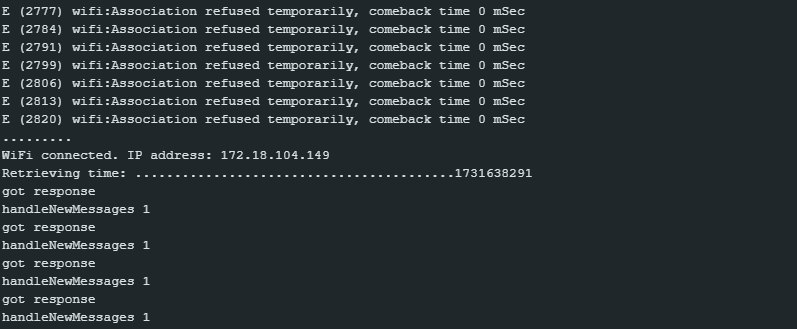
|  |  |  |
| --- | --- | --- |
| Nama | : Ludang Prasetyo Nugroho | [Teknik Komputer ( S1)](https://www.utdi.ac.id/) |
| Nim Matkul | 225510017  : Prak system IOT |  |
|  |  | **PRAKTEK** |

>>>> Code Program 1

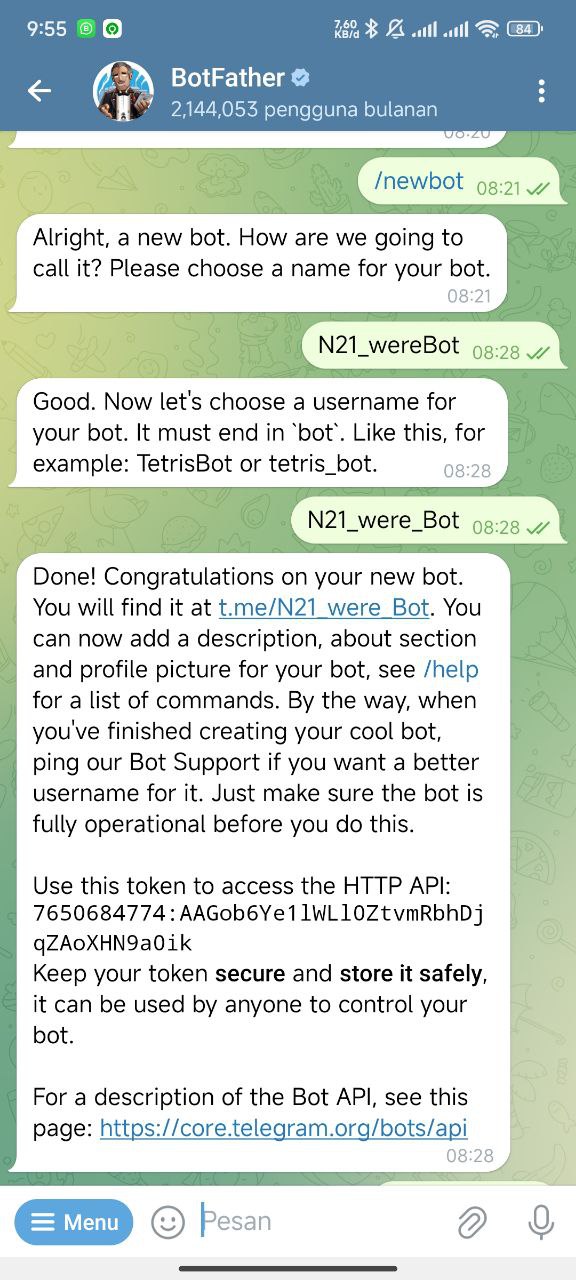
|  |
| --- |
| #include <WiFi.h> |
| #include <WiFiClientSecure.h> |
| #include <UniversalTelegramBot.h> |
|  |
| // Wifi network station credentials |
| #define WIFI\_SSID "RPLA\_2.4" |
| #define WIFI\_PASSWORD "utdijogja" |
|  |
| // Telegram BOT Token (Get from Botfather) |
| #define BOT\_TOKEN "7650684774:AAGob6Ye1lWLl0ZtvmRbhDjqZAoXHN9a0ik" |
|  |
| #define LED1 19 |
| #define LED2 18 |
| #define LED3 5 |
| #define LED4 17 |
|  |
| const unsigned long BOT\_MTBS = 1000; // mean time between scan messages |
|  |
| WiFiClientSecure secured\_client; |
| UniversalTelegramBot bot(BOT\_TOKEN, secured\_client); |
| unsigned long bot\_lasttime; // last time messages' scan has been done |
|  |
| int led1Status = 0; |
| int led2Status = 0; |
| int led3Status = 0; |
| int led4Status = 0; |
|  |
| void handleNewMessages(int numNewMessages) |
| { |
| Serial.print("handleNewMessages "); |
| Serial.println(numNewMessages); |
|  |
| for (int i = 0; i < numNewMessages; i++) |
| { |
| String chat\_id = bot.messages[i].chat\_id; |
| String text = bot.messages[i].text; |
|  |
| String from\_name = bot.messages[i].from\_name; |
| if (from\_name == "") |
| from\_name = "Guest"; |
|  |
| // Commands for LED 1 |
| if (text == "/led1on") |
| { |
| digitalWrite(LED1, HIGH); |
| led1Status = 1; |
| bot.sendMessage(chat\_id, "Led1 ON", ""); |
| } |
| if (text == "/led1off") |
| { |
| digitalWrite(LED1, LOW); |
| led1Status = 0; |
| bot.sendMessage(chat\_id, "Led1 OFF", ""); |
| } |
|  |
| // Commands for LED 2 |
| if (text == "/led2on") |
| { |
| digitalWrite(LED2, HIGH); |
| led2Status = 1; |
| bot.sendMessage(chat\_id, "Led2 ON", ""); |
| } |
| if (text == "/led2off") |
| { |
| digitalWrite(LED2, LOW); |
| led2Status = 0; |
| bot.sendMessage(chat\_id, "Led2 OFF", ""); |
| } |
|  |
| // Commands for LED 3 |
| if (text == "/led3on") |
| { |
| digitalWrite(LED3, HIGH); |
| led3Status = 1; |
| bot.sendMessage(chat\_id, "Led3 ON", ""); |
| } |
| if (text == "/led3off") |
| { |
| digitalWrite(LED3, LOW); |
| led3Status = 0; |
| bot.sendMessage(chat\_id, "Led3 OFF", ""); |
| } |
|  |
| // Commands for LED 4 |
| if (text == "/led4on") |
| { |
| digitalWrite(LED4, HIGH); |
| led4Status = 1; |
| bot.sendMessage(chat\_id, "Led4 ON", ""); |
| } |
| if (text == "/led4off") |
| { |
| digitalWrite(LED4, LOW); |
| led4Status = 0; |
| bot.sendMessage(chat\_id, "Led4 OFF", ""); |
| } |
|  |
| if (text == "/status") |
| { |
| String statusMessage = "Status:\n"; |
| statusMessage += "Led1: " + String(led1Status ? "ON" : "OFF") + "\n"; |
| statusMessage += "Led2: " + String(led2Status ? "ON" : "OFF") + "\n"; |
| statusMessage += "Led3: " + String(led3Status ? "ON" : "OFF") + "\n"; |
| statusMessage += "Led4: " + String(led4Status ? "ON" : "OFF") + "\n"; |
| bot.sendMessage(chat\_id, statusMessage, ""); |
| } |
|  |
| if (text == "/start") |
| { |
| String welcome = "Welcome to Universal Arduino Telegram Bot library, " + from\_name + ".\n"; |
| welcome += "This is Flash Led Bot example.\n\n"; |
| welcome += "/led1on : to switch Led1 ON\n"; |
| welcome += "/led1off : to switch Led1 OFF\n"; |
| welcome += "/led2on : to switch Led2 ON\n"; |
| welcome += "/led2off : to switch Led2 OFF\n"; |
| welcome += "/led3on : to switch Led3 ON\n"; |
| welcome += "/led3off : to switch Led3 OFF\n"; |
| welcome += "/led4on : to switch Led4 ON\n"; |
| welcome += "/led4off : to switch Led4 OFF\n"; |
| welcome += "/status : Returns current status of all LEDs\n"; |
| bot.sendMessage(chat\_id, welcome, ""); |
| } |
| } |
| } |
|  |
| void setup() |
| { |
| Serial.begin(115200); |
| Serial.println(); |
|  |
| pinMode(LED1, OUTPUT); |
| pinMode(LED2, OUTPUT); |
| pinMode(LED3, OUTPUT); |
| pinMode(LED4, OUTPUT); |
| delay(10); |
|  |
| digitalWrite(LED1, LOW); |
| digitalWrite(LED2, LOW); |
| digitalWrite(LED3, LOW); |
| digitalWrite(LED4, LOW); |
|  |
| Serial.print("Connecting to Wifi SSID "); |
| Serial.print(WIFI\_SSID); |
| WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD); |
| secured\_client.setCACert(TELEGRAM\_CERTIFICATE\_ROOT); |
|  |
| while (WiFi.status() != WL\_CONNECTED) |
| { |
| Serial.print("."); |
| delay(500); |
| } |
| Serial.print("\nWiFi connected. IP address: "); |
| Serial.println(WiFi.localIP()); |
|  |
| Serial.print("Retrieving time: "); |
| configTime(0, 0, "pool.ntp.org"); |
| time\_t now = time(nullptr); |
| while (now < 24 \* 3600) |
| { |
| Serial.print("."); |
| delay(100); |
| now = time(nullptr); |
| } |
| Serial.println(now); |
| } |
|  |
| void loop() |
| { |
| if (millis() - bot\_lasttime > BOT\_MTBS) |
| { |
| int numNewMessages = bot.getUpdates(bot.last\_message\_received + 1); |
|  |
| while (numNewMessages) |
| { |
| Serial.println("got response"); |
| handleNewMessages(numNewMessages); |
| numNewMessages = bot.getUpdates(bot.last\_message\_received + 1); |
| } |
|  |
| bot\_lasttime = millis(); |
| } |
| } |

**# Serial monitor**



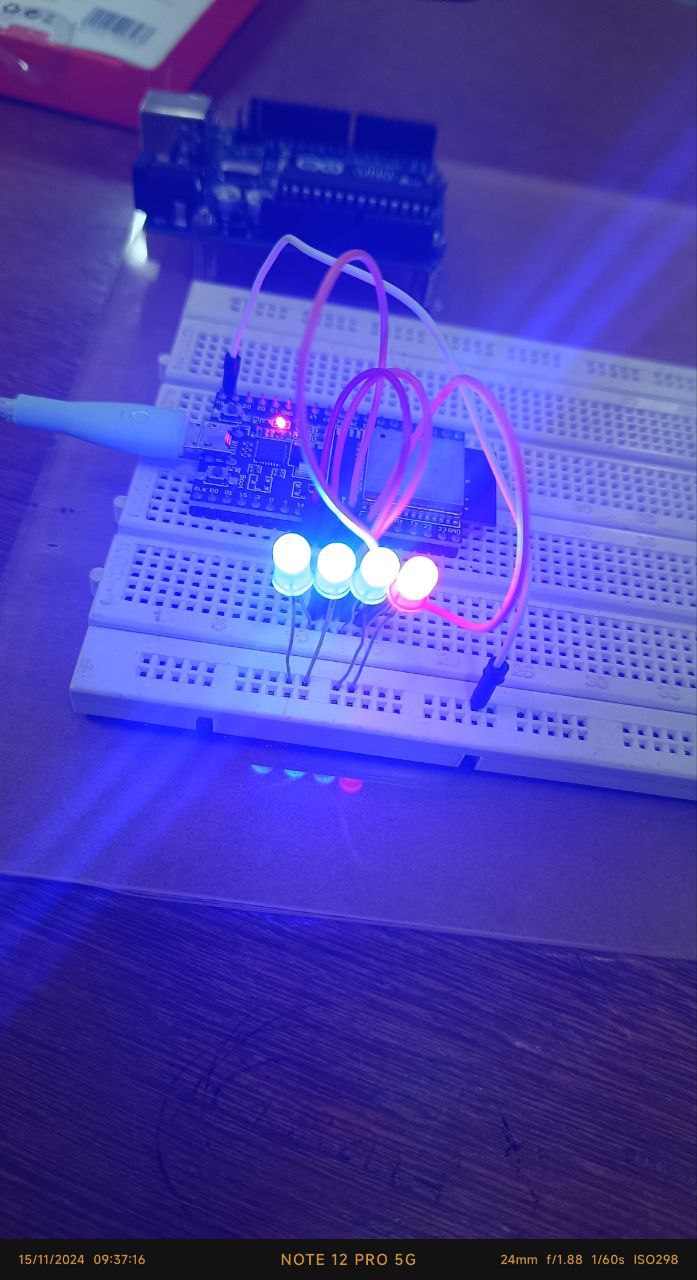
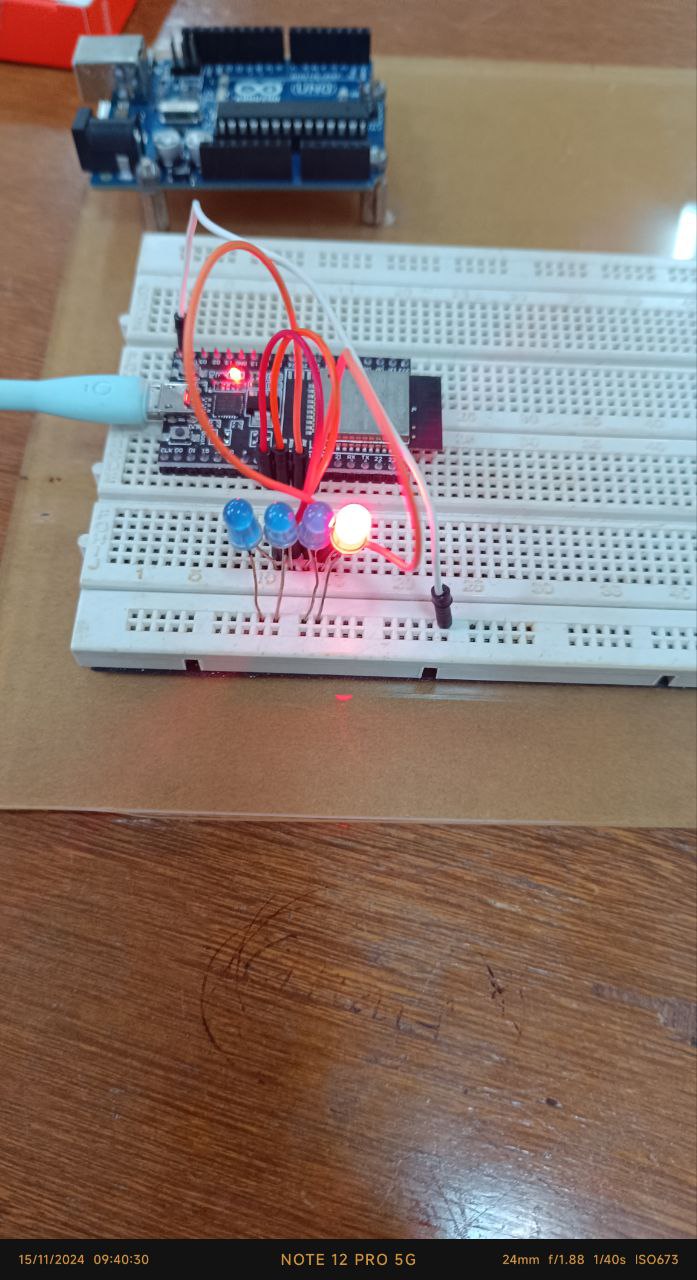
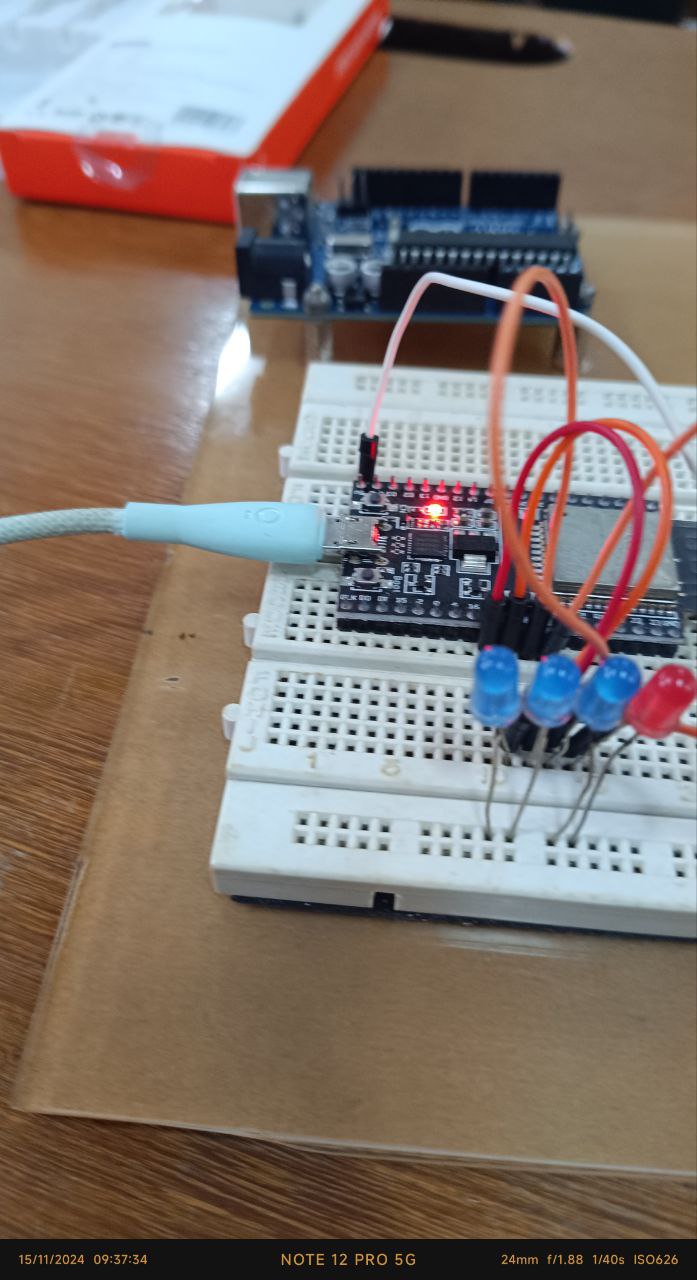
Code berjalan sesuwai arahan saat menerima pesan dari telegram akan di munculkan di serial monitor

**# Membuat bod dan mengambil toket Bod**

Membuat sebuah token bod dan memberikan Input untuk mengontrol led

**# Kontrol led**



Ini saat led tidak di berikan masukan atau lednya masih mati semua dan di kananya saat led di berikan masukan untuk di nyalakan akan menyalakan led 1 dan led seterusnya akan menyala juga

>>>> Code Program 2

|  |
| --- |
| #include <WiFi.h> |
| #include <WiFiClientSecure.h> |
| #include <UniversalTelegramBot.h> |
|  |
| // Wifi network station credentials |
| #define WIFI\_SSID "ssid" |
| #define WIFI\_PASSWORD "passwd" |
| // Telegram BOT Token (Get from Botfather) |
| #define BOT\_TOKEN "token" |
|  |
| const unsigned long BOT\_MTBS = 1000; // mean time between scan messages |
|  |
| WiFiClientSecure secured\_client; |
| UniversalTelegramBot bot(BOT\_TOKEN, secured\_client); |
| unsigned long bot\_lasttime;          // last time messages' scan has been done |
|  |
| void handleNewMessages(int numNewMessages) |
| { |
| for (int i = 0; i < numNewMessages; i++) |
| { |
| // Inline buttons with callbacks when pressed will raise a callback\_query message |
| if (bot.messages[i].type == "callback\_query") |
| { |
| Serial.print("Call back button pressed by: "); |
| Serial.println(bot.messages[i].from\_id); |
| Serial.print("Data on the button: "); |
| Serial.println(bot.messages[i].text); |
| bot.sendMessage(bot.messages[i].from\_id, bot.messages[i].text, ""); |
| } |
| else |
| { |
| String chat\_id = bot.messages[i].chat\_id; |
| String text = bot.messages[i].text; |
|  |
| String from\_name = bot.messages[i].from\_name; |
| if (from\_name == "") |
| from\_name = "Guest"; |
|  |
| if (text == "/options") |
| { |
| String keyboardJson = "[[{ \"text\" : \"Go to Google\", \"url\" : \"https://www.google.com\" }],[{ \"text\" : \"Send\", \"callback\_data\" : \"This was sent by inline\" }]]"; |
| bot.sendMessageWithInlineKeyboard(chat\_id, "Choose from one of the following options", "", keyboardJson); |
| } |
|  |
| if (text == "/start") |
| { |
| String welcome = "Welcome to Universal Arduino Telegram Bot library, " + from\_name + ".\n"; |
| welcome += "This is Inline Keyboard Markup example.\n\n"; |
| welcome += "/options : returns the inline keyboard\n"; |
|  |
| bot.sendMessage(chat\_id, welcome, "Markdown"); |
| } |
| } |
| } |
| } |
|  |
| void setup() |
| { |
| Serial.begin(115200); |
| Serial.println(); |
|  |
| // attempt to connect to Wifi network: |
| Serial.print("Connecting to Wifi SSID "); |
| Serial.print(WIFI\_SSID); |
| WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD); |
| secured\_client.setCACert(TELEGRAM\_CERTIFICATE\_ROOT); // Add root certificate for api.telegram.org |
| while (WiFi.status() != WL\_CONNECTED) |
| { |
| Serial.print("."); |
| delay(500); |
| } |
| Serial.print("\nWiFi connected. IP address: "); |
| Serial.println(WiFi.localIP()); |
|  |
| Serial.print("Retrieving time: "); |
| configTime(0, 0, "pool.ntp.org"); // get UTC time via NTP |
| time\_t now = time(nullptr); |
| while (now < 24 \* 3600) |
| { |
| Serial.print("."); |
| delay(100); |
| now = time(nullptr); |
| } |
| Serial.println(now); |
| } |
|  |
| void loop() |
| { |
| if (millis() - bot\_lasttime > BOT\_MTBS) |
| { |
| int numNewMessages = bot.getUpdates(bot.last\_message\_received + 1); |
|  |
| while (numNewMessages) |
| { |
| Serial.println("got response"); |
| handleNewMessages(numNewMessages); |
| numNewMessages = bot.getUpdates(bot.last\_message\_received + 1); |
| } |
|  |
| bot\_lasttime = millis(); |
| } |
| } |

>>>> Code Program 3

|  |
| --- |
| #include <WiFi.h> |
| #include <WiFiClientSecure.h> |
| #include <UniversalTelegramBot.h> |
|  |
| const unsigned long BOT\_MTBS = 1000; // mean time between scan messages |
|  |
| // Wifi network station credentials |
| #define WIFI\_SSID "ssid" |
| #define WIFI\_PASSWORD "paswd" |
| // Telegram BOT Token (Get from Botfather) |
| #define BOT\_TOKEN "token" |
|  |
| unsigned long bot\_lasttime;          // last time messages' scan has been done |
| WiFiClientSecure secured\_client; |
| UniversalTelegramBot bot(BOT\_TOKEN, secured\_client); |
|  |
| void handleNewMessages(int numNewMessages) |
| { |
| for (int i = 0; i < numNewMessages; i++) |
| { |
| String chat\_id = bot.messages[i].chat\_id; |
| String text = bot.messages[i].text; |
|  |
| String from\_name = bot.messages[i].from\_name; |
| if (from\_name == "") |
| from\_name = "Guest"; |
|  |
| if (bot.messages[i].longitude != 0 || bot.messages[i].latitude != 0) |
| { |
| Serial.print("Long: "); |
| Serial.println(String(bot.messages[i].longitude, 6)); |
| Serial.print("Lat: "); |
| Serial.println(String(bot.messages[i].latitude, 6)); |
|  |
| String message = "Long: " + String(bot.messages[i].longitude, 6) + "\n"; |
| message += "Lat: " + String(bot.messages[i].latitude, 6) + "\n"; |
| bot.sendMessage(chat\_id, message, "Markdown"); |
| } |
| else if (text == "/start") |
| { |
| String welcome = "Welcome to Universal Arduino Telegram Bot library, " + from\_name + ".\n"; |
| welcome += "Share a location or a live location and the bot will respond with the co-ords\n"; |
|  |
| bot.sendMessage(chat\_id, welcome, "Markdown"); |
| } |
| } |
| } |
|  |
| void setup() |
| { |
| Serial.begin(115200); |
| Serial.println(); |
|  |
| // attempt to connect to Wifi network: |
| Serial.print("Connecting to Wifi SSID "); |
| Serial.print(WIFI\_SSID); |
| WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD); |
| secured\_client.setCACert(TELEGRAM\_CERTIFICATE\_ROOT); // Add root certificate for api.telegram.org |
| while (WiFi.status() != WL\_CONNECTED) |
| { |
| Serial.print("."); |
| delay(500); |
| } |
| Serial.print("\nWiFi connected. IP address: "); |
| Serial.println(WiFi.localIP()); |
|  |
| Serial.print("Retrieving time: "); |
| configTime(0, 0, "pool.ntp.org"); // get UTC time via NTP |
| time\_t now = time(nullptr); |
| while (now < 24 \* 3600) |
| { |
| Serial.print("."); |
| delay(100); |
| now = time(nullptr); |
| } |
| Serial.println(now); |
| } |
|  |
| void loop() |
| { |
| if (millis() - bot\_lasttime > BOT\_MTBS) |
| { |
| int numNewMessages = bot.getUpdates(bot.last\_message\_received + 1); |
|  |
| while (numNewMessages) |
| { |
| Serial.println("got response"); |
| handleNewMessages(numNewMessages); |
| numNewMessages = bot.getUpdates(bot.last\_message\_received + 1); |
| } |
|  |
| bot\_lasttime = millis(); |
| } |
| } |

>>>> Code Program 4

|  |
| --- |
| /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
| A telegram bot for your ESP32 that demonstrates sending an image |
| from URL. |
|  |
| Parts: |
| ESP32 D1 Mini stlye Dev board\* - http://s.click.aliexpress.com/e/C6ds4my |
| (or any ESP32 board) |
|  |
| = Affilate |
|  |
| If you find what I do useful and would like to support me, |
| please consider becoming a sponsor on Github |
| https://github.com/sponsors/witnessmenow/ |
|  |
| Example originally written by Vadim Sinitski |
|  |
| Library written by Brian Lough |
| YouTube: https://www.youtube.com/brianlough |
| Tindie: https://www.tindie.com/stores/brianlough/ |
| Twitter: https://twitter.com/witnessmenow |
| \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ |
| #include <WiFi.h> |
| #include <WiFiClientSecure.h> |
| #include <UniversalTelegramBot.h> |
|  |
| // Wifi network station credentials |
| #define WIFI\_SSID "harjolukito" |
| #define WIFI\_PASSWORD "ami1971da" |
| // Telegram BOT Token (Get from Botfather) |
| #define BOT\_TOKEN "6392538293:AAEwF1QytFs2JKM1fqoWZWIwI0spCVtnDjU" |
|  |
| const unsigned long BOT\_MTBS = 1000; // mean time between scan messages |
|  |
| unsigned long bot\_lasttime;          // last time messages' scan has been done |
| WiFiClientSecure secured\_client; |
| UniversalTelegramBot bot(BOT\_TOKEN, secured\_client); |
|  |
| String test\_photo\_url = "https://www.arduino.cc/en/uploads/Trademark/ArduinoCommunityLogo.png"; |
|  |
| void handleNewMessages(int numNewMessages) { |
| Serial.print("handleNewMessages "); |
| Serial.println(numNewMessages); |
|  |
| for (int i=0; i<numNewMessages; i++) { |
| String chat\_id = bot.messages[i].chat\_id; |
| String text = bot.messages[i].text; |
|  |
| String from\_name = bot.messages[i].from\_name; |
| if (from\_name == "") from\_name = "Guest"; |
|  |
| if (text == "/get\_test\_photo") { |
| bot.sendPhoto(chat\_id, test\_photo\_url, "Caption is optional, you may not use photo caption"); |
| } |
|  |
| if (text == "/start") { |
| String welcome = "Welcome to Universal Arduino Telegram Bot library, " + from\_name + ".\n"; |
| welcome += "This is Send Image From URL example.\n\n"; |
| welcome += "/get\_test\_photo : getting test photo\n"; |
|  |
| bot.sendMessage(chat\_id, welcome, ""); |
| } |
| } |
| } |
|  |
| void setup() |
| { |
| Serial.begin(115200); |
| Serial.println(); |
|  |
| // attempt to connect to Wifi network: |
| Serial.print("Connecting to Wifi SSID "); |
| Serial.print(WIFI\_SSID); |
| WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD); |
| secured\_client.setCACert(TELEGRAM\_CERTIFICATE\_ROOT); // Add root certificate for api.telegram.org |
| while (WiFi.status() != WL\_CONNECTED) |
| { |
| Serial.print("."); |
| delay(500); |
| } |
| Serial.print("\nWiFi connected. IP address: "); |
| Serial.println(WiFi.localIP()); |
|  |
| Serial.print("Retrieving time: "); |
| configTime(0, 0, "pool.ntp.org"); // get UTC time via NTP |
| time\_t now = time(nullptr); |
| while (now < 24 \* 3600) |
| { |
| Serial.print("."); |
| delay(100); |
| now = time(nullptr); |
| } |
| Serial.println(now); |
| } |
|  |
| void loop() |
| { |
| if (millis() - bot\_lasttime > BOT\_MTBS) |
| { |
| int numNewMessages = bot.getUpdates(bot.last\_message\_received + 1); |
|  |
| while (numNewMessages) |
| { |
| Serial.println("got response"); |
| handleNewMessages(numNewMessages); |
| numNewMessages = bot.getUpdates(bot.last\_message\_received + 1); |
| } |
|  |
| bot\_lasttime = millis(); |
| } |
| } |