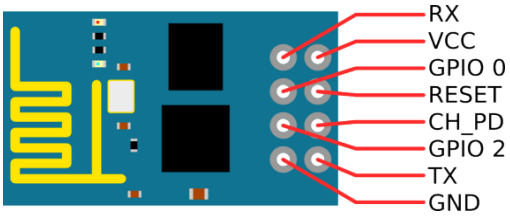
NEC

NodeMCU

Veneth S
DEPT. OF EEE
NEC

ESP8266:

ESP8266 is a low-cost, WiFi Module chip that can be configured to connect to the Internet for Internet of Things (IoT) and similar Technology Projects. Basically, Your normal Electrical and Mechanical equipment's cannot connect to the Internet on their own. They don't have the in-built setup to do so. This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using Hayes-style commands.



Pros and Cons:

+ built-in Wi-Fi

+ smaller size

+ Arduino IDE supported

— Not all Arduino libraries supported and work

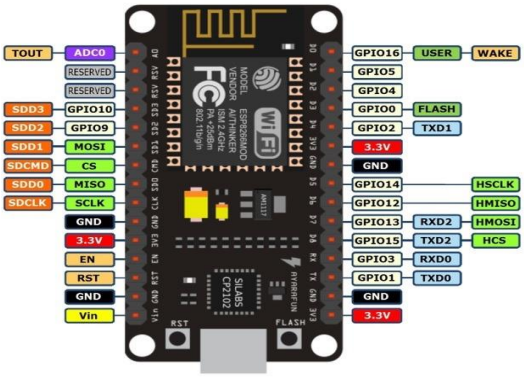
— Not as many GPIO pins

— ESP-1 does NOT have ANY analog pins!

— Only supports 3.3V power rail (5V devices would need level shifter)

NodeMCU:

NodeMCU is a Firmware on ESP8266. It's basically an SoC (System on Chip). A System on a Chip or System on Chip (SoC) is an integrated circuit that integrates all components of a computer or other electronic systems. The NodeMCU board is based on an ESP8266-12 but features a built-in serial over USB interface and other amenities like 2 buttons and 2 LEDs. Simply said NodeMCU = Arduino + ESP8266.



Features:

- 802.11 b/g/n
- Wi-Fi 2.4 GHz, support WPA/WPA2
- Integrated low power 32-bit MCU
- Operating Voltage: 3.0 ~ 3.6 V
- Operating Current: 80mA
- Integrated 10-bit ADC

Programming Used:

- Arduino
- Python
- Lua Script

Pros and Cons:

+ ESP is cheaper than Arduino or clones

+ ESP has built-in WiFi

+ ESP is smaller than most Arduinos

+ ESP can now be programmed with Arduino IDE so you can leverage your Arduino experience

+ Supports Python program and Lua script also.

— It is a 3.3V device, so it may not be compatible with some peripherals

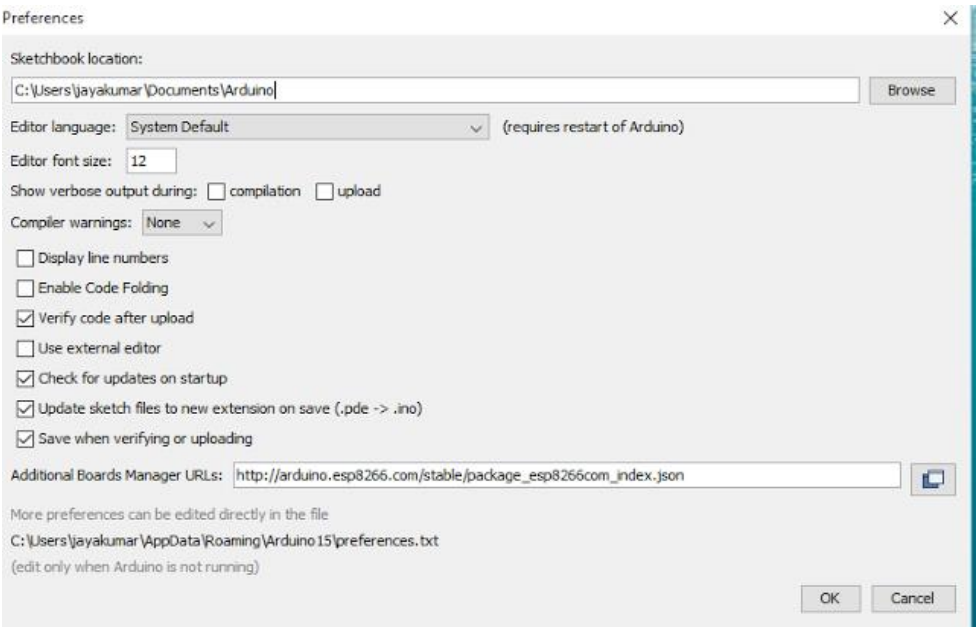
— Less # of analogy pin

— Lack of official documentation

Get Started with NodeMCU:

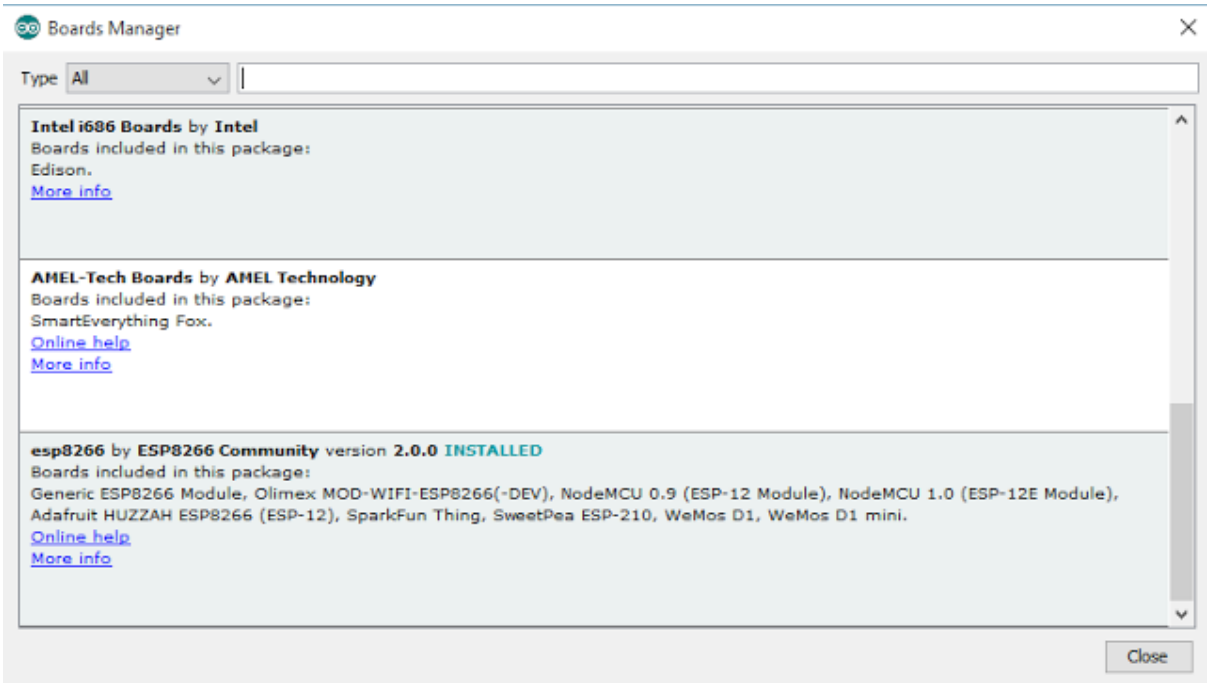
Let's begin with installing the Esp8266 support for the Arduino. Step by step tutorial on how to get started with our favourite chip Esp8266.

Firstly open the Arduino IDE -> Go to files and click on the preference in the Arduino IDE



copy the below code in the Additional boards Manager http://arduino.esp8266.com/stable/package_esp8266com_index.json click OK to close the preference Tab.

After completing the above steps , go to Tools and board, and then select board Manager



Navigate to esp8266 by esp8266 community and install the software for Arduino. Once all the above process been completed we are read to program our esp8266 with Arduino IDE.

Blink LED:

```
void setup() {
  pinMode(D4, OUTPUT);     // Initialize the LED_BUILTIN pin as an output
}
void loop() {
  digitalWrite(D4, LOW);
  delay(1000);
  digitalWrite(D4, HIGH);
  delay(1000);
}
```

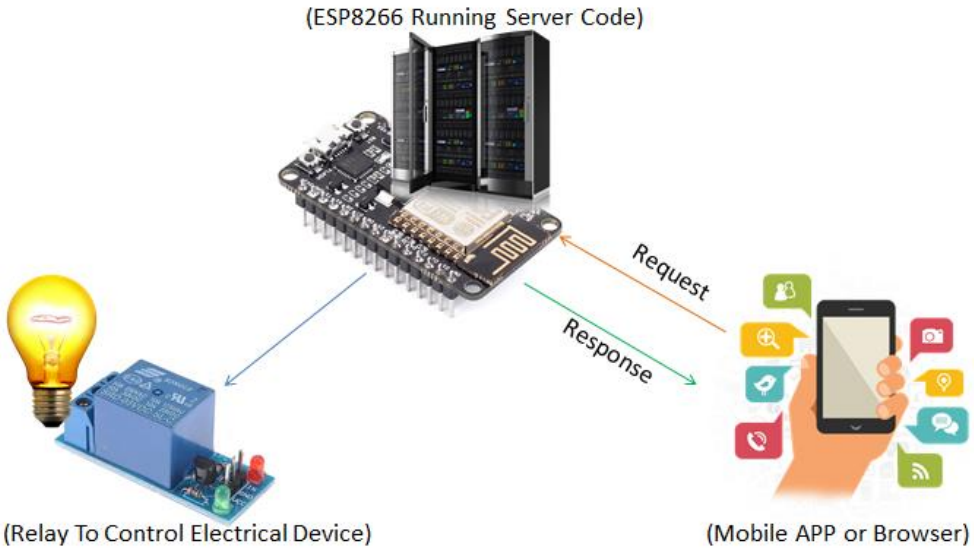
WiFi Client:

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

```
const char* ssid = " type your WiFi name";
const char* password = " type your password ";
 WiFiServer server(80);//Service Port
void WiFiEvent(WiFiEvent_t event) {
    Serial.printf("[WiFi-event] event: %d\n", event);
    switch(event) {
        case WIFI_EVENT_STAMODE_GOT_IP:
            Serial.println("WiFi connected");
            Serial.println("IP address: ");
            Serial.println(WiFi.localIP());
            break;
        case WIFI_EVENT_STAMODE_DISCONNECTED:
            Serial.println("WiFi lost connection");
            break;
    }
}
void setup() {
Serial.begin(115200);
WiFi.disconnect(true);

    delay(1000);

    WiFi.onEvent(WiFiEvent);

    WiFi.begin(ssid, password);

    Serial.println();
    Serial.println();
```

```
  Serial.println("Wait for WiFi... ");
 Serial.println("");
 Serial.println("WiFi connected");
  }
```

Home Automation Using Web Page:



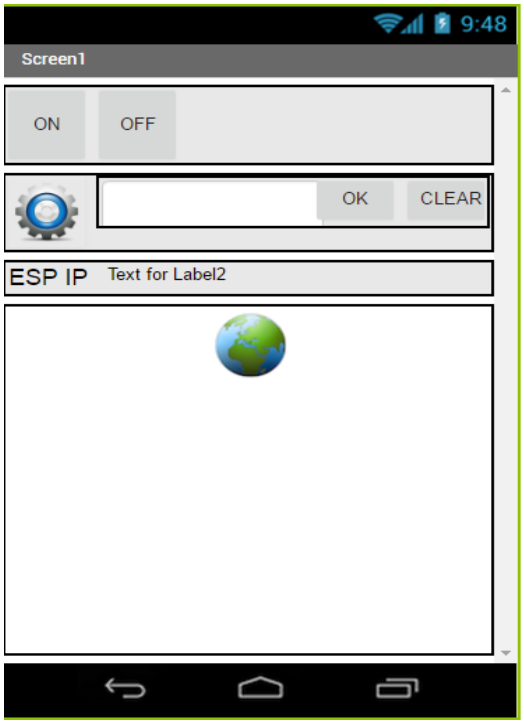

```
#include <ESP8266WiFi.h>
const char* ssid = " type your WiFi name";
const char* password = " type your password ";
        WiFiServer server(80);//Service Port
        void WiFiEvent(WiFiEvent_t event) {
          Serial.printf("[WiFi-event] event: %d\n", event);
          switch(event) {
            case WIFI_EVENT_STAMODE_GOT_IP:
              Serial.println("WiFi connected");
              Serial.println("IP address: ");
              Serial.println(WiFi.localIP());
              break;
            case WIFI_EVENT_STAMODE_DISCONNECTED:
              Serial.println("WiFi lost connection");
```

```
        break;
    }
}

void setup() {
  Serial.begin(115200);
 pinMode(2,OUTPUT);
digitalWrite(2, LOW);
   // delete old config
   WiFi.disconnect(true);
   delay(1000);
   WiFi.onEvent(WiFiEvent);
   WiFi.begin(ssid, password);
   Serial.println();
   Serial.println();
   Serial.println("Wait for WiFi... ");
 Serial.println("");
 Serial.println("WiFi connected");
 // Start the server
 server.begin();
 Serial.println("Server started");
 // Print the IP address
 Serial.print("Use this URL to connect: ");
 Serial.print("http://");
 Serial.print(WiFi.localIP());
 Serial.println("/");
}
 void loop() {
 // Check if a client has connected
```

```
WiFiClient client = server.available();
if (!client) {
  return;
}
// Wait until the client sends some data
Serial.println("new client");
while(!client.available()){
  delay(1);
}
// Read the first line of the request
String request = client.readStringUntil('\r');
Serial.println(request);
client.flush();
   // Match the request
 int value = LOW;
if (request.indexOf("/LED=ON") != -1) {
  digitalWrite(2, HIGH);
  value = HIGH;
}
if (request.indexOf("/LED=OFF") != -1){
  digitalWrite(2, LOW);
  value = LOW;
}
 //Set ledPin according to the request
//digitalWrite(ledPin, value);
// Return the response
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println(""); //  do not forget this one
```

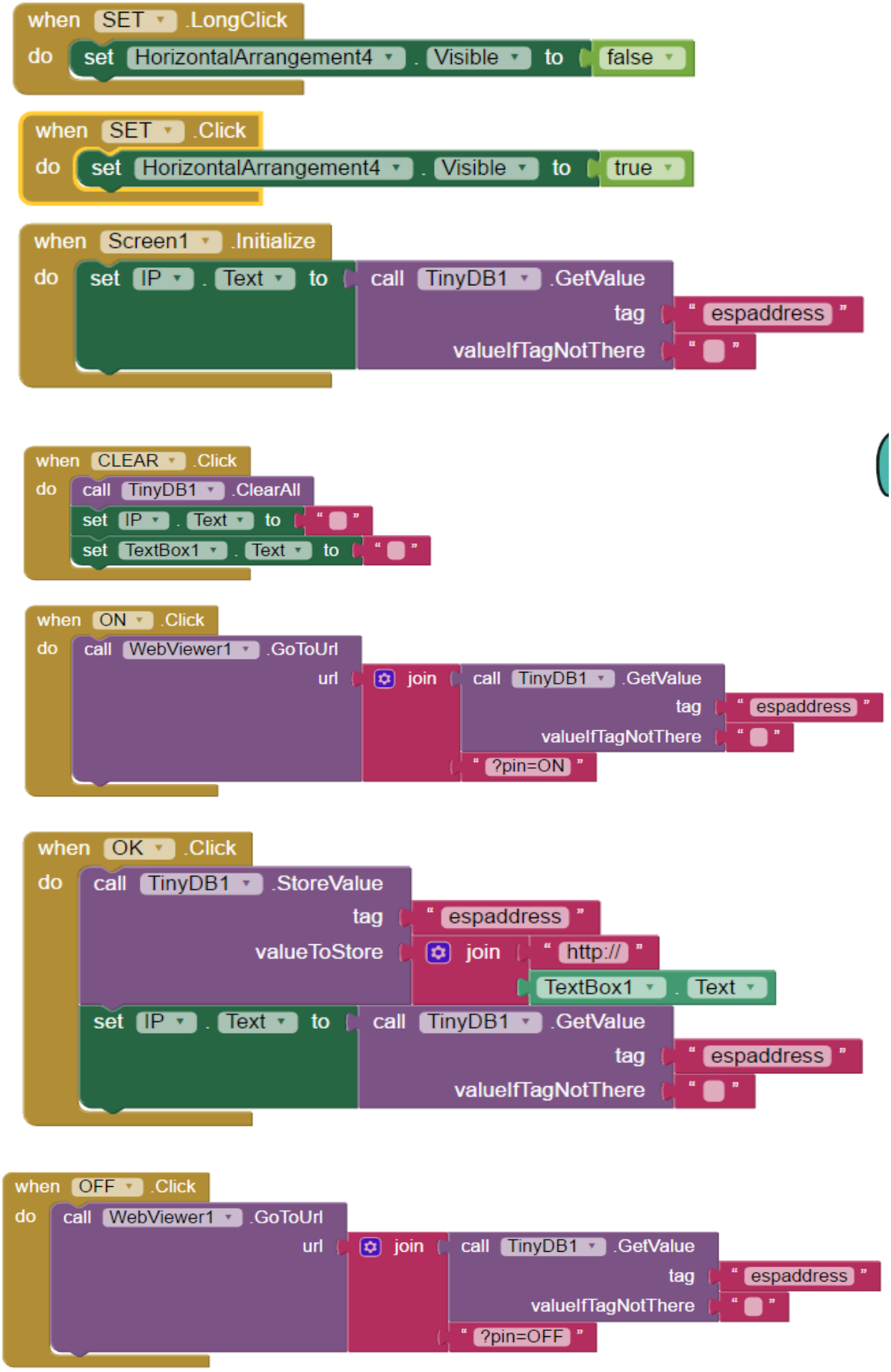
```
client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.print("Led pin is now: ");
if(value == HIGH) {
  client.print("On");
} else {
  client.print("Off");
}
client.println("<br><br>");
client.println("Click <a href=\"/LED=ON\">here</a> turn the LED on pin 2 ON<br>");
client.println("Click <a href=\"/LED=OFF\">here turn the LED on pin 2 OFF<br>");
client.println("</html>");
delay(1);
Serial.println("Client disconnected");
Serial.println("");
}
```

App Creation:

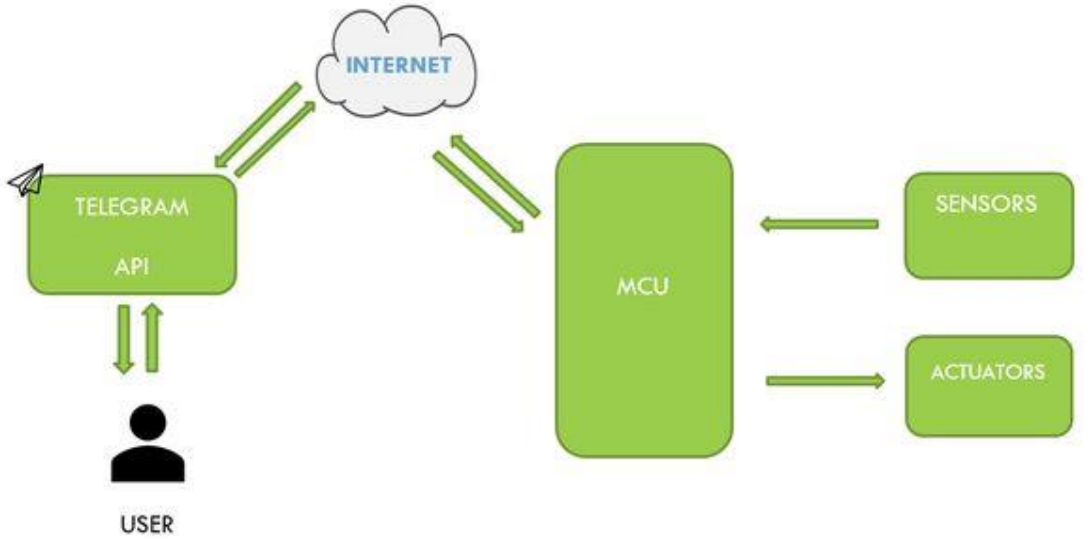


We can easily create the app by using MIT App Inventorhttp://ai2.appinventor.mit.edu/

The app front end is created by just drag the required components. In text box the IP Address of NodeMCU is entered with the help of WebViewer (Globe Logo) we can control it like web page. The backend code as follows,


when SET .LongClick
do set HorizontalArrangement4 . Visible to false
when SET .Click
do set HorizontalArrangement4 . Visible to true
when Screen1 .Initialize
do set IP . Text to call TinyDB1 .GetValue
tag " espaddress "
valueIfTagNotThere " "
when CLEAR .Click
do call TinyDB1 .ClearAll
set IP . Text to " "
set TextBox1 . Text to " "
when ON .Click
do call WebViewer1 .GoToUrl
url join call TinyDB1 .GetValue
tag " espaddress "
valueIfTagNotThere " "
" ?pin=ON "
when OK .Click
do call TinyDB1 .StoreValue
tag " espaddress "
valueToStore join " http:// "
TextBox1 . Text
set IP . Text to call TinyDB1 .GetValue
tag " espaddress "
valueIfTagNotThere " "
when OFF .Click
do call WebViewer1 .GoToUrl
url join call TinyDB1 .GetValue
tag " espaddress "
valueIfTagNotThere " "
" ?pin=OFF "

Home Automation Using Telegram:



This API allows you to connect bots to our system. **Telegram Bots** are special accounts that do not require an additional phone number to set up. These accounts serve as an interface for code running somewhere on your server.

To use this, you don't need to know anything about how our MTProto encryption protocol works — our intermediary server will handle all encryption and communication with the Telegram API for you. You communicate with this server via a simple HTTPS-interface that offers a simplified version of the Telegram API.

The follow YouTube link will help how setup Telegram to Control NodeMCU
https://www.youtube.com/watch?v=8TIbn3bax8s&t=402s

Library file https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot

Sample Code:

```
#include <ESP8266WiFi.h>

#include <WiFiClientSecure.h>

#include <ESP8266TelegramBOT.h>

// Initialize Wifi connection to the router

const char* ssid     = "WiFi Name";

const char* password = "Password";

// Initialize Telegram BOT

#define BOTtoken "API Key"  //token of TestBOT
```

```
#define BOTname "Your Bot Name"
#define BOTusername "Bot User Name"
TelegramBOT bot(BOTtoken, BOTname, BOTusername);
int Bot_mtbs = 1000; //mean time between scan messages
long Bot_lasttime;   //last time messages' scan has been done
bool Start = false;
/*****************************************
 * EchoMessages - function to Echo messages *
 *****************************************/
void Bot_ExecMessages() {
  digitalWrite(2, LOW);
  for (int i = 1; i < bot.message[0][0].toInt() + 1; i++)     {
    //bot.message[i][5]=bot.message[i][5].substring(1,bot.message[i][5].length());
    if (bot.message[i][5] == "\/ledon") {
      digitalWrite(2, HIGH);   // turn the LED on (HIGH is the voltage level)
      bot.sendMessage(bot.message[i][4], "Led is ON", "");
    }
    if (bot.message[i][5] == "\/ledoff") {
      digitalWrite(2, LOW);    // turn the LED off (LOW is the voltage level)
      bot.sendMessage(bot.message[i][4], "Led is OFF", "");
    }
    if (bot.message[i][5] == "\/start") {
      digitalWrite(2, LOW);
      String wellcome = "Welcome from FlashLedBot, your personal Bot on ESP8266 board";
```

```
      String wellcome1 = "/ledon : to switch the Led ON";
      String wellcome2 = "/ledoff : to switch the Led OFF";
      bot.sendMessage(bot.message[i][4], wellcome, "");
      bot.sendMessage(bot.message[i][4], wellcome1, "");
      bot.sendMessage(bot.message[i][4], wellcome2, "");
      Start = true;
    }
  }
  bot.message[0][0] = "";   // All messages have been replied - reset new messages
}
void WiFiEvent(WiFiEvent_t event) {
    Serial.printf("[WiFi-event] event: %d\n", event);
    switch(event) {
        case WIFI_EVENT_STAMODE_GOT_IP:
            Serial.println("WiFi connected");
            Serial.println("IP address: ");
            Serial.println(WiFi.localIP());
            break;
        case WIFI_EVENT_STAMODE_DISCONNECTED:
            Serial.println("WiFi lost connection");
            break;
    }
}
void setup() {
```

```
  Serial.begin(115200);
  pinMode(2,OUTPUT);
    // delete old config
    WiFi.disconnect(true);
    delay(1000);
    WiFi.onEvent(WiFiEvent);
    WiFi.begin(ssid, password);
    Serial.println();
   Serial.println();
    Serial.println("Wait for WiFi... ");
  bot.begin();      // launch Bot functionalities
   // initialize digital pin 2 as an output.
}
void loop() {
  if (millis() > Bot_lasttime + Bot_mtbs)  {
    bot.getUpdates(bot.message[0][1]);   // launch API GetUpdates up to xxx message
    Bot_ExecMessages();  // reply to message with Echo
    Bot_lasttime = millis();
  }
}
```

_____ XXX _____