Maleta de la Innovación 4.0 IoT



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)
 - <u>https://shop.innovadidactic.com/es/standard-placas-shields-y-kits/1455-keyestudio-sma</u> <u>rt-home-para-arduino-con-placa-keyestudio-plus.html</u>
- Material extra para este curso:
 - Placa ESP32 STEAMakers
 - Pantalla OLED
 - Reloj RTC (+pila)
 - Lector RFID
 - Tarjeta uSD

¿Qué es loT?

IoT viene del inglés "Internet Of Things", es decir, "Internet de las cosas".

La definición de loT 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.

loT 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&f eature=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.

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

<u>Subir (upload)</u>: 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"

<u>Chromebook</u>: 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-1vQSrOKHpbLQHVbGFdAvp7DcndoftoHDI20nvw GMaxu_7bGc1bUCmi4U6DZrJWRSudc2iXBg43QMuzCT/pub

Presentación "Introducción a ArduinoBlocks"

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

Empezando con ArduinoBlocks

- Registro / Inicio de sesión
 - o http://www.arduinoblocks.com/web/site/login
- Mis proyectos
 - o 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)



-<u>Bus I2C</u>

(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-direc cionable-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

L Nuevo proyecto personal

| Tipo de proyecto | ESP32 STEAMakers |
|---------------------|--|
| Nombre | blink |
| Descripción | Normal ≑ ▲ B I <u>U</u> S = = = = = % |

3) Crear el siguiente programa:



 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



Reloj RTC i2c + Pila

DS3231 R2 R1 Linium Ceil 31 W8 W1 Keuestudip Ceil Pantalla OLED i2c 128x64 píxeles



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):

| icializ | zar | |
|---------|---|---------------------------------------|
| >_ | Iniciar Baudios 115200 🔹 | |
| | | |
| ucle | | |
| Est | ablecer temperatura T = C DHT-22 | jo Temperatura ⁰C ▼ Pin (26 (D2) ▼ |
| Est | ablecer humedad 🔹 = 🕻 DHT-22 💉 | Humedad % 🔻 Pin 26 (D2) 💌 |
| | | + |
| >_ | Enviar 🤇 🤲 Temperatura actual (C): 🥲 | Salto de línea |
| >_ | Enviar 🕻 temperatura 🔹 🗸 Salto de línea | · · · · · · · · · |
| ۶_ | Enviar (44 Humedad actual (%): 🤊 🔲 | Salto de línea |
| ۶_ | Enviar 🚺 humedad 🕥 🗸 Salto de línea | · · · · · · · · · · · · · · · · · · · |
| >_ | Enviar 🤇 🤲 💶 💴 🗸 Salto de línea | |
| | perar 5000 milisegundos | ••••• |
| Esp | | |

Programa de ejemplo (serial plotter)

| + + + + + + + + + | • |
|-------------------------|--------|
| D2) 🔹 | + |
| | + |
| | |
| | |
| | |
| | (D2) • |

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)



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):

| Inicializar | | | | | | | |
|--|---------------------------------|-----------------------|--------------------------------|----|--------|-------------|--------|
| OLED # 1 V Iniciar I2C 0x3 | C 🔻 🗸 Mo | strar auton | náticamente | | | | |
| | | | | | | | |
| | | | | | | | |
| Bucle | | | | | | | |
| UNDER # 1 V Limpiar | | | | | | | |
| | | | | | | | |
| contar con radio v desde D | 0 hasta | 4 0 de | a 🛯 1 | | | | |
| contar con radio v desde b hacer bled # 1 v Circulo | 0 hasta 0 X C 64 | 40 de Y 3 2 | a 11 R (radio | Le | d ON 🔻 | 🗸 Re | llenar |
| contar con radio desde hacer # 1 Circulo Esperar 100 mi | 0 hasta 0 X 64 lisegundos | 40 de Y 3 2 | a 1 R 1 radio | Le | d ON 🔻 | √ Re | llenar |
| contar con radio v desde hacer Esperar 100 mi Esperar 2000 miliseguno | 0 hasta | 40 de Y 532 | a 1 R (radio | Le | d ON 🔻 | √ Re | llenar |

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"

| 🔸 🕞 🔁 👬 esp32 iot | - 0 | le | d I | oit | ma | ap | |
|------------------------------|-----|----|-----|-----|----|----|--|
| LCD - Symbol editor | | | | | | | |
| III OLED - Bitmap editor | | | | | | | |
| LedMatrix - Bitmap editor | | | | | | | |
| III NeoPixel - Bitmap editor | | | | | | | |
| | | | | | | | |
| SRTTTL Info | | | | | | | |
| | | | | | | | |

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-A rduino_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):

| # 1 ▼ Iniciar I2C 0x3C ▼ ✓ Mostrar automáticamente | |
|--|-----------------|
| OLED | |
| | |
| | |
| | |
| | |
| Bucle | |
| | |
| | |
| | |
| | |
| | |
| | |
| I I I Bitmap X C O Y C O 128,64,0x0,0x0,0x0,0x0,0x7,0xf0,0 | x0,0x0,0x0,0x0, |
| Image: Second | x0,0x0,0x0,0x0, |
| Image: Weight of the second | x0,0x0,0x0,0x0, |
| OLED # 1 Bitmap X 0 Y 0 128,64,0x0,0x0,0x0,0x0,0x7,0xf0,0 Esperar 2000 milisegundos ULED # 1 V Limpiar | x0,0x0,0x0,0x0, |
| OLED # 1 Bitmap X 0 Y 0 128,64,0x0,0x0,0x0,0x0,0x7,0xf0,0 Esperar 2000 milisegundos # 1 Limpiar # 1 Limpiar | x0,0x0,0x0,0x0, |
| 0LED # 1 Bitmap X 0 Y 0 128,64,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x | x0,0x0,0x0,0x0, |
| 0LED # 1 Bitmap X 0 Y 0 128,64,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x | x0,0x0,0x0,0x0, |
| Image: Weight of the second | x0,0x0,0x0,0x0, |

Programa de ejemplo (icono moviéndose por la pantalla)



https://img.freepik.com/vector-premium/logo-icono-copo-nieve-azul-simbolo-nieve-vector-navidad_231 786-4028.jpg

OLED - Bitmap Data







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:



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.<u>4.-RTC</u>

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



| Iniciar Baudios 1152 | 200 - | | | | | | | | | | | | | | | | | |
|--|---------|-------|-------|------------|-------|--------------|------|------|-------|-----|---|---|--------------|-----|------|----|-------|----|
| | | | | | | | | | | | | | | | | | | |
| + | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| ucle | | | | | | | | | | | | | | | | | | |
| | | | | | | 1 | | | | | | | | | | | | |
| Enviar 🕻 🧐 Fijando | fecha y | hora | de R1 | Г С | " | \checkmark | Sal | to d | le lí | nea | a | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| Reloj (I2C) | | | | | | | | | | | | | | | | | | |
| 1 March 1 Marc | | | | | | | | | | | | | | | | | | |
| | 47 | | | | | | | | | | | | | | | | | |
| Dia= | 17 | | | | | | | | | | | | | | | | | |
| Mes= | 10 | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| Año= | 2022 | 1.1 | | | | | | | | | | | | | | | | |
| Hora= | 16 | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| Minuto= | 24 | | | | | | | | | | | | | | | | | |
| Segundo= | 20 | | | | | | | | | | | | | | | | | |
| | 20 | | | | | | | | | | | | | | _ | | | |
| Enviar C 🧉 RTC pu | esto en | hora! | desc | onec | ta el | mo | dulo | o pa | ara (| ∋v. | | " | \checkmark |) s | alto | de | e lír | ne |
| | | | | | | | | | | | _ | | | | | | | |

2.4.2.-Fecha y hora RTC en pantalla OLED



2.<u>5.-WiFi</u>

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.



ArduinoBlocks :: Consola serie

| Baudrate: 115200 V Conectar Desconectar Limpiar | |
|--|-------------------------------------|
| Enviar | L. |
| CONTENIDO DEL ARCHIVO DATOS.TXT En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivil 🕮 a un hidalgo de los de l astillero, adarga antigua, rocil 🕮 n flaco y galgo corredor. Una olla de algo mal 🕮 s vaca que carnero, salpicol 🛍 n las mal 🕮 | lanza en Is noches, |
| duelos y quebrantos los salimbados, lentejas los viernes, algulimn palomino de animadidura los domingos, consumiliman de su hacienda. El resto della concluitiman sayo de velarte, calzas de velludo para las fiestas con sus pantuflos de lo mis | n las tres partes smo, los diÌas |

3.1.2.-Escritura en archivos de texto

| ializar | | | | | | | | | | | | | | | | | | | |
|--|--------------------------------------|---|---------------------------------|--------|---------------------|----------------------------------|-------------------------|--------------|-------|------------------|-------------------|-----|-----------|-------------|------------------|----|------|------|---------------------|
| Iniciar Baud | ios [115 | 200 - | | | | | | | | | | | | | | | | | |
| | | 200 | | | | | | | | | | | | | | | | | |
| SD Iniciar | | | | | | | | | | | | | | | | | | | |
| | | + + + + | | | | | | | | | | | | | | | | | |
| sperar 1000 | milise | gundos | | | | | | | | | | | | | | | | | |
| | | | | | | | | - | | | | | | | | | | | |
| – Enviar 🗘 🕫 | Escribi | endo un | valor | aleato | rio ca | da 5s | " | V | Salto | de | líne | a | | | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| <u></u> | | | | | | | | | | | | | | | | | | | |
| + + + + + + + + + + + + + + + + + + + | | | | | | | | | | | | | | | | | | | |
| cle Sole Escribir (| " ale | eatorio.tx | t " | Texto | • | entero | aleato | orio c | le 🎙 | 1 | a | | 100 | | • | Sa | ilto | de l | líne |
| le 52 Escribir (| " ale | eatorio.tx o.del.arc | t " | Texto | ۰ (e Salt | entero : o de lí | aleato | prio c | le 🎙 | 1 | a | | 100 | | | Sa | ilto | de l | líne |
| le SS Escribir (S- Enviar (4 | Tamañ | eatorio.tx o del arc | tt " hivo: | Texto | o (e) Salt | entero : o de lín | aleato | prio c | le 🖣 | - - - 1 | • • • • • • | | 100 | + + + | * * * * | Sa | ilto | de l | líne |
| le SSE Escribir (SEnviar () SEnviar () | Tamañ | eatorio.tx o del arc amaño de | t) " hivo:) e arch | Texto | e Salt | entero a o de lín | aleato nea | orio c | | 1 Salt | a | | 100 ea | | | Sa | ilto | de l | + líne |
| le SS Escribir (S Enviar (S Enviar (| Tamaño | eatorio.tx o del arc amaño de | t) " hivo: e arch | Texto | ۹ Salt | entero a o de lín leatoria | aleato nea o.txt) | orio d | | 1 Salt | a o de | e C | 100 ea | | | Sa | ilto | de l | + líne + + |
| e Escribir (Enviar (() Enviar () | Tamañ Tamañ | eatorio.tx o del arc amaño de | t) " hivo:) e arch | Texto | ۹ (و Salt | entero a o de lín leatori | aleato nea o.txt) | orio c 22 | le 🖣 | 1 Salt | a o de | • C | 100 ea | | | Sa | ilto | de I | líne |
| e Escribir (Enviar (() Enviar () Esperar () 5000 | (Tamañ (Tamañ (Tamañ milise | eatorio.tx o del arc amaño de gundos | t) " hivo: e arch | Texto | o (∉ Salt Ƙa | entero a o de lín leatori | aleato nea o.txt) | 27 | | 1 Salt | a o de | • C | 100 ea | | | Sa | ilto | del | líne |

ArduinoBlocks :: Consola serie

| Baudrate: 115200 V | Conectar | Desconectar | Limpiar |
|-----------------------|----------|-------------|---------|
| | | ✓ Enviar | |
| Tamaño dol archivo 14 | | | |
| Tamaño del archivo:21 | | | |
| Tamaño del archivo:28 | | | |
| Tamaño del archivo:35 | | | |
| Tamaño del archivo:42 | | | |
| Tamaño del archivo:49 | | | |
| Tamaño del archivo:56 | | | |
| | | | |
| | | | |

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

| Inicializa | ar | | | | |
|------------|--|--------------------|--------------|----------|------------------------------------|
| | Iniciar Baudios 115200 | | | | |
| | | | | | |
| 5.0 | SD Iniciar | | | | |
| | | | | | |
| | | | | | |
| Espei | erar 500 milisegundos | | | | |
| | ······································ | | | | |
| | | | | | |
| | | | | | |
| Bucle | | | | | |
| + si | si 🔰 🚺 Jatos recibidos? | | | | |
| • 51 | | | | | |
| hacer | er 🛛 Establecer comando 🔨 = 🕻 🔊 Recibir texto 🗸 Hasta salto de línea | | | | |
| | | | | | |
| | + si (annanda - igual a P) ((harrar)) | | | | |
| | | | | | |
| 2 - 1 | hacer Eliminar archive (46 Force and 1) | | | | |
| | | para registrar dat | tos | | |
| | | Escribir | | | |
| | Enviar 🕻 44 *** Registro de datos eliminado *** 🤍 🗸 Salto de línea | | | | |
| | | | | | |
| | + si (, , , , , , , , , , , , , , , , , , | | sensores.csv | | |
| | comando V Igual a V mostrar // | Texto | CSV ; 🗸 | | |
| | hacer have a first and the second sec | | | DUT 22 | Tomporatura (C. z. Din (26 (D2) z. |
| | Enviar Matos registrados: 77 V Salto de línea | | | | Pin 20 (D2) |
| | | | | | |
| | Volcar por puerto serie (" sensores.csv ?? | | | DHT-22 🦯 | K Humedad % Pin 26 (D2) |
| * • | | | | | |
| | Enviar 🕻 🤲 *** Fin datos 🥨 🗸 Salto de línea | | | • | |
| | | ✓ Salto de línea | | | |
| | <u> </u> | | | | |
| Eiecu | utar cada (5000) me | | | | |
| Ejecu | | | | | |
| re | egistrar datos | | | | |
| | · · · · · · · · · · · · · · · · · · · | | | | |
| | <u>-</u> | | | | |
| | | | | | |

Comandos mediante consola serie:

ArduinoBlocks :: Consola serie

| Baudrate: 115200 V | Conectar | Desconectar | Limpiar |
|-------------------------|--------------|-------------|---------|
| borrar | Line feed | - Enviar | |
| *** Registro de datos e | liminado *** | | |
| | | | |
| | | | |

ArduinoBlocks :: Consola serie


Archivo CSV leído en el PC mediante Libre Office Calc:

| 📄 se | nsores.csv - L | ibreOffice Calc |
|--------------|--------------------|--|
| <u>A</u> rch | ivo <u>E</u> ditar | <u>V</u> er <u>I</u> nsertar <u>F</u> ormatc |
| | • 🖻 • 📑 | • 🛃 🚍 🔯 💑 🏾 |
| Libe | eration Sans | ✓ 10 ✓ a a |
| H28 | | - 🐇 Σ = |
| | A | В |
| 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.

| cializar Teclado 👔 Nombre ABlocksKey | · · · · · | | | | | | | | | | | | | |
|---|-------------|-----|------|-----|------|------|------|------|---|-----|------|----|------|----|
| Teclado State Nombre ABlocksKey | de a se a d | | | | | | | | | | | | | |
| | vooaro | | | | | | | | | | | | | |
| | yboard | | | | | | | | | | | | | |
| · · · · · · · · · · · · · · · · · · | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| Ejecutar cada 🔰 5000 ms 🐳 🐇 | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| Teclado 👔 Pulsación 🔹 🗍 | ← Tab • | | | | | | | | | | | i. | | i. |
| | | | + + | * * | | - | ÷ + | + + | - | + + | | + | + + | |
| Teclado 👔 Enviar 🕻 DHT-22 | 2 🚿 | Hun | neda | d % | V) P | in 2 | 6 (D | 2) 🔻 | | s | alto | de | líne | a |
| | - - | | | | | | | | | | | | | |

3.2.2 Simular movimiento y click del ratón

| Inicializar | |
|-----------------|---|
| Ratón 💦 | Nombre ABlocksMouse |
| + + + + + + | · · · · · · · · · · · · · · · · · · · |
| · · · · · · · · | |
| Bucle + + + | |
| + si | Pulsador Pin 14 (D7) v se ha pulsado v Invertir |
| hacer R | atón 🚯 Botón Click 🔹 Izquierda 🔹 |
| Ejecutar ca | ida (100) ms |
| + si | Nivel de luz (LDR) |
| hacer | Ratón 🚯 Mover Y 🛛 📭 |
| sino | Ratón 🚯 Mover Y 🔹 🗾 |
| | |

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 |
|------------|---|
| MY MQTT | bd Image: Constraint of the stages 1909610 Bashboard 1909610 Bashboard 1909610 Bashboard 14 Bashboard 14 Bashboard 14 Bashboard 15 Bashboard 14 Bashboard 15 Bashboard 14 Bashboard 15 Bashboard 14 Bashboard 15 Bashboard 16 Bashboard 17 Bashboard 183 Bashboard |
| | MQTT Dash |
| | Home Image: Second |
| | IoT MQTT Panel |
| | $ \begin{array}{ c c c c c c } \hline & & & & & & & & & & & & & & & & & & $ |

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

Servidor de pruebas: broker.hivemq.com

Mensaje publicado:

<u>Topic</u>:

<u>Valor</u>:

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

| MQTT Broker | |
|---------------------|---------|
| broker.hivemq.com | |
| Port | SSL |
| MQTT V3 | MQTT V5 |
| Credentials | |
| Username (optional) | |
| Password (optional) | |
| | Connect |

| Subscribe | |
|-------------------------------|-----------|
| Торіс | |
| | Subscribe |
| ablocks/juanjo/luz Enabled | |

| broker.hive | mq.com | • |
|-------------|-------------------|-----------------------|
| 969 | ablocks/juanjo/lu | z QoS 0 |
| 966 | ablocks/juanjo/lu | z QoS 0 |
| 945 | ablocks/juanjo/lu | z QaS O |
| 943 | ablocks/juanjo/lu | z QoS 0 |
| 944 | ablocks/juanjo/lu | z QoS O |
| | | ⊳× |
| Dashboard | Subscribe | ≡ , Publish |

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

| Servidor de pruebas: | broker.hivemq.com |
|----------------------|---------------------------------------|
| Mensaje publicado: | |
| <u>Topic</u> : | <u>Valor</u> : |
| ablocks/juanjo/t | Nivel de temperatura del sensor DHT22 |
| ablocks/juanjo/h | Nivel de humedad del sensor DHT22 |

Programa en la placa ESP32 STEAMakers

| cializar Conectar a una red W SSID (arduinoblocks) | iFi | Bu | cle Ejecuta | r cada 🔰 | 5000 | ns | IT 22 | | Tem | peratu | ra °C | | | | · · · |
|--|-------|-------------------|----------------|----------|---------|-------|--------|---------|-------|--------|-------|------|-------|-----|-------|
| Clave 123456blocks | · + + | + + + + + + | Loid | Diecer | | | | J. | , Tem | peratu | | | | (04 | ') |
| Broker broker.hivemq.com | | + + + + + + | MQT | Publica | ar Tema | ۲ " (| ablock | s/juanj | o/t " | Valo | r 🕻 | temp | | | |
| Puerto 1883 | | | | | | | | | | | | | | | |
| Cliente Id AB_esp32juanjo | | | | | | | | | | | | | | | |
| Usuario | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | · · · | + + | |

Utilizaremos la aplicación lot Panel para crea un panel de monitorización

| ← Edit Conne | ction | | |
|---|-------------------------|------|--------|
| Connection name pruebas ablocks | | | 0 |
| Client ID juanjoesp32 | | | 0 |
| Broker Web/IP address broker.hivemq.co | m | | 0 |
| Port number 1883 | Network protocol TCP | * | 0 |
| Add Dashboard | | | + |
| sensores | | 5 | : |
| Additional options | | | \sim |
| | CANCEL | SAVI | |

| ← Add a Line Graph panel | |
|---|-----------|
| Panel name * Historico | |
| Disable dashboard prefix topi | с |
| Topic for graph 1 * ablocks/juanjo/t | |
| Label for graph 1 Temperatura | |
| Factor Graph color | |
| Show plot area | |
| Show points and tooltip | Unit C |
| Enable notification | 0 |
| Payload is JSON Data | |
| Add more graph | + |
| Max persistence 10 | 0 |
| Smooth curve | 0 |



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



| | | ← Add a Text Input panel |
|-------------------------|---|----------------------------------|
| ← Add Dashboard | | Panel name * Mostrar texto: |
| Dashboard name OLED | | Disable dashboard prefix topic 🕜 |
| | | Topic * ablocks/juanjo/oled |
| Dashboard prefix topic | 0 | Clear text on publish |
| Dashboard theme color | • | Payload is JSON Data |
| Choose dashboard icon | | Show sent timestamp |
| Set as connection home | | Confirm before publish |
| Set as connection nome | | 🗌 Retain 🛛 QoS 🛛 👻 |
| CANCEL | | |
| | | i |
| Mostrar texto: Hola! | | |

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 <u>How to Buy</u> 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://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

| # 1 Niciar I2C 0x3 | C 🔽 🗸 Mostrar automáticamente | 111111111111 | http:// | Solicitud | GET 🔹 / | | |
|-------------------------------|---------------------------------------|--------------|-----------------------|------------------|-------------|----------------------------|------------------------|
| Conceptor o uno rod W | | | | Server Enviar | respuesta | | |
| | | | | | | (000) | |
| GID arduinoblocks | | | | | | | |
| ave 123456blocks | | | | Co | oratasida | | |
| Iniciar servidor web | Puerto 80 | | | | Contenido (| Hola, soy tu esp32 web ser | ver, aime io que quier |
| # 1 J Limpiar | · · · · · · · · · · · · · · · · · · · | | · · · · · · · · · · · | | | | |
| | | | | server Solicitud | GET V / er | cender | |
| # 1 Texto X 0 | Y DO C 🚓 Dirección IP 🔹 | Led ON 🔹 sma | | | | | |
| | | | Le | d 📈 | Pin 25 (D3) | Estado ON T | |
| · · · · · · · · · · · · · | | | | | | | |
| | | | | server Enviar | respuesta | | |
| <mark>-</mark> | | | | | our d | 000 | |
| | | | | | | 200 | |
| server Colicitud no oncontrad | | | | 🕎 Co | Intent type | Content type text/plain | |
| | | | | | | Lea encendido 37 | |
| Enviar respuesta | | | | | | | |
| | · · · · · · · · · · · · · · · · · · · | | | server Solicitud | GET 🔹 / ap | agar | |
| Código | 404 | | http:// | 1 | | | |
| Content type | Content type text/plain | | Le | d 🔬 | Pin 25 (D3) | Estado OFF | |
| Contenido (| Esta peticion no es valida! | | | | | | |
| | | | | server Enviar | respuesta | | |
| | | | | | | | |
| | | | | | 0/1 | 222 | |
| | | | | | Código 🕽 | 200 | |

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:



Hola, bienvenido a mi web

Podemos crear contenido HTML básico mediante bloques:

| | and a second second | | | | | | | | | | | | | | | | | | | | |
|---------------|------------------------------------|--------------------------|------------------------|---------------|---------------|------------|------------|-------------|--------------|-------------|-------------|-----------|-----|------|-----|-----|-----|-----|----|------------|--|
| HTML Document | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | |
| Head | HTML Head | | | | | | | | | | | | | | | | | | | | |
| | | | | 4.14 | | | | | | | | | | | | | | | | | |
| | Title | C 66 | Mi w | /eb | ?? • | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | |
| | | A | | | | | | | | | | | | | | | | | | | |
| | Meta | 6 | char | set= | "ISO | -88 | 59- | 1" r | nam | e= | "vie | ewp | oor | t" c | cor | nte | nt= | ="W | /i |) 2 | |
| | Meta Style (css) | | char | set= v {ba | "ISO | -88 วนท | 59- d-c | 1" r olo | nam r: #f | e= | "vie f:} | ewp >> | oor | t" c | cor | nte | nt= | ="W | /i |) * | |
| | Meta Style (css) |) "(| char body | set= y {ba | "ISO | -88 oun | 59- d-c | 1" r olo | nam r: #1 | e= | "vie f;} | ewp | oor | t" c | cor | nte | nt= | ="W | vi |) ? | |
| | Meta Style (css) Script (is) |) ((| char body | set= y {ba | "ISO Ickgr | -88 oun | 59- d-c | 1" r olo | nam r: #1 | e= fffff | "vie [;} | ewp | oor | t" c | cor | nte | nt= | ="W | vi |) ? | |
| | Meta Style (css) Script (js) | ••• ••• ••• | char body | set= y {ba | "ISO | -88 oun | 59- d-c | 1" r olo | nam r: #1 | e= fffff | "vie f;} | ewp | oor | t" c | cor | nte | nt= | •"W | vi |) , | |
| Body | Meta Style (css) Script (js) | C C Io a n | char body ni web | set= y {ba | "ISO | -88 oun | 59- d-c | 1" r olo | nam r: #1 | e= fffff | "vie [;} | ewp | bor | t" c | cor | nte | nt= | •"W | vi |] | |
| Body (| Meta Style (css) Script (js) | C (() C (() Io a n | char body ni web | set= y {ba | "ISO | -88 oun | 59- d-c | 1" r olo | nam r: #1 | e= fffff | "vie f;} | ewp | oor | t" c | cor | nte | nt= | •"W | vi | | |

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



TEMPERATURA

La temperatura actual es de 24.60 grados C

(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> <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:

| server | Solicitud GET / | logo.png | Archivo S | D / web1/logo.png |
|----------|-----------------|----------|-----------|-------------------|
| U | | | | |

Ejemplo completo:



Ejemplo encendido/apagado via web

| Inicializar | | | |
|---------------------------------|-------------------------------|---------------------------------------|---|
| Conectar a una red WiFi | | http:// | olicitud GET · / encender |
| SSID arduinoblocks | | | |
| Clave 123456blocks | | | Pin 26 (D2) Estado ON |
| http:// Iniciar servidor web Pu | ierto 80 | http:// | Enviar respuesta |
| Solicitud GET / | Archivo SD / web2/inicio.html | | Content type Content type text/html 🔹 |
| Selicitud GET / bombi | las ppg | mbillas nng | Archivo SD (Web2/inicio.html) |
| | | | |
| Lco # 1 • Iniciar 2x16 • 12 | C ADDR 0x27 * 🔹 | * * * * * * * * * * * * * * * * | |
| | + + + + + + + + | http:// | olicitud GET 🗸 / apagar |
| LCD # 1 Limpiar | | | |
| LCO # 1 V Imprimir Columna | a O 🗸 Fila O 🗸 🕻 🚒 Dirección | IP 🕤 | Pin 26 (D2) Y Estado OFF Y |
| | | Server | Enviar respuesta |
| | | | |
| | | | Content type 🕻 Content type text/html 🔻 |
| Bucle | | | Archivo SD 🗸 44 web2/inicio.html ?? |
| | | · · · · · · · · · · · · · · · · · · · | |
| | | | |

Código HTML "inicio.html"

<HTML> <HEAD> <TITLE>Web2</TITLE> </HEAD> <BODY> <HR> Web2 - Control encencido/apagado (io26) <HR>

 Encender el led
<
 Encender el led

 Apagar el led </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:

| Inicializar | | | | | | | | | | | | | |
|---|-------------------------------------|--|--|-----------------|--|------------------|-------|---------|--------------|---------------|-----|------|-------|
| | | | | | | | | | | | | | |
| OLED # 1 V Iniciar I2C 0x3C V | 🖌 Mos | strar auto | mática | amer | nte | | | | | | | | |
| | | | | | | | | | | | | | |
| •••••• # 1 . Limpiar | | | | | | 1 | | | | | | | |
| OLED # CIMPIAN | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| Conectar a una red WiFi | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| SSID arduinoblocks | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| Clave 123456blocks | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| Bucle | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| Eiecutar cada 15000 ms | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| Establecer ip publica • = (| http:// | Petición | GET | | | | | | | | | | |
| Establecer (ip publica •) = (| http:// | Petición | GET | | ht | the | //20 | inif | | /2for | mat | -toy | |
| Establecer (ip publica •) = (| http:// | Petición | GET URL |) | f [ht | tps: | //ap | i.ipify | .org | /?for | mat | =tex | t " |
| Establecer (ip publica) = (| http:// | Petición Auth: Us | GET URL uario | | f ht | tps: | //ap | i.ipify | /.org | /?for | mat | =tex | t " |
| Establecer (ip publica) = (| http:// | Petición Auth: Us | GET URL uario | | í (ht | tps: | //ap | i.ipify | /.org | /?for | mat | =tex | t " |
| Establecer (ip publica) = (| http:// | Petición Auth: Us | GET URL uario Clave | | 6 6 6 7 7 8 7 9 7 9 7 9 10 < | tps: | //ap | i.ipify | /.org | /?for | mat | =tex | t) " |
| Establecer ip publica • = (| http:// Basic | Petición Auth: Us | GET URL uario Clave | | ht 0 1 | tps: | //ap | i.ipify | .org | /?for | mat | =tex | t) " |
| Establecer (ip publica v = (| http:// Basic | Petición Auth: Us | GET URL uario Clave | | (ht) () | tps: >> | //ap | i.ipify | /.org | /?for | mat | =tex | t) >> |
| Establecer ip publica · = (| http:// Basic | Petición Auth: Us | GET URL uario Clave | | Int Int Int | tps: | //ap | i.ipify | /.org | /?for | mat | =tex | t) " |
| Establecer (ip publica v = (| http:// Basic | Petición Auth: Us (| GET URL uario Clave | | | tps: " | //ap | i.ipify | /.org | /?for | mat | =tex | t) " |
| Establecer ip publica • = () | http:// Basic | Petición Auth: Us (| GET URL uario Clave | | | tps: " " | //ap | i.ipify | /.org. | /?for nall | mat | =tex | t))) |
| Establecer ip publica • = (| http:// Basic | Petición Auth: Us (| GET URL uario Clave | | (ht) () <li< th=""><th>tps:))))</th><th>()/ap</th><th>i.ipify</th><th>/.org.</th><th>/?for nall</th><th>mat</th><th>=tex</th><th>t) "</th></li<> | tps:)))) | ()/ap | i.ipify | /.org. | /?for nall | mat | =tex | t) " |
| Establecer ip publica v = (OLED # 1 v Limpiar OLED # 1 v Texto X 0 OLED # 1 v Texto X 0 | http:// Basic Y C 0 Y (30 | Petición Auth: Us (((Pi () | GET URL uario Clave ublic IF | | í (ht | tps:)) .ed | ()/ap | i.ipify | /.org sn | /?for nall | mat | =tex | |
| Establecer ip publica v = (OLED # 1 v Limpiar OLED # 1 v Texto X 0 OLED # 1 v Texto X 0 | http:// Basic | Petición Auth: Us (((Petición (| GET URL uario Clave ublic IF | | í (ht | tps:)) .ed | (ON | i.ipify | sn: small | /?for nall | mat | =tex | |

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

| In | icializar 🔸 🔸 👘 | | | | | | | | | | | | | | | | | |
|----|---|--|---|--------------------------------|---------------------------------------|---|----|-----|------|----|-------------|----|-----|-----|---|-----------------|-----------------------------------|--|
| | Conecta | r a una re | d W | iFi | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | SSID arduinobl | ocks | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | Clave 1234560 | IOCKS | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| D. | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | http:// Petició | n GET | | | | | | | | | | | | | | | | |
| | | | 66 1 | http | .//1 | 22 | 16 | 0 1 | n 1 | 12 | | no | or | d | 25 | ,, | | |
| | | URL U | | пцр | :// 13 | 92. | 10 | 0. | J. I | 12 | <i>.</i> /e | nu | er | lae | 21 | | | |
| | | | | | | | | | | | | | | | | | 1 | |
| | BasicAuth: U | Isuario 🖡 | | | | | | | | | | | | | | | - | |
| | BasicAuth: U | Isuario | | | | | | | | | | | | | | | - | |
| | BasicAuth: U | Isuario 🕽 Clave 🏮 | ••• ••• | • " • " | • | | | | | | | | | | | | • | |
| | BasicAuth: U Esperar (500 | Isuario D Clave D 0 milise |) |] "] " dos | | | | | | | | | | | | | * * * * | |
| | BasicAuth: U | Usuario Clave 0 milise |) ۵۰ ۵۰۰ gun |] "] " dos | * * * * | | | | | | | | | | | | * * * * * | |
| | BasicAuth: U Esperar (500 http:// Petició | Isuario D Clave D O milise n GET |) ›› egun |) ") " dos | | | | | | | | | | | | | * * * * * * * * | |
| | BasicAuth: U Esperar (500 http:// Petició | Isuario |) ›› egune | dos | ·//10 | - - - - - - - - - - - - - - - - - - - | 16 | 8.1 | 1 | 12 | | | | | · · · · · | | * * * * * * * * * | |
| | BasicAuth: U Esperar (500 http:// Petició | Isuario Clave 0 milise n GET URL (|) ›› cc (eguno cc (| dos http | ://1 | 92. | 16 | 8.0 | D.1 | 12 | 2/a | pa | iga | ar | · · · · · | | * * * * * * * * * * | |
| | BasicAuth: U Esperar (500 http:// Petició BasicAuth: U | Isuario Clave 0 milise n GET URL 0 Isuario |) ›› egune • • • |) ?? dos http | · · · · · · · · · · · · · · · · · · · | 92. | 16 | 8.0 | 0.1 | 12 | 2/a | pa | iga | ar | · · · · · · · · · · · · · · · · · · · | | * * * * * * * * * * * * | |
| | BasicAuth: U Esperar (500 http:// Petició BasicAuth: U | Isuario Clave O milise n GET URL Isuario Clave | ، ۵۵ egune ، ۵۵ ، ۵۵ ، ۵۵ ، ۵۵ | dos http | ://19 | 92. | 16 | 8.0 | D.1 | 12 | 2/a | pa | iga | ar | · · · · · · · · · · · · · · · · · · · | | * * * * * * * * * * * * * | |
| | BasicAuth: U Esperar (500 http:// Petició BasicAuth: U | Isuario Clave O milise n GET URL Isuario Clave |) >> egune >> >> >> >> |) ?? dos http) ?? | ://1 | 92. | 16 | 8.(| D.1 | 12 | 2/a | pa | ıga | | · · · · · · · · · · · · · · · · · · · | | * * * * * * * * * * * * * | |
| | BasicAuth: U Esperar (500 http:// Petició BasicAuth: U Esperar (500 | Isuario Clave O milise n GET URL Usuario Clave O milise | ۰۰ (egun) ۰۰ (۰۰ (۰۰ (۰۰ (| dos http יי יי dos | ://1 | € 92. | 16 | 8.0 | 0.1 | 12 | 2/a | pa | iga | | · · · · · · · · · · · · · · · · · · · | * * * * * * * * | * * * * * * * * * * * * * * * | |
| | BasicAuth: U Esperar () 500 (http://) Petició BasicAuth: U Esperar () 500 | Isuario Clave O milise n GET URL USUARIO Clave O milise | egune «« « « « « | dos http % | ://1 | 9 2 . | 16 | 8.0 | | 12 | 2/a | pa | iga | | · · · · · · · · · · · · · · · · · · · | | * * * * * * * * * * * * * * * * * | |

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-stea makers-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.

| clienteweb_led | Screen1 • Añadir ventana Eliminar ventana Publish to Gallery | | Diseñador Bloques |
|-----------------------|--|-----------------------|----------------------------|
| Paleta | Visor | Componentes | Propiedades |
| Search Components | Mostrar en el Visor los componentes ocultos | 😑 📃 Screen1 | textoDireccionIP |
| Interfaz de usuario | Tamaño del teléfono (505,320) 🗸 | A Etiqueta1 | ColorDeFondo |
| Disposición | | A Etiqueta2 | Por defecto Habilitado |
| Medios | ☞ 1 9:48 | textoDirectionIP | |
| Dibujo y animación | Screen1 | botonApagar | Negrita |
| Maps | Cliente web - Control led | Web1 | Cursiva |
| Charts | IP de la ESP32: | | Tamaño de letra |
| Sensores | | | 14.0 |
| Social | ENCENDER | | TipoDeLetra |
| Almacenamiento | APAGAR | | por defecto • |
| Conectividad | | | Automático |
| ActivityStarter 🤊 | | | Ancho |
| ClienteBluetooth 🔊 | | | Automático |
| ServidorBluetooth 🛛 🔊 | | | Pista |
| 🥺 Serial 💿 | | | Multilinea |
| Neb 📀 | | Cambiar nombre Borrar | |
| LEGO® MINDSTORMS® | | Medios | SóloNúmeros |
| Experimental | | Subir archivo | ReadOnly |
| Extension | | | Texto |

| 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 V.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/496531508354 7867/

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-comman ds-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 (*jmuy 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: <u>Markdown</u>, 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.

| In | icializar Conectar a una red WiFi SSID [jp Clave [jp Telegram Iniciar API Token 5 | Bucle | |
|----|--|--|----------------|
| | Telegram Nuevo mensaje recibido | | |
| | + si I Telegram Mensaje Texto • Igual a • L • • (/temp) ?? | | |
| | hacer Telegram Enviar a Chat ID | Mensaje I + - crear texto con (" La temperatura actual es de " DHT-11 Temperatura °C • Pin 16 (D5) • (" grados " | Formato None 🔹 |
| | + si 🕻 🖌 Telegram Mensaje Texto 🔹 igual a 🕤 🕻 🕊 /hum " | | |
| | hacer Telegram Enviar a Chat ID | Mensaje (+ - crear texto con (" La humedad actual es del " For DHT-11 Humedad % Pin 16 (D5) " % " | mato None 🔪 |

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:

| | Proyectos Conectar Generar Settings Ayuda Mis View • • • • • • • Proyectos Trash | Guía Informar de un Esp problema • | añol arduinoblocks@gmail.com * |
|---------------------|---|---------------------------------------|--|
| clienteweb_led | Screen 1 • Añadir ventana Eliminar ventana Publish to Gallery | | Diseñador Bloques |
| Paleta | Visor | Componentes | Propiedades |
| Search Components | Mostrar en el Visor los componentes ocultos | 😑 🔲 Screen1 | Screen1 |
| Interfaz de usuario | Tamaño del teléfono (505,320) V | A Etiqueta1 | PantallaAcercaDe |
| Disposición | | A Etiqueta2 | |
| Medios | ☞』 2:48 | botonEncender | AccentColor Por defecto |
| Dibujo y animación | Screen1 | botonApagar | DispHorizontal |
| Maps | Cliente web - Control led | 🔵 Web1 | Centro : 3 • |
| Charts | IP de la ESP32: | | Arriba : 1 • |
| Sensores | 192.168.0.100 | | AppName |
| Social | ENCENDER | | clienteweb_led |
| Almacenamiento | APAGAR | | ColorDeFondo |
| Conectividad | | | ImagenDeFondo |
| ActivityStarter 💿 | | | Ninguno |
| ClienteBluetooth 💿 | | | BigDefaultText |
| ServidorBluetooth 🕝 | | 2 | BlocksToolkit |
| 🥺 Serial 🕝 | | | |
| 🜒 Web 💿 | | Cambiar nombre Borrar | AnimaciónCierreDePantalla Por defecto 🔹 |
| LEGO® MINDSTORMS® | | Medios | DefaultFileScope |
| Experimental | | Subir archivo | App • |
| Extension | | | |
| | | | Icono |
| cuando | botonEncender V. Clic |
|----------|---|
| ejecutar | poner Web1 🔻 . Url 🔻 como Ӷ 😂 unir () " (http://) " |
| | C textoDireccionIP v . Texto v |
| | (/encender) " |
| | Ilamar Web1 . Obtener |
| | |
| | |
| cuando | botonApagar V. Clic |
| ejecutar | poner Web1 🔻 . Url 🔻 como 🚺 😂 unir 🕻 " (http://) " |
| | textoDireccionIP • . Texto • |
| | (apagar " |
| | Ilamar Web1 V. Obtener |

Servidor web en ESP32 STEAMakers + led:



Proyectos Conectar Generar Settings Español arduinoblocks@gmail.com Ayuda Mis proyectos Informar de un problema View Trash Guía Screen1 • Añadir ventana Eliminar ventana Publish to Gallery Diseñador Bloques clienteweb_servo Visor Paleta Componentes Propiedades Mostrar en el Visor los componentes ocultos Screen1 Deslizador1 Tamaño del teléfono (505,320) 🗸 Interfaz de usuario A Etiqueta1 ColorIzquierda Por defecto A Etiqueta2 Botón ? ColorDerecha T textoDireccionIP Por defecto CasillaDeVerificación ? Deslizador1 9:48 📓 🧟 Ancho SWeb1 SelectorDeFecha ? en1 200 pixels.. **Cliente web - Control servo** 🌌 Imagen ? ValorMáximo IP de la ESP32: 180 A Etiqueta (?) 192.168.0.100 ValorMínimo E SelectorDeLista ? 0 VisorDeLista ? ThumbEnabled A Notificador ? \checkmark PosiciónDelPulgar CampoDeContraseña ? 90 Deslizador Visible Desplegable ? Switch ? CampoDeTexto (?) Cambiar nombre Borrar 2 SelectorDeHora ? Medios VisorWeb ? Subir archivo... \triangleleft 0 Disposición

6.1.2.-Aplicación appinventor para mover servomotor

Programa de control del servo y servidor web en ESP32 STEAMakers

| Inicializar |
|---|
| Conectar a una red WiFi |
| |
| SSID arduinoblocks |
| Clave 123456blocks |
| http:// Iniciar servidor web Puerto 80 |
| |
| Servo 🔉 Pin 25 (D3) 🗸 Grados 🕻 Ángulo 0° Retardo (ms) 🚺 0 |
| |
| |
| |
| Server Solicitud CET - / Sonro |
| |
| Establecer pos servo = Valor del parámetro (Número) |
| Servo 🎽 Pin 25 (D3) 🗸 Grados (pos servo 🗸 Retardo (ms) |
| |

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

| WIFI Enviar a todos Mensaje ('temperatura ' Data | 5 C DHT-11 Temperatura °C 🔹 Pin (25 (D3) 🔹 |
|---|--|
|---|--|

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

| Inicializar • • • • • • • • • • • | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---------------------------------------|----|---|---------------------------------------|-----|------|------|----------------------------|-------|----|------|---|-------------|------|-----|-----|-----|-----|-----|--------------|---|---|-----|-----|---|-----|-----|---|--|
| WIFI Crear o unirse a una red | Ľ | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Nombre ABLOCKS | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 01 10045070 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Clave 12345678 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Puerto 5555 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| • • • • • • • • • • • • • • • • • • • | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bucie | | | | | | | | | N | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| B WiFi Enviar a todos Mensaj | je | 4 | " [| ale | eato | orio |) " | , | Da | atos | (| ſ | ente | ero | ale | eat | ori | o d | le | ٢ | 0 |] a | a (| 1 | 00 | | | |
| | 4 | | 0.0 | 4 | 4 | 0.0 | | - | | 0 0 | 0 | | 4 | 0 0 | | | | | | • | | | | | 4 | 0 0 | - | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Ejecutar cada 5000 ms | je | | · · · · · · · · · · · · · · · · · · · | ale | eato | orio |)))))) | + + + | Da | atos | - - - - - - - - - - - - - - - - - - - | • • • | ente | ero | ale | eat | ori | o d | * * le | | 0 | | a (| | 100 | | | |

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

| icializar • • • • • • • • • • • • | | | | | | | | | | | | | | | | | | | | | | | |
|--|--------------|---|-----|-----|-----|-----|------|-----|-----|-----|----|---|---|----|---|---|---|---|---|----|------|---|---|
| Sear o unirse a una red | | | | | | | | | | | | | | | | | | | | | | | |
| Nombre ABLOCKS | | | | | | | | | | | | | | | | | | | | | | | |
| Clave 12345678 | | | | | | | | | | | | | | | | | | | | | | | |
| Puerto 5555 | | | | | | | | | | | | | | | | | | | | | | | |
| # 1 Iniciar I2C 0x3C V | \checkmark | M | ost | rar | au | ton | náti | cai | me | nte | ÷ | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | |
| • • • • • • • • • • • • • • • • • • • | | | | | | | | | | | | | | | | | | | | | | | |
| ucle | | | | | | | | | | | | | | | | | | | | | | | |
| ······································ | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | |
| WIFL Mensaje recibido: aleatorio | | | | | | | | | | | | | | | | | | | | | | | |
| nien # 1 Limpiar | + + | | | | | | | | | | | | | | | | | | | | | | |
| | | | 1 | | | - | ÷. | | | - | - | | 1 | + | | - | + | - | - | 1 | | | + |
| oled # 1 Texto X 0 Y | 0 | C | 1 | | WiF | i I | Dat | os | rec | ibi | do | s | | Le | d | O | N | • | m | ed | liur | n | ¥ |

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/3blynk

Bibliografía y enlaces de interés:

http://www.arduinoblocks.com

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

https://www.youtube.com/watch?list=PL1pKD-Bz2QBAgfy580m8OaQ2Z60v6DOhC&v=MQjIEI7I4ik&f eature=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 loT con ESP32 STEAMakers



CAMPUS D'ALCOI

Juanjo López ArduinoBlocks.com arduinoblocks@gmail.com