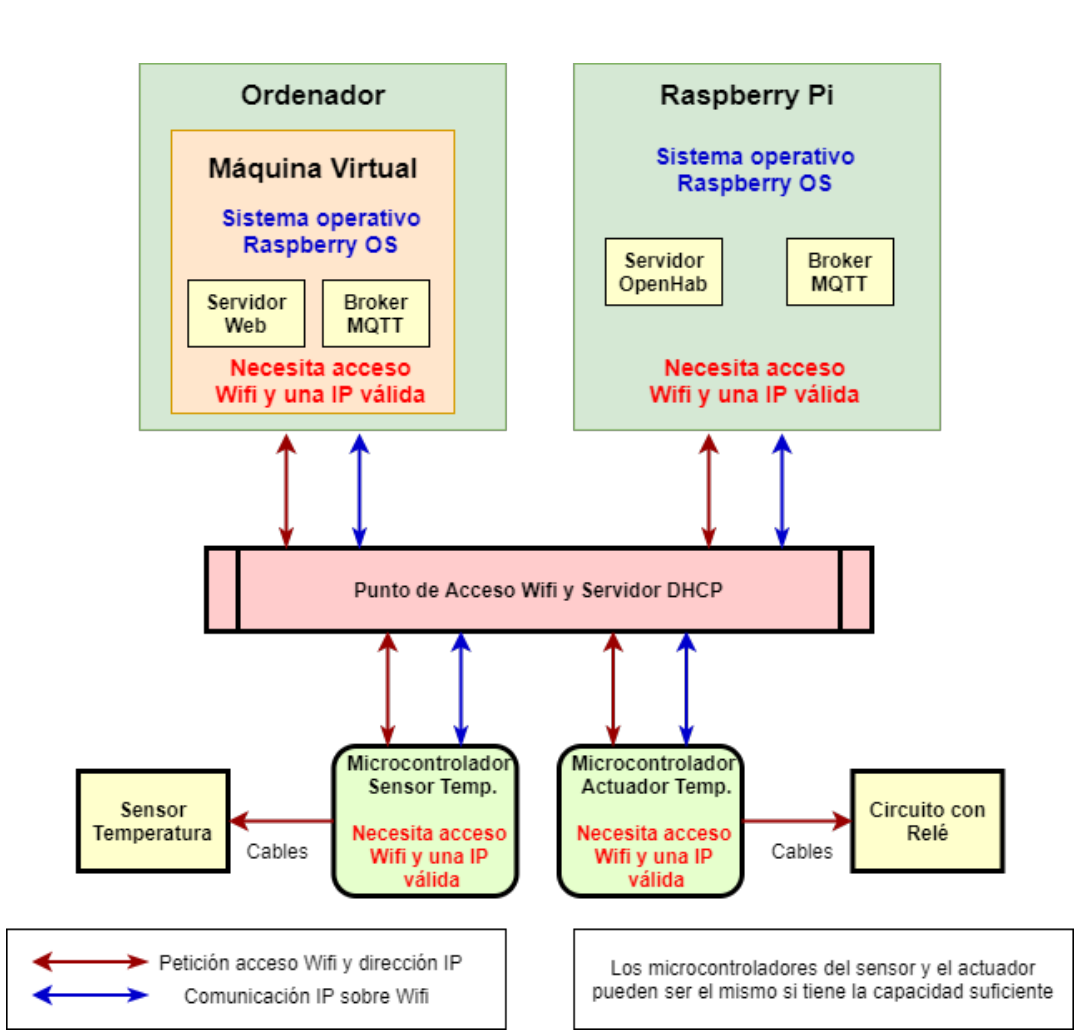
Universidad Politécnica de Madrid

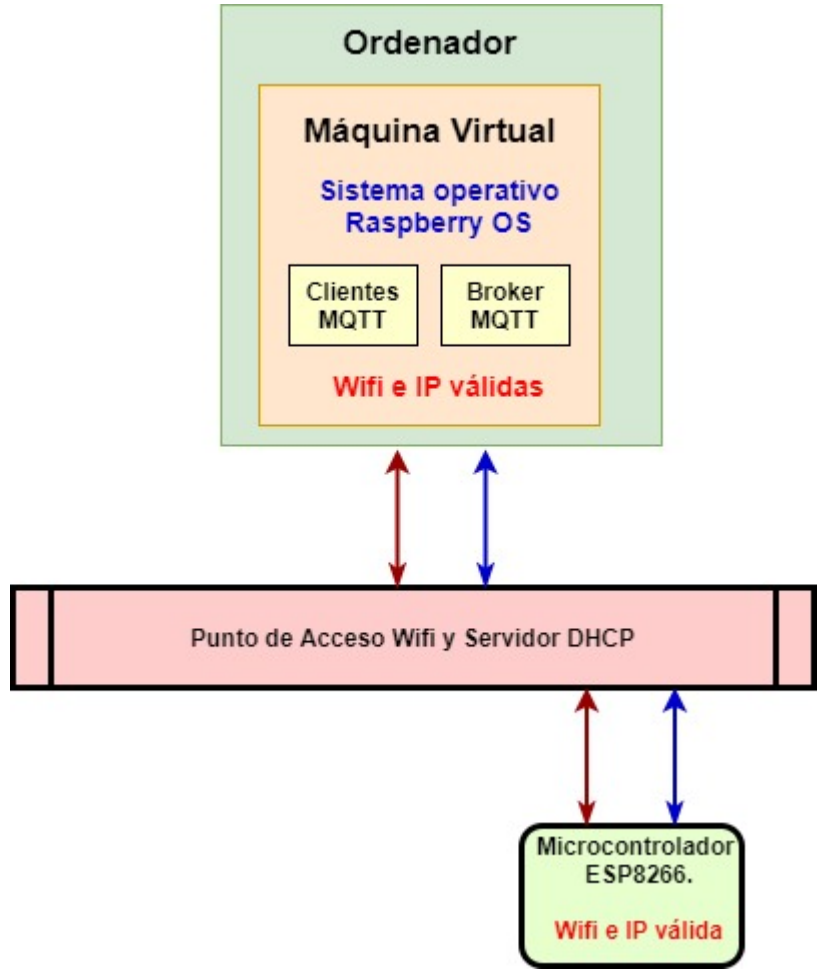
Uso del IoT para construir tú mismo un hogar digital

Envío de mensajes MQTT mediante ESP8266

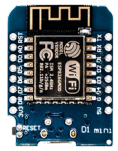
Iván Pau

Configuración inicial

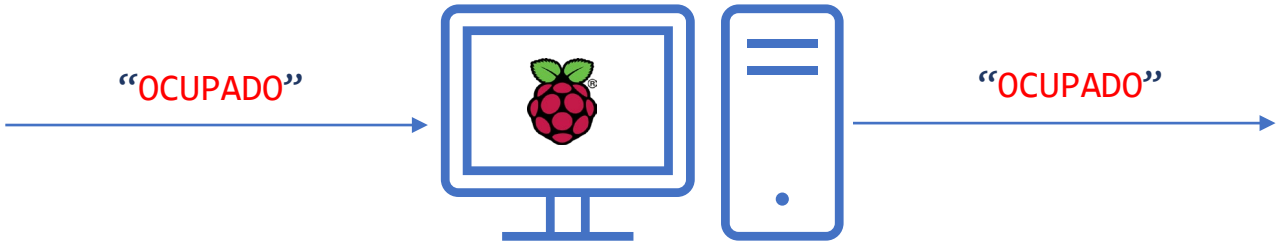


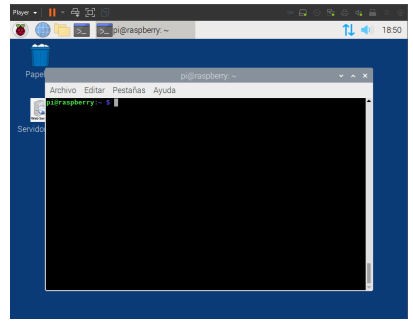


Escenario

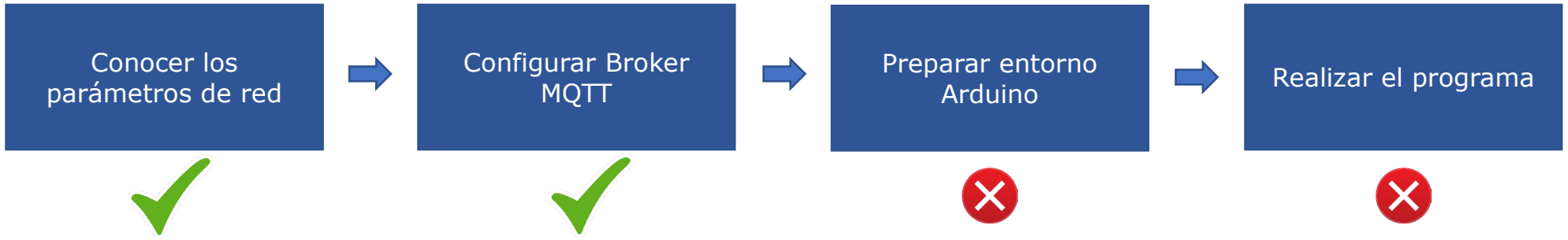


Publicando en:
casa/salon/presencia





Suscrito a:
casa/salon/presencia



Biblioteca PubSubClient

pubsubclient.knolleary.net

Arduino Client for MQTT | API Documentation

Arduino Client for MQTT

This library provides a client for doing simple publish/subscribe messaging with a server that supports MQTT

For more information about MQTT, visit MQTT.org.

Download

The latest version of the library can be downloaded from GitHub.

Documentation

The library comes with a number of example sketches. See `File > Examples > PubSubClient` within the Arduino application.

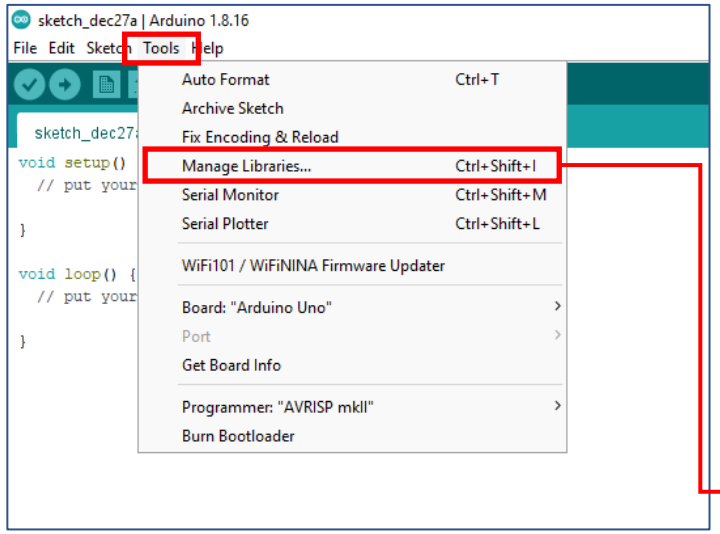
Full API Documentation is available.

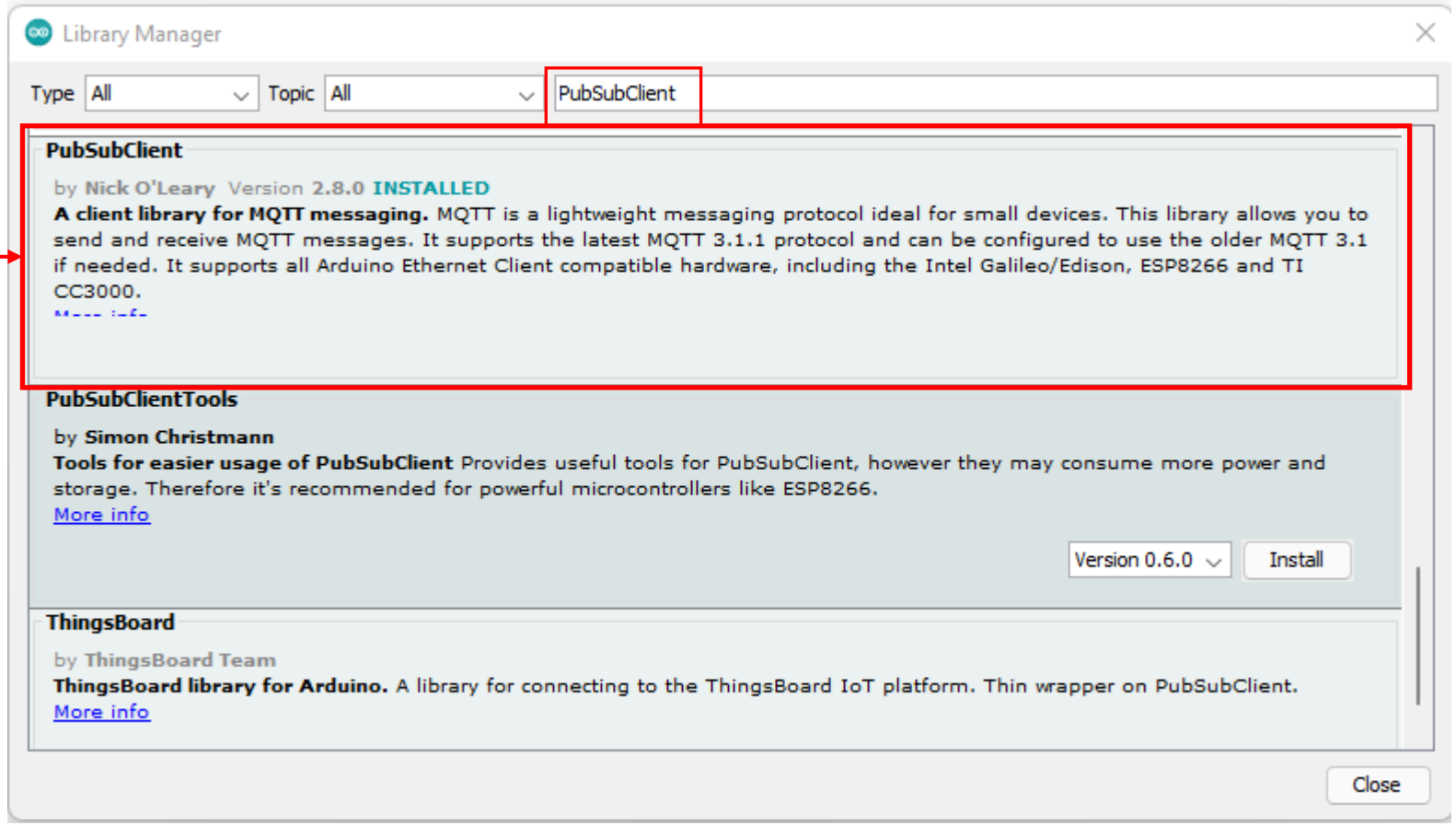
Author

- Nick O'Leary - @knolleary

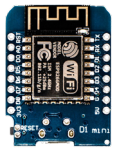
License

Preparar el entorno Arduino



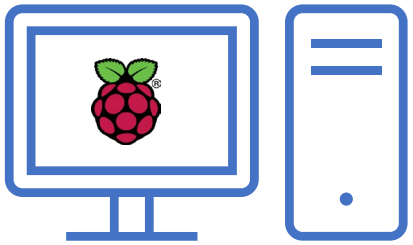


Escenario

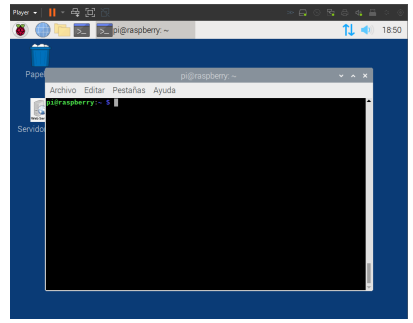


Publicando en:
casa/salon/presencia

"OCUPADO"



"OCUPADO"

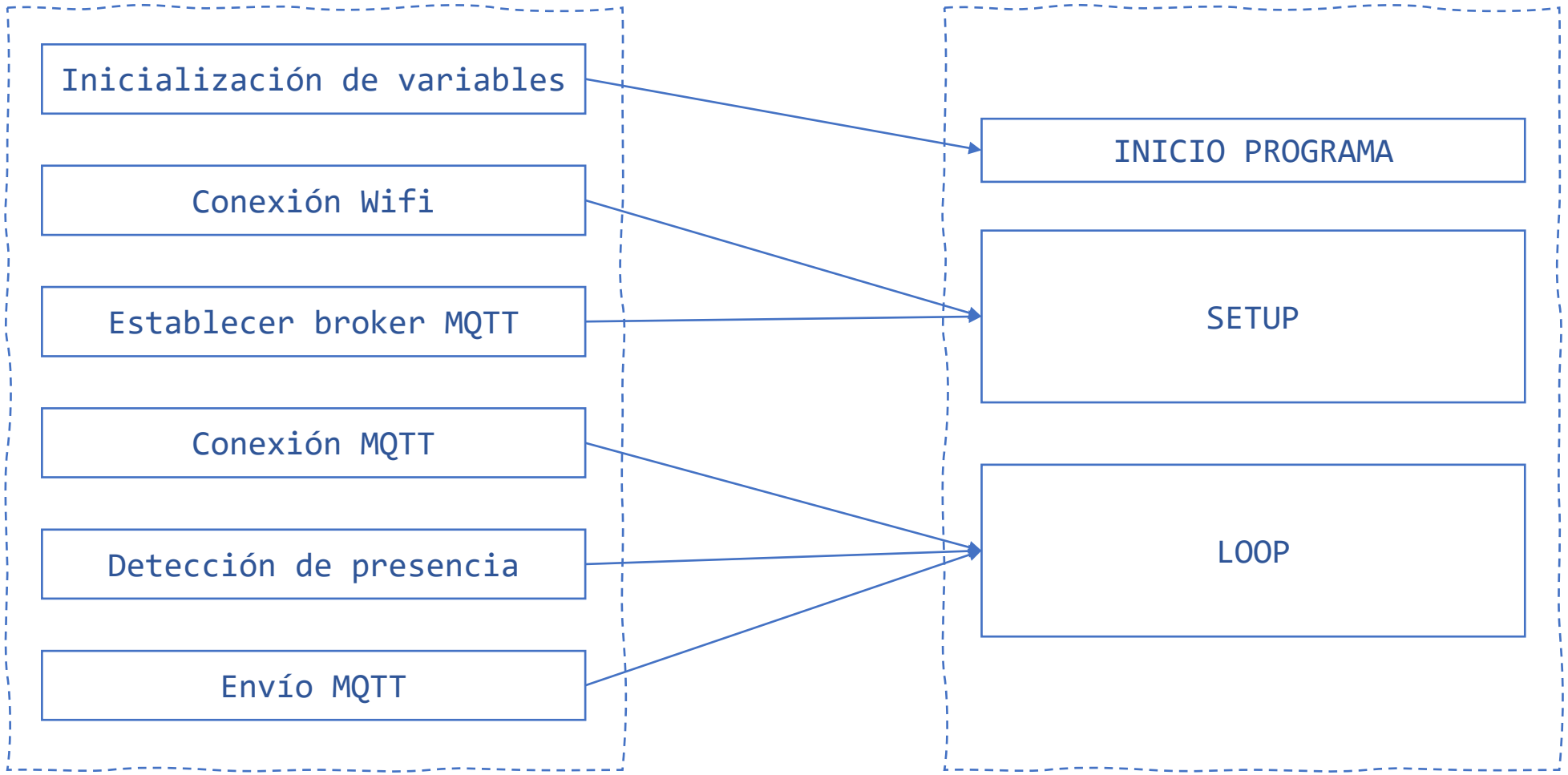


Suscrito a:
casa/salon/presencia

Conocer los parámetros de red → Configurar Broker MQTT → Preparar entorno Arduino → Realizar el programa

✓ ✓ ✓ ✗

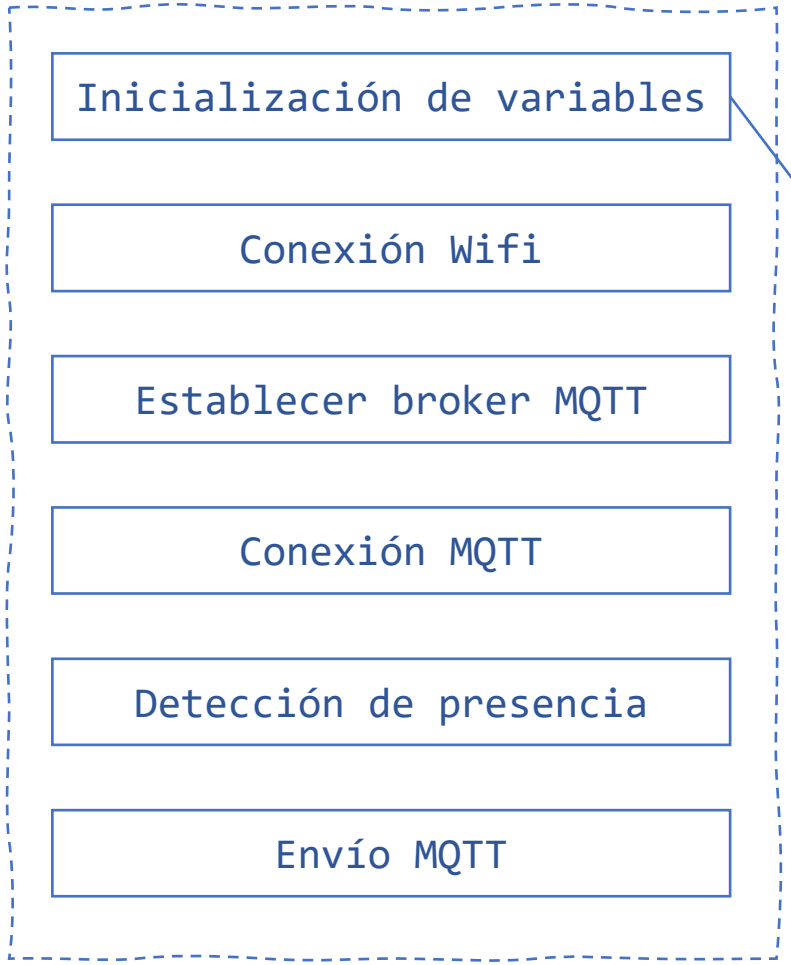
Programa de envío



Estructura programa

Bloques programa Arduino

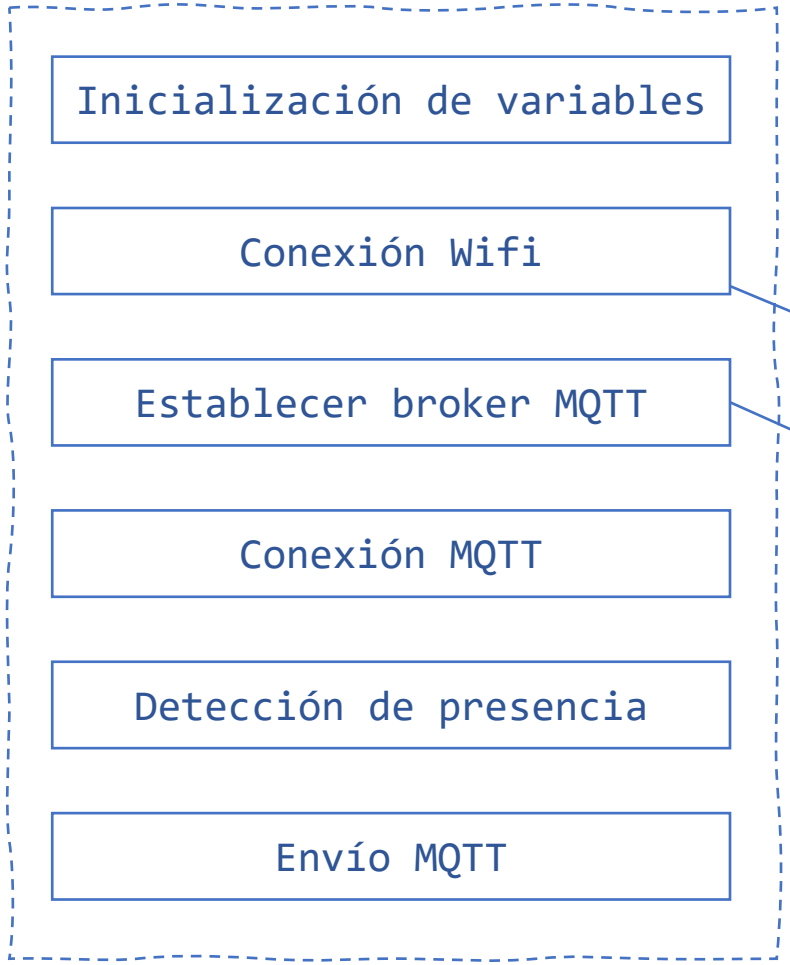
Programa de envío



Estructura programa

```
#include <ESP8266WiFi.h> // Biblioteca para esp8266
#include <PubSubClient.h> // Biblioteca para el cliente MQTT

// Constantes programa
#define MSG_BUFFER_SIZE  (50)
#define TIEMPO_ENTRE_MENSAJES (2000)
#define NUMERO_REINTENTOS (5)
#define TIEMPO_RECONEXION (5000)
#define PRESENCIA "OCUPADO"
#define NO_PRESENCIA "VACIO"
#define TOPIC_PUBLICACION "casa/salon/presencia"

// Parametros de conexión Wifi
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);
//IP del broker
IPAddress mqtt_server (192, 168, 1, 145);

// Variables globales
WiFiClient clienteWIFI;
PubSubClient clienteMQTT(clienteWIFI);
char mensaje[MSG_BUFFER_SIZE];
long tiempoUltimoMensaje = 0;
```

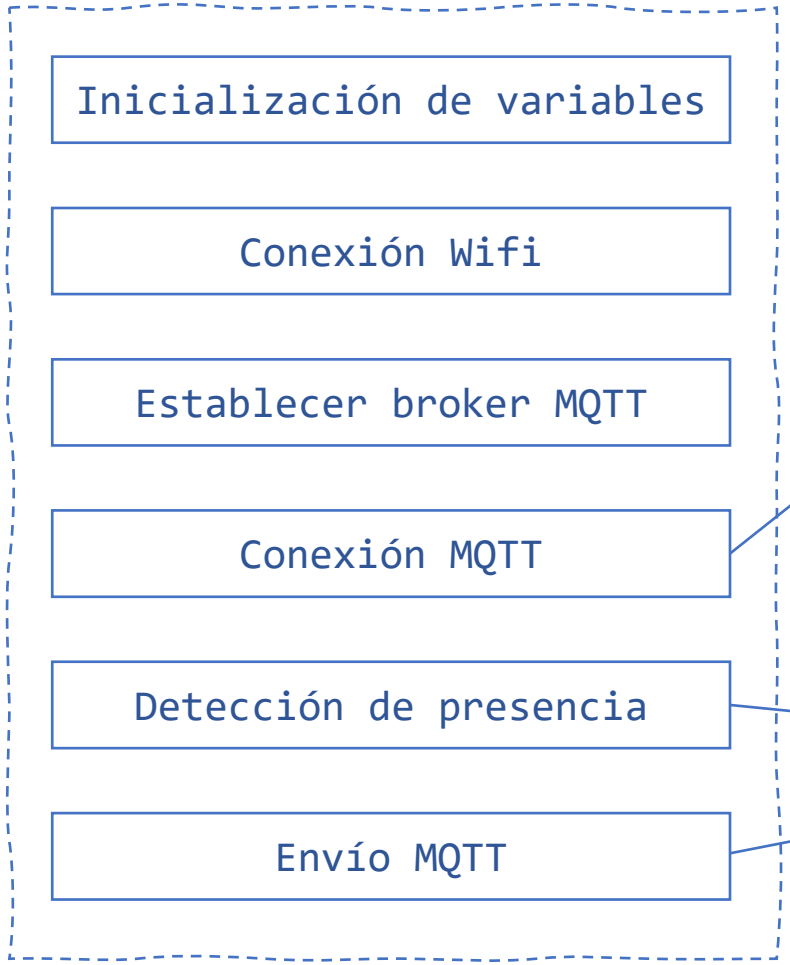
Programa de envío



Estructura programa

```
void setup() {
  // Puerto serie
  Serial.begin(115200);

  // Se habilita la conexión Wifi
  conectar_WIFI();

  // Se establece la dirección del Broker para MQTT
  clienteMQTT.setServer(mqtt_server, 1883);

  // Se inicializa el generador de pseudoaleatoriedad
  randomSeed(micros());

}
```

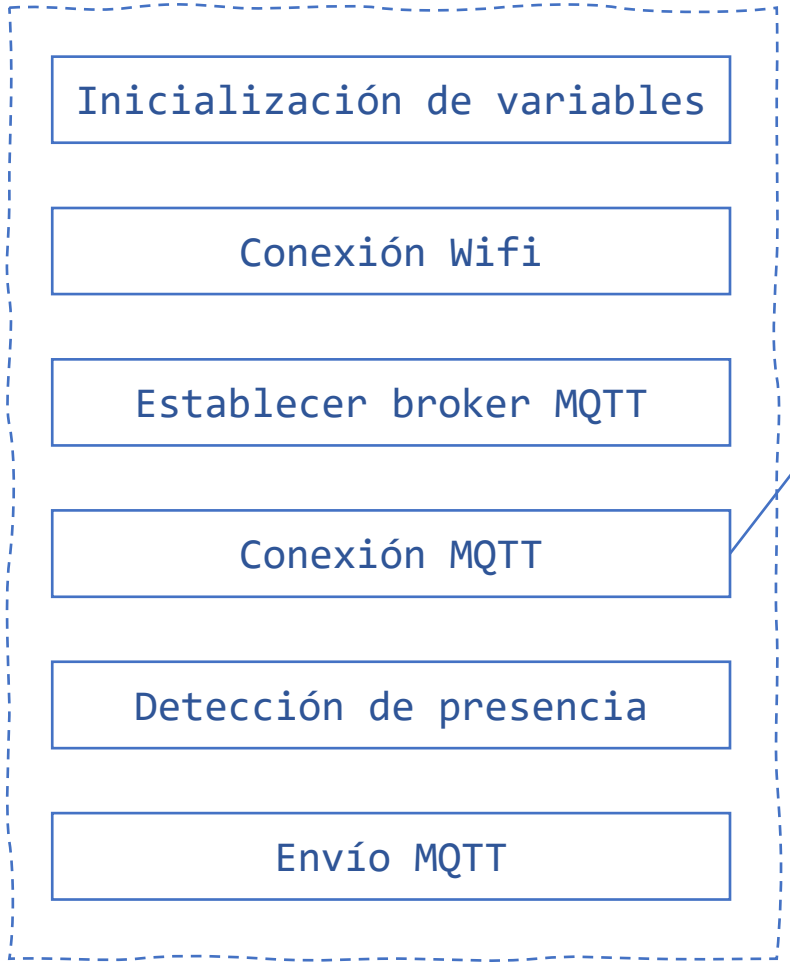
Programa de envío




```
void loop() {

  // Se asegura de que existe conexión con el broker
  if (!clienteMQTT.connected()) {
    conectar_MQTT();
  }

  // Control de envíos espaciados en el tiempo (TIEMPO_ENTRE_MENSAJES)
  // Se obtiene una referencia de tiempo actual
  long now = millis();
  if (now - tiempoUltimoMensaje > TIEMPO_ENTRE_MENSAJES) {
    // Se actualiza el último mensaje a la referencia de tiempo actual
    tiempoUltimoMensaje = now;
    detecta_presencia ();
    mqtt_EnviarMensaje();
   }

}
```

Estructura programa

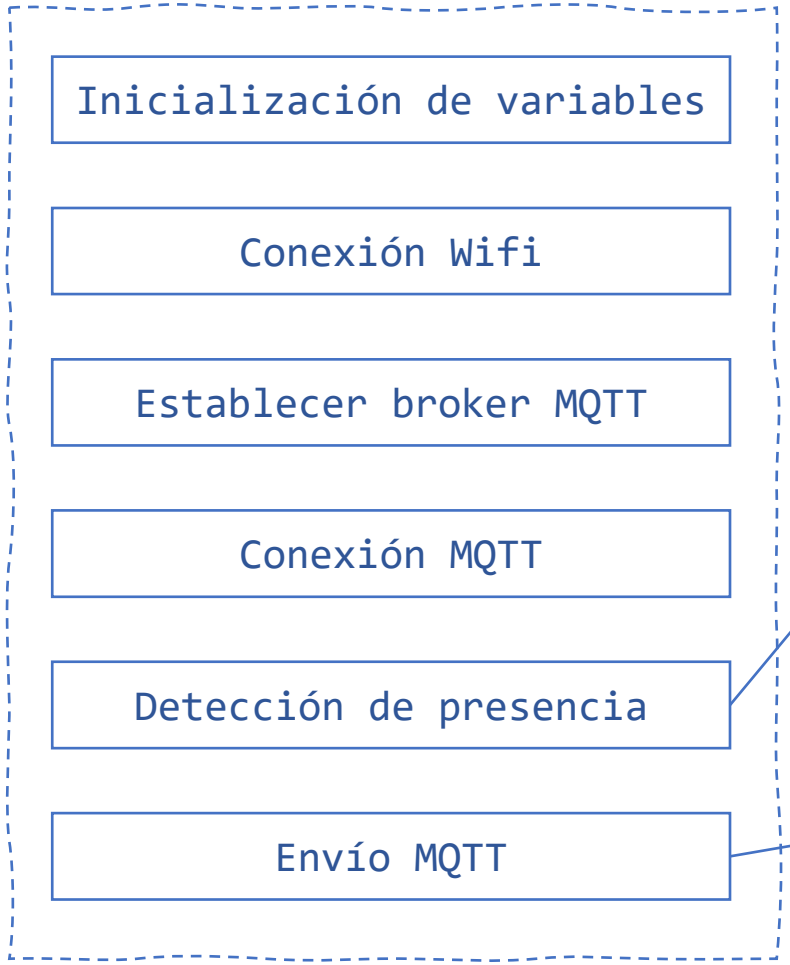
Programa de envío



Estructura programa

```
void conectar_MQTT() {

  for (int i=0 ; i<NUMERO_REINTENTOS ; i++) {

    // Creación de un ID de cliente
    String clientId = "esp32Client-1";

    // Intento de conexión
    if (clienteMQTT.connect(clientId.c_str())) {
      Serial.println("Conectado");
      i=NUMERO_REINTENTOS;

    } else {
      Serial.print("Error al conectar. Resultado: ");
      Serial.print(clienteMQTT.state());
      Serial.println(". Se volverá a intentar en 5 segundos. ");

      delay(TIEMPO_RECONEXION);
    }
  }
}
```

Programa de envío



Estructura programa

```
void detecta_presencia () {
   long numeroAleatorio = random(0, 2);
   Serial.print("Numero seleccionado: ");
   Serial.println (numeroAleatorio);
   if (numeroAleatorio) {
    snprintf (mensaje, MSG_BUFFER_SIZE, PRESENCIA);
   } else {
    snprintf (mensaje, MSG_BUFFER_SIZE, NO_PRESENCIA);
   }
   Serial.print("Mensaje a publicar: ");
   Serial.println(mensaje);
}

void mqtt_EnviarMensaje(){
    clienteMQTT.publish(TOPIC_PUBLICACION, mensaje);
}
```