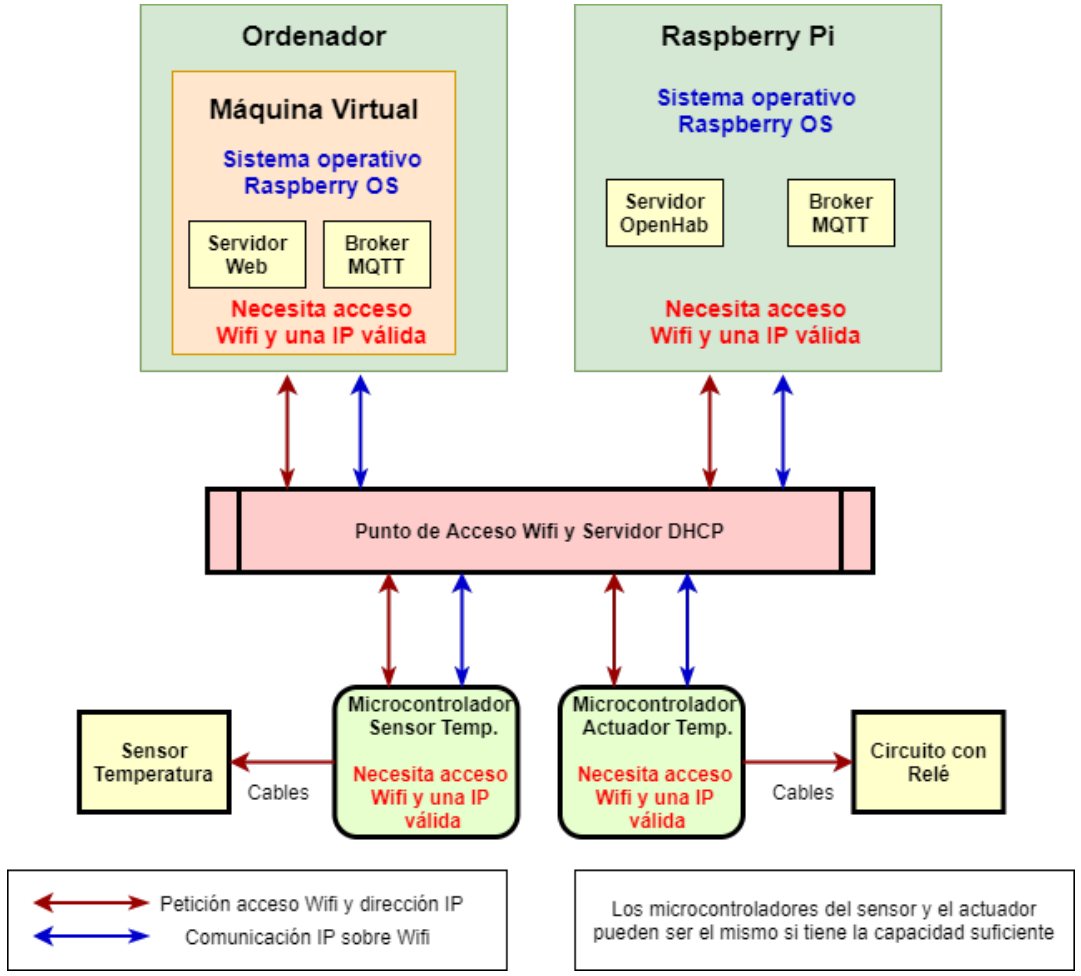
Universidad Politécnica de Madrid

Uso del IoT para construir tú mismo un hogar digital

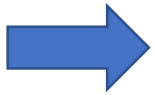
Diseño software del actuador de enchufe

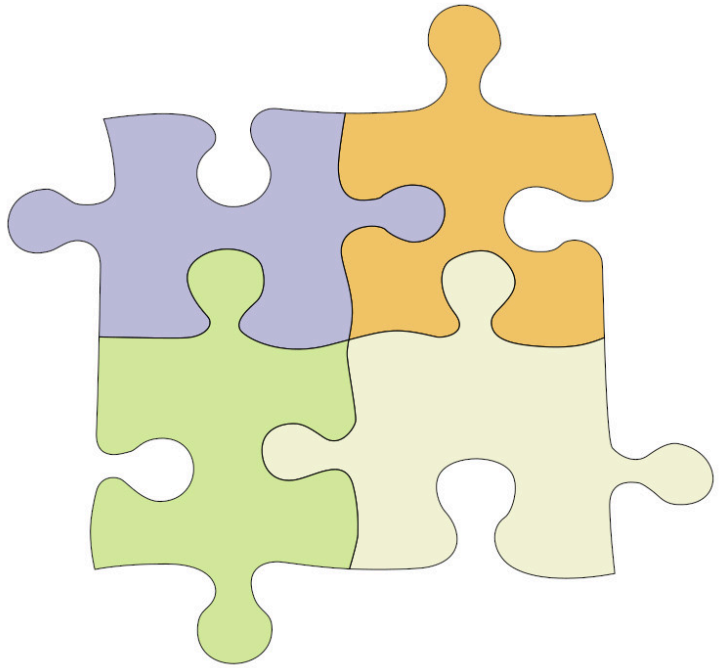
Iván Pau

Configuración inicial



Actuador de enchufe: SW





HW del actuador de enchufe

SW del actuador de enchufe

Requisitos

Navegadores Web
(cualquier cliente HTTP)

http://192.168.1.50/**enchufe/encender**

OK, actuador de enchufe encendido

http://192.168.1.50/**enchufe/apagar**

OK, actuador de enchufe apagado

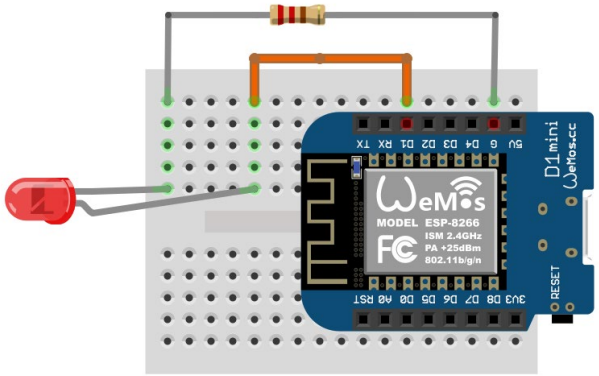
http://192.168.1.50/**enchufe/estado**

encendido

apagado

http://192.168.1.50/

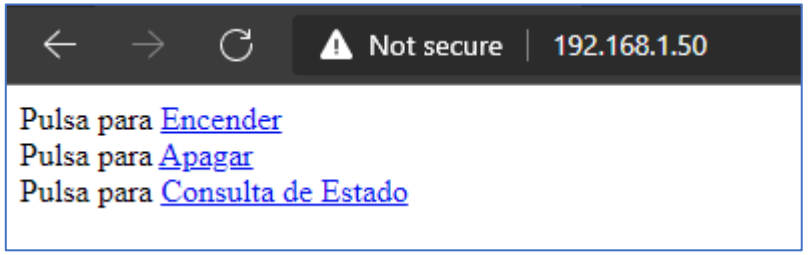
<menú con opciones>

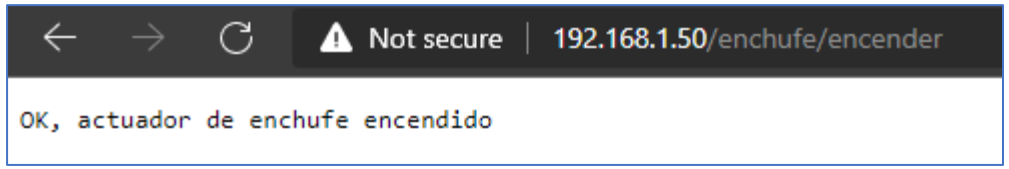


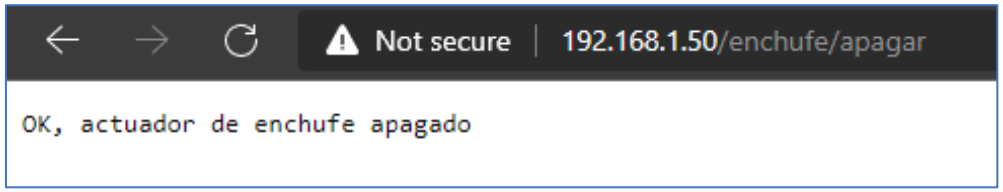
Navegadores Web
Actuador de enchufe

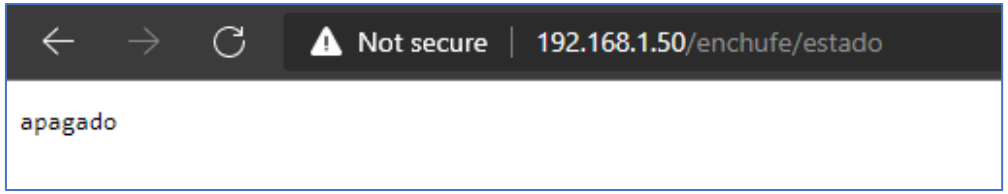
Dir. IP: 192.168.1.50

Requisitos

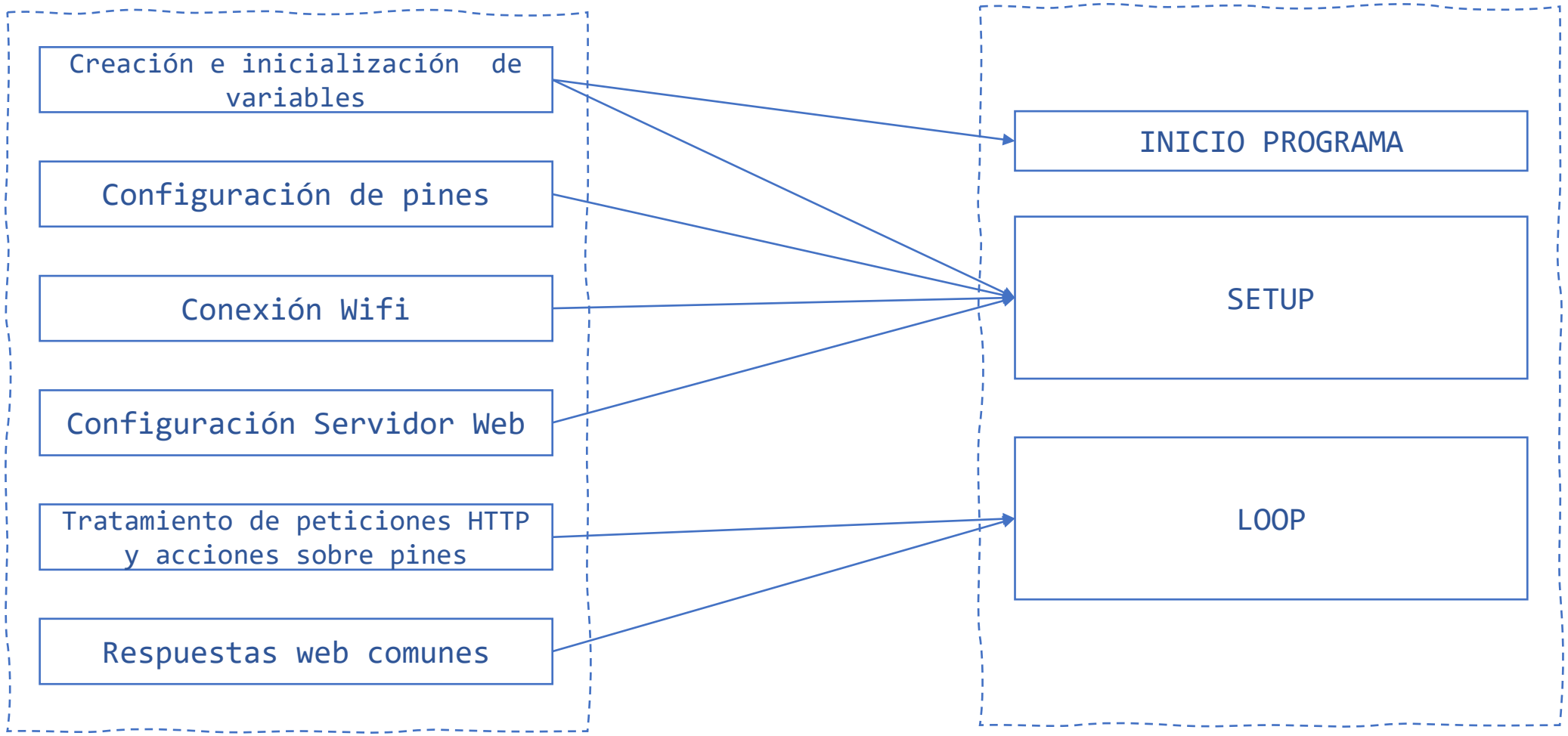

Not secure
192.168.1.50
Pulsa para Encender
Pulsa para Apagar
Pulsa para Consulta de Estado


Not secure
192.168.1.50/enchufe/encender
OK, actuador de enchufe encendido


Not secure
192.168.1.50/enchufe/apagar
OK, actuador de enchufe apagado


Not secure
192.168.1.50/enchufe/estado
apagado

Actuador enchufe



Estructura programa

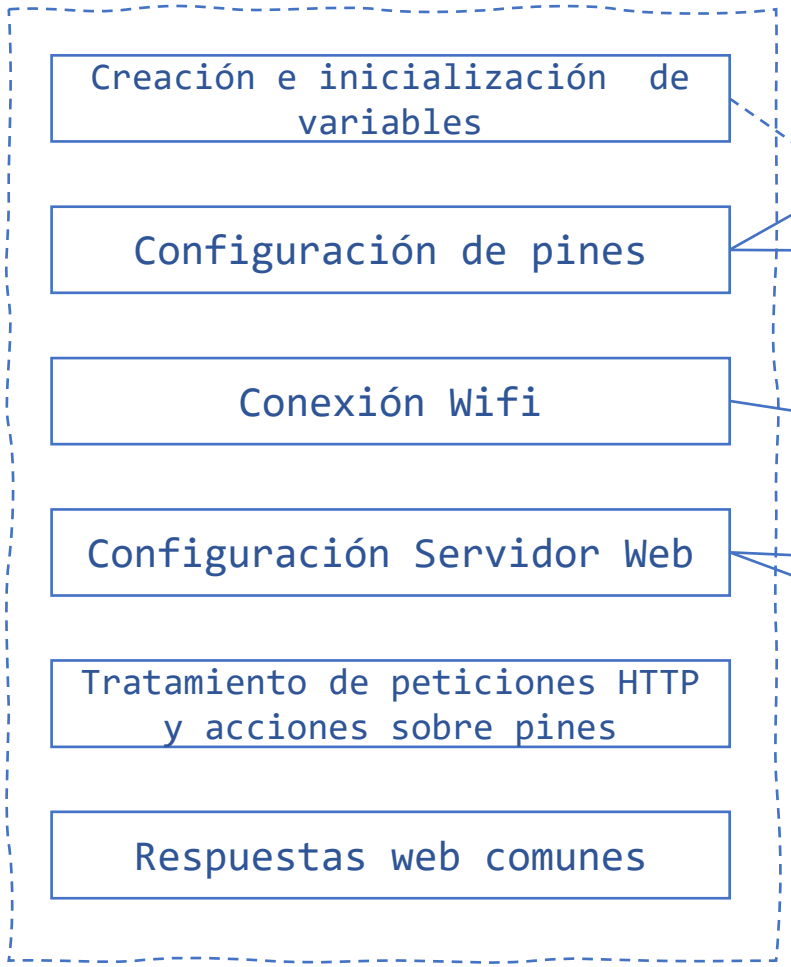
Bloques programa Arduino

Actuador enchufe

Estructura programa

```
#include <ESP8266WiFi.h>          // biblioteca wifi de esp8266
#include <ESP8266WebServer.h>     // Servidor web

// Definición de constantes
// Led interno - Solo para pruebas
#define LEDINTERNO LED_BUILTIN
#define ENCENDER LOW
#define APAGAR HIGH
// Relé
#define APAGAR_ENCHUFE LOW
#define ENCENDER_ENCHUFE HIGH
#define PINRELE_ENCHUFE D1  // Pin donde irá conectado el relé

// Parámetros de la red wifi de la casa
const char* ssid = "LabIoT1";
const char* password = "XXXXXXXX";

// IP de la ESP8266
IPAddress wifiIP(192, 168, 1, 50);
// Máscara de red
IPAddress wifiNET(255, 255, 255, 0);
// Dirección IP del encaminador
IPAddress wifiON(192, 168, 1, 1);

// Objecto servidor
ESP8266WebServer servidorWeb(80);
bool enchufeEncendido;
```

Actuador enchufe



Estructura programa

```
void setup(void) {
  // Configuración de pines
  pinMode(LEDINTERNO, OUTPUT);                // Configurar pin como salida
  digitalWrite(LEDINTERNO, APAGAR);           // Apagar led interno

  pinMode(PINRELE_ENCHUFE, OUTPUT);           // Configurar pin como salida
  digitalWrite(PINRELE_ENCHUFE, APAGAR_ENCHUFE);  // Desactivar relé
  enchufeEncendido = false;

  Serial.begin(115200);           // Establecer velocidad consola serie

  conectar_WIFI ();

  // Configuración del servidor web:
  configura_ServidorWEB ();

  // Arrancar servidor web
  servidorWeb.begin();

  Serial.println("Servidor web arrancado");
  Serial.println("Listo. Conectarse a un navegador y usar estas URLs:");
  Serial.print("Para activar: ");
  Serial.print (WiFi.localIP());
  Serial.println("/activar");
  Serial.print("Para desactivar: ");
  Serial.print (WiFi.localIP());
  Serial.println("/desactivar");

}
```

Actuador enchufe

Estructura programa

```
void configura_ServidorWEB () {

  /*
   * POSIBLE MOFICACIÓN: Con distintas URLs tendríamos posibilidad de controlar
   *  distintos pines (circuitos) de forma sencilla
   */

  servidorWeb.on("/", manejadorRaiz);
  servidorWeb.on("/enchufe/encender", manejadorEncenderEnchufe);
  servidorWeb.on("/enchufe/apagar", manejadorApagarEnchufe);
  servidorWeb.on("/enchufe/estado", manejadorEstado);
  servidorWeb.onNotFound(paginaNoEncontrada);
}
```

Actuador enchufe

Estructura programa

```
// Página inicial con el menú de opciones
void manejadorRaiz() {
  String mensaje;
  mensaje = "<!DOCTYPE HTML>\r\n<html>\r\n";
  mensaje += "<head><meta http-equiv=\"Content-Type\" content=\"text/html; charset=utf-8\">";
  mensaje += "<title>Actuador de enchufe</title></head>\r\n";
  mensaje += "<body>";
  mensaje += "Pulsa para <a href=""/enchufe/encender"">Encender</a><br>";
  mensaje += "Pulsa para <a href=""/enchufe/apagar"">Apagar</a><br>";
  mensaje += "Pulsa para <a href=""/enchufe/estado"">Consulta de Estado</a>";
  mensaje += "</body>";
  mensaje += "</html>\n";
  servidorWeb.send(200, "text/html; charset=UTF-8", mensaje);
}


void paginaNoEncontrada() {
  String mensaje = "Página no encontrada\n\n";
  mensaje += "URI: ";
  mensaje += servidorWeb.uri();
  mensaje += "\nMetodo: ";
  mensaje += (servidorWeb.method() == HTTP_GET) ? "GET" : "POST";
  mensaje += "\nArgumentos: ";
  mensaje += servidorWeb.args();
  mensaje += "\n";
  for (uint8_t i = 0; i < servidorWeb.args(); i++) {
    mensaje += " " + servidorWeb.argName(i) + ": " + servidorWeb.arg(i) + "\n";
  }
  servidorWeb.send(404, "text/plain", mensaje);
}
```

Actuador enchufe

Estructura programa

```
void loop(void) {
  // Consultar si se ha recibido una petición al servidor web
  servidorWeb.handleClient();

  // Control de pulsador (para otros actuadores)
  // comprobar_pulsacion ();

}

void comprobar_pulsacion () {
  // Lectura del pin de control del pulsados
  // Filtrar por software los rebotes del pulsador
  // Si se ha activado el pulsador actualizar los pines y variables de estado
}
```

Actuador enchufe

Estructura programa

```
void manejadorEstado() {
  if (enchufeEncendido) {
    servidorWeb.send(200, "text/plain", "encendido");
  } else {
    servidorWeb.send(200, "text/plain", "apagado");
  }
}

void manejadorApagarEnchufe() {
  digitalWrite(LEDINTERNO, APAGAR);           // Apagar led interno
  digitalWrite(PINRELE_ENCHUFE, APAGAR_ENCHUFE);  // Desactivar relé
  enchufeEncendido = false;
  servidorWeb.send(200, "text/plain", "OK, actuador de enchufe apagado");
  Serial.println("Relé desactivado");
}

void manejadorEncenderEnchufe() {
  digitalWrite(LEDINTERNO, ENCENDER);       // Encender led interno
  digitalWrite(PINRELE_ENCHUFE, ENCENDER_ENCHUFE);   // Activar relé
  enchufeEncendido = true;
  servidorWeb.send(200, "text/plain", "OK, actuador de enchufe encendido");
  Serial.println("Relé activado");
}
```

Retos: diseña tus actuadores

Reto: Diseñar dos actuadores

Actuador 1: Actuador de interruptor

- La parte Hardware ya comentada en la sesion de diseño hardware de sensor de enchufe
- A nivel software también reponderá a peticiones HTTP al igual que el actuador de enchufe (a distintas URLs y con distintas respuestas)
- Es imprescindible que se usen las URLs indicadas a continuación y sus respuestas (ya que serán interpretadas en el módulo 6)
- Nuevo en el software: hay que usar un pin de E/S para lectura y controlar cuando se ha realizado una pulsación (controlando rebotes)

Actuador 2: actuador de persiana

- La parte Hardware ya comentada en la sesion de diseño hardware de sensor de enchufe
- A nivel software también reponderá a peticiones HTTP al igual que el actuador de enchufe (a distintas URLs y con distintas respuestas)
- Es imprescindible que se usen las URLs indicadas posteriormente y sus respuestas (ya que serán interpretadas en el módulo 6)
- Nuevo en el software: hay que usar nuevos pines de E/S ya que en total hay que controlar dos pulsadores (uno por relé) y dos relés (uno por fase del motor: subida o bajada)

Actuador Interruptor

SOFTWARE

Navegadores Web
(cualquier cliente HTTP)

http://192.168.1.51/interruptor/encender

OK, interruptor encendido

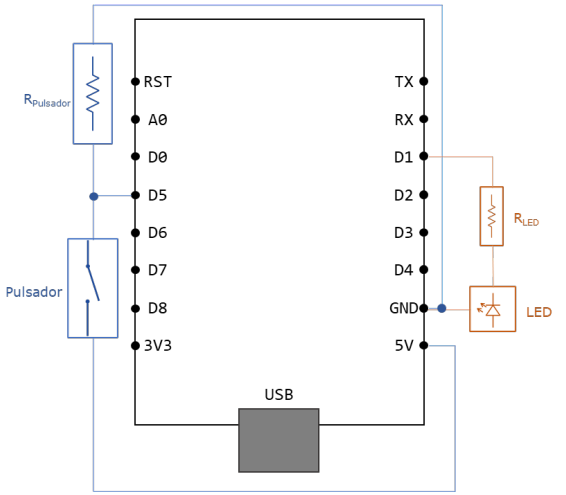
http://192.168.1.51/interruptor/apagar

OK, interruptor apagado

http://192.168.1.51/interruptor/estado

encendido

apagado



Actuador de interruptor

Actuador interruptor

POLITÉCNICA
Universidad Politécnica de Madrid

```
void loop(void) {
  // Consultar si se ha recibido una petición al servidor web
  servidorWeb.handleClient();

  // Control de pulsador (para otros actuadores)
  // comprobar_pulsacion ();

}

void comprobar_pulsacion () {
  // Lectura del pin de control del pulsados
  // Filtrar por software los rebotes del pulsador
  // Si se ha activado el pulsador actualizar los pines y variables de estado
}
```

Actuador persiana

SOFTWARE

Navegadores Web
(cualquier cliente HTTP)

http://192.168.1.52/persiana/subir

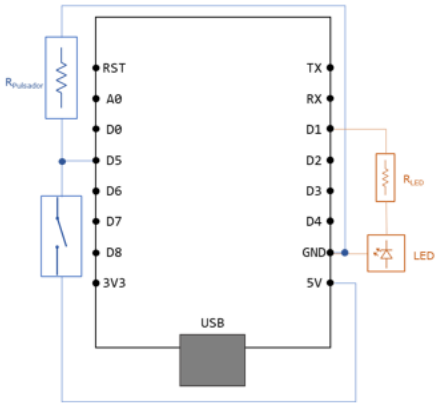
SUBIENDO LA PERSIANA!

http://192.168.1.52/persiana/bajar

BAJANDO LA PERSIANA!

http://192.168.1.52/persiana/parar

MOTOR PARADO!



X 2

Actuador de persiana

Actuador persiana


```
void loop(void) {
  // Consultar si se ha recibido una petición al servidor web
  servidorWeb.handleClient();

  // Control de pulsador (para otros actuadores)
  // comprobar_pulsacion ();

}

void comprobar_pulsacion () {
  // Lectura del pin de control del pulsados
  // Filtrar por software los rebotes del pulsador
  // Si se ha activado el pulsador actualizar los pines y variables de estado
}
```