

Arduino IDE

- [Software: Arduino IDE](#)
- [Programas Arduino IDE sin IoT](#)
- [Escaneo Wifi](#)
- [Arduino Cloud](#)
- [Coche teledirigido](#)
- [ESP32 + Sensores externos + IoT](#)

Software: Arduino IDE

Aunque Arduino ALVIK está diseñado para utilizar con Micropython, se puede utilizar el Aruido IDE

OJO, TEN EN CUENTA QUE TE CARGAS EL COMPILADOR, SI QUIERES VOLVER A PROGRAMAR CON MICROPYTHON TIENES QUE VOLVERLO A CARGAR Mira <https://libros.catedu.es/books/arduino-alvik/page/instalar-micropython>

COMENZAMOS CON ARDUINO IDE

Necesitarás el **entorno de desarrollo Arduino IDE** (IDE, Integrated development environment) (aquí <https://www.arduino.cc/en/Main/Software> para descargártelo)

En Linux puede salir este mensaje "can't open device "/dev/ttyUSB0": Permission denied" donde 0 puede ser otro número, la solución [aquí](#)

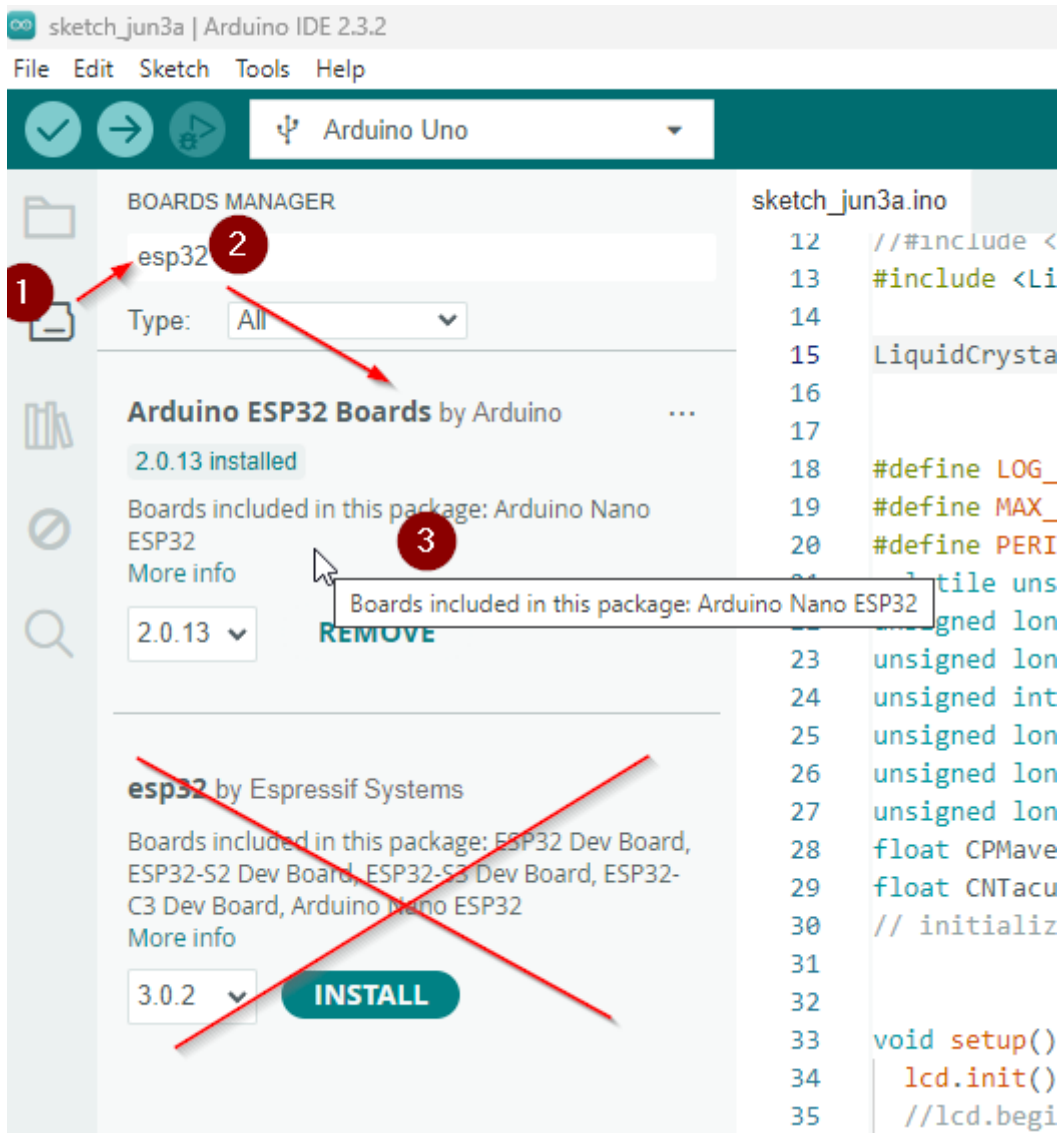
Está constituido por un **editor de texto** para escribir el código, un **área de mensajes**, una barra de herramientas con botones para las funciones comunes, y una serie de menús.

Arduino utiliza para escribir el código fuente o programa de aplicación lo que denomina "sketch" (programa). Estos programas son escritos en el editor de texto. Existe la posibilidad de cortar/pegar y buscar/remplazar texto.



Board manager: Arduino ESP32 Boards by Arduino

Lo primero que tenemos que hacer es instalar la placa Arduino ESP32 tal y como dice esta captura



O este vídeo a partir de 9:30 (pongo el vídeo pues es interesante si quieres aprender más sobre Arduino ESP32)

Programas Arduino IDE sin IoT

En la pagina https://www.arduinolibraries.info/libraries/arduino_alvik o desde https://github.com/arduino-libraries/Arduino_Alvik podemos descargarnos multitud de ejemplos de código escrito en Arduino IDE para manejar este robot

Librería Arduino_Alvik.h

Las funciones que tiene la librería son prácticamente las vistas en las APIs, ver <https://libros.catedu.es/books/arduino-alvik/page/arduino-alvik-api>

Para ejecutarlo en el Arduino IDE tenemos que tener esta librería que es fácilmente instalable:

drive | Arduino IDE 2.3.2

File Edit Sketch Tools Help

Arduino Nano ESP32

LIBRARY MANAGER

Arduino_Alvik.h

Type: All

Topic: All

Arduino_Alvik by Arduino, Giovanni di Dio... 1.0.1 installed

Library to code Arduino Alvik robot This library is used to code Arduino Alvik, examples show y... More info

1.0.1 REMOVE

Arduino_AlvikCarrier by Arduino, Giovanni di Dio... 1.0.1 INSTALL

drive.ino

```
1  /*
2  2  T
3  3
4  4  C
5  5
6  6  T
7  7  L
8  8  f
9  9
10 */
11
12 #incl
13
14 Ardui
15
16 void·
17 ··alv
18 }
19
20 void·
21 ··alv
22 ··del
23 ··alv
24 ··del
25 }
```

Ejemplo Drive

Este sencillo programa hace mover el robot a una velocidad de 10 y va cambiando el giro de 45º a -45º cada segundo

```
#include "Arduino_Alvik.h"
```

```
Arduino_Alvik alvik;
```

```
void setup() {
```

```
    alvik.begin();  
}  
  
void loop() {  
    alvik.drive(10, 45);  
    delay(10000);  
    alvik.drive(10, -45);  
    delay(10000);  
}
```

y da este error NO DEU dfu-util: No DFU capable USB device available Failed uploading: uploading error: exist status 74 ¿Por qué?

Lee <https://libros.catedu.es/books/arduino-alvik/page/modo-bootloader>

Resultado

<https://www.youtube.com/embed/1zG26a3Lf1g>

Experimenta si tienes dos Arduino Alviks Con https://github.com/arduino-libraries/Arduino_Alvik/blob/main/examples/remote_control/remote_control.ino

Escaneo Wifi

Desde [https://github.com/espressif/arduino-](https://github.com/espressif/arduino-esp32/blob/master/libraries/WiFi/examples/WiFiScan/WiFiScan.ino)

[esp32/blob/master/libraries/WiFi/examples/WiFiScan/WiFiScan.ino](https://github.com/espressif/arduino-esp32/blob/master/libraries/WiFi/examples/WiFiScan/WiFiScan.ino) podemos encontrar este programa para escanear las redes wifi desde nuestro ESP32 Arduino

<https://app.arduino.cc/sketches/54b6f875-2961-4ec5-8a48-608d9dde5feb?view-mode=preview>

<https://app.arduino.cc/sketches/54b6f875-2961-4ec5-8a48-608d9dde5feb?view-mode=embed>

Instalando la librería Wifi.h

Te dará un error de compilación pues no tiene esta librería. Puedes descargar la versión última desde <https://www.arduino.cc/reference/en/libraries/wifi/>

FUNCTIONS

VARIABLES

STRUCTURE

LIBRARIES

IOT CLOUD API

GLOSSARY

WiFi

Communication

Enables network connection (local and Internet) using the Arduino WiFi shield. With this library you can instantiate Servers, Clients and send/receive UDP packets through WiFi. The shield can connect either to open or encrypted networks (WEP, W The IP address can be assigned statically or through a DHCP. The library can also m. DNS.

[Go to repository](#)

Note: this library was retired and is no longer maintained.

Compatibility

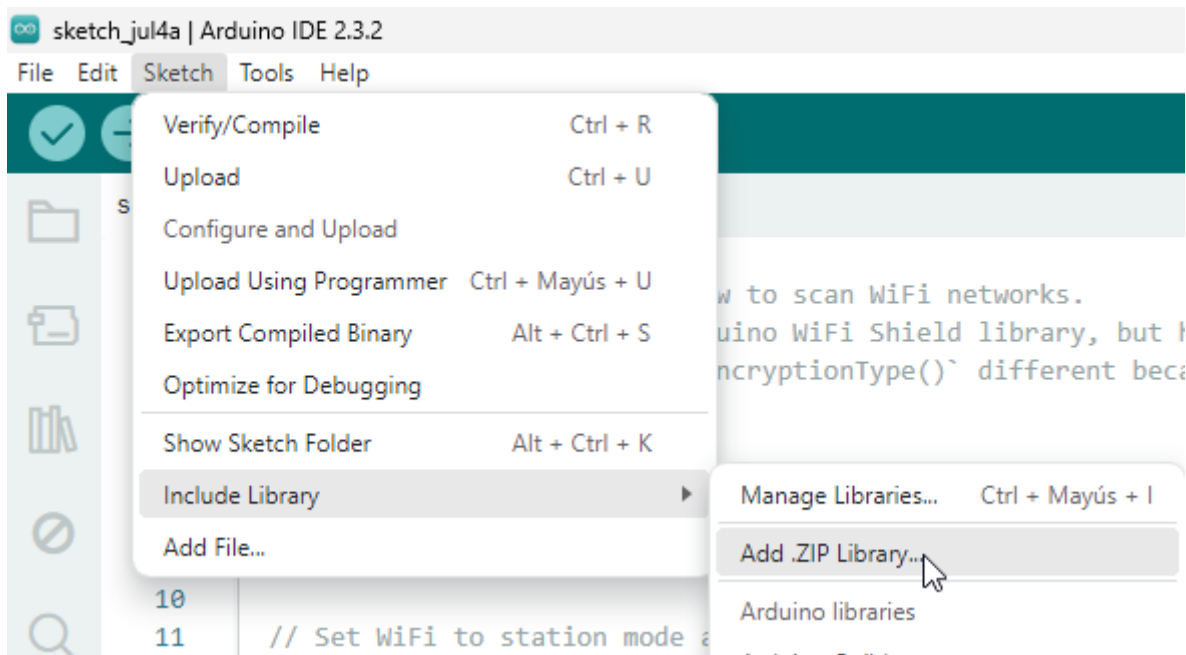
This library is compatible with **all** architectures so you should be able to use it on al Arduino boards.

Releases

To use this library, open the [Library Manager](#) in the Arduino IDE and install it from t

- 1.2.7 (latest)
- 1.2.6
- 1.2.5

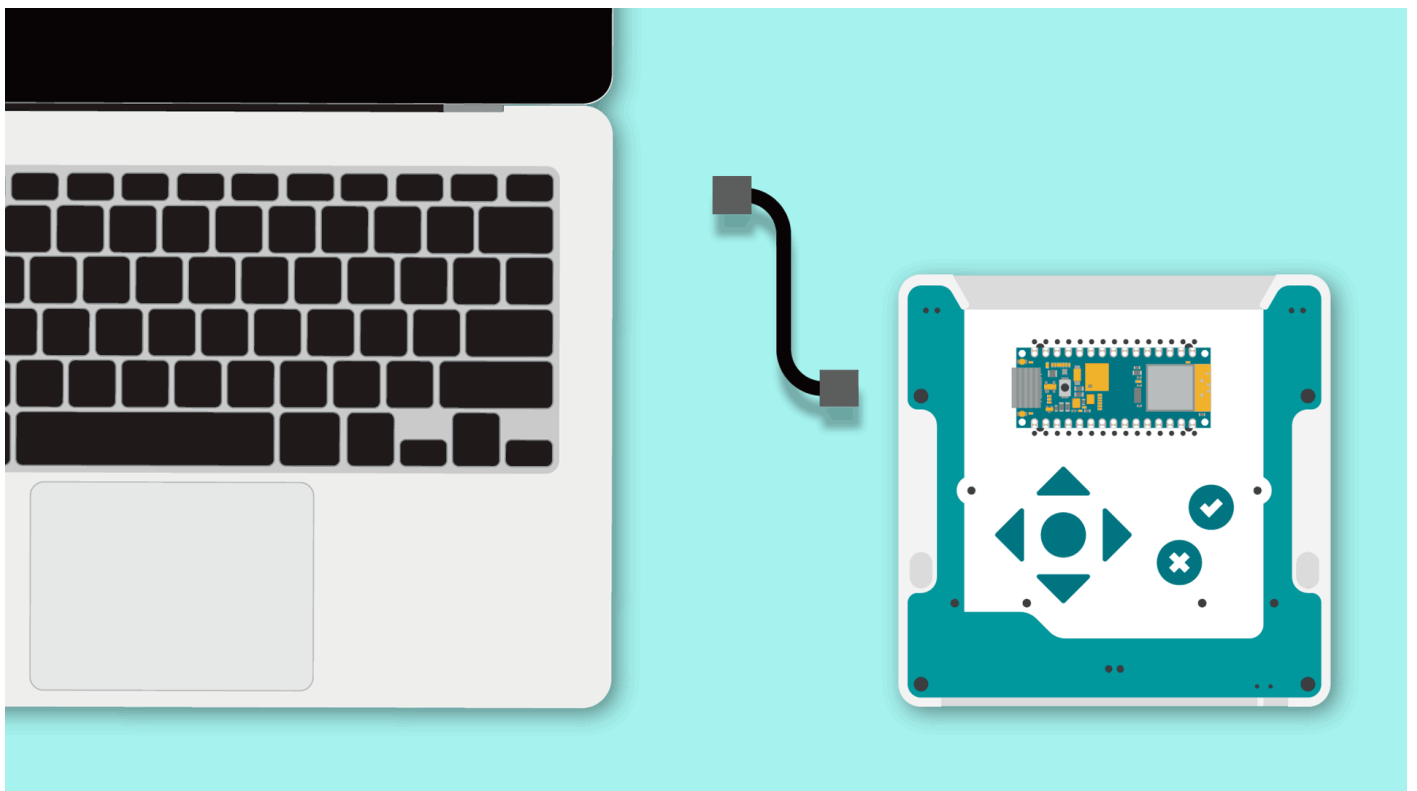
Una vez descargada (un fichero ZIP no lo descomprimas) en el editor Arduino IDE se instala desde este menú



Seleccionamos el fichero Zip que has descargado y ya tenemos la librería instalada

Compilamos

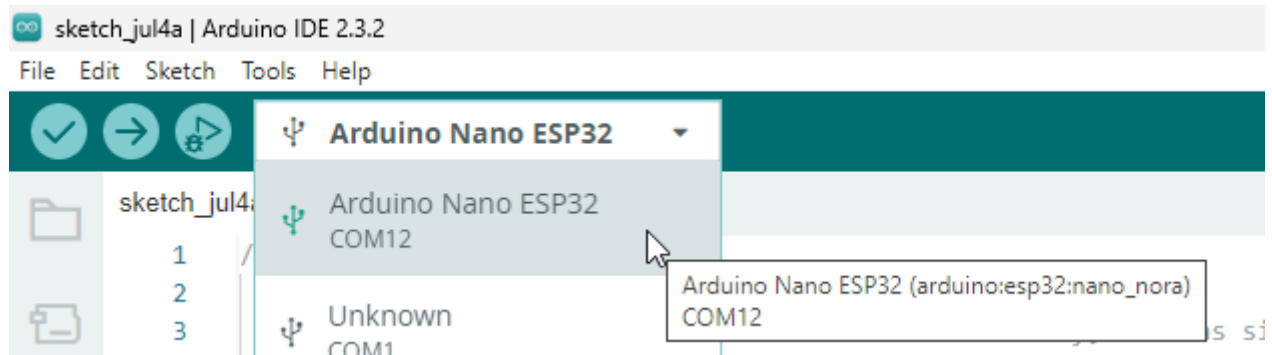
Antes de compilar CONECTAMOS NUESTRO ESP32



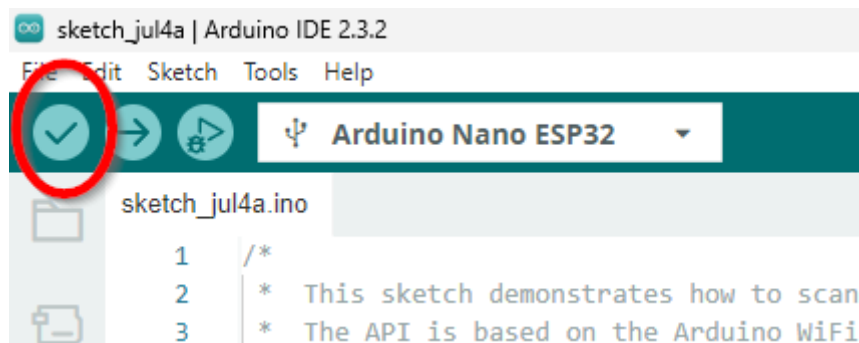
Licencia CC-BY-NC-SA origen <https://courses.arduino.cc/explore-robotics-micropython/lessons/getting-started/>

No hace falta encender el robot Arduino Alvik

Y seleccionamos la placa que ha reconocido

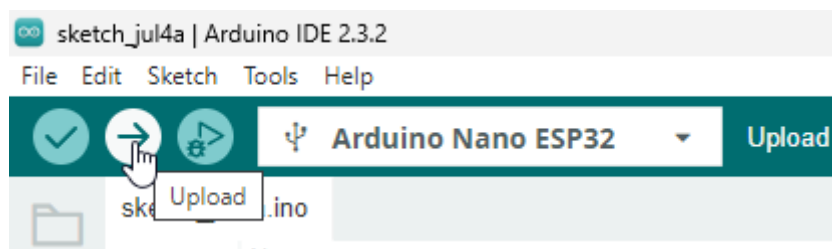


Y ya se puede compilar !!! no tiene que dar ningún fallo



Subirlo al ESP32

Pues si lo intentas subir

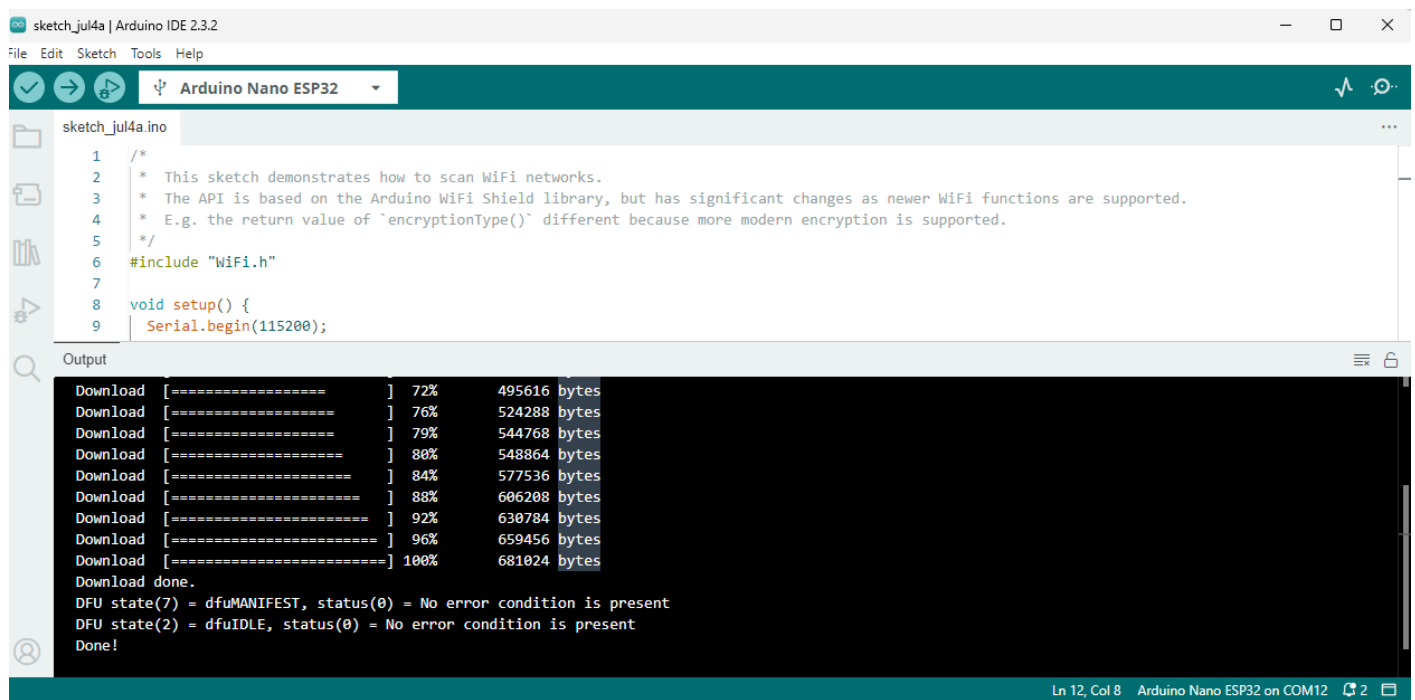


y da este error **NO DEU dfu-util: No DFU capable USB device available Failed uploading: uploading error: exist status 74 ¿Por qué?**

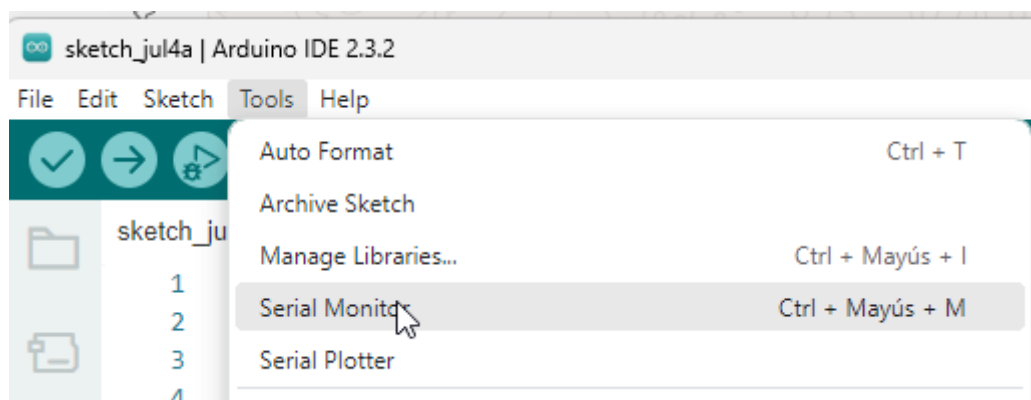
Lee <https://libros.catedu.es/books/arduino-alvik/page/modo-bootloader>

Resultado

Le damos a subir, y en la ventana de Output da como correcto



Y si nos vamos a la ventana del monitor serie



No nos sale nada !!! le das al botón de reset y ya sale :

```

sketch_jul4a.ino
1  /*
2  * This sketch demonstrates how to scan WiFi networks.
3  * The API is based on the Arduino WiFi Shield library, but has significant changes as new
4  * E.g. the return value of `encryptionType()` different because more modern encryption is
5  */
6  #include "WiFi.h"
7
8  void setup() {
9      Serial.begin(115200);

```

Output Serial Monitor x

Message (Enter to send message to 'Arduino Nano ESP32' on 'COM12')

```

Scan done
13 networks found
Nr | SSID                                | RSSI | CH | Encryption
1  | DIRECT-gd-EPSON-ET-4800 Series      | -59  | 11 | WPA2
2  | catedu                              | -69  | 6  | WPA+WPA2
3  | INVITADOS_ARAGON                   | -75  | 1  | open
4  | COLABORADORES_ARAGON               | -76  | 1  | WPA2-EAP
5  | EMBOU_76B6                         | -86  | 5  | WPA+WPA2
6  | COLABORADORES_ARAGON               | -87  | 6  | WPA2-EAP
7  | MIWIFI_F6Yh                        | -88  | 6  | WPA2
8  | INVITADOS_ARAGON                   | -88  | 6  | open
9  | COLABORADORES_ARAGON               | -88  | 11 | WPA2-EAP

```

Dar previamente
al botón de Reset

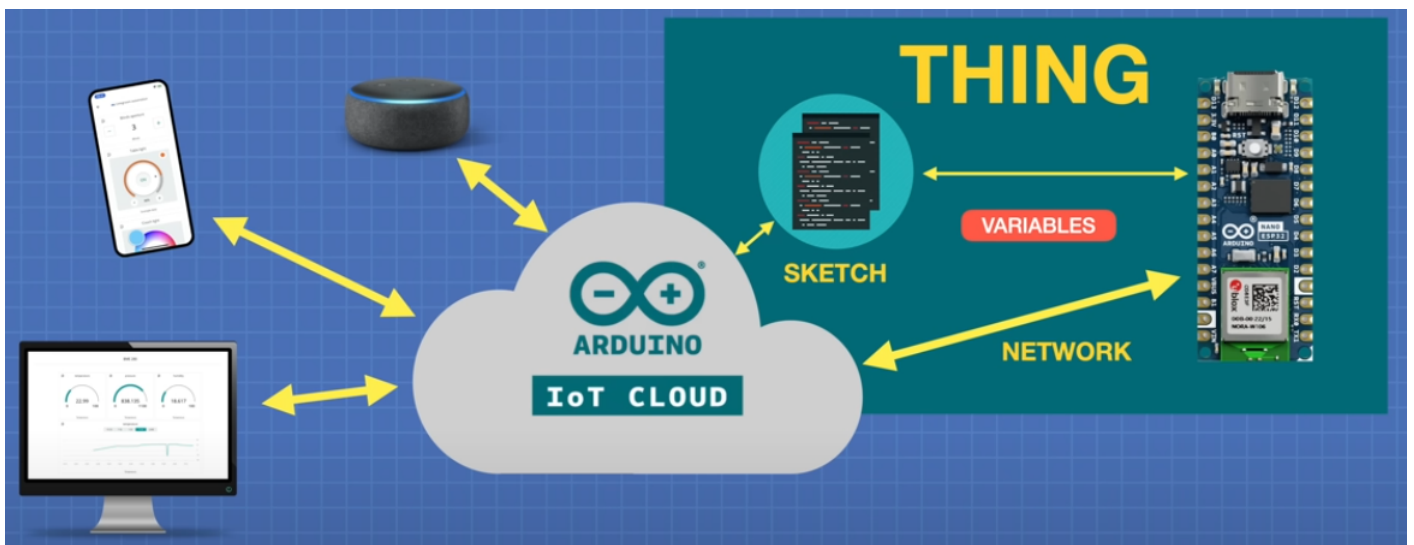
¿Puedo ahora ejecutar un programa en MicroPython?

No, tal y como dice aquí <https://libros.catedu.es/books/arduino-alvik/page/instalar-micropython> tienes que instalar el interpretador/compilador de Micropython dentro del ESP32, sino Arduino Lab for Micropython no se podrá conectar porque no lo encontrará.

Arduino Cloud

Esta plataforma <https://docs.arduino.cc/arduino-cloud/> nos permite conectar nuestras placas (Arduino v4, ESP32, et...) con un panel de control **Dashboard** y así controlarlos a distancia por Internet.

El mecanismo es sencillo, el ESP32 conectado por internet, pasa variables a un código (Sketch), a este conjunto se le llama **Thing**, y este se lo comunica a **IoT CLOUD** y la plataforma lo comunica a los paneles de control. **Dashboard** que se puede ver desde el PC o desde el móvil. El proceso también funciona al revés.



Extraído de [Youtube Exploring the Arduino Nano ESP32](#)

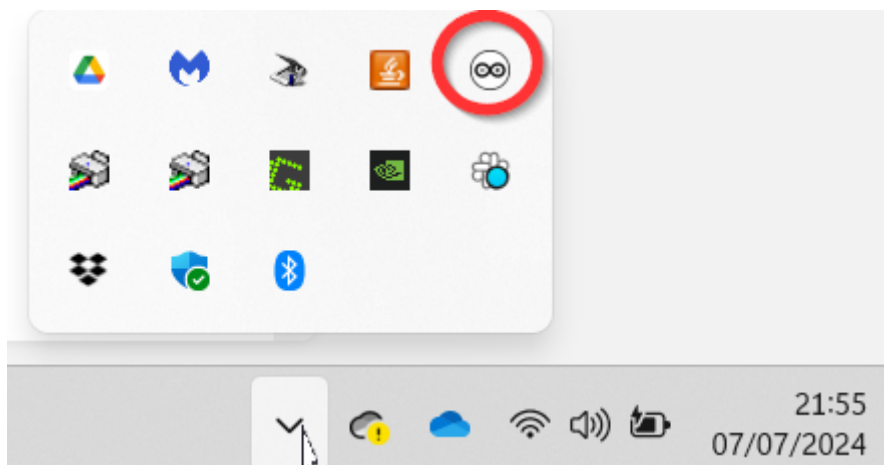
1. Creamos una cuenta en **Arduino Cloud**
2. Instalamos **Arduino Create Agent**
3. **Build the Thing** es decir preparamos nuestra placa ESP32 con el Sketch
 1. Creamos the device
 2. Creamos the thing
 3. Añadimos las variables
 4. Creamos el sketch y lo grabamos en el ESP32
4. Construimos un **Dashboard** o panel de control

PASO 1 LOGUEARSE EN ARDUINO CLOUD

En Plan permite una cuenta gratuita sólo se pueden 2 **things** ver <https://cloud.arduino.cc/plans>

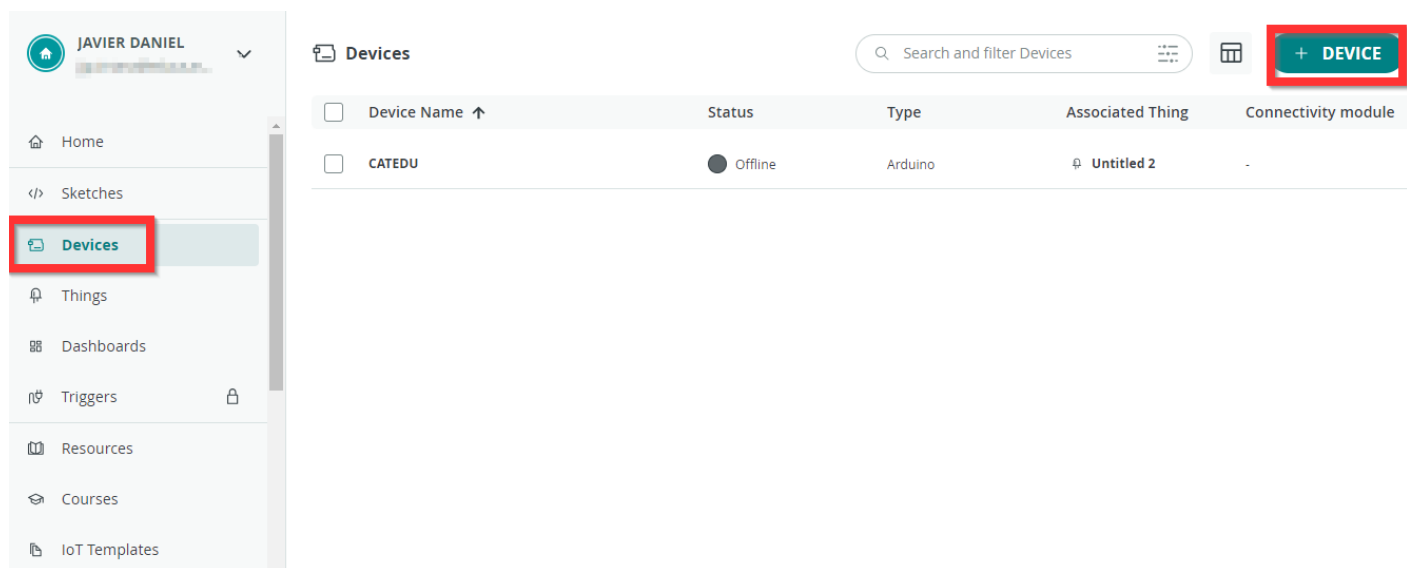
PASO 2 ARDUINO CREATE AGENT

Arduino Create Agent te lo puedes descargar desde <https://cloud.arduino.cc/download-agent>, se descarga, se ejecuta, hay que seguir los pasos, se queda en segundo plano en el PC y no tienes que preocuparte



PASO 3 Build the Thing: CREATE DEVICE

Primero añadimos un Device o placa en <https://app.arduino.cc/devices>



Elegimos placa Arduino

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

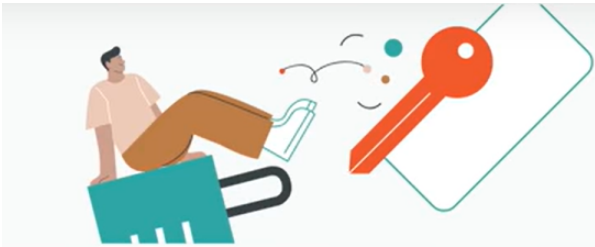


Any Device ⓘ

⌘ Python, MicroPython, JavaScript (NodeJS)

Si falla, ponemos la placa en modo Bootloader (ver qué es eso en <https://libros.catedu.es/books/arduino-alvik/page/instalar-micropython>) y entonces detectará el puerto

Conectamos nuestro Arduino Alvik y saldrá un diálogo con un TOKEN on **Secret key** que lo guardaremos **ante todo no hacerlo público**



Make your device IoT-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

XXXXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

Secret Key



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

PASO 3 Build the Thing: CREATE THING

Una vez creada la placa, nos vamos a Thing, crear

Home

Sketches

Devices

Things

Dashboards

Triggers

Resources

Courses

IoT Templates

Follow these simple steps to create your first Thing on Arduino Cloud

Create a Thing

A Thing is a virtual entity that lets you link your physical device to the Cloud: it includes variables, sketch and metadata

Associate Device and Network


Select the Device you want to use and enter your network credentials, so you can send and receive data remotely


Start creating!


Easily add Cloud Variables that will be automatically included in the sketch, and that are used to exchange data with the Cloud

+ CREATE THING

Asociamos el Thing al Device, y le configuramos una red wifi (te predirá el Secret Key)


 Setup


 Sketch

 Metadata


DD


Associated Device


 ARDUINO ALVIK

ID: d70751da-68fb-494e-b16a-0578...

Type: Arduino Nano ESP32

Status:  Offline

 Change


 Detach

Network

Wi-Fi Name: catedu

Password:

Secret Key:

 Change

PASO 3 Build the Thing: CREATE THING-VARIABLES

Luego añadimos variables, por ejemplo RGBverde que va a encender y apagar la luz verde, va a ser tipo Bool y Read&Write

RGBverde

Declaration

`bool` rGBverde

Type

Boolean

Variable Permission

Read & Write

Update Policy

On change

ID

72371e05-f94d-487a-bb77-f20b97f4dc18



Last Value

false

Last Update

11 Jul 2024 12:12:50

PASO 3 Build the Thing: CREATE THING-SKETCH

Dentro de Thinks nos vamos a SKETCH

Things > micosa-alkvik ▾

Setup

Sketch

Metadata

Cloud Variables

ADD

Name ↓	Last Val...	Last Update
RGBverde <code>bool</code> rGBverde;	false	11 Jul 2024 12:12:50

Associated Device

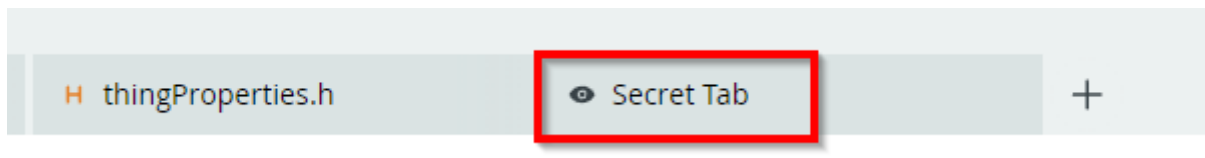
ARDUINO ALVIK

ID: d70751da-68fb-494e-b16a-0578...

Type: Arduino Nano ESP32

Status: Offline

y vemos que ha creado un código ***thingProperties.h*** que tiene que tener el SSID de la wifi, su contraseña y la palabra clave de nuestro ESP32, podemos ponerlo manualmente o nos fijamos y en **Secret Tab** estan ya puestos :



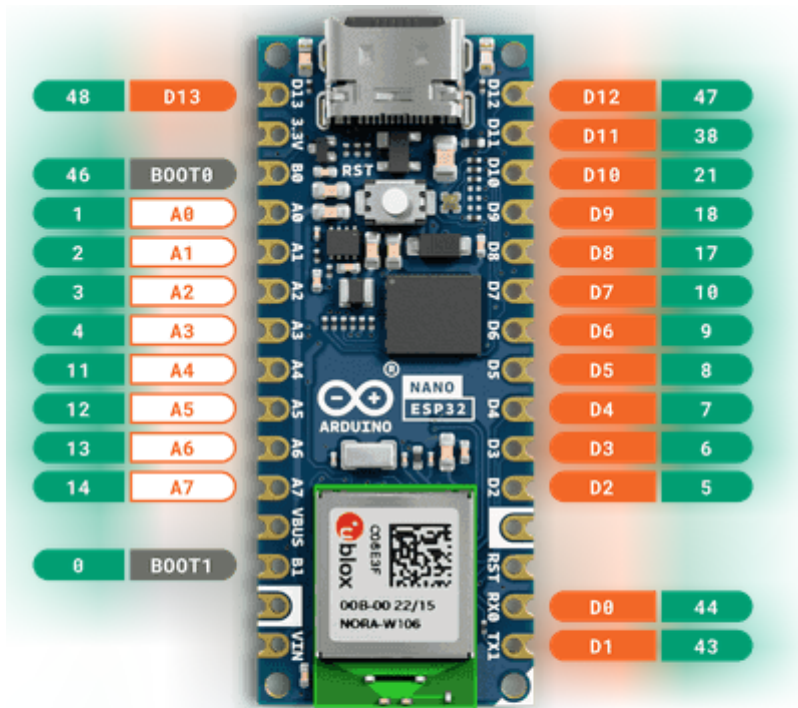
```
1 // Code generated by Arduino IoT Cloud, DO NOT EDIT.
2
3 #include <ArduinoIoTCloud.h>
4 #include <Arduino_ConnectionHandler.h>
5
6 const char DEVICE_LOGIN_NAME[] = "d70751da-68fb-494e-b16a-0578bae10fa2";
7
8 const char SSID[] = SECRET_SSID; // Network SSID (name)
9 const char PASS[] = SECRET_OPTIONAL_PASS; // Network password (use for WPA, or use as
  key for WEP)
10 const char DEVICE_KEY[] = SECRET_DEVICE_KEY // Secret device password
11
12 void onRGBverdeChange();
13
14 bool rGBverde;
15
16 void initProperties(){
```

El otro script es el nombre que hemos creado en Thing y vemos que :

- LINEA 9 Esta declarada la variable que hemos añadido
- LINEA 16 Incluye la librería **thingProperties.h**
- LINEA 41 Añadimos en setup() la declaración que D13 SERÁ SALIDA **pinMode(D13,OUTPUT);**
- LINEA 60 AL 66 Añadimos en **onRGBverdeChange** una condicional, de tal manera que si la variable es cierta, que encienda el led y si es falsa que lo apague

¿Por qué es D13? ¿NO TENDRÍA QUE SER 48?

Eso ya lo hemos visto en <https://libros.catedu.es/books/arduino-alvik/page/parpadeo-led-esp32>



i ESP32 pin numbers

i Nano pin numbers

Fuente <https://docs.arduino.cc/tutorials/alvik/user-manual/>

- SI USAMOS MICROPYTHON TENEMOS QUE USAR LAS VERDES
- SI USAMOS CÓDIGO ARDUINO IDE TENEMOS QUE USAR LAS ROJAS

/*

Sketch generated by the Arduino IoT Cloud Thing "Untitled"

<https://create.arduino.cc/cloud/things/34a0aae1-c7b9-42ab-92d4-0e37bd51031f>

Arduino IoT Cloud Variables description

The following variables are automatically generated and updated when changes are made to the Thing

bool rGBverde;

Variables which are marked as READ/WRITE in the Cloud Thing will also have functions which are called when their values are changed from the Dashboard.

These functions are generated with the Thing and added at the end of this sketch.

```
*/

#include "thingProperties.h"

void setup() {
  // Initialize serial and wait for port to open:
  Serial.begin(9600);
  // This delay gives the chance to wait for a Serial Monitor without blocking if none is found
  delay(1500);

  // Defined in thingProperties.h
  initProperties();

  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
   The following function allows you to obtain more information
   related to the state of network and IoT Cloud connection and errors
   the higher number the more granular information you'll get.
   The default is 0 (only errors).
   Maximum is 4
  */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();

  /// MI CODIGO
  pinMode(D13,OUTPUT);
}

void loop() {
  ArduinoCloud.update();
  // Your code here

}
```

```

/*
  Since RGBverde is READ_WRITE variable, onRGBverdeChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onRGBverdeChange() {
  // Add your code here to act upon RGBverde change
  if (rGBverde){
    digitalWrite(D13,HIGH);

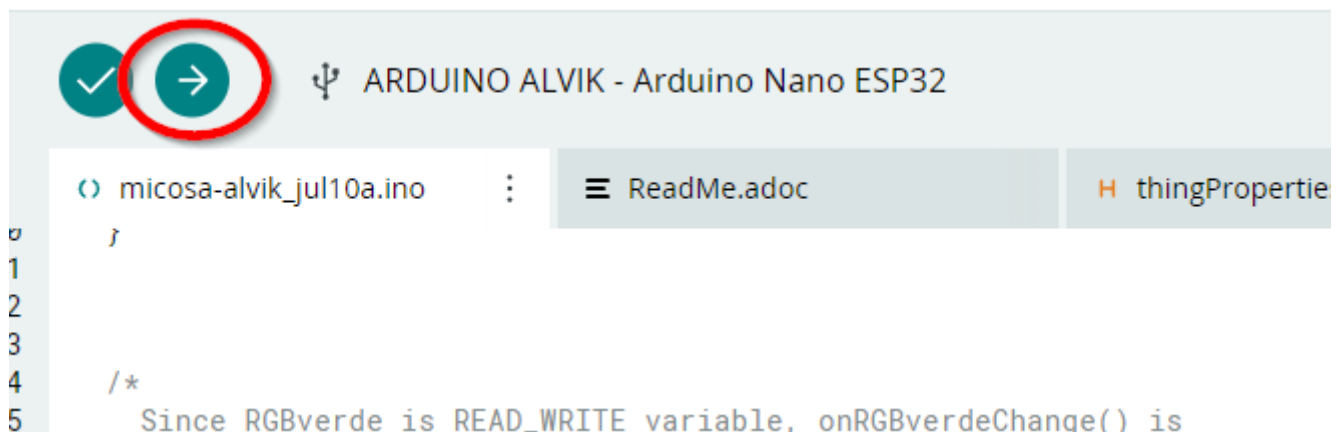
  }else{
    digitalWrite(D13,LOW);

  }
}

/*
  Since RGBrojo is READ_WRITE variable, onRGBrojoChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onRGBrojoChange() {
  // Add your code here to act upon RGBrojo change
}

```

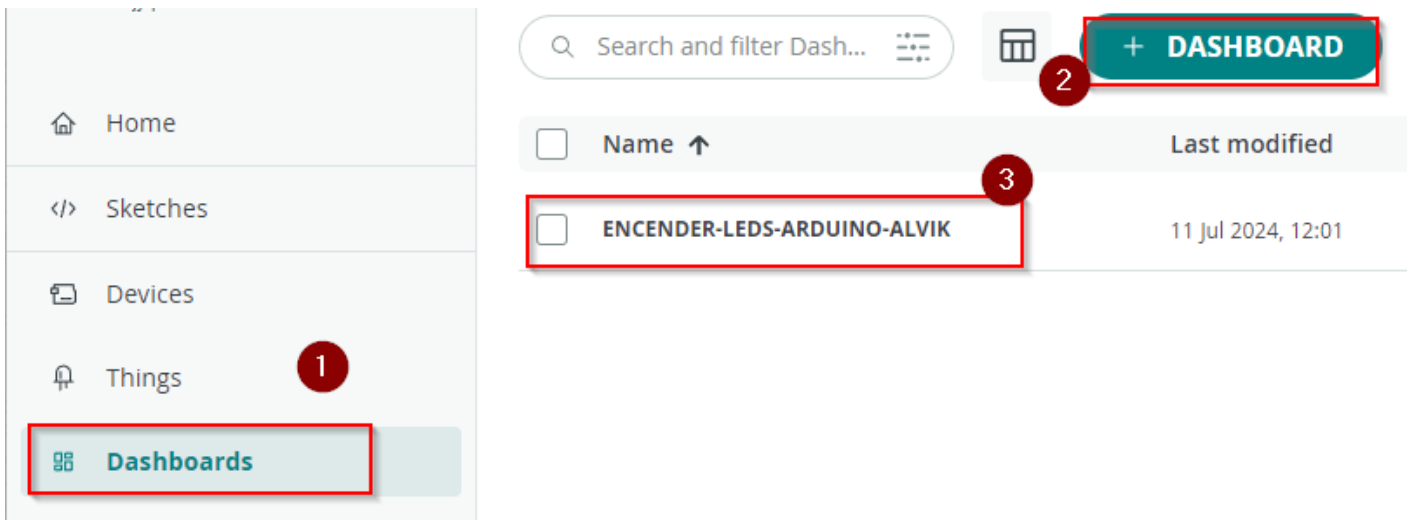
Lo subimos



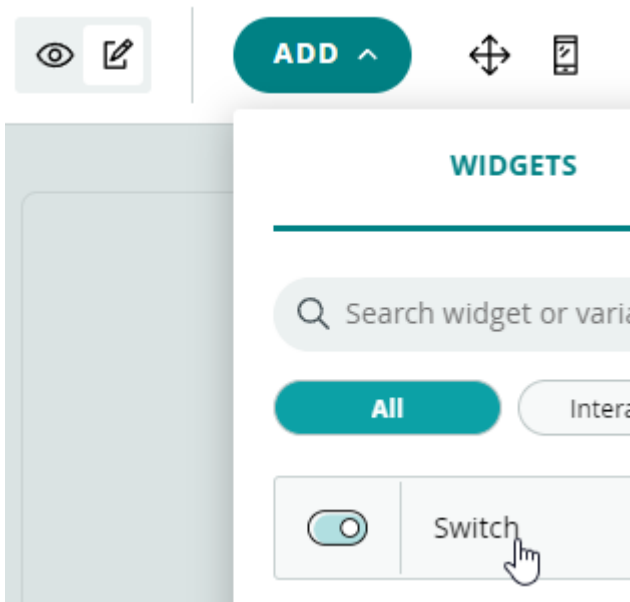
Ojo, tienes que tener el Arduino **Arduino Create Agent paso 2**

PASO 4 Dashboard

Creamos un panel de control



Y le añadimos un Switch asociado a la variable RGBverde



Podemos ver el dashboard en un teléfono móvil instalando la [APP Arduino IoT Cloud Remote](#)

Arduino IoT Cloud Remote

Arduino

La aplicación Arduino para controlar sus proyectos de Internet de las cosas desde cualquier lugar.

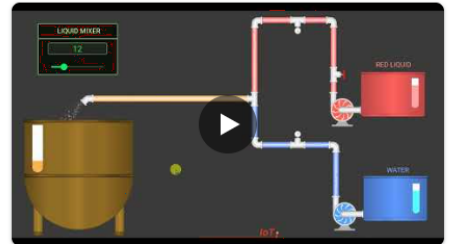
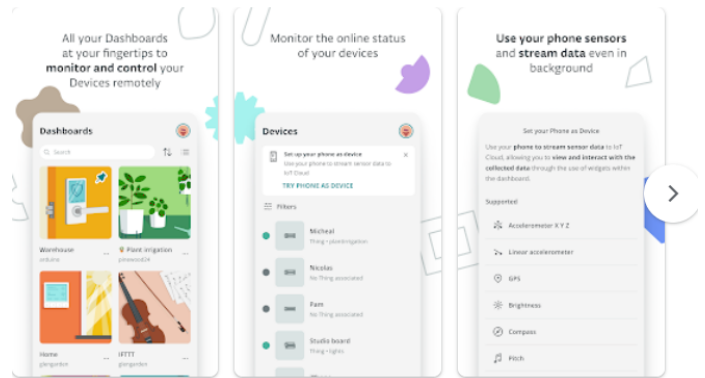


3,8★
992 reseñas

100 mil+
Descargas

PEGI 3

Descargar



Al loguearse con tu cuenta, ya nos aparece el Dashboard

Resultado

<https://www.youtube.com/embed/YDVMYbJtWUU>

Coche teledirigido

Aprovechamos el programa que enciende y apaga un led por Arduino Cloud

Variables

Le añadimos tres variables más :

1. **velocidad** tipo entero Read&Write
2. **giro** tipo entero Read&Write
3. **distancia** tipo float Read

Cloud Variables

ADD

	Name ↓	Last Value	Last Update
<input type="checkbox"/>	distancia float distancia;	0.3	11 Jul 2024 22:32:38
<input type="checkbox"/>	giro int giro;	1	11 Jul 2024 22:32:31
<input type="checkbox"/>	RGBverde bool rGBverde;	true	11 Jul 2024 22:31:00
<input type="checkbox"/>	velocidad int velocidad;	0	11 Jul 2024 22:32:01

Sketch

En thingProperties.h añade **automáticamente** estas variables y funciones, **no tienes que añadirlas** :

```
void onGiroChange();
void onVelocidadChange();
void onRGBverdeChange();

float distancia;
int giro;
int velocidad;
bool rGBverde;
```

Pero en la función principal, nosotros vamos a poner el siguiente código :

- Línea 2 **#include "Arduino_Alvik.h"** para que incluya la librería de manejo del robot

- Línea 4 Creamos un objeto alvik **Arduino_Alvik alvik;**
- Línea 6 Creamos una variable tipo array de 5 elementos para almacenar las distancias que lee el sensor de distancia **float distances[5];**
- Línea 15 arrancamos el objeto alvik **alvik.begin();**
- Línea 41 que el alvik se mueva según la velocidad y el giro **alvik.drive(velocidad,giro);**
 - Es la instrucción principal y qué sencilla ☐
- Línea 42 leemos el array de distancias **alvik.get_distance(distances[0], distances[1], distances[2], distances[3], distances[4]);**
- Línea 43 de todas las distancias, sólo nos importa la 2 **distancia=distances[2];**

Nota: la instrucción 41 se han colocado dentro de loop() pero también se podría haber colocado dentro de onGiroChange();
onVelocidadChange();

```
#include "thingProperties.h"
#include "Arduino_Alvik.h"

Arduino_Alvik alvik;

float distances[5];

void setup() {
  // Initialize serial and wait for port to open:
  Serial.begin(9600);
  // This delay gives the chance to wait for a Serial Monitor without blocking if none is found
  delay(1500);

  alvik.begin();

  // Defined in thingProperties.h
  initProperties();

  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
  The following function allows you to obtain more information
  related to the state of network and IoT Cloud connection and errors
  the higher number the more granular information you'll get.
  The default is 0 (only errors).
```

```

    Maximum is 4
*/
setDebugMessageLevel(2);
ArduinoCloud.printDebugInfo();

/// MI CODIGO
pinMode(D13,OUTPUT);
}

void loop() {
    ArduinoCloud.update();
    // Your code here

    alvik.drive(velocidad,giro);
    alvik.get_distance(distances[0], distances[1], distances[2], distances[3], distances[4]);
    distancia=distances[2];

}

/*
    Since RGBverde is READ_WRITE variable, onRGBverdeChange() is
    executed every time a new value is received from IoT Cloud.
*/
void onRGBverdeChange() {
    // Add your code here to act upon RGBverde change
    if (rGBverde){
        digitalWrite(D13,HIGH);

    }else{
        digitalWrite(D13,LOW);

    }
}

/*
    Since RGBrojo is READ_WRITE variable, onRGBrojoChange() is

```

executed every time a new value is received from IoT Cloud.

*/

```
void onRGBrojoChange() {
```

```
    // Add your code here to act upon RGBrojo change
```

```
}
```

/*

Since Velocidad is READ_WRITE variable, onVelocidadChange() is executed every time a new value is received from IoT Cloud.

*/

```
void onVelocidadChange() {
```

```
    // Add your code here to act upon Velocidad change
```

```
}
```

/*

Since Giro is READ_WRITE variable, onGiroChange() is executed every time a new value is received from IoT Cloud.

*/

```
void onGiroChange() {
```

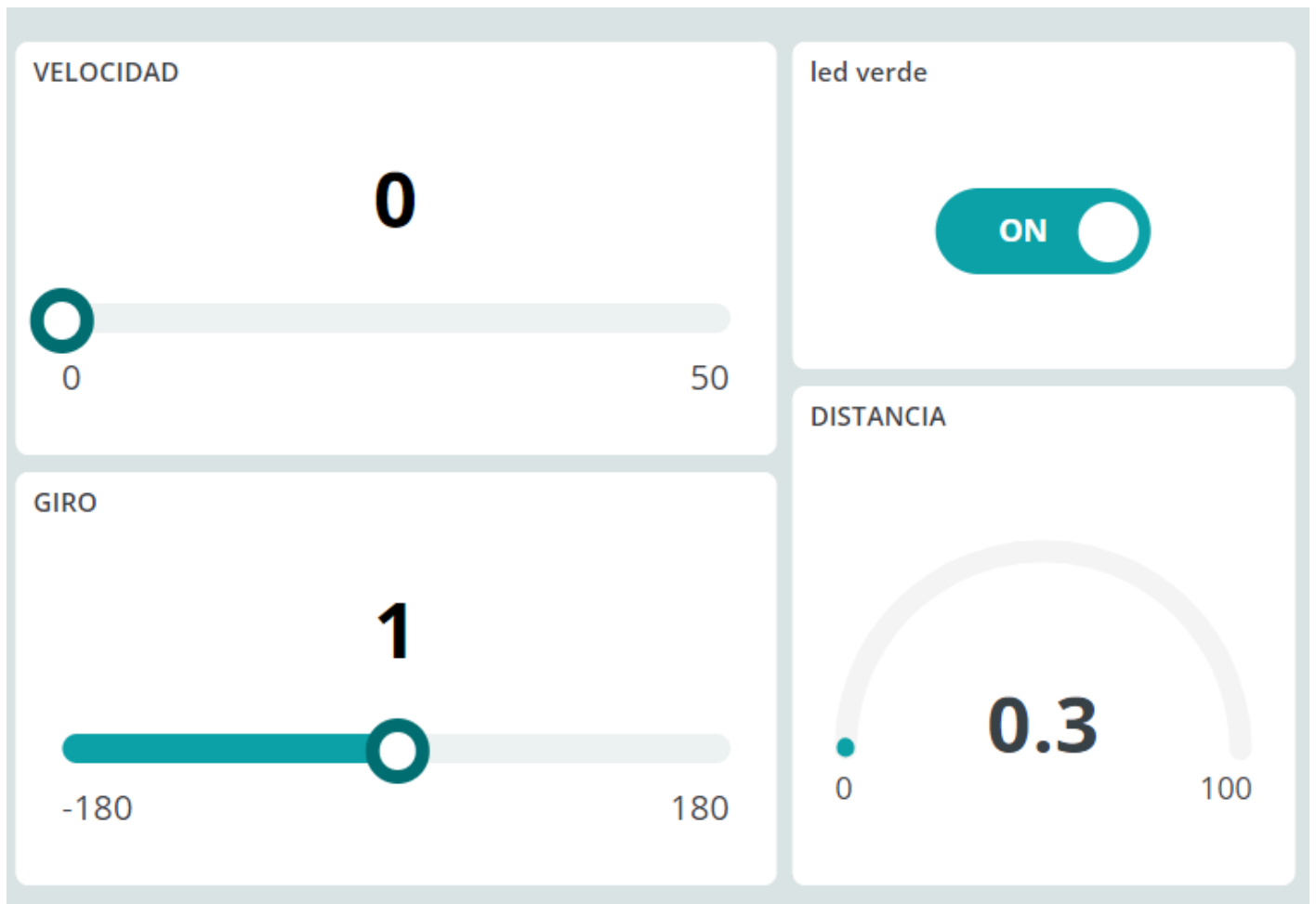
```
    // Add your code here to act upon Giro change
```

```
}
```

Dashboard

Creamos un panel de control con:

- Un **slider** para velocidad de 0 a 50
- Un **slider** para el giro de -180 a +180
- Un **gauge** para distancia



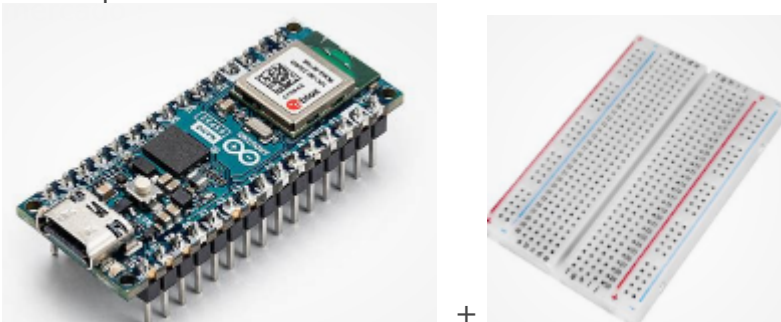
Resultado

<https://www.youtube.com/embed/5cWI3y3A3z8>

ESP32 + Sensores externos + IoT

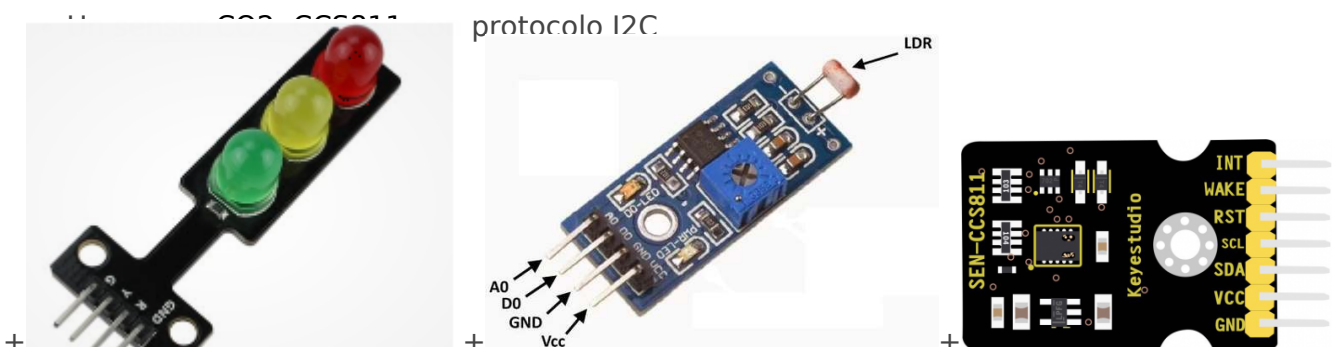
OBJETIVO

Ahora vamos a utilizar el ESP32 SIN EL ARDUINO ALVIK podemos sacar la placa microcontroladora y ponerlo en una placa *protoboard* y experimentar con sensores y actuadores estándares en el



Para ver varias posibilidades, vamos a ver estos sensores y actuadores (recomendamos ver estas páginas [actuadores](#) y [sensores](#))

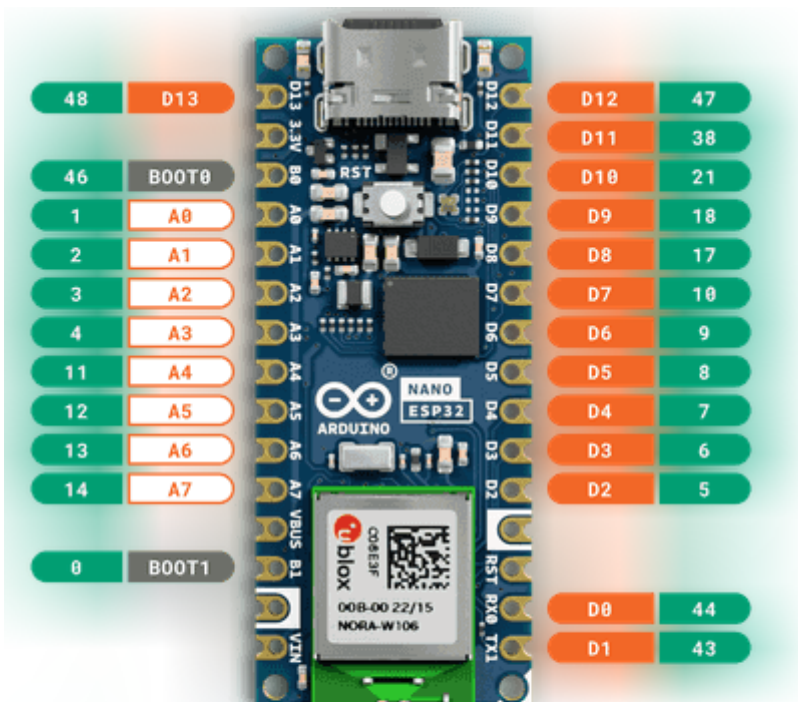
- Un led de salida simple, para practicar salida digital en mi caso voy a elegir este gracioso [semáforo](#)
- Un [sensor LDR](#) pero para practicar los dos tipos de señal,
 - con salida analógica
 - con salida digital.



ESQUEMA DE CONEXIONES

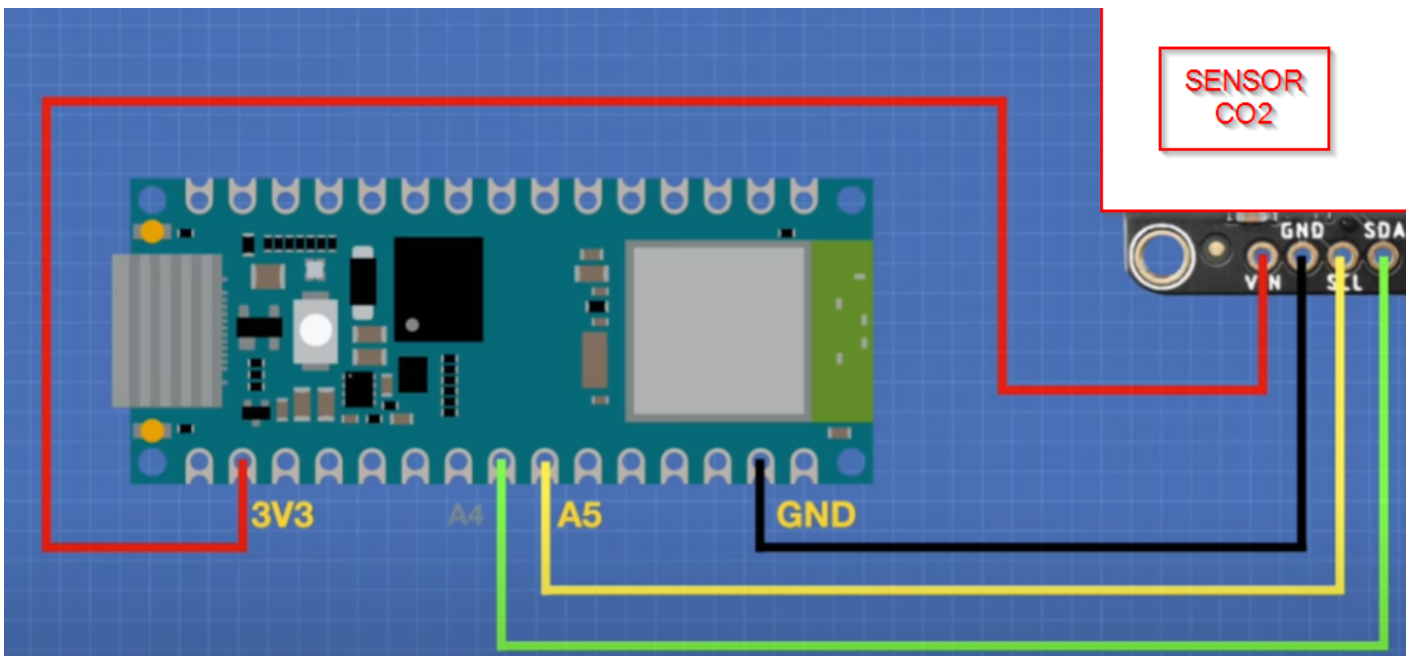
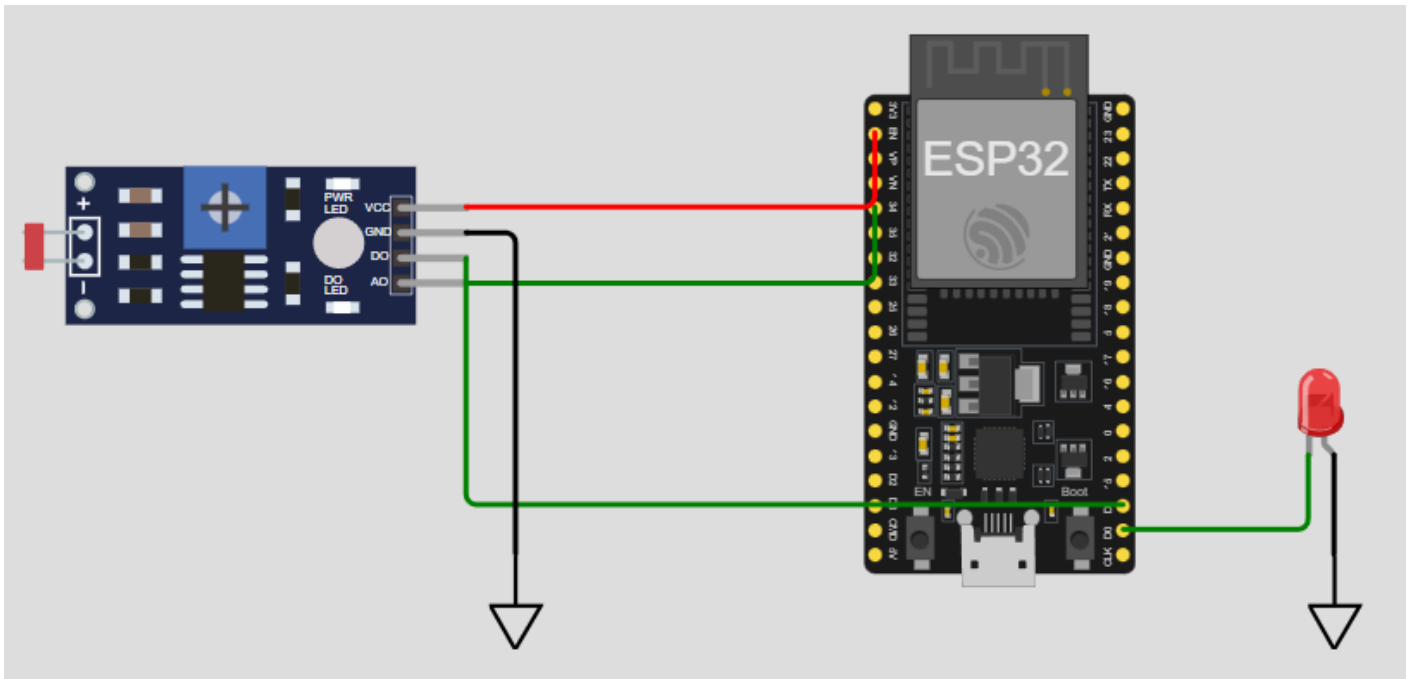
- **SEMAFORO**

- LED ROJO al D1 del ESP32
- GND a GND
- **MODULO SENSOR LDR**
 - SEÑAL DIGITAL al D0 del ESP32
 - SEÑAL ANALÓGICA al A0 del ESP32
 - VCC a 3V3
 - GND A GND
- **MODULO SENSOR CO2**
 - SCL al pin A5 del ESP32
 - SDA al pin A4 del ESP32
 - PIN WAKE a GND
 - VCC a 3V3
 - GND A GND



i ESP32 pin numbers

i Nano pin numbers



DEVICES

Nos vamos a Arduino Cloud, y en **DEVICES** añadimos el ESP32 y obtenemos el TOKEN o palabra secreta (si has hecho la práctica anterior, no es necesario pues ya tenemos el TOKEN o palabra secreta) como es similar al caso anterior, no lo desarrollamos. (Nos pedirá también el SSID y la contraseña de la red wifi)

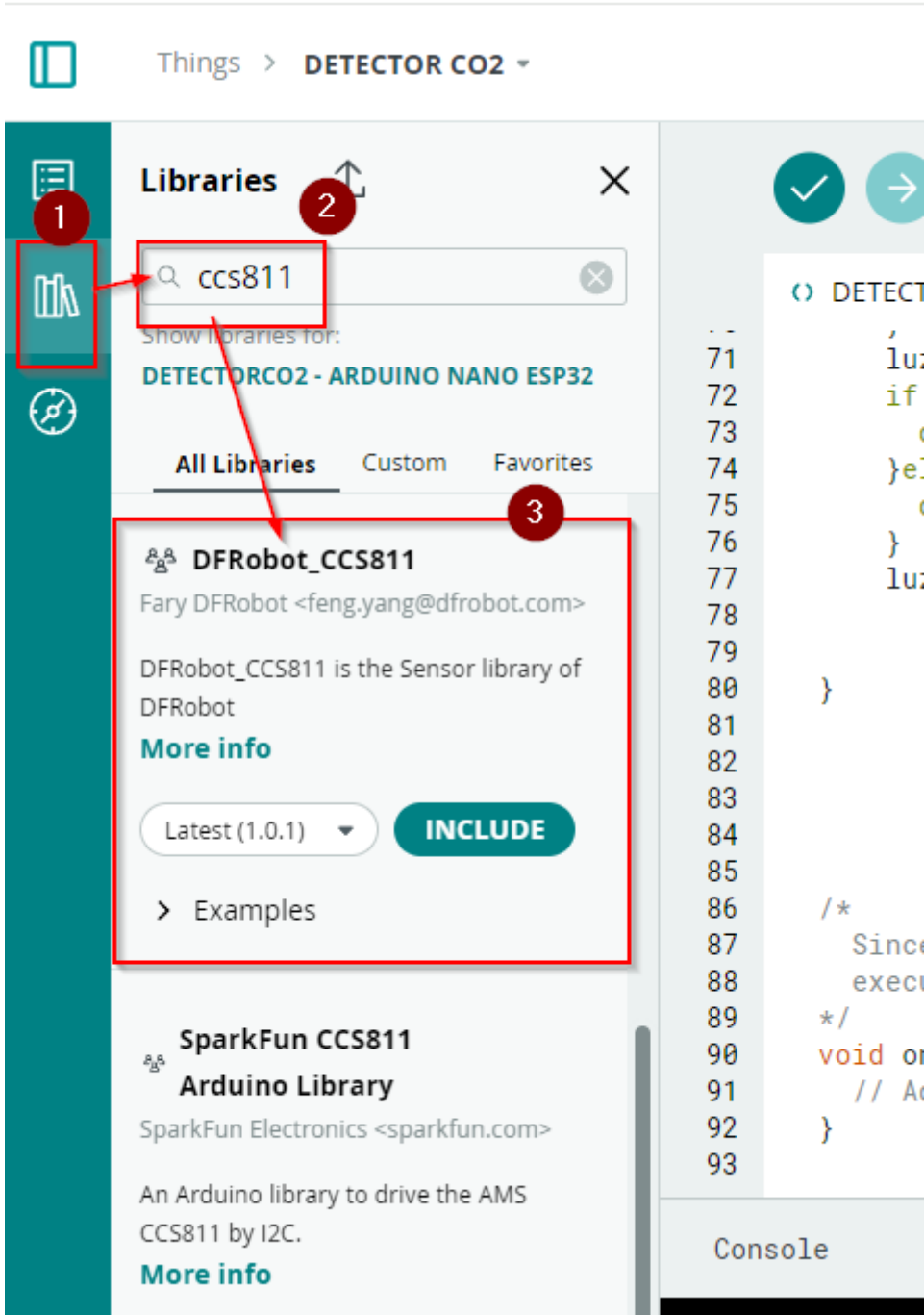
VARIABLES

Añadimos las siguientes **variables** :

- CO2 tipo int y Read
- luz tipo int y Read
- luzdigital tipo bool y Read
- rojo tipo bool y Read&Write

EL SCKETCH -LIBRERIA CCS811

Primero añadiríamos la librería de keystudio <https://fs.keyestudio.com/KS0457> pero no lo permite Arduino Cloud, viendo las instrucciones, vemos que son las mismas que en los ejemplos de esta librería la de DF que es la que instalamos :



esto provoca la incorporación de la línea 1 **#include <DFRobot_CCS811.h>**

EL SCKETCH -EL CÓDIGO

- Tenemos las variables definidas en las líneas 10-13 :
 - **int cO2;**
 - **int luz;**
 - **bool luzdigital;**
 - **bool rojo;**
- Definimos una variable de tipo el sensor CCS811 en la línea 23 **DFRobot_CCS811 CCS811;**
- En Setup en las líneas 48-21 arrancamos ese sensor:
 - **while(CCS811.begin() != 0){**
 Serial.println("failed to init chip, please check if the chip connection is fine");
 delay(1000);
 }
- Definimos los pines digitales 0 y 1 como entrada y salida respectivamente:
 - **pinMode(1,OUTPUT);**
 pinMode(0,INPUT);
- En las líneas 60-70 que lea el CCS811 y la parte de CO2 que lo meta en la variable CO2 (línea 63)
 - **if(CCS811.checkDataReady() == true){**
 Serial.print("CO2: ");
 Serial.print(CCS811.getCO2PPM());
 cO2=CCS811.getCO2PPM();
 Serial.print("ppm, TVOC: ");
 Serial.print(CCS811.getTVOCPPB());
 Serial.println("ppb");

 } else {
 Serial.println("Data is not ready!");
 }
- En las línea 71 que luz sea la lectura del pin A0 **luz = analogRead(A0);**
- En las líneas 72-76 que según rojo se encienda o no el led
 - **if (rojo){**
 digitalWrite(1,HIGH);
 }else{
 digitalWrite(1,LOW);
 }
- En la línea 77 que luzdigital sea la lectura de la salida digital del sensor LDR
 - **luzdigital=digitalRead(0);**

```
#include <DFRobot_CCS811.h>
```

```
/*
```

Sketch generated by the Arduino IoT Cloud Thing "Untitled"

<https://create.arduino.cc/cloud/things/17c10209-3874-430a-877c-c082ff7dd38d>

Arduino IoT Cloud Variables description

The following variables are automatically generated and updated when changes are made to the Thing

```
int cO2;
```

```
int luz;
```

```
bool luzdigital;
```

```
bool rojo;
```

Variables which are marked as READ/WRITE in the Cloud Thing will also have functions which are called when their values are changed from the Dashboard.

These functions are generated with the Thing and added at the end of this sketch.

```
*/
```

```
#include "thingProperties.h"
```

```
//DFRobot_CCS811 CCS811(&Wire, /*IIC_ADDRESS=*/0x5A);
```

```
DFRobot_CCS811 CCS811;
```

```
void setup() {
```

```
  // Initialize serial and wait for port to open:
```

```
  Serial.begin(9600);
```

```
  // This delay gives the chance to wait for a Serial Monitor without blocking if none is found
```

```
  delay(1500);
```

```
  // Defined in thingProperties.h
```

```
  initProperties();
```

```
  // Connect to Arduino IoT Cloud
```

```
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
```

```
/*
```

The following function allows you to obtain more information

related to the state of network and IoT Cloud connection and errors

the higher number the more granular information you'll get.

The default is 0 (only errors).

Maximum is 4

*/

```
setDebugMessageLevel(2);
```

```
ArduinoCloud.printDebugInfo();
```

```
while(CCS811.begin() != 0){
```

```
    Serial.println("failed to init chip, please check if the chip connection is fine");
```

```
    delay(1000);
```

```
}
```

```
pinMode(1,OUTPUT);
```

```
pinMode(0,INPUT);
```

```
}
```

```
void loop() {
```

```
    ArduinoCloud.update();
```

```
    // Your code here
```

```
    if(CCS811.checkDataReady() == true){
```

```
        Serial.print("CO2: ");
```

```
        Serial.print(CCS811.getCO2PPM());
```

```
        cO2=CCS811.getCO2PPM();
```

```
        Serial.print("ppm, TVOC: ");
```

```
        Serial.print(CCS811.getTVOCPPB());
```

```
        Serial.println("ppb");
```

```
    } else {
```

```
        Serial.println("Data is not ready!");
```

```
    }
```

```
    luz = analogRead(A0);
```

```
    if (rojo){
```

```
        digitalWrite(1,HIGH);
```

```
    }else{
```

```
        digitalWrite(1,LOW);
```

```
    }
```

```
    luzdigital=digitalRead(0);
```

```
}
```

```
/*
```

```
Since Rojo is READ_WRITE variable, onRojoChange() is  
executed every time a new value is received from IoT Cloud.
```

```
*/
```

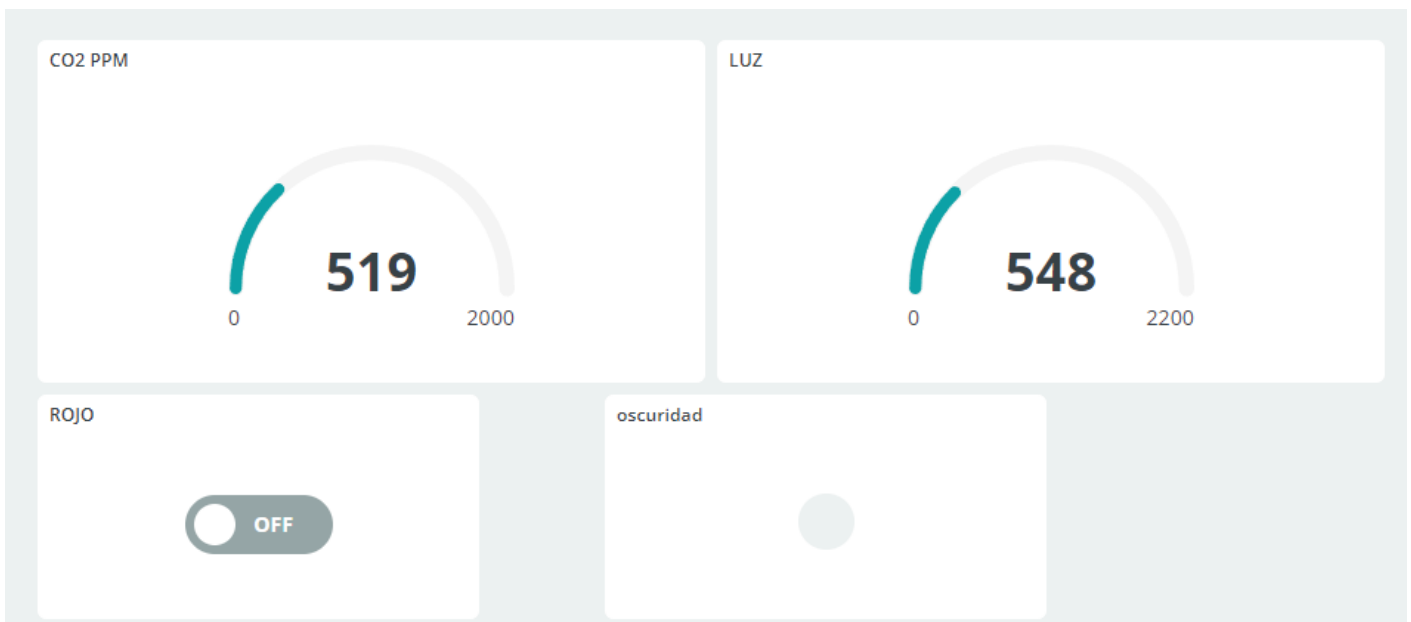
```
void onRojoChange() {
```

```
// Add your code here to act upon Rojo change
```

```
}
```

DASHBOARD

- Un gauge ligado a **CO2** desde 0 a 2000
- Un gauge ligado a **Luz** de 0 a 2.200
- Un Switch ligado a **rojo**
- Un led de oscuridad ligado a **luzdigital**



Alternativa : en vez de luz tendría que llamarse "oscuridad" que sea luz pero que vaya al revés

RESULTADO

<https://www.youtube.com/embed/WS02iGCux4U>

ALTERNATIVA: Que el semáforo visualice los niveles peligrosos de CO2, por ejemplo el umbral del amarillo 600-1.000

¿Te atreves a poner un servomotor?