**Descripción del código servo**

Esta placa ESP8266 tiene conectado un servo SG90, cuya propósito es comportarse como una hipotética barrera de garaje.

El código comienza con la inclusión de librerías y subcódigos. Aquí se incluyen:

* ESP8266WiFi.h : Permite la conexión de la placa a la red WiFi.
* PubSubClient.h : Permite el uso de MQTT.
* ArduinoJson.h : Facilita el formateo de mensajes en estructura JSON.
* Servo.h : Facilita el control de servos.
* “actualiza\_dual.h” : Contiene las instrucciones necesarias para realizar la actualización FOTA.
* “pulsador\_int.h” : Contiene las instrucciones necesarias para usar el botón Flash de la placa como un botón genérico.

Posteriormente, se definen e inicializan las variables y parámetros necesarios, desde las credenciales para WiFi o MQTT, hasta las relacionadas con el servo.

**Función callback**

Esta se encarga de responder ante los mensajes recibidos por los topics MQTT a los que la placa se encuentra suscritos (se comenta más adelante). Se observa el topic concreto del mensaje recibido y, en función del mismo, se realizan las acciones pertinentes:

* Si el topic es “II7/Entrada/FOTA”: Se llama a la función asociada a la actualización FOTA (dentro de “actualiza\_dual.h”) y se actualiza el momento de la última actualización al actual (mediante la función millis()).
* Si el topic es “II7/Entrada/config”: Se actualiza la frecuencia de actualización FOTA a la indicada en el mensaje.
* Si el topic es “II7/Entrada/BarreraCMD”: Se analiza el mensaje recibido.
  + Si el mensaje es “subir”: Se sube la “barrera” a 90º y se publica el estado “subido” en el topic “II7/Entrada/BarreraEstado”.
  + Si el mensaje es “bajar”: Se baja la “barrera” hasta 0º progresivamente (10º menos cada 0,5 s) y se publica el estado “bajado” en el topic “II7/Entrada/BarreraEstado”.

En caso de que ocurra algún error al analizar el topic o el mensaje, o que sencillamente el mensaje no corresponda con los descritos, no se realizará ninguna de las acciones comentadas, y el programa seguirá adelante.

**Función reconnect**

Su función consiste en intentar conectarse al broker MQTT indicado al inicio del código. Si no lo consigue, lo vuelve a intentar en bucle.

Si logra conectarse, procede a suscribirse a los topics indicados en el apartado anterior. Además, publica un mensaje en “II7/Entrada/conexion” indicando que se ha conectado.

**Función setup \_MQTT**

Inicializa el servicio MQTT, indicando el servidor y puerto empleados y la función de callback.

**Función setup**

Es una de las funciones principales de Arduino, en ella se incluyen las acciones a realizar solo al comienzo del programa.

En este caso, se comienza realizando la conexión WiFi mediante las credenciales incluidas al principio del código. Mientras la placa no logre conectarse, el programa no avanzará.

Posteriormente se realiza un primer intento de actualización FOTA, con su correspondiente actualización de la variable asociada al momento del último intento (“lastFOTA”).

A continuación se procede con el intento de conexión al broker MQTT, llamando a la función anterior.

Para terminar, se configura el servo indicando el pin al que se encuentra conectado y los anchos de pulso adecuados para lograr que el motor empleado abarque su recorrido completo (180º). También se llama a la función que permite el funcionamiento correcto del pulsador (incluida en “pulsador\_int.h”.

**Función loop**

Es la otra función principal y predeterminada de Arduino, y en ella se incluyen las acciones que la placa debe realizar en bucle.

En este caso, se comienza comprobando si la placa se ha conectado correctamente al servidor MQTT. En caso contrario, se llama a la función “reconnect” hasta lograrlo.

Posteriormente, se comprueba si el tiempo transcurrido desde el último intento de actualización FOTA es superior al umbral indicado al inicio (o de manera externa mediante MQTT). En caso afirmativo, se vuelve a intentar y se actualiza “lastFOTA”.

A continuación, se comprueba si ha ocurrido un evento de pulsador, es decir, si se ha pulsado o soltado. En caso afirmativo, se comprueba cual de los dos casos es.

* Si el botón se ha pulsado, se guarda en una variable el instante en el que ha ocurrido.
* Si el botón se ha soltado, se comprueba el tiempo transcurrido desde que se pulsó.
  + Si el tiempo es superior a 3 s, se intenta una actualización FOTA y se actualiza “lastFOTA”.
  + En caso contrario (tiempo inferior a 3 s), se interpreta como que un vehículo desea acceder al garaje, y se publica en el topic “II7/Entrada/Pulsador” el mensaje “1”, para así indicárselo al sistema, que se encargará de autorizar o no la subida de la barrera.