
  - **Formato**: Se puede indicar tres tipos de formato para aplicar estilos al mensaje (negrita, títulos, etc.). Formato: Markdown, MarkdownV2 o HTML



- **Evento de cuando se recibe un nuevo mensaje desde Telegram**:
  - Dentro de este bloque procesaremos el mensaje recibido, pudiendo acceder al texto del mensaje, al remitente (nombre) y al Chat-ID de donde procede el mensaje.
  - Aquí responderemos o actuaremos según el mensaje que nos llegue.

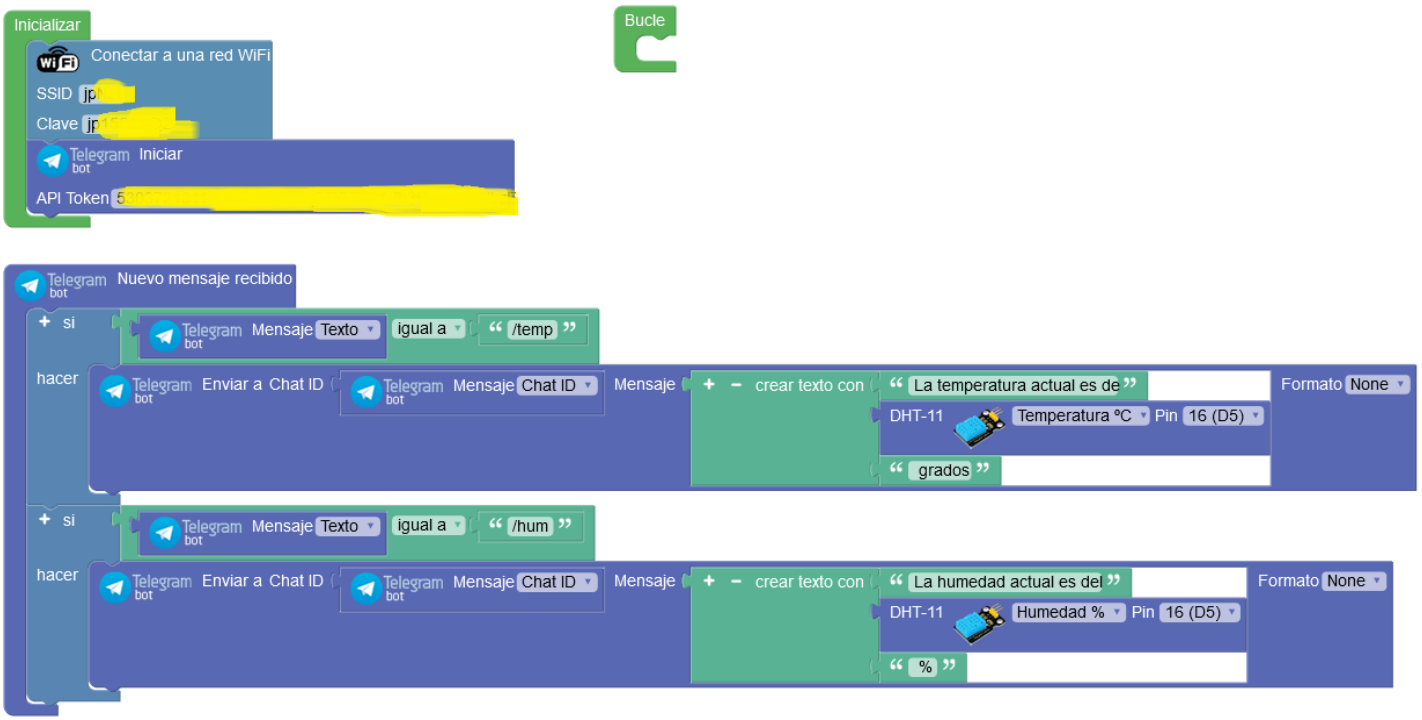


- **Mensaje recibido**: Este bloque sólo tiene sentido usarlo dentro del evento de "nuevo mensaje recibido", si se usa fuera de él su contenido estará vacío.
  - Texto: texto del mensaje recibido
  - Chat ID: ID del chat de donde procede el mensaje. Muy útil para responder a ese mismo chat (pueden llegar mensajes de distintos chats, grupos, etc. por lo que de esta forma identificamos el origen y nos sirve para responder al chat correcto)
  - Remitente: El nombre del usuario remitente del mensaje.
  - Remitente ID: El ID del remitente del mensaje.

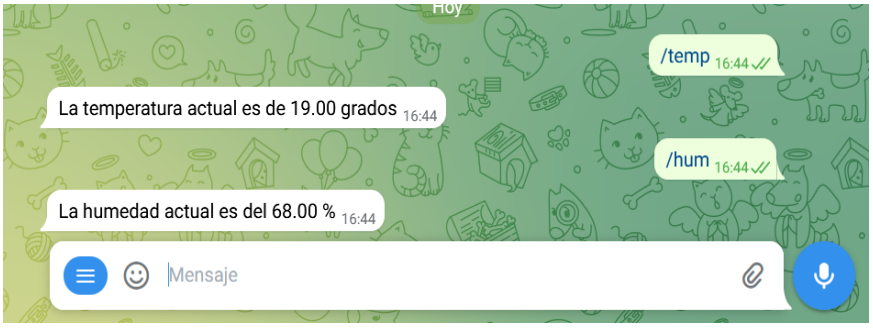


Ejemplo 1: Bot de Temperatura y humedad con DHT11

El Bot responderá con la temperatura y humedad de un sensor DHT-11 conectado a la ESP32 STEAMakers.

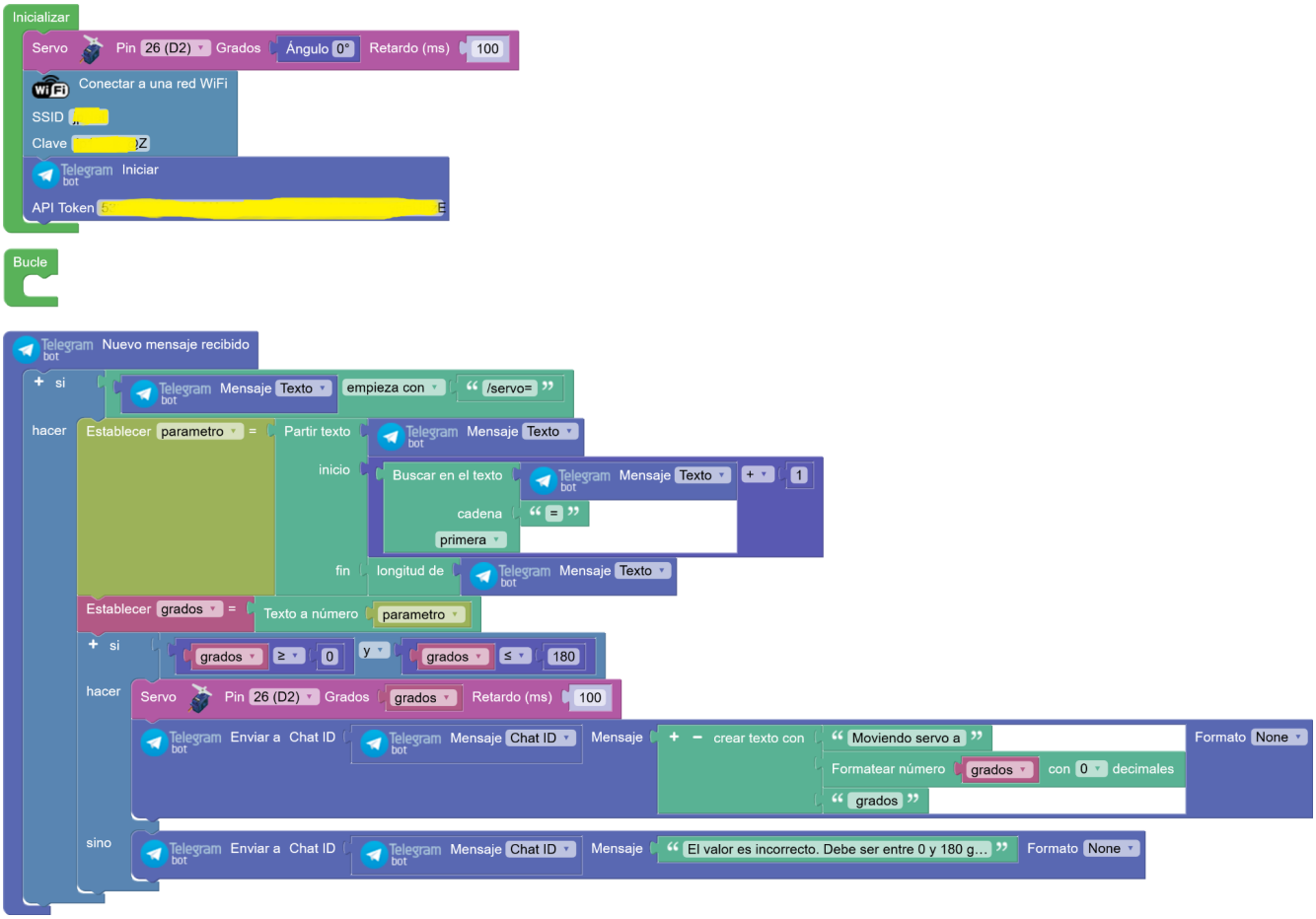


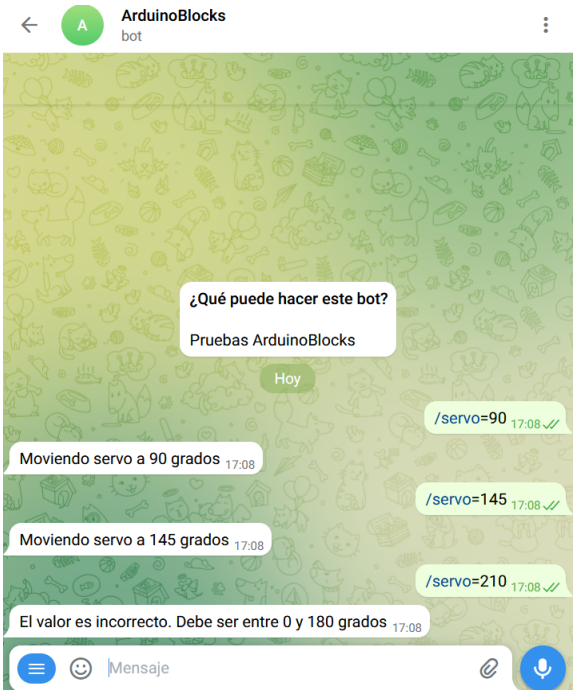
Desde telegram “chateamos” con el Bot de la ESP32 STEAMakers y obtenemos la temperatura y humedad del sensor en tiempo real.



Ejemplo 2: Situar un servo en la posición enviada desde Telegram

Mediante el comando **/servo=GRADOS** situaremos el servo en la posicion deseada con el valor GRADOS





Sesión 6

- AppInventor + HTTP cliente
    - Aplicación appinventor para control Led on/off
    - Aplicación appinventor para mover servomotor

- WifiMesh

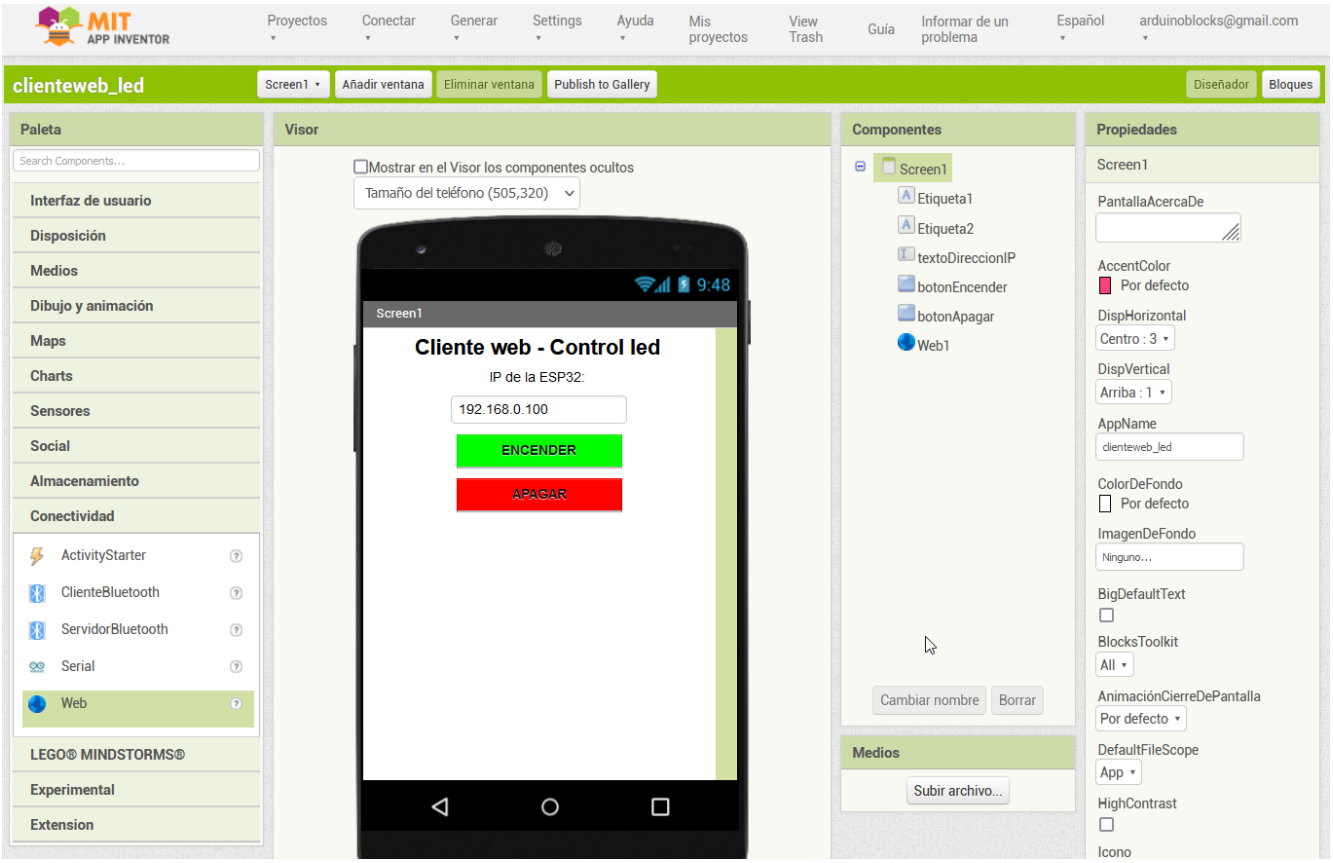
- Proyectos de ejemplo / repaso

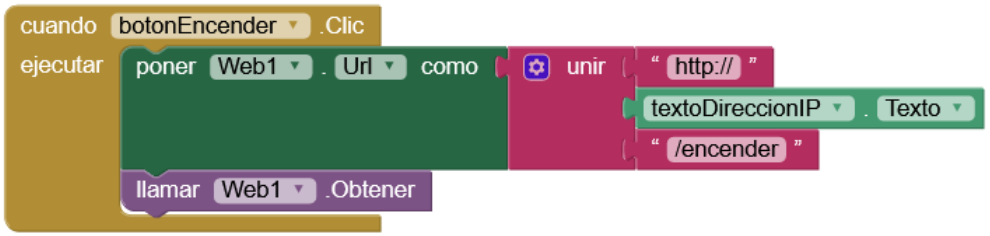
6.1.-AppInventor + HTTP

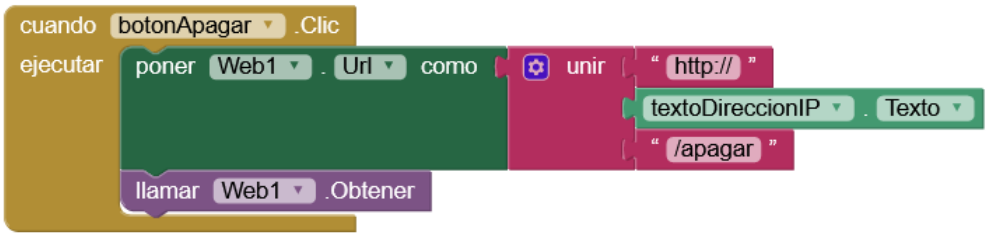
Usaremos un servidor web en la placa ESP32 como interfaz para conectar remotamente desde una aplicación AppInventor que hará peticiones HTTP para controlar remotamente

6.1.1.-Aplicación appinventor para control Led on/off

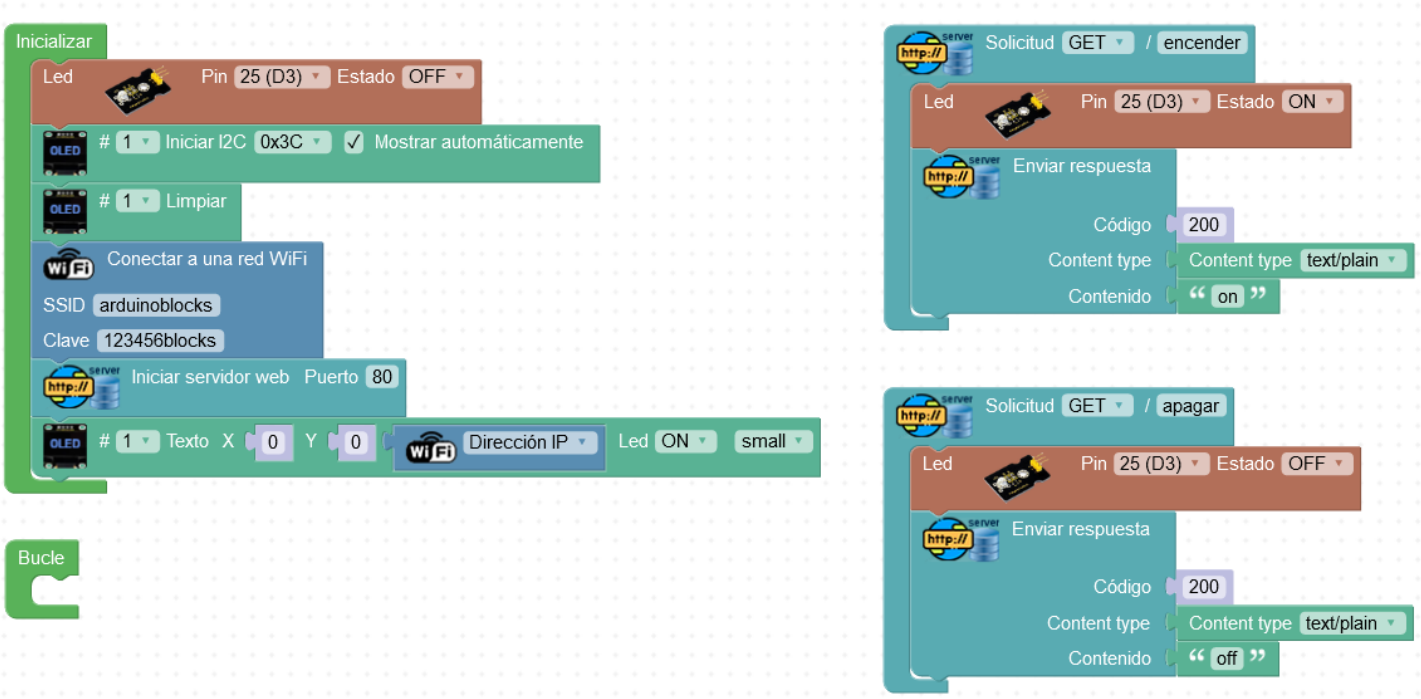
Aplicación en AppInventor:



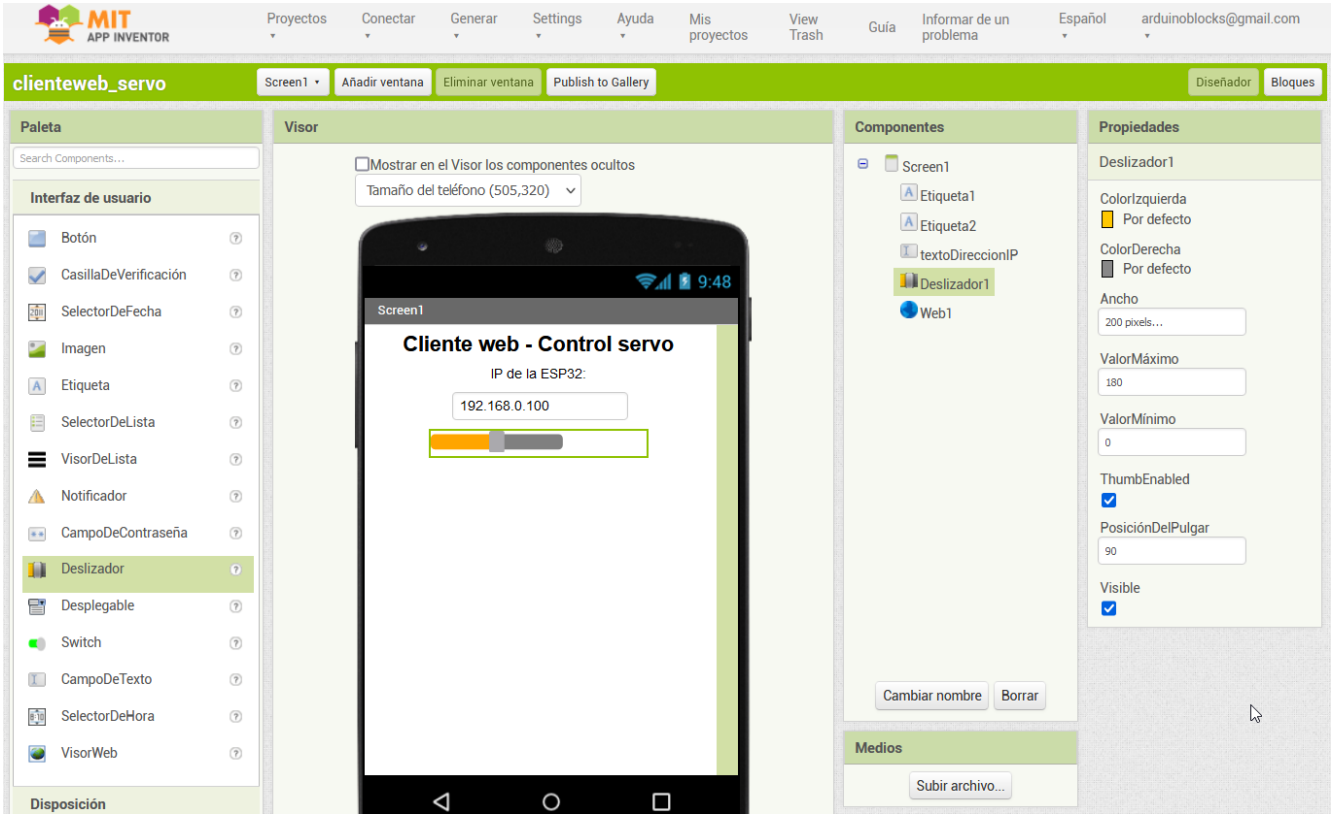




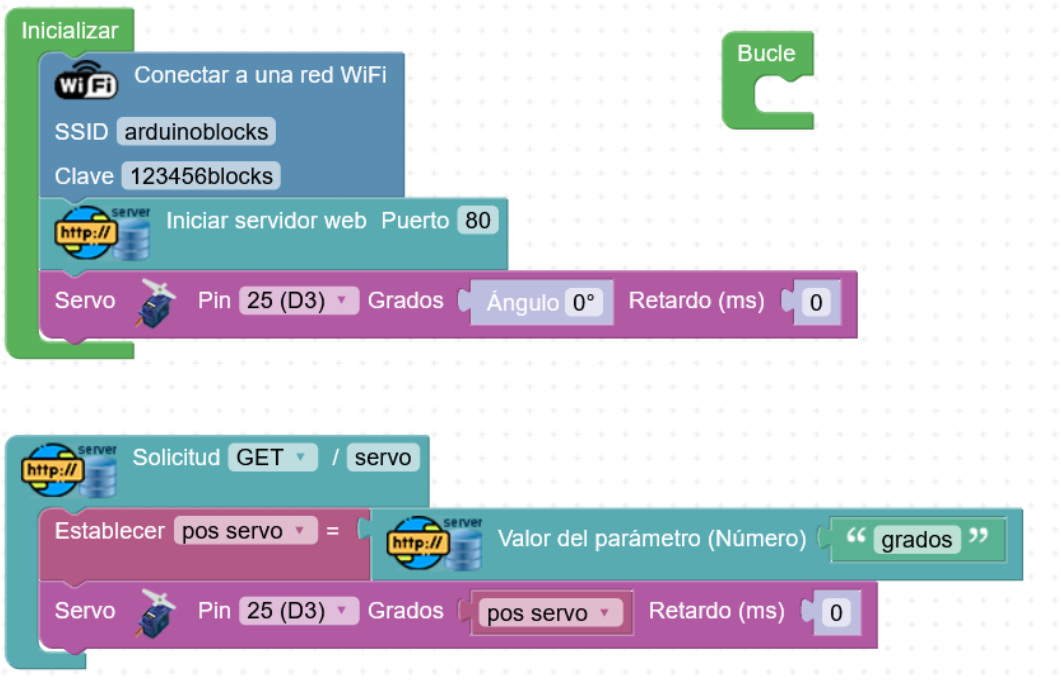
Servidor web en ESP32 STEAMakers + led:



6.1.2.-Aplicación appinventor para mover servomotor



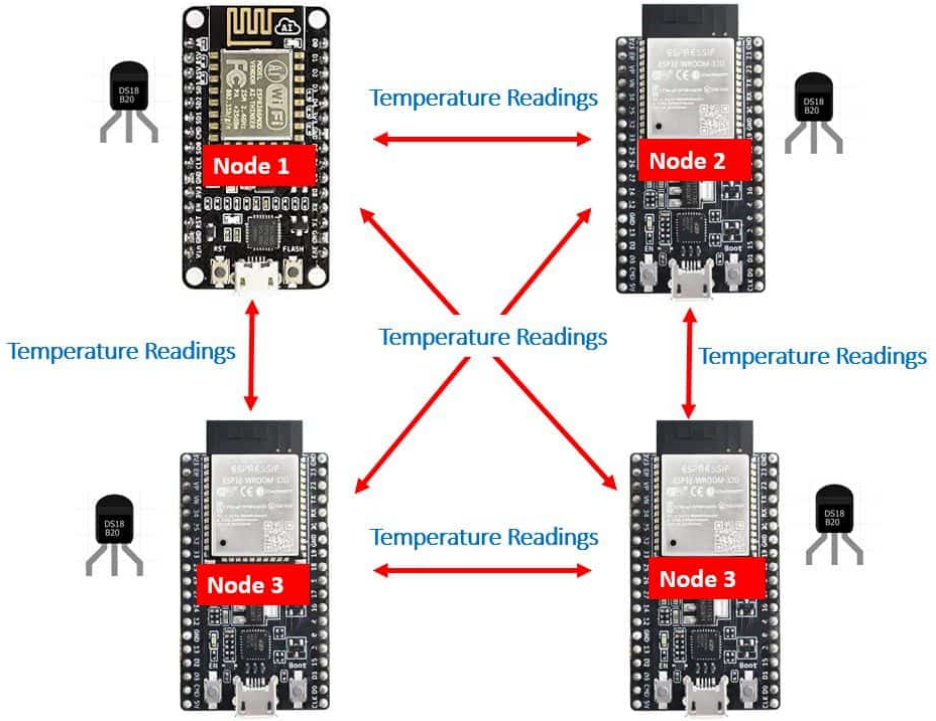
Programa de control del servo y servidor web en ESP32 STEAMakers



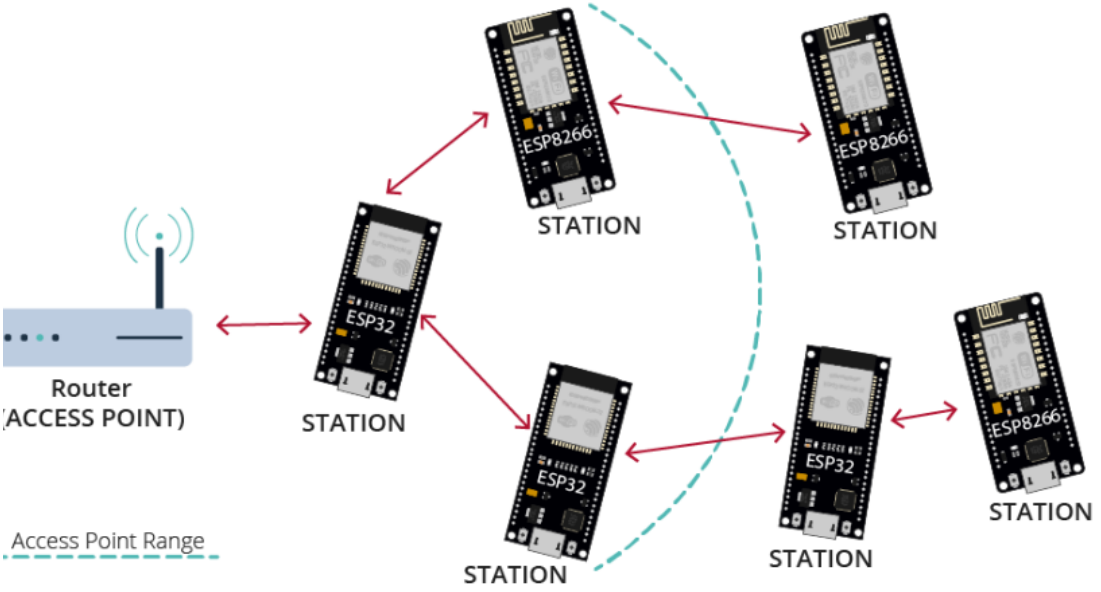
6.2.-WifiMesh

WifiMesh permite comunicar varias placas ESP32 STEAMakers entre ellas sin necesidad de conectar a una red WiFi previa. Son los propios dispositivos los que hacen de punto de acceso WiFi (AP) y de clientes WiFi (STA) a la vez.

Entre ellos podremos enviar mensajes básicos para intercambiar información todos con todos.



La red WifiMesh puede distribuirse usando como “repetidores” a los mismos dispositivos



Todos los dispositivos deben iniciar el WifiMesh con el mismo ID, la misma clave y el mismo puerto.

Al enviar mensajes podemos simplemente enviar un mensaje, o combinar un mensaje + datos para de forma sencilla tener comando + parametros (por ejemplo)

Ejemplo: envía el mensaje “hola” a todos los nodos conectados a la red mesh

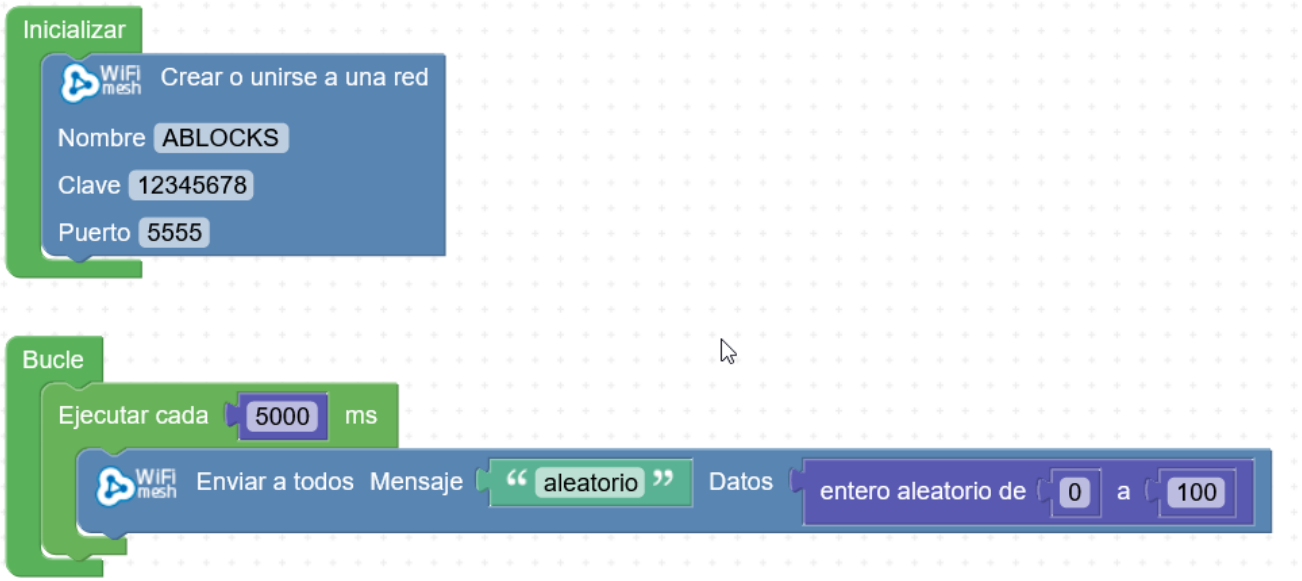
Ejemplo: envía el mensaje “temperatura” junto a un valor de temperatura numérico a todos los nodos conectados a la red mesh

Cualquier nodo conectado a la red mesh puede recibir los mensajes de otros nodos:

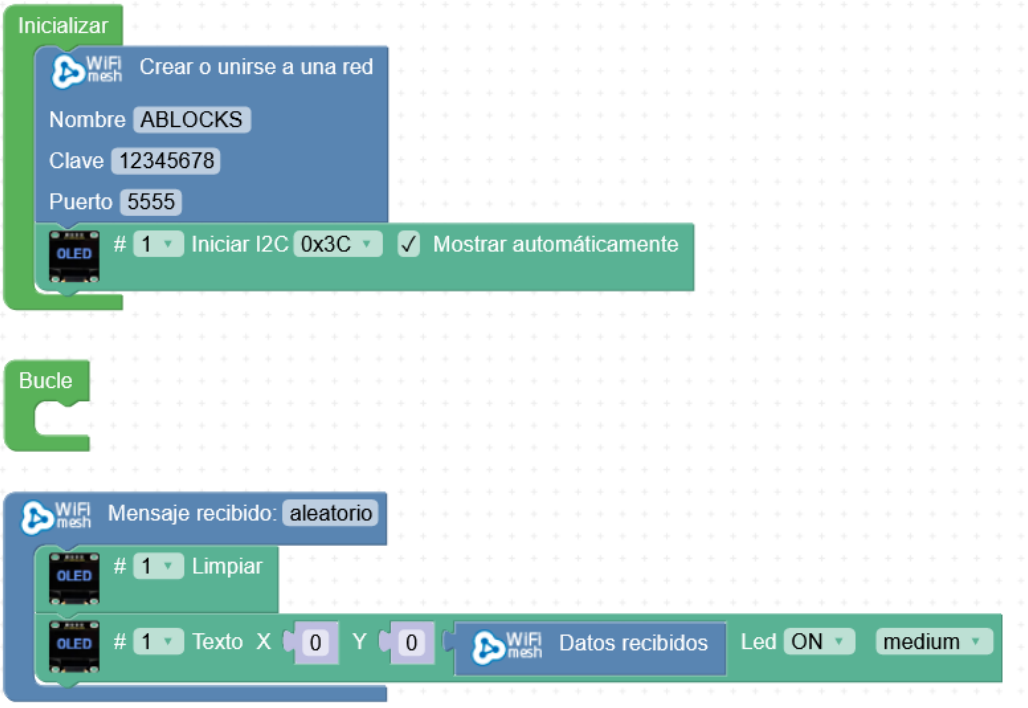
Dentro del evento podemos obtener los datos del remitente, el mensaje y los datos (si los hay) :

También podemos responder a un evento para un mensaje en concreto:

Ejemplo: dispositivo “maestro” que envía al resto un valor “aleatorio” cada 5s



Nodos esclavos, al recibir el valor lo visualizan en una pantalla OLED:

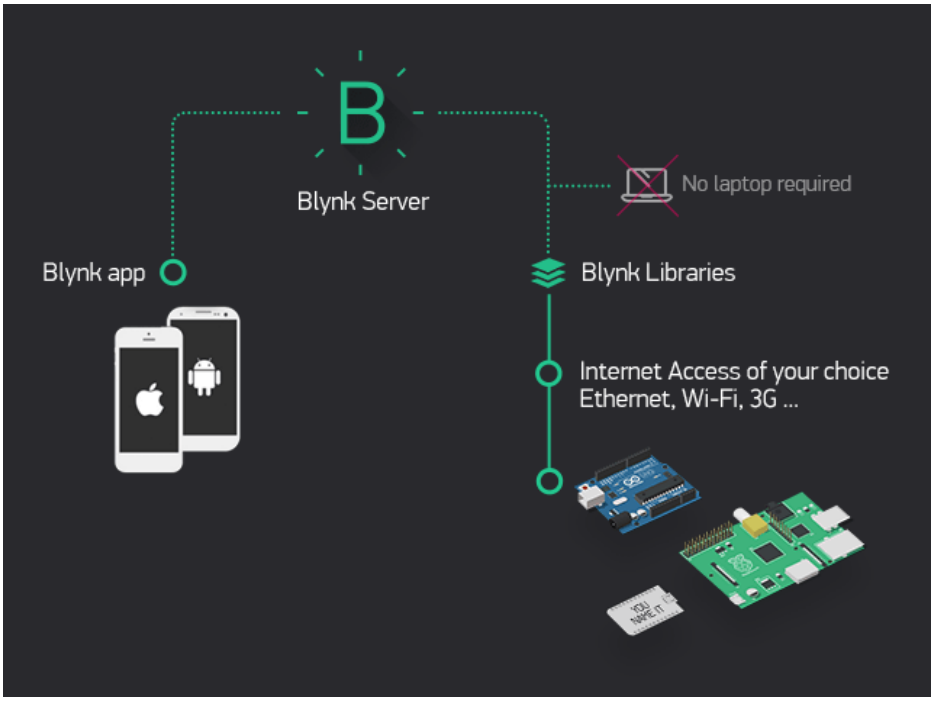


6.3.-Blynk (legacy)

Blynk es un entorno basado en un servidor propio y una aplicación (iOS, Android) que permite fácilmente crear aplicaciones de control y monitorización para IoT.

https://blynk.io/

La versión “legacy” ha quedado sustituida por una versión actual (mucho más restringida y de pago en muchas características), pero la versión anterior la podemos instalar en nuestro propio servidor y podemos instalar de igual forma la aplicación Blynk para dispositivos móviles.



Noticia sobre el cambio de Blynk Legacy a Blynk IoT
https://blynk.io/blog/what-will-happen-to-the-legacy-blynk-platform

Para más información sobre como instalar y usar “Blynk Legacy” podéis consultar la siguiente web:
https://libros.catedu.es/books/rover-marciano-con-arduinoblocks-e-internet-de-las-cosas-iot/chapter/3-blynk

Bibliografía y enlaces de interés:

http://www.arduinoblocks.com

https://shop.innovadidactic.com/es/

https://www.youtube.com/watch?list=PL1pKD-Bz2QBAgfy580m8OaQ2Z60v6DOhC&v=MQjIEI7I4ik&feature=youtu.be

https://wiki.keyestudio.com/Category:Arduino_Board

https://thingspeak.com/

https://io.adafruit.com

https://blynk.io/

http://ai2.appinventor.mit.edu

https://es.wikipedia.org/wiki/ESP32

https://www.ardutaller.com.es/

https://didactronica.com/arduino-desde-cero-arduinoblocks/

https://www.luisllamas.es/esp32/

https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

https://www.luisllamas.es/arduino-i2c/

https://www.youtube.com/watch?v=QRQoWKgjhgU

https://www.youtube.com/c/ArduinoBlocks

Redes sociales:

https://twitter.com/arduinoblocks

https://www.youtube.com/@arduinoblocks

https://t.me/arduinoblocks

https://t.me/innovadidactic_comunidad

arduinoblocks@gmail.com

Noviembre 2022

Maleta de la Innovación 4.0 IoT con ESP32 STEAMakers

Juanjo López
ArduinoBlocks.com
arduinoblocks@gmail.com