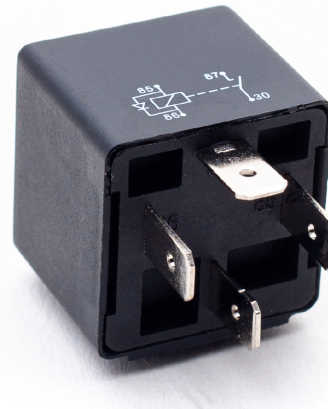


INTRODUCTION TO RELAY



What is our GOAL for this CLASS?

In this class, we were introduced to Relay and we learned to operate AC appliances with Relay as per Google Assistant instructions.

What did we ACHIEVE in the class TODAY?

- We were introduced to Relay
- We learned about IFTTT.

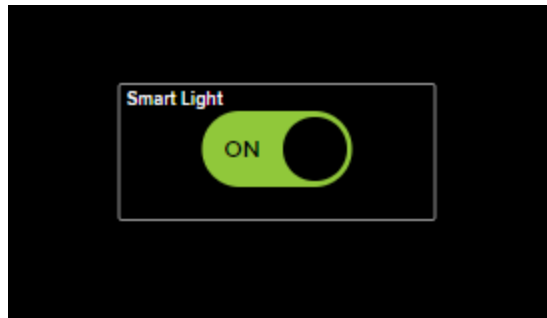
Which CONCEPTS/ CODING BLOCKS did we cover today?

- We learned how a button works.
 - When we toggle the level to the ON position, the contact closes completes the circuit and allows current to flow through the switch.
 - When we toggle the level to the OFF position, the contact opens, restricting current to flow through the switch.
- We learned about Relay
 - A relay is an electromagnetic switch operated by a relatively small electric current that can turn on or off a much larger electric current.
 - Electromagnet: A coil of wire that becomes a temporary magnet when electricity flows through it.
 - In relay two circuits are there, one for small electric current and the second for large current
- We learned how to make a circuit with Relay.
- We learned about IFTTT
 - IFTTT stands for 'If this, then that.' It is an open-source service that allows users

to set up responses to events according to their preferences. We can create chains of conditional statements which are known as Applets. When the user says a specific phrase the data will be sent to Adafruit IO

How did we DO the activities?

1. Gather the material from the IoT kit:
 - 1 x ESP32
 - 1 x USB Cable
 - 1 x Breadboard
 - 4 x Jumper wires
 - 1 x Relay
 - 1 x Bulb, Lamp, Mosquito Repellent Machine
2. Connections for **Circuit** Diagram
 - **Relay VCC(Red) pin:** Connect with 5 V PIN of the ESP32
 - **Relay GND(Black) pin:** Connect with GND of the ESP32
 - **Relay Input(Yellow) pin:** Connect with GPIO PIN 23
3. Create a feed on the Adafruit cloud server. Open Adafruit account
 - Click on Sign in
 - Click on IO
 - Go to Dashboards
 - Click on New Dashboard
 - Write the Name (Lamp)
 - Write Description if needed. Click on Create
 - Click on Create New Block
 - Select the Toggle Button.
 - Enter new feed name (Light)
 - Note: This same to be use in program
 - Click the "Create" button to proceed further.
 - Select the Feed which you have just created and then click on Next Step
 - Select the "default" values for "Toggle" button. Write the name which need to be displayed (Smart Light)
 - Click on "Create block"
 - On creating Toggle switch block, your dashboard will be looks like below

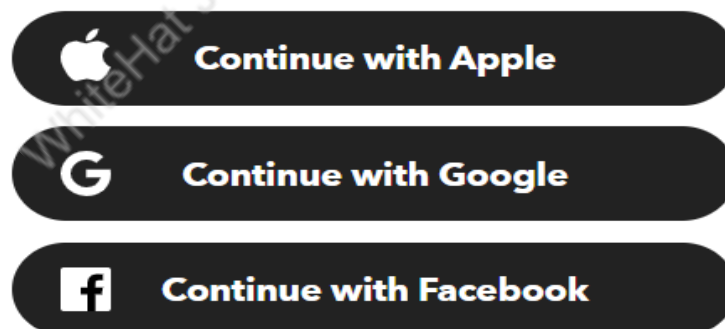


4. Connecting to Google Assistant through IFTTT to control the onboard LAMP/LED/Device of ESP32 through voice commands.
5. Open the [link](#) and click on **Get started**



6. Click the '**sign up**'. The window will pop up. Enter your email address and password to start working in IFTTT.

Get started with IFTTT



Or use your email to [sign up](#) or [log in](#)

7. After creating an account, we will be directed to the page where we will create our applet. Click on 'Create'.

IFTTT

Applets for Hue lights

My Applets

Explore

Developers ▾

Create

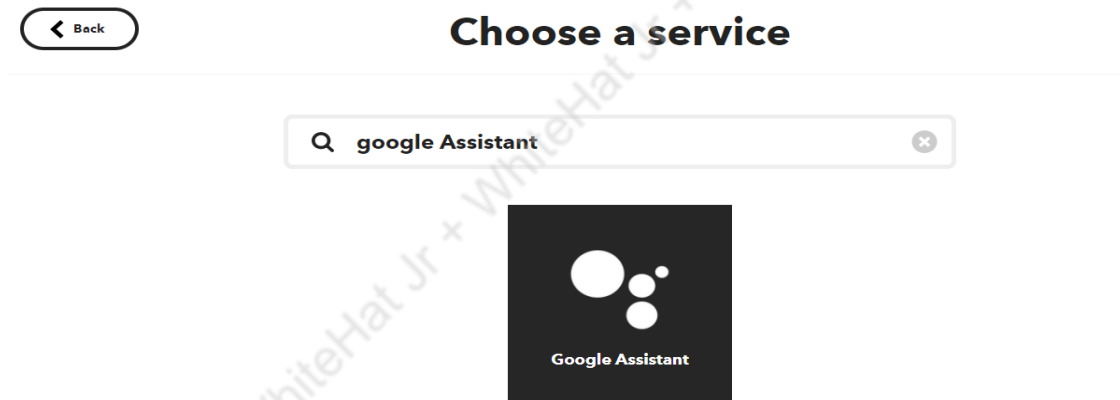
Upgrade



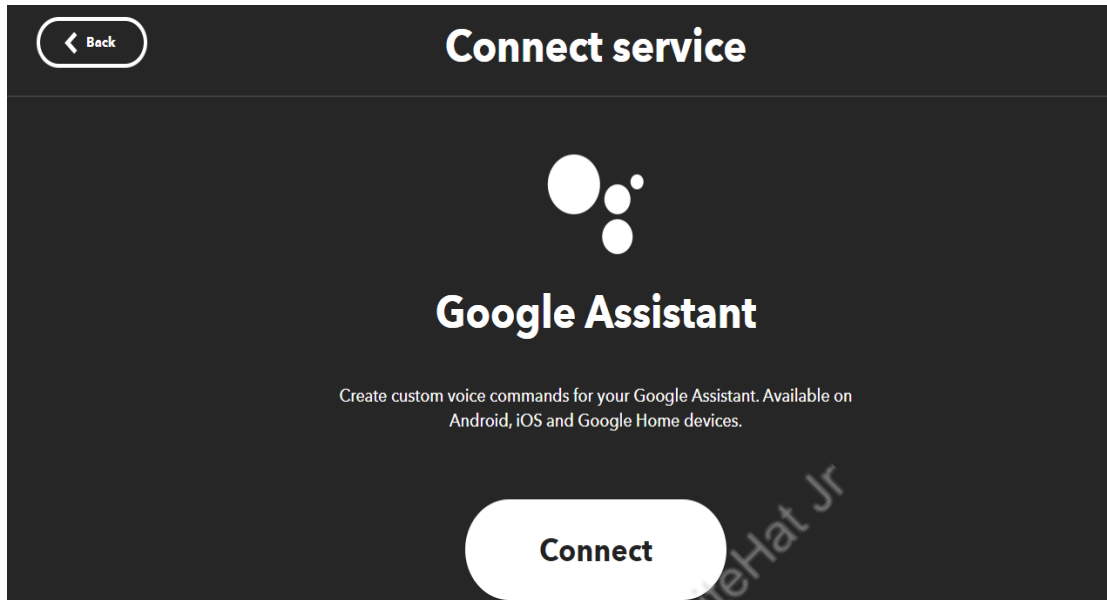
8. Click the Add button in the If This section.



9. Write down 'Google Assistant' in the search option and its icon will appear:



10. Click on "Connect"



11. Click the icon and choose your trigger. Use '**Say a simple phrase**' as your trigger.
 - Click the '**Connect**' button to establish the connection.

12. write a program

- The top of the program has includes. We will write all supporting header files and libraries
- **include keyword** is used to import libraries in embedded language
- **WiFi.h**: WiFi library will be able to answer all HTTP request
- **Adafruit BMP085.h** Adafruit **BMP085** is used to connect the BMP180 sensor with the Adafruit dashboard.
- Adafruit supports a protocol called **MQTT**, or **message queue telemetry transport**, for communication with devices. To send and receive feed data,
- **Adafruit_MQTT** header file tells about sending and receiving packets.
- **Adafruit_MQTT_Client** allows publishing to a feed.

```
#include <WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
```

13. Connect with ESP32 with the WiFi. For that, use SSID(Wi-Fi credentials i.e WiFi name and WiFi Password)
 - **WLAN_SSID**, **WLAN_PASS** are the variables that are used to save WiFi credentials. Set the **SSID** and **password**

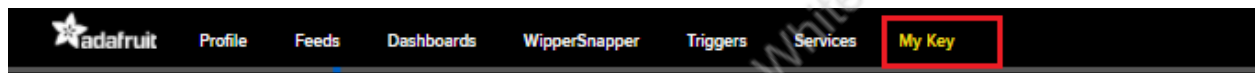
```
#define WLAN_SSID      "WR3005N3-757E"
#define WLAN_PASS      "70029949"
```

14. Adafruit Setup and Authentication

- Set the Adafruit AIO_SERVER , AIO_SERVERPORT, AIO_USERNAME, AIO_KEY

```
#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                // use 8883 for SSL
#define AIO_USERNAME    "Tamtap"
#define AIO_KEY         "aio_ozIw67GZ2KKiwjN7Uvz5HDQeiuY0"
```

- To get AIO_USERNAME & AIO_KEY go to adafruit, Click on My Key



- Note down IO_USERNAME & IO_KEY, paste the same in the program where AIO_USERNAME & AIO_KEY is mentioned.

YOUR ADAFRUIT IO KEY ✕

Your Adafruit IO Key should be kept in a safe place and treated with the same care as your Adafruit username and password. People who have access to your Adafruit IO Key can view all of your data, create new feeds for your account, and manipulate your active feeds.

If you need to regenerate a new Adafruit IO Key, all of your existing programs and scripts will need to be manually changed to the new key.

Username

Active Key REGENERATE KEY

[Hide Code Samples](#)

Arduino

```
#define IO_USERNAME    "Tamtap"
#define IO_KEY         "aio_ozIw67GZ2KKiwjN7Uvz5HDQeiuY0"
```

Linux Shell

```
export IO_USERNAME="Tamtap"
export IO_KEY="aio_ozIw67GZ2KKiwjN7Uvz5HDQeiuY0"
```

Scripting

```
ADAFRUIT_IO_USERNAME = "Tamtap"
ADAFRUIT_IO_KEY = "aio_ozIw67GZ2KKiwjN7Uvz5HDQeiuY0"
```

- Define relay at GPIO pin 23

- Setup the MQTT client class by passing Adafruit server set up details like **AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY**
- To control LED from the server we use **Adafruit_MQTT_Subscribe**
- **AIO_USERNAME/feeds/light** is feed name

```
const int relay=23;

WiFiClient client;
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);
Adafruit_MQTT_Subscribe Light = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/light");
```

15. Define GPIO pins

- Define LED's pin along with data type **int**
- Define Variable **p** with datatype **float**
- Define **String** variable to store value
- Define **bmp** object for **Adafruit_BMP085**

```
const int led1 = 18;
const int led2 = 19;

float p;

String stringOne, stringTwo;

Adafruit_BMP085 bmp;
```

16. Initialize the setup()

- **digitalWrite()** function is used to change the beginning stage LED/LAMP in OFF position.
- **Serial.begin(9600)** is used for data exchange speed.
- Set a **delay** of **10 ms**
- **PinMode()** configures the specified pin to behave either as an input or an output.
- **digitalWrite()** function helps to change the state of LED from **HIGH to LOW** or vice versa
- **Serial.println** is used to print the statement

```

void setup() {
    digitalWrite(relay, LOW);
    Serial.begin(9600);
    delay(10);
    pinMode(relay, OUTPUT);
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(WLAN_SSID);
    WiFi.begin(WLAN_SSID, WLAN_PASS);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println();
    Serial.println("WiFi connected");
    Serial.println("IP address: "); Serial.println(WiFi.localIP());
    mqtt.subscribe(&Light);
}

```

17. Create the **Adafruit_MQTT_Subscribe** object **subscription**, object subscription is used to interact and subscribe. Use this to determine which subscription was received.

- Prints out the received data but also compares the data to determine whether the string received is **ON** or **OFF**
- **digitalWrite()** will change the state of LED using **HIGH** or **LOW**
- If it is **ON** then make it **HIGH**, otherwise make it **LOW**

```

void loop() {
    MQTT_connect();
    Adafruit_MQTT_Subscribe *subscription;
    //digitalWrite(relay, LOW);
    while ((subscription = mqtt.readSubscription(5000))) {

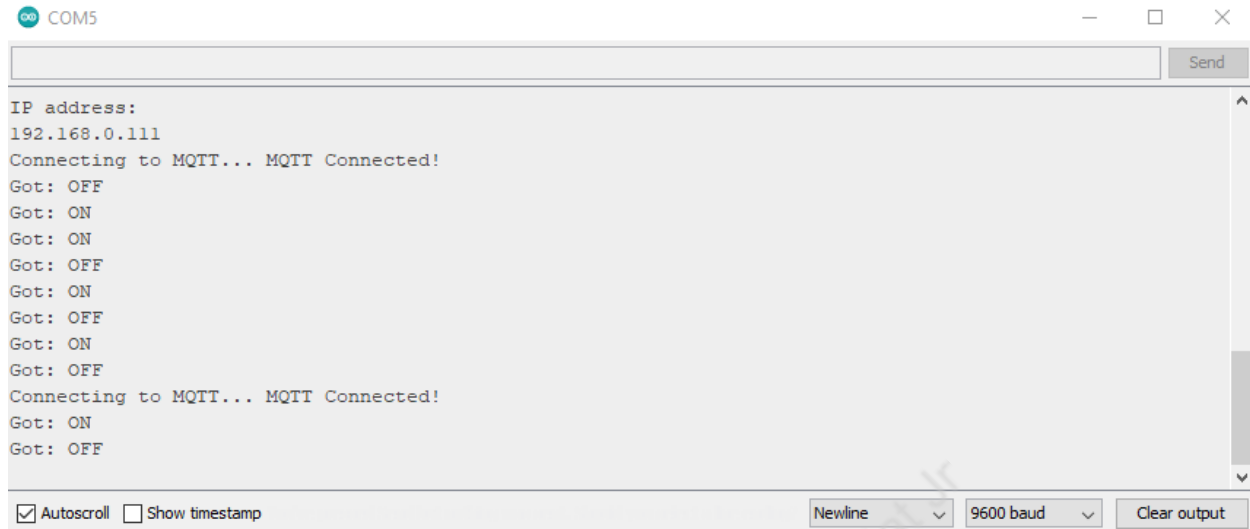
        if (subscription == &Light) {
            Serial.print(F("Got: "));
            Serial.println((char *)Light.lastread);
            if (!strcmp((char*) Light.lastread, "OFF"))
            {
                digitalWrite(relay, HIGH);
            }
            else
            {
                digitalWrite(relay, LOW);
            }
        }
    }
}

```


18. Function to connect and reconnect as necessary to the MQTT server alive.
- It should be called in the loop function and it will take care of connecting.
 - If it gets disconnected then try to connect it again.
 - **ret** is a variable which will save temporary return value in data type **int**

```
void MQTT_connect() {  
    int8_t ret;  
    if (mqtt.connected()) {  
        return;  
    }  
  
    uint8_t retries = 3;  
    while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected  
        mqtt.disconnect();  
        delay(5000); // wait 5 seconds  
        retries--;  
        if (retries == 0) {  
            while (1);  
        }  
    }  
}
```

19. Compile and upload the program to ESP32 board using Arduino IDE
- Verify the program by clicking the Tick option
 - Upload the program by clicking the arrow option
20. Note: If the port is not selected, insert the USB cable in Computer's port and select the port
- Go to Tools and select Serial Monitor



```
COM5
Send
IP address:
192.168.0.111
Connecting to MQTT... MQTT Connected!
Got: OFF
Got: ON
Got: ON
Got: OFF
Got: ON
Got: OFF
Got: ON
Got: OFF
Connecting to MQTT... MQTT Connected!
Got: ON
Got: OFF
Autoscroll Show timestamp Newline 9600 baud Clear output
```

- **Error Message:** If an error message comes like no such file or directory then use the below method to resolve this error
- Go to Tools
 - Click on Manage Libraries
 - Write the component name which needs to install like **adafruit mqtt**
 - Click on Install

21. Go to Tools and select Serial **Monitor**

- See the **LED status**
- Open your Adafruit server and check the live values of Pressure and Temperature. Click the Toggle buttons to turn ON and OFF your LED's

What's NEXT?

In the next class, we will learn **about Weather Monitoring System**.

Expand Your Knowledge

To know more about **Relays** [click here](#).