Smart Open Lab

Curso Wemos (Arduino)

Internet of Things

#IoT

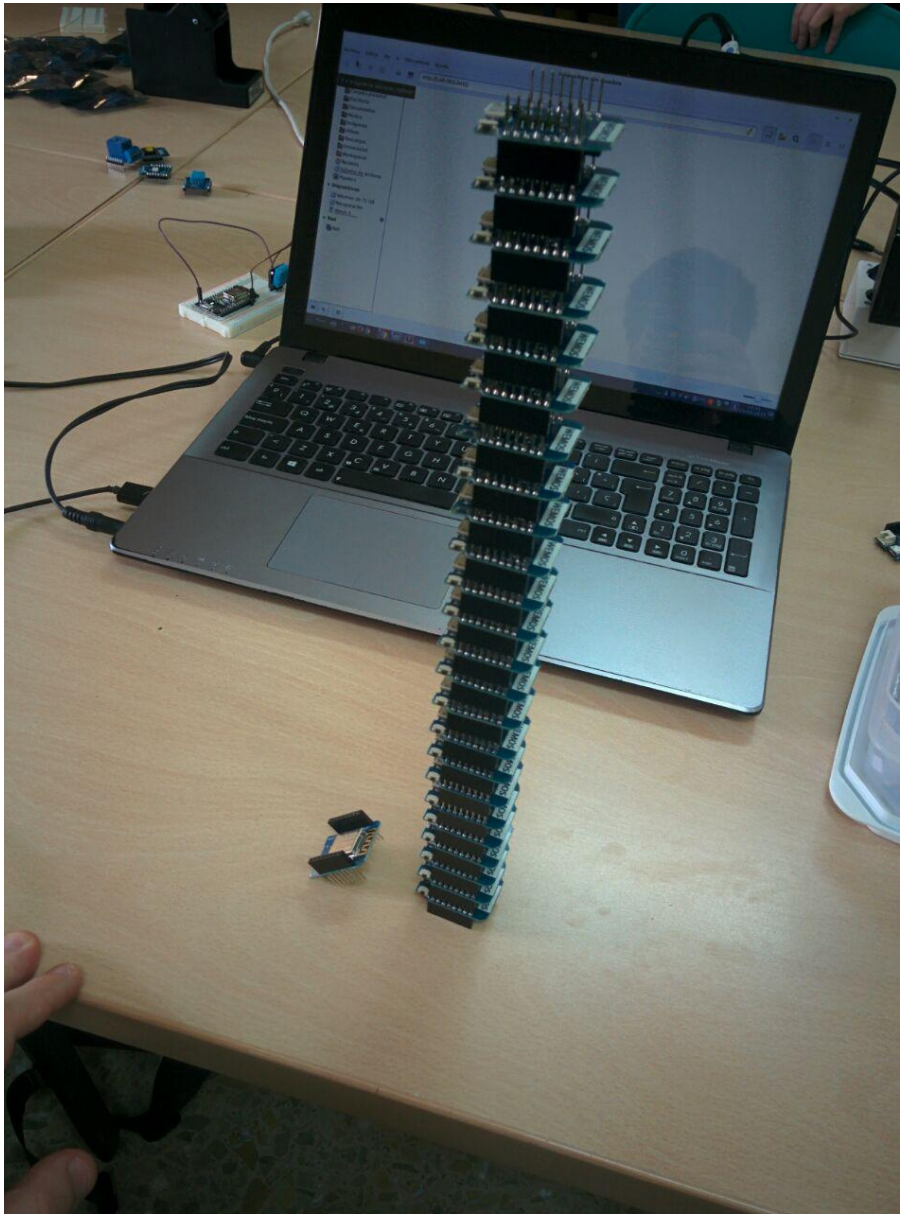
ARDUINO

Objetivo

EJ1

- **Que al pulsar un botón en casa o en el trabajo trabajo se active la WiFi del móvil, esté donde esté**
- **Soltar el botón IoT en casa/trabajo y que sea fácil de configurar**
  - Que no requiera conocimientos técnicos para conectarlo la primera vez a Internet…
- **Que el botón sea muy barato**

Wemos es una placa que usa el chip ESP8266



- **Chip WiFi por unos 3€**
- **11 I/O pines**
  - Todos (excepto D0) soportan interrupciones, PWM, I2C y one-wire
- **1 entrada analógica**
- **Micro USB**
  - (alimentación y datos)
- **Compatible:**
  - Arduino
  - Nodemcu

Solución

EJ1

- Usar un portal cautivo si el dispositivo no puede conectar a Internet, de modo que cuando me conecte a su red me permita configurarlo
- Invocar un webhook (enlace) en un servidor remoto (IFTTT) cuando se pulse un botón
- Mantener el dispositivo IoT en bajo consumo dormido, y activarlo mediante interrupciones

Modos de ahorro de energía en ESP8266

	Modem-sleep	Light-sleep	Deep-sleep
Wi-Fi	OFF	OFF	OFF
System clock	ON	OFF	OFF
RTC	ON	ON	ON
CPU	ON	Pendiente	OFF
Corriente	~15 mA	~0.4 mA	~20 µA

Automáticamente activo cuando conectado a un AP y en modo STA WiFi.mode(WIFI_STA)

Automáticamente activo cuando conectado a Wi-Fi y la CPU sin uso. Se puede despertar con interrupciones

Solo se puede despertar reseteándolo (WDT o Pin RST)

ESP.deepSleep(sleepTimeuS);

Portal cautivo: Librería “WiFiManager”

EJ1

```
#include <ESP8266WiFi.h>
#include <DNSServer.h>

#include <ESP8266WebServer.h>

#include <WiFiManager.h>
void setup() {
  WiFiManager wifiManager;
  wifiManager.autoConnect("MARINO_ESP", "1234");

  //Serial.begin(115200);
 //Serial.println(WiFi.localIP());
}

void loop() {
}
```

← Librería base ESP8266

← Servidor DNS local para redireccionarnos al portal cautivo

← Para montar el servidor y servir la web de configuración

← Hace todo el tema de portal y contraseñas automáticamente

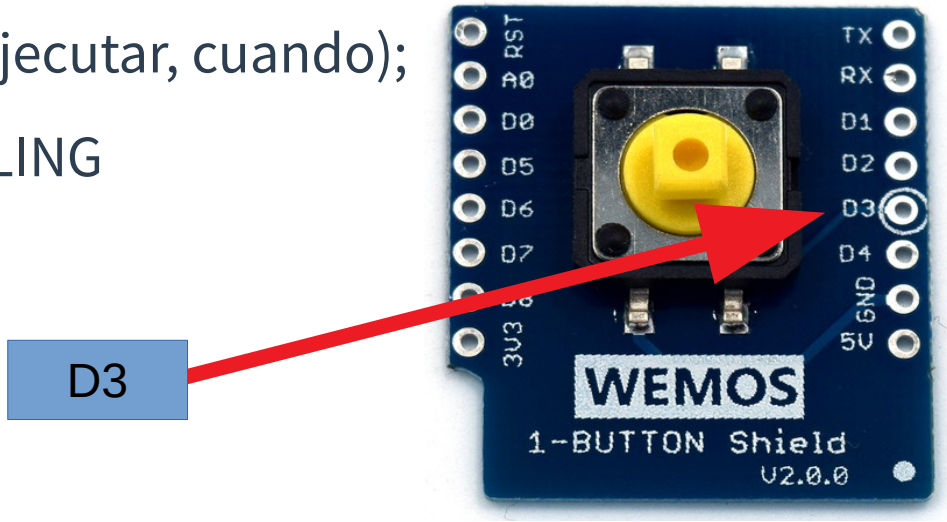
← Se puede imprimir algo para probar cuando pasa

Bajo consumo: activación por interrupciones

EJ1

- **Consejos:**
  - Las interrupciones deben ser lo más cortas posibles
    - si es posible simplemente “seteando” banderas
    - las banderas declararlas **volatile**
      - **E.g., volatile bool BUTTON_state**
  - Para añadir interrupciones en arduino (setup):

    attachInterrupt(GPIO_Pin, funcionAEjecutar, cuando);
    - //cuando = CHANGE, RISING, FALLING
      - Lógica inversa ESP8266



Usad el modo de ahorro LIGHT para...

- **… que el ESP8266 espere dormido y al presionar el botón se active de nuevo**

```
while (condición)  ← ¿¿¿¿¿¿ usando BUTTON_state ???????
    yield();
```

Usando el modo de ahorro LIGHT y despertando al clickear el botón

EJ1

```
volatile bool BUTTON_state = false;


void button_pressed() {
 noInterrupts();
 BUTTON_state = true;
 interrupts();
}


void setup() {
 Serial.begin(115200);
 attachInterrupt(D3, button_pressed,
FALLING); //Lógica inversa
}
```

```
void loop() {
 while (!BUTTON_state)
  yield();


 Serial.println("botón pulsado");
 BUTTON_state = false;
}
```

Invocar un webhook (de IFTTT): Librería “IFTTTWebhook”

- #include <IFTTTWebhook.h>
- #define IFTTT_API_KEY "la tuya"
- #define IFTTT_EVENT_NAME "button_pressed"

- IFTTTWebhook myWebHook(IFTTT_API_KEY, IFTTT_EVENT_NAME);
- myWebHook.trigger();

TODO JUNTO

EJ1

```
#include <ESP8266WiFi.h>
#include <DNSServer.h>
#include <ESP8266WebServer.h>
#include <WiFiManager.h>

#include <IFTTTWebhook.h>
#define IFTTT_API_KEY "la tuya"
#define IFTTT_EVENT_NAME "button_pressed"

volatile bool BUTTON_state = false;

void button_pressed() {
 noInterrupts();
 BUTTON_state = true;
 interrupts();
}
```

```
void setup() {
 WiFiManager wifiManager;
 wifiManager.autoConnect("MARINO_ESP", "1234");
 attachInterrupt(D3, button_pressed, FALLING);
 Serial.begin(115200);
 Serial.println(WiFi.localIP());
}

void loop() {
 while (!BUTTON_state)
  yield();

 IFTTTWebhook myWebHook(IFTTT_API_KEY,
IFTTT_EVENT_NAME);
 myWebHook.trigger();

 Serial.println("botón pulsado");
 BUTTON_state = false;
}
```