



Instituto Tecnológico Nacional de México
Instituto Tecnológico de Colima



Materia: APPS PARA EL INTERNET DE LAS COSAS

Docente: ANGEL VEJAR CORTES

Carrera: Ingeniería en informática

Equipo:

Daniel Ezequiel Alvarado Almánzar

Christian Rafael Legorreta Avalos

José Salvador Orozco Gutiérrez

Rogelio Valdez Macías

Diego Rodolfo Diaz Morfin

Jueves 5 de noviembre del 2023

INDICE

| | |
|--|----|
| Introducción | 3 |
| Desarrollo..... | 4 |
| Código: | 4 |
| Configuración inicial | 4 |
| Configuración del pin del LED | 5 |
| Funcionamiento: | 5 |
| Lectura de datos al presionar los botones y lectura de humedad | 6 |
| Funcionando:..... | 7 |
| Conclusión | 15 |

Introducción

La introducción a este desarrollo del código nos adentra en el mundo de la programación y configuración de un dispositivo IoT basado en un ESP32, que utiliza el servicio de Arduino Cloud para comunicarse y controlar sus funcionalidades. En esta descripción, se han presentado los elementos esenciales del código y su funcionamiento. Desde la inclusión de bibliotecas y configuraciones iniciales hasta el proceso de lectura de datos de sensores y la interacción con botones. Este código permite al dispositivo conectarse a la plataforma en la nube y actualizar sus propiedades de manera remota.

Además, se proporciona una visión general de la interfaz de Arduino Cloud, donde se pueden configurar y gestionar las propiedades del dispositivo, así como establecer conexiones con la red Wi-Fi y obtener claves secretas para la comunicación. La narración destaca la importancia de la vinculación de variables con componentes de la interfaz para el control adecuado del dispositivo.

En el conjunto de instrucciones, se desglosan los pasos necesarios para configurar un proyecto en Arduino Cloud, desde la creación del dispositivo hasta la adición de variables y su vinculación con el código del ESP32. Esta descripción ofrece una guía inicial para comprender el proceso de desarrollo y configuración de dispositivos IoT con Arduino Cloud.

Desarrollo

Código:

```
#include "thingProperties.h"
#include <DHT.h>
// Pin del sensor DHT11
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht (DHTPIN, DHTTYPE);
```

Aquí se incluye la biblioteca "thingProperties.h", que generalmente es generada por Arduino Cloud para describir las propiedades del dispositivo, También se incluye la biblioteca "DHT" para trabajar con un sensor DHT11 y se define el pin (DHTPIN) al que está conectado el sensor DHT11 y el tipo de sensor (DHTTYPE) como DHT11. Luego, se crea un objeto "DHT" llamado "dht" usando estos valores.

Configuración inicial

```
void setup () {
  Serial.begin(9600);
  delay(1500);
  initProperties();
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}
```

En esta parte del código nos encargamos de inicializar la comunicación serial, configurar las propiedades del dispositivo, establecer la conexión con Arduino Cloud y configurar la información de depuración para el monitoreo y diagnóstico. Es una parte esencial para que el dispositivo pueda funcionar y comunicarse correctamente con la plataforma en la nube.

Configuración del pin del LED

```
// Configuración del pin del LED (D21 en un ESP32)
pinMode (21, OUTPUT);

// Configuración de los pines de los botones del contador
pinMode (4, INPUT); // Botón Aumentar (D4)
pinMode (5, INPUT); // Botón Disminuir (D5)

// Inicialización del sensor DHT11
dht.begin ();
}
```

Con esta parte del código se leen los estados actuales de los botones aumentar y disminuir, los cuales están conectados como el código lo especifica en el pin4 y pin5. Además de la inicialización del sensor DHT11

Funcionamiento:

```
void loop () {
  ArduinoCloud.update();
```

Arduinicloud.update es el que se encarga de actualizar y mantener la comunicación entre los dispositivos de la plataforma Arduino cloud. Sin esto no podríamos recibir y enviar datos desde la nube de Arduino, lo cual nos permite el monitoreo y control remoto con nuestro dispositivo sp32.

```
// Leer el estado actual de los botones
bool buttonAumentarState = digitalRead (4);
bool buttonDisminuirState = digitalRead (5);
```

Con estas 2 líneas de Código nos permite mandar la información de si los botones están presionados o no a los pines correspondientes de nuestro Arduino, cada bool que tenemos es para ya sea aumentarlo o disminuir.

Lectura de datos al presionar los botones y lectura de humedad

La función loop () es el núcleo del programa y se ejecuta continuamente

```
void loop () {  
  ArduinoCloud.update();  
  // Leer el estado actual de los botones  
  bool buttonAumentarState = digitalRead (4);  
  bool buttonDisminuirState = digitalRead (5);
```

Se leen los estados actuales de los botones Aumentar y Disminuir y se almacenan en las variables "buttonAumentarState" y "buttonDisminuirState", respectivamente.

```
  // Actualizar el estado del LED (D21)  
  digitalWrite (21, led);  
  // Leer la temperatura y humedad del sensor DHT11  
  float temperature = dht.readTemperature ();  
  float humidity = dht.readHumidity();  
  // Incrementar el contador si se presiona el botón de aumento (D4)  
  if (buttonAumentarState == HIGH && lastButtonAumentarState == LOW) {  
    contador += 1;  
  }
```

Se lee la temperatura y humedad del sensor DHT11, y estos valores se almacenan en las variables "temperature" y "humidity". Si la temperatura es válida (no es un valor indefinido "NaN" y está dentro de un rango específico), se actualizan las variables "temperatura" y "humedad".

```
  // Decrementar el contador si se presiona el botón de disminuir (D5)  
  if (buttonDisminuirState == HIGH && lastButtonDisminuirState == LOW) {  
    contador -= 1;  
  }
```

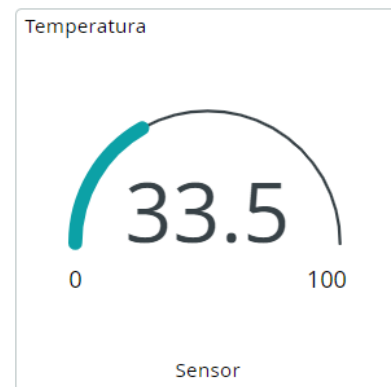
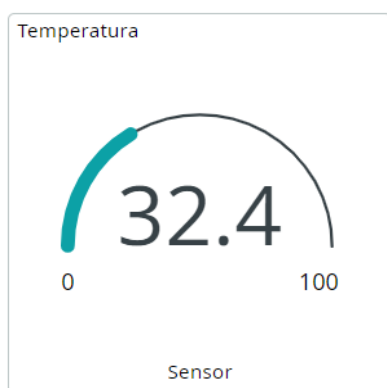
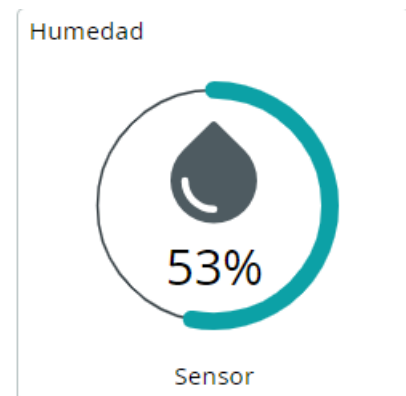
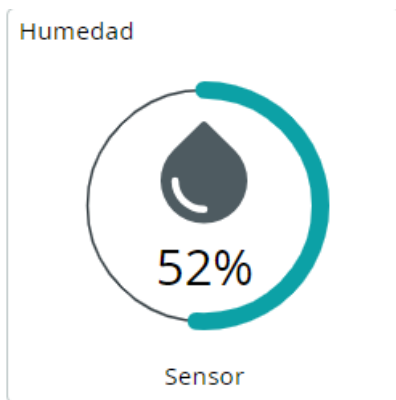
```
  // Actualizar el estado anterior de los botones  
  lastButtonAumentarState = buttonAumentarState;  
  lastButtonDisminuirState = buttonDisminuirState;
```



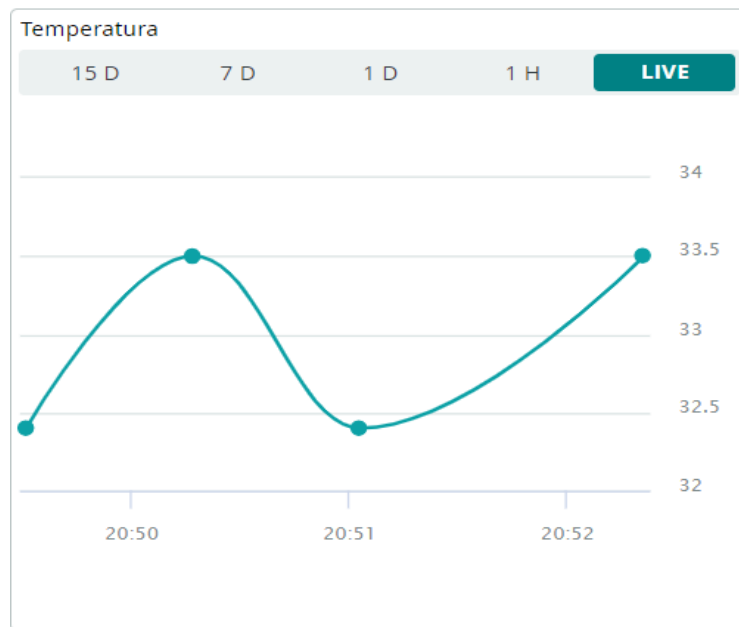
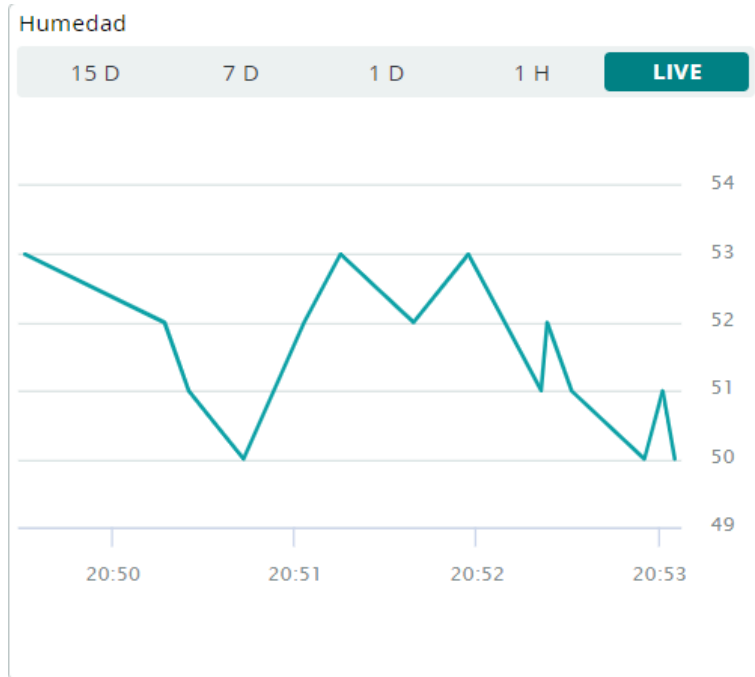
Además, el código incluye funciones como `onLedChange()`, `onContadorChange()`, `onTemperaturaChange()`, y `onHumedadChange()`, que están destinadas a manejar cambios en las propiedades del dispositivo. Estas funciones deben personalizarse para realizar acciones específicas cuando estas propiedades cambian.

Funcionando:

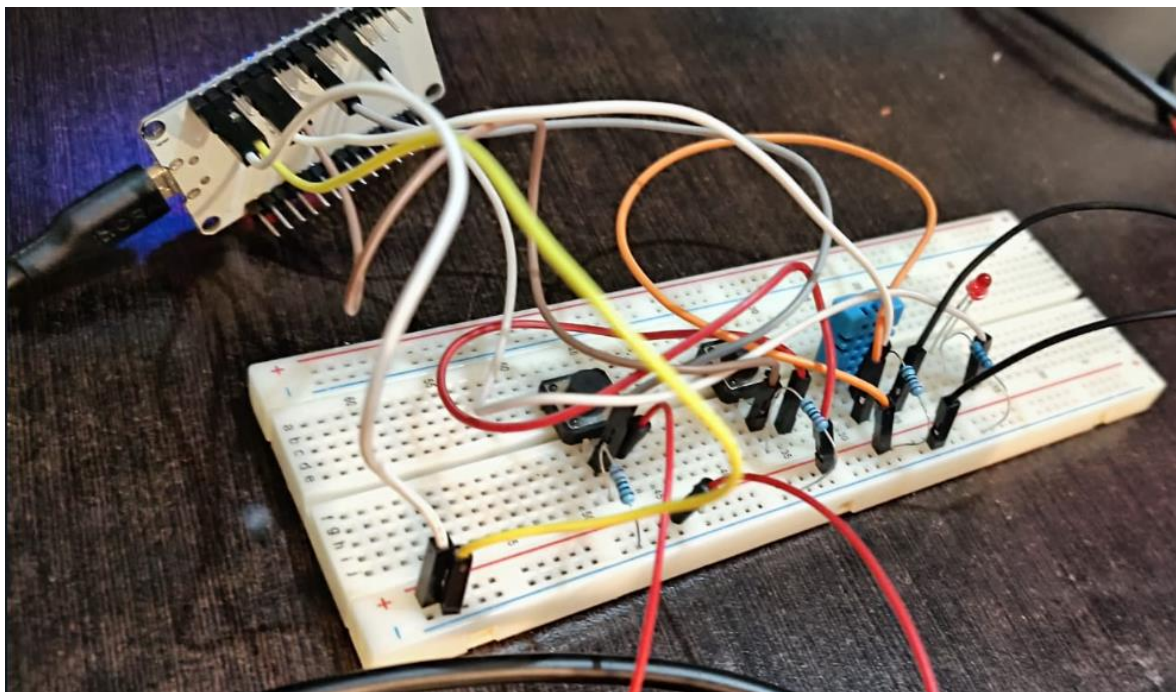
Aquí se ve el funcionamiento del sp32 en la página de Arduino cloud del como si se mandan los datos por medio del sensor de humedad del antes y el después al igual que la temperatura ambiente que detecta.



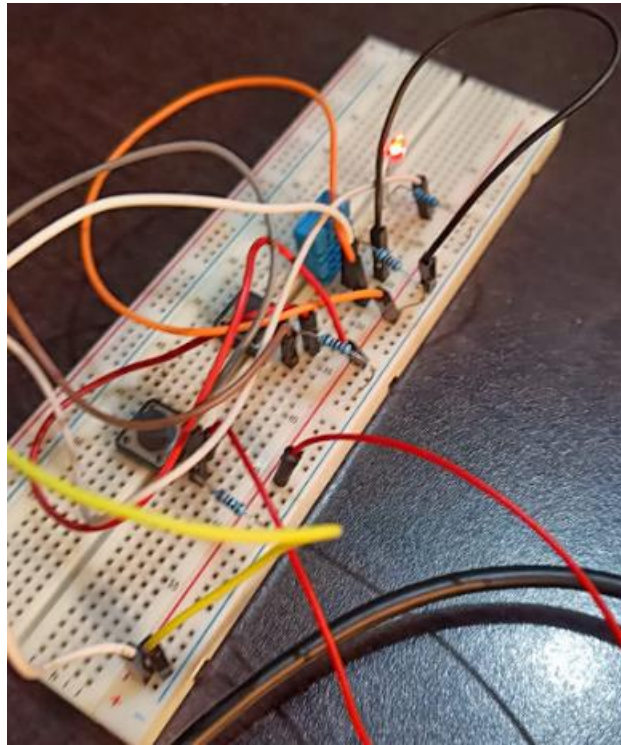
Aquí se muestran los mismos datos de arriba, pero de forma grafica



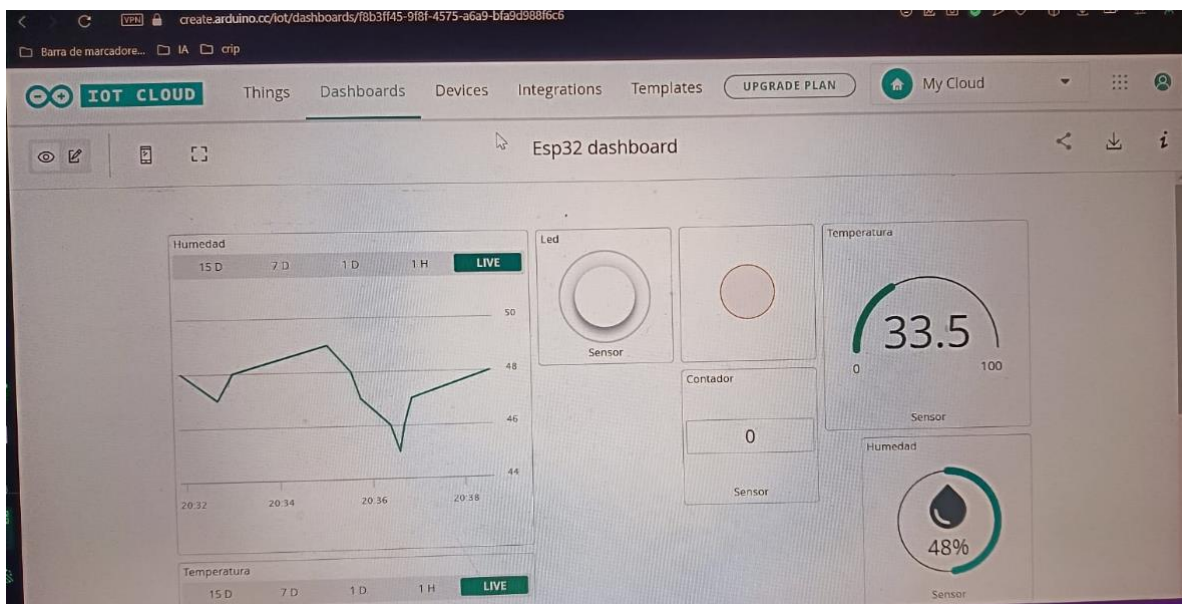
Led Apagado



Led Encendido



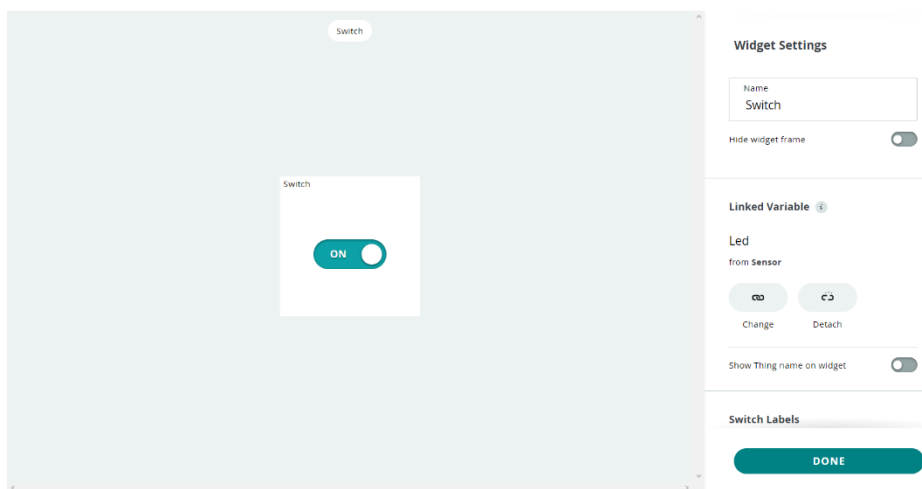
Todo Junto funcionando



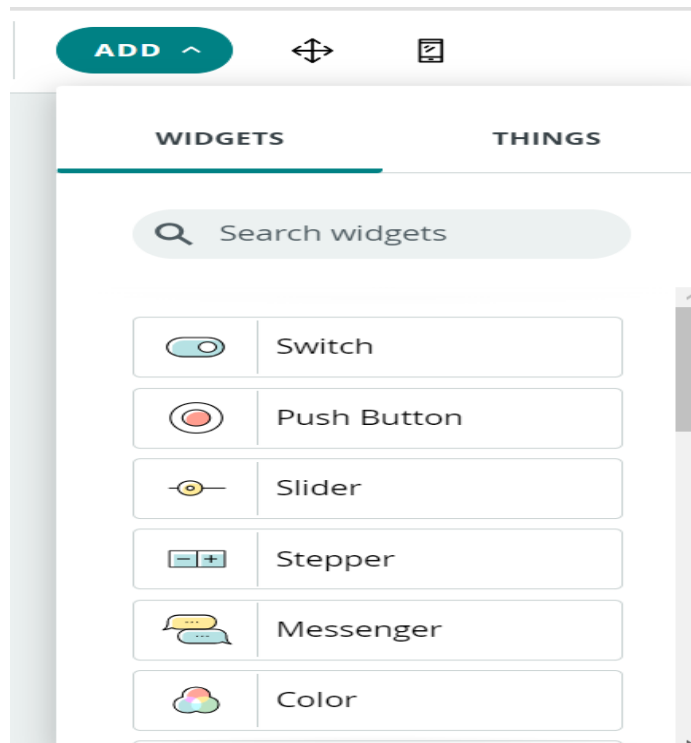
También explicando un poco el apartado de Arduino cloud tenemos lo siguiente

Estos son los botones o dependiendo lo que necesitemos para que se pueda utilizar desde el sp32

En este caso es un botón switch para encender o apagar aquí también se alcanza a observar varias opciones que tenemos al momento de colocarlo lo cual al darle change es para cambiarlo y el link es para enlazarlo con nuestra variable led que es en este caso para prender o apagarlo



Tenemos muchas opciones al presionar el botón ADD es para agregar lo que comentábamos anteriormente colocar un botón, las graficas o los encargados de ver la temperatura de humedad o temperatura



Esta parte es para una vez puesto lo anterior que necesitas se link con la variable que creamos y mas adelante explicare donde es, retomando lo de la variable y el link esto es para que donde tenemos nuestro Código la parte encargada de controlar el led funcione si no hacemos esto no funcionara.

| | | | | | | | | | | | | | | |
|---|--|--|-------|--------|------|---------|------------|------|------------|------------|---------------|--|-------------|----------------------|
| Things Sensor Goldarina - ESP32 Dev Module Untitled No associated device | Variables Contador Int Humedad Relative Humidity Led Boolean Temperatura Temperature Sensor (°C) | Led <table><tr><td>Thing</td><td>Sensor</td></tr><tr><td>Type</td><td>Boolean</td></tr><tr><td>Last Value</td><td>true</td></tr><tr><td>Permission</td><td>Read/Write</td></tr><tr><td>Update Policy</td><td></td></tr><tr><td>Last Update</td><td>24 Oct 2023 21:03:03</td></tr></table> LINK VARIABLE | Thing | Sensor | Type | Boolean | Last Value | true | Permission | Read/Write | Update Policy | | Last Update | 24 Oct 2023 21:03:03 |
| Thing | Sensor | | | | | | | | | | | | | |
| Type | Boolean | | | | | | | | | | | | | |
| Last Value | true | | | | | | | | | | | | | |
| Permission | Read/Write | | | | | | | | | | | | | |
| Update Policy | | | | | | | | | | | | | | |
| Last Update | 24 Oct 2023 21:03:03 | | | | | | | | | | | | | |

En el apartado de Dashboard es donde creamos nuestro Arduino por así decirlo es donde lo configuramos y agregaremos el Código, además de la secret key y red wifi para que funcione

[Things](#) [Dashboards](#) [Devices](#) [Integrations](#) [Templates](#) [UPGRADE PLAN](#) [My Cloud](#)


Dashboards [CREATE](#)


| Name ↓ | Last modified | Owned by |
|---|----------------------|----------|
| <input type="checkbox"/> Esp32 dashboard | 24 Oct 2023 20:54:50 | diaz12 |
| <input type="checkbox"/> Untitled | 24 Oct 2023 20:56:52 | diaz12 |

Creamos uno nuevo en donde dice create y nos aparecerá lo siguiente donde seleccionamos el que está a la derecha


Setup Device

RECOMMENDED **AUTOMATIC** Auto-sketch generation, online editing, easy upload


Arduino board ⓘ
</> Arduino language (C++)


Third party device ⓘ
</> Arduino language (C++)

MANUAL Manual programming and upload, offline editing


Any Device ⓘ
</> Python, MicroPython, JavaScript (NodeJS)

Paso seguido saldrá lo siguiente, seleccionamos Esp 32 y continuamos nos dará la llave secreta y un Código que tenemos que guardar bien porque lo pide en la configuración de wifi para que funcione todo en orden

← Setup Device X

Select device type

Please select the device type and model you want to configure

☐ ESP8266 ☒ ESP32 ☐ LoRaWAN

ESP32 Dev Module

CONTINUE

Setup Device X

make your device ready

To use this board you will need a Device ID and a Secret Key, please copy and save them or [download the PDF](#).

Also, keep in mind that this device authentication has a lower security level compared to other Arduino devices.

Device ID
84c07436-141c-49eb-8522-bd57ae658ef0

Secret Key
HJXW4MGZII2EZAS5YQSA8

⚠ Secret key cannot be recovered
Please keep it safe, if you lose it you will have to delete and setup your device again.

☐ I saved my device ID and Secret Key

CONTINUE

Una vez que lo tengamos creado nos aparecerá con el nombre que le hallamos llamado y agregamos con ADD las variables que utilizaremos

LOUD

Things Dashboards Devices Integrations Templates

UPGRADE PLAN

My Cloud

Sensor

Setup Sketch Metadata

Cloud Variables

ADD

| | Name ↓ | Last Value | Last Update | |
|--------------------------|---|------------|----------------------|---|
| <input type="checkbox"/> | Contador <code>int contador;</code> | 0 | 24 Oct 2023 20:49:31 | ⋮ |
| <input type="checkbox"/> | Humedad <code>CloudRelativeHumidity humedad;</code> | 52 | 24 Oct 2023 21:00:25 | ⋮ |
| <input type="checkbox"/> | Led <code>bool led;</code> | false | 24 Oct 2023 20:54:49 | ⋮ |
| <input type="checkbox"/> | Temperatura <code>CloudTemperatureSensor temperatura;</code> | 31.3 | 24 Oct 2023 21:00:27 | ⋮ |

Associated Device

Goldarina

ID: c880c665-8896-4279-908f-5...

Type: ESP32 Dev Module

Status: ☒ Offline

Change

Detach

Network

Configuramos en el botón Change los datos y listo

Network

Wi-Fi Name: INFINIT...

Password:

Secret Key:



Change

Conclusión

José Salvador Orosco Gutiérrez

La realización de esta actividad me ha permitido adquirir una comprensión más profunda del Internet de las cosas, proporcionándonos una alternativa distinta a la creación de nuestro propio servidor y página web. Utilizando Arduino Cloud, pudimos prescindir fácilmente del servidor y la página web, y en su lugar aprovechar el panel de control que Arduino Cloud ofrece de forma gratuita. Sin embargo, es importante destacar que esta ventaja también puede convertirse en una desventaja en el futuro si el proyecto requiere escalabilidad, la incorporación de más sensores o una mayor robustez, ya que su desarrollo se torna más complejo. Además, en el caso de proyectos más extensos, sería necesario adquirir una membresía, dado que la versión gratuita se queda corta.

Personalmente, esta práctica ha representado un valioso aprendizaje. A pesar de algunos obstáculos que enfrentamos durante el proceso, como la configuración del ESP32 para conectarlo a la nube, logré dominar el uso de Arduino Cloud.

Diego Rodolfo Diaz Morfin

El código con Arduino Cloud es más fácil de desarrollar, ya que Arduino Cloud proporciona una plataforma simplificada para la comunicación entre el dispositivo y la nube. No necesitas lidiar con detalles de HTTP ni configurar un servidor web.

Por otra parte, la implementación es más sencilla, ya que no es necesario configurar ni mantener un servidor web adicional. Arduino Cloud se encarga de la comunicación entre el dispositivo y la nube.

Para terminar, podemos decir qué es menos complejo en términos de comunicación y configuración del servidor, ya que Arduino Cloud se encarga de gran parte de la infraestructura. Sin embargo, la funcionalidad puede ser más limitada en comparación con una implementación personalizada.

Rogelio Valdez Macias

En conclusión, la experiencia de trabajar con Arduino Cloud para implementar el Internet de las cosas ha sido enriquecedora y ha proporcionado una valiosa perspectiva sobre las ventajas y desventajas de esta plataforma. A través de esta actividad, se ha demostrado que Arduino Cloud ofrece una solución accesible y fácil de usar para proyectos IoT, eliminando la necesidad de crear un servidor y una página web desde cero. Esto ha permitido una comprensión más profunda de la tecnología.

En lo personal, esta experiencia ha sido un valioso aprendizaje, ya que ha implicado superar obstáculos y dominar el uso de una plataforma importante en el mundo del IoT. En última instancia, esta actividad ha contribuido a mi crecimiento y conocimiento en el campo de la tecnología y la conectividad.

Daniel Ezequiel Alvarado Almanza

Lo que aprendí de este proyecto es que ahora en vez de usar un servidor local donde otra persona del equipo tiene que estar conectada, usamos un servidor web que es el ArduinoCloud que es mucho más fácil de manejar y que resulta más económico en casos de que el proyecto no requiera de tanta escalabilidad, aparte de que se adapta muy bien a cualquier tipo de proyecto ya que cuenta con un diseño adaptable para que se vea presentable, aparte de que nos ahorramos algo de funciones en código, por ejemplo: con el Arduino se inicia la comunicación con la plataforma a través de la función "ArduinoCloud.begin" y se establece el nivel de mensajes de depuración lo cual esto permite enviar datos del dispositivo a la nube de Arduino para su monitoreo y control, por lo que no requiere de más agregados ya que el servidor lo hace por el, por lo que su complejidad es menor a comparación de lo que se inició inicialmente, donde dependíamos de que la conexión con la otra persona "Servidor" no se perdiera y que fallara el proyecto.

Christian Rafael Legorreta Avalos

En conclusión, esta actividad implicó el empleo de un ESP32, un sensor de humedad y temperatura, así como dos botones para controlar el contador y transmitir estas mediciones a la nube a través de Arduino Cloud. Esta experiencia nos permitió aplicar conceptos de IoT en una aplicación práctica en la que gestionamos los datos medidos así como las pulsaciones del contador en el ámbito del Internet de las cosas.