Guia completo do NodeMCU – ESP12 – Alarme Residencial IOT (3)

22-28 minutes

Nesse tutorial NodeMCU – ESP12 – Alarme Residencial IOT (3), você verá :

O que é IOT – Internet das Coisas ?

Para quem nunca ouviu falar em Internet das Coisas – IOT (*Internet of Things*) é uma nova tecnologia que interliga aparelhos às pessoas, através da internet e que esta se tornando muito popular. Os aparelhos ou dispositivos podem enviar ou receber dados e comunicar-se com os usuários . A Internet das Coisas emergiu dos avanços de várias áreas como sistemas embarcados, comunicação e sensoriamento. Os estudos indicam que tudo e todos estarão interconectados através da Internet em um breve futuro.

Muita gente no Brasil já tem o seu Smartphone. Os aplicativos como todo mundo sabe, tem facilitado muito a nossa vida. Várias empresas tem criando aplicativos para se aproximarem dos clientes e facilitar a comunicação com os mesmos.

IOT faz a conexão entre aparelhos e pessoas, pode ser através de aplicativos, Plataformas IOT ou mesmo através de paginas da WEB. Muitas Plataformas IOT estão sendo criadas para servirem de link entre as pessoas e os seus dispositivos. Grandes empresas como a Google, IBM e Microsoft já criaram suas plataformas IOT. A previsão é de que bilhões de dispositivos sejam interligados e essas empresas estão de olho nesse mercado. Veja que algumas plataformas são grátis e outras são pagas para uso, além de um limite determinado. Para você ter uma ideia das Plataformas IOT hoje existentes, veja essa lista :

Lista de Plataformas IOT – 2018

Eu imagino aplicações práticas e úteis para IOT, como :

– Monitoramento de alarme residencial – imagine você receber uma mensagem no Smartphone – alguém tentou invadir a sua casa – veja a foto do invasor. Pode ficar tranquilo, o invasor desistiu e foi embora.

– Porteiro residencial remoto – Você esta no trabalho e recebe essa mensagem no Smartphone- O entregador do correios esta na sua porta – quer falar com ele ? Posso autorizar o recebimento ? Ai abre uma caixa grande automática, para o carteiro deixar a encomenda.

– Monitoramento de água em casa – Você recebe essa mensagem no smartphone e no email – Sr morador, baseando-se em gastos mensais de água em sua residência, existe uma possibilidade de vazamento. Sugiro que faça uma verificação.

– Monitoramento de energia : Você recebe essa mensagem no smartphone – Sr morador, a casa esta vazia e o ferro de passar continua ligado. Deseja que ele seja desligado ?

Acho que é por aí ! Usem a criatividade e façam projetos úteis que possam facilitar as nossas vidas.

Aplicativo TELEGRAM:

Ao estudar sobre IOT para desenvolver esse tutorial, fiquei em dúvida sobre qual plataforma IOT eu usaria. Existem centenas de Plataformas IOT, e cada uma tem um jeito de usá-la. Não existe ainda uma Plataforma muito popular ou padrão. Essa Tecnologia é nova e está em pleno desenvolvimento.

Por isso escolhi usar o TELEGRAM . **TELEGRAM** é um aplicativo de troca de mensagens concorrente do Whatsapp. Você pode usar o aplicativo no smartphone (Android e IOS) e também no seu PC (Windows, MAC ou Linux) ! O uso é grátis e muito simples ! E para ajudar na escolha, o **TELEGRAM** tem um **BOT** que permite fazer a conexão IOT . É muito legal e fácil de usar.

Definição de BOT – Wikipedia:

“**Bot**, diminutivo de ***robot***, também conhecido como **Internet bot** ou **web robot**, é uma aplicação de software concebida para simular ações humanas repetidas vezes de maneira padrão, da mesma forma como faria um robô.”

Plataforma TELEGRAM BOT

Com o BOT do TELEGRAM , você poderá receber mensagens dos dispositivos (nesse caso, o NodeMCU) e enviar comandos, através do seu Smartphone ou PC !

Instalando o TELEGRAM:

Se você usa um Smartphone Android ou um IPHONE, instale o aplicativo através da Google Play ou da Apple Store:

TELEGRAM na Google Play

TELEGRAM na Apple Store

Após a instalação do TELEGRAM, se preferir usar a língua portuguesa, entre em **Configurações** > **Idioma** e altere para Portugês (Brasil). O **Menu** fica na parte superior lado esquerdo (três riscos). Crie o seu usuário – o uso é grátis.

Para facilitar a instalação e configuração do **TELEGRAM BOT** , instale o Aplicativo Desktop também no seu PC :

TELEGRAM para Windows, MAC e LINUX

No **Telegram Desktop** para alterar para português, entre em **Configurations> change language.**

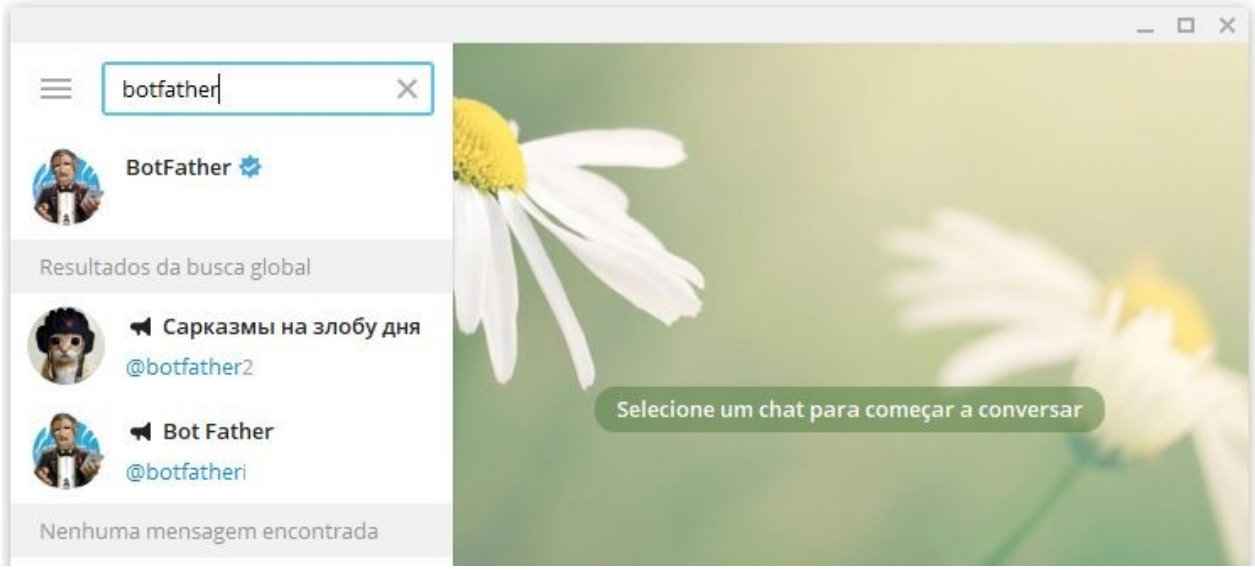
Após a instalação e configuração do TELEGRAM , vai descobrir que muitos amigos e conhecidos seus já são usuários do TELEGRAM. Da forma que você pode conversar com os seus amigos, você poderá "conversar" também com o **BOT**, através de comandos. No link abaixo, você poderá entender melhor como funciona o **BOT**.

BOTs – introdução para desenvolvedores

Configurando o BOT do TELEGRAM :

Para poder criar um novo usuário **BOT** , é necessário acessar o usuário BOT Pai (**BotFather**) . Sugiro que faça todo esse procedimento no **TELEGRAM Desktop** no seu PC. Pois assim ficará mais fácil, copiar as informações necessárias.

Na janela do TELEGRAM Desktop, procure o usuário **BotFather**. Veja que existem outros usuários com nomes similares. Clique no usuário **BotFather**.



Na janela do BotFather, na linha de comandos (parte inferior) clique em **Começar**. Vai aparecer uma lista de comandos do **BotFather**. Digite **/help** se precisar de ajuda. Todos os comandos do BOT devem começar com uma barra /. Veja alguns comandos :

- **/newbot** – para criar um novo BOT
- **/mybots** – para editar seus BOTs
- **/setcommands** – criar e alterar os comandos
- **/deletebot** – para apagar um BOT

Para criar o seu BOT , digite o comando **/newbot** . Após a primeira pergunta, insira o nome do seu BOT. Se o seu nome já existir, use um outro nome , como por exemplo (no meu caso):

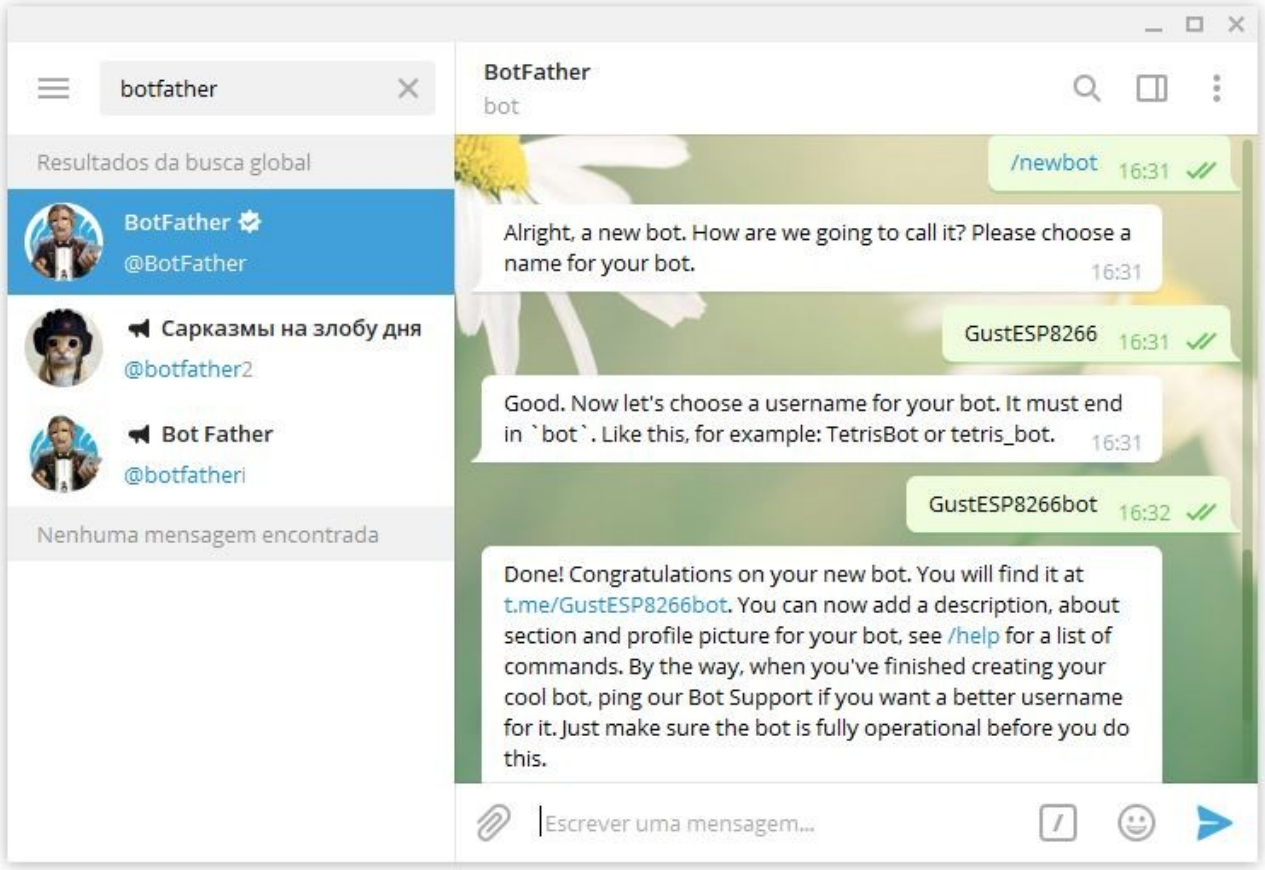
GustESP8266

Na segunda pergunta, digite o nome do seu novo usuário. O nome do usuário deve terminar com bot. Se o seu nome já existir, use um outro nome , como por exemplo :

GustESP8266bot

Se o usuário foi criado com sucesso, aparecerá a mensagem ” Done!” . **Veja que uma chave TOKEN foi criada. Essa chave deverá ser copiada, pois será inserida no Sketch do NodeMCU ESP12 !**

Use this token to access the HTTP API (exemplo):
123456789:ABCDEFGHIJKLMOPQRST-ABCDEFGHIJKLMNO



OK ! Já criamos o BOT e o usuário no TELEGRAM.

Instalando as Bibliotecas Arduino para o TELEGRAM:

Para usar o **NodeMCU-ESP12** com o TELEGRAM, será necessário a instalação de duas Bibliotecas na **Arduino IDE:**

- *ArduinoJson*
- *Universal Arduino Telegram Bot*

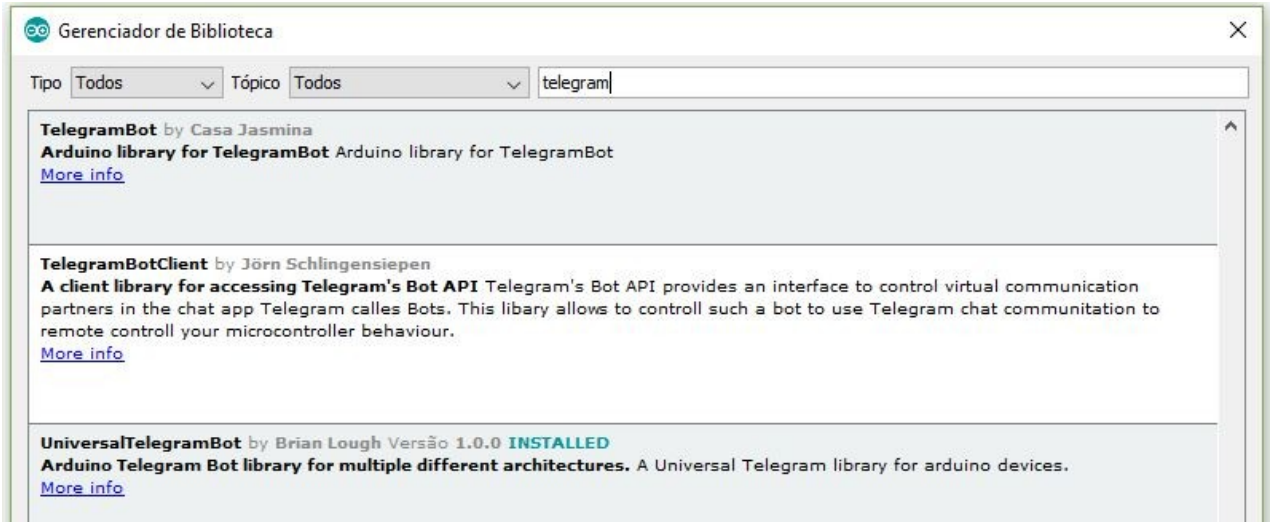
Vamos instalar essas duas Bibliotecas, usando o **Gerenciador de Bibliotecas**. Abra a Arduino IDE, já configurada para o NodeMCU-ESP12. Se ainda não configurou, veja o segundo tutorial :

NodeMCU – ESP12 – Usando Arduino IDE (2)

Para instalar a nova Biblioteca, clique em **Sketch > Incluir Biblioteca > Gerenciar Bibliotecas .** Na janela do Gerenciador de Bibliotecas, refine a busca digitando **arduinojson.** Clique em **more info** e depois em **instalar**.

Vamos instalar agora a segunda biblioteca **UniversalTelegramBot** com o mesmo procedimento acima. Refine a busca digitando **telegram**. Clique em **more info** e depois em **instalar**.

Após a instalação das duas Bibliotecas, é necessário que feche e abra novamente o programa Arduino IDE, para efetivar as Bibliotecas.



Pronto, Arduino IDE preparada !

Testando o NodeMCU-ESP12 com o TELEGRAM:

Para testarmos a comunicação do Aplicativo **TELEGRAM** com o **NodeMCU-ESP12**, usaremos um Sketch baseado em um exemplo da Biblioteca **Universal Arduino Telegram Bot.** Sabemos que o **Led azul** da placa NodeMCU está conectado no pino **GPIO_16**. Um pulso LOW(0V) acionará esse led. Portanto, através de comandos no TELEGRAM, faremos o Led azul acender e apagar.

Sketch exemplo – Flash Led

Programa de teste do NodeMCU-ESP12 com o TELEGRAM :

Sketch Flash Led Telegram.ino

IMPORTANTE : No sketch abaixo, você deverá alterar o nome do seu roteador WIFI (**ssid**) e a senha do roteador (**password**) também. Digite entre as aspas, exemplo = “eletrogate”.

Na linha do programa **#define BOTtoken,** insira a **chave Token** (entre aspas) que foi copiada quando gerou o seu usuário **BOT** através do **BotFather.** Exemplo:

“123456789:ABCDEFGHIJKLMOPQRST-ABCDEFGHIJKLMNO”

```
/*  ESP822 - Flash LED com Telegram
    Arduino IDE 1.8.5 - ESP8266
    Gustavo Murta   13/mar/2018
Baseado em:
https://github.com/witnessmenow/Universal-Arduino-Telegram-
Bot/blob/master/examples/ESP8266/FlashLED/FlashLED.ino

Blog Eletrogate:
http://blog.eletrogate.com/nodemcu-esp12-alarme-residencial-iot-3/
*/

#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>

// Inicializando a conexao WIFI com o Roteador - digite entre aspas
char ssid[] = "roteador WIFI";           // nome do seu roteador WIFI (SSID)
char password[] = "senha do WIFI";       // senha do roteador WIFI

// Inicializa o BOT Telegram - copie aqui a chave Token quando configurou o seu
BOT - entre aspas
#define BOTtoken "123456789:ABCDEFGHIJKLMOPQRST-ABCDEFGHIJKLMNO"  // sua chave
Token Bot

WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);

int Bot_mtbs = 1000;      // tempo entre a leitura das mensagens
long Bot_lasttime;        // ultima mensagem lida
bool Start = false;

const int ledPin = 16;    // GPIO_16 do LED azul
int ledStatus = 0;

void handleNewMessages(int numNewMessages)
{
  Serial.print("Mensagem recebida = ");
  Serial.println(String(numNewMessages));

  for (int i = 0; i < numNewMessages; i++)
  {
    String chat_id = String(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 == "/ledon")                              //  comando Ledon
    {
      digitalWrite(ledPin, LOW);                       // acende LED azul
      ledStatus = 1;
      bot.sendMessage(chat_id, "LED esta aceso", "");  // envia mensagem
    }

    if (text == "/ledoff")                                 //  comando Ledoff
```

```
    {
      ledStatus = 0;
      digitalWrite(ledPin, HIGH);                          // apaga LED azul
      bot.sendMessage(chat_id, "LED esta apagado", "");    // envia mensagem
    }

    if (text == "/status")                                 // comando estado do
LED
    {
      if (ledStatus) {
        bot.sendMessage(chat_id, "LED esta aceso", "");
      } else {
        bot.sendMessage(chat_id, "LED esta apagado", "");
      }
    }

    if (text == "/start")                                   // comando comeca
    {
      String welcome = "Bem-vindo a Biblioteca Universal Arduino Telegram Bot, "
+ from_name + ".\n";
      welcome += "Esse é um exemplo de controle do Led.\n\n";
      welcome += "/ledon : para acender o LED\n";
      welcome += "/ledoff : para apagar o LED\n";
      welcome += "/status : mostra o estado do LED\n";
      bot.sendMessage(chat_id, welcome, "Markdown");
    }
  }
}

void setup() {
  Serial.begin(115200);

  WiFi.mode(WIFI_STA);          // Configura o WIFI do NodeMCU para modo estação
  WiFi.disconnect();            // desconecta o WIFI
  delay(100);                   // atraso de 100 milisegundos

  Serial.print("Conectando Wifi: ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED)   // aguardando a conexão WEB
  {
    Serial.print(".");
    delay(500);                           // atraso de 0,5 segundos
  }
  Serial.println("");
  Serial.println("WiFi conectado");       // WIFI conectado
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  pinMode(ledPin, OUTPUT);                // configura o pino do LED como saida
  delay(10);
  digitalWrite(ledPin, HIGH);             // inicializa o LED como apagado
}

void loop()
{
  if (millis() > Bot_lasttime + Bot_mtbs)    // controlando as mensagens
  {
    int numNewMessages = bot.getUpdates(bot.last_message_received + 1);

    while (numNewMessages)                  // numero de novas mensagens
    {
      Serial.println("Resposta recebida do Telegram");
```

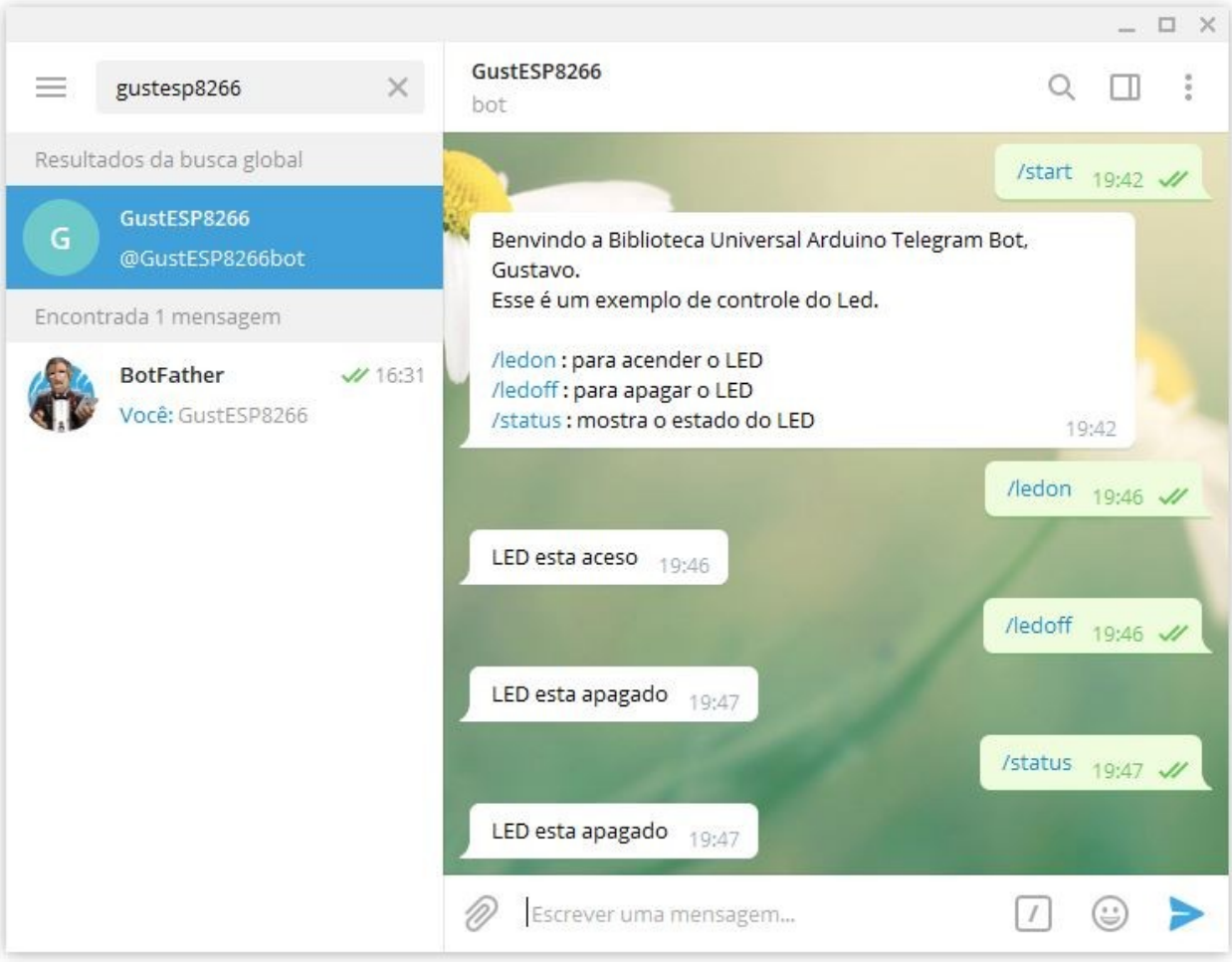
```
      handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    }

    Bot_lasttime = millis();
  }
}
```

Copie o Sketch acima na **Arduino IDE**. Conecte a placa NodeMCU no seu PC. Certifique-se que ela foi reconhecida. Após compilar e carregar o programa no **NodeMCU-ESP12** (clique no botão **Carregar**), abra a janela da console (Serial Monitor) da IDE. Clique no botão **Monitor serial** e altere a velocidade para **115200 Bps** na barra inferior da janela.

No aplicativo TELEGRAM (poderá testar no seu PC ou no seu Smartphone) , procure pelo nome do seu usuário BOT . No meu caso **GustESP8266**. Clique no nome para abrir a janela de comunicação. Clique em **Começar** para iniciar os testes. Sempre que executar algum comando, aguarde alguns segundos para obter a resposta !

Comandos para teste (pode clicar em cima dos comandos já digitados) :

- **/ledon** – para acender o LED
- **/ledoff** – para apagar o LED
- **/status** – para mostrar o estado do LED
- **/start** – para inciar a comunicação



Projeto NodeMCU ESP-12 – Alarme Residencial IOT:

Esse projeto de Alarme Residencial IOT é experimental. Verifique questões de segurança, e crie soluções se houver necessidade.

Nesse projeto, a placa NodeMCU ESP-12 será transformada em um Alarme Residencial IOT. Será usado somente um sensor de disparo, no caso um botão (conectado no pino D5 / GPIO_14). Mas esse botão poderá ser substituído por um outro tipo de sensor, como um sensor Infra-vermelho de movimento ou um sensor com contato magnético, etc. É possível acrescentar um número maior de sensores, para isso deverá fazer a implementação dos mesmos no programa. O LED Azul da placa NodeMCU será usado para visualização do alarme disparado. Quando o alarme for disparado, o LED azul acenderá. O projeto é o mais básico possível para facilitar o entendimento do mesmo. Mas à partir dele, poderá incluir um buzzer sonoro e muito mais outros recursos! Use sua criatividade.

Esse Alarme NodeMCU poderá ser controlado remotamente através do aplicativo TELEGRAM, tanto pelo smartphone como pelo seu PC. Você receberá uma mensagem, quando o Alarme for disparado. Poderá desativá-lo ou ativá-lo também através de comandos no TELEGRAM. É claro, para que ele possa enviar as mensagens, deverá ter acesso à internet (NodeMCU deverá estar conectado através do roteador WIFI).

Veja a foto do NodeMCU-ESP12 montado em Protoboard:

- Alimentação 3,3V – fio amarelo => fileira superior do Protoboard
- Terra(GND) – fio preto => fileira inferior do Protoboard
- Botão (sensor) – fio preto => fileira inferior do Protoboard
- Botão (sensor) – fio azul => pino D5 do NodeMCU
- Resistor 10K ohms – fileira superior do Protoboard => botão

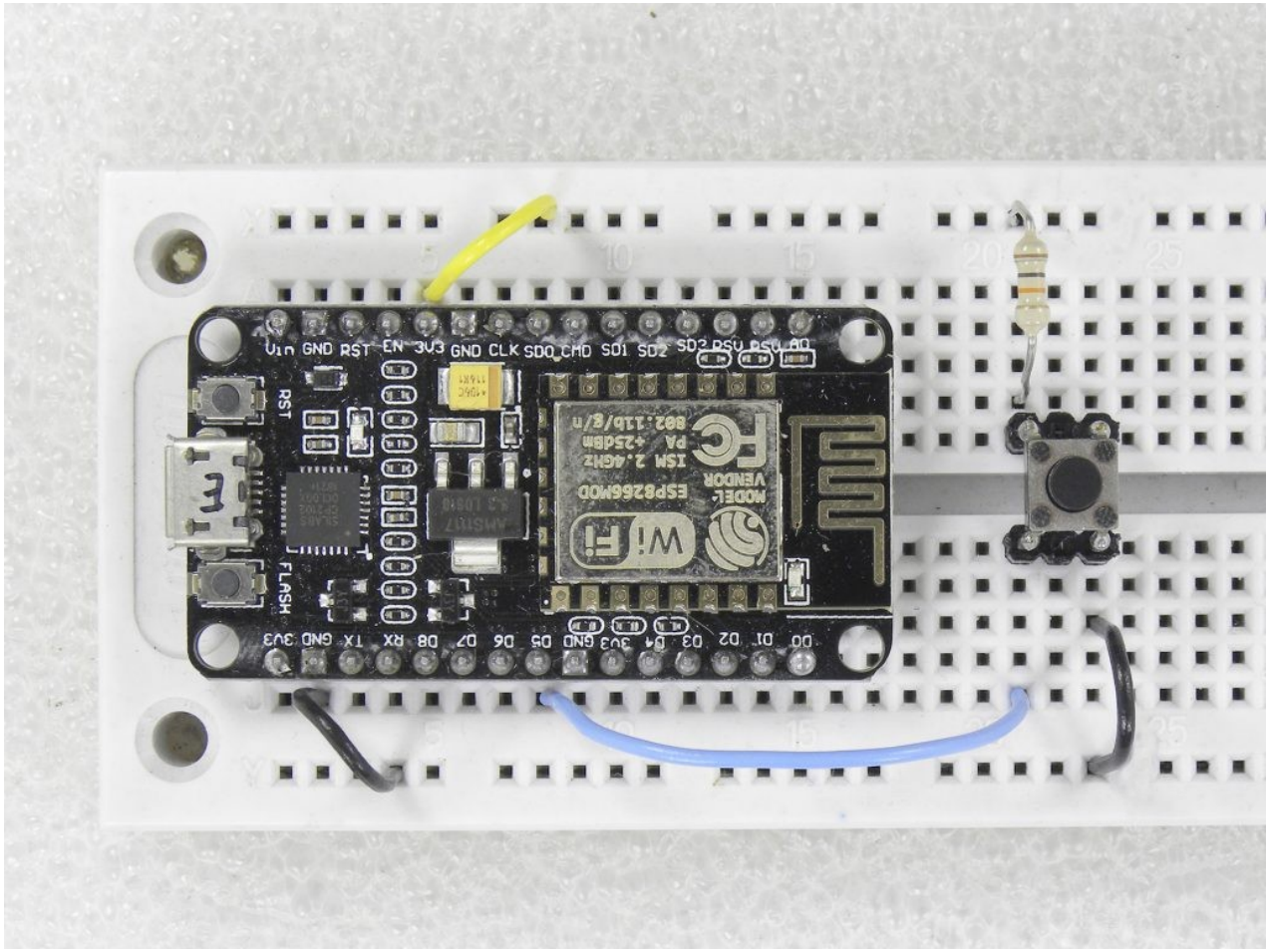


Foto : Gustavo Murta

Esse é o Sketch de exemplo que eu me baseei para fazer uma parte do programa que eu desenvolvi para o Alarme. A outra parte eu me baseei no exemplo anterior – Flash Led.

Sketch exemplo – Push-notifications-Arduino-ESP8266

Programa Alarme Residencial IOT – NodeMCU-ESP12 / TELEGRAM :

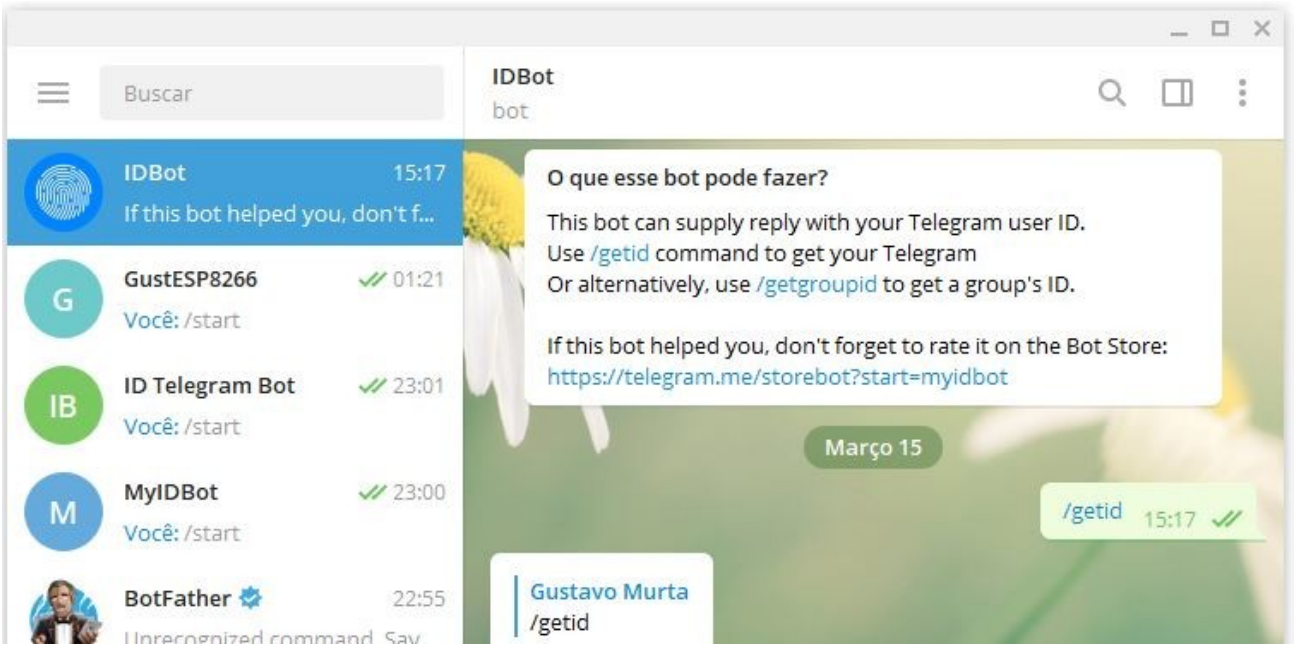
ESP8266AlarmeIOT.INO

IMPORTANTE : No sketch abaixo, você deverá também alterar o nome do seu roteador WIFI (**ssid**) e a senha do roteador (**password**) . Digite entre as aspas, exemplo = “eletrogate”.

Na linha do programa **#define BOTtoken,** insira a **chave Token** (entre aspas) que foi copiada quando gerou o seu usuário **BOT** através do **BotFather.** Exemplo:

“123456789:ABCDEFGHIJKLMOPQRST-ABCDEFGHIJKLMNO”

Você deverá preencher o **CHAT ID** na linha de programa **#define CHAT_ID “123456789”.** Insira o número entre as aspas. Para obter o CHAT ID, acesse o usuário **IDBot** no Telegram e digite **/getid.**




```
/*  ESP822 - NodeMCU Alarme Residencial IOT
    Arduino IDE 1.8.5 - ESP8266
    Gustavo Murta   14/mar/2018
    Baseado em:
    https://github.com/witnessmenow/push-notifications-arduino-esp8266/blob/master/PushNotificaitonDemo.ino
    https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot/blob/master/examples/ESP8266/FlashLED/FlashLED.ino

    Blog Eletrogate:
    http://blog.eletrogate.com/nodemcu-esp12-alarme-residencial-iot-3/

    Você pode copiar, distribuir e modificar o software desde que as modificações sejam descritas e licenciadas gratuitamente em LGPL-3.
    As obras derivadas (incluindo modificações ou qualquer coisa vinculada estaticamente à biblioteca) só podem ser redistribuídas no LGPL-3,
    mas as aplicações que utilizam a biblioteca não precisam ser.
*/


#include <UniversalTelegramBot.h>
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>

// Configurando a conexao WIFI com o Roteador
char ssid[] = "roteador WIFI";        // nome do seu roteador WIFI (SSID)
char password[] = "senha do WIFI";    // senha do roteador WIFI

#define TELEGRAM_SENSOR_PIN D5        // Sensor de disparo do alarme => D5 = GPIO_14

// Inicializa o BOT Telegram - copie aqui a chave Token quando configurou o seu BOT - entre aspas
#define BOT_TOKEN "123456789:ABCDEFGHIJKLMOPQRST-ABCDEFGHIJKLMNO"          // sua chave Token Bot

#define CHAT_ID "123456789" // Para obter o Chat ID, acesse Telegram => usuario IDBot => comando /getid

// cliente SSL necessario para a Biblioteca
WiFiClientSecure client;
```

```
UniversalTelegramBot bot(BOT_TOKEN, client);

String ipAddress = "";
volatile bool telegramSensorPressedFlag = false;
const int ledPin = 16;    // GPIO_16 do LED azul
int alarmeAtive = 0;      // Alarme Ativado
int alarmeTriggered = 0;  // Alarme disparado
int Bot_mtbs = 1000;      // tempo entre a leitura das mensagens
long Bot_lasttime;        // ultima mensagem lida
//bool Start = false;

void setup()
{
  Serial.begin(115200);

  // Inicializa o botao e o LED
  pinMode(TELEGRAM_SENSOR_PIN, INPUT);      // define o Sensor de disparo como
entrada
  pinMode(ledPin, OUTPUT);                  // configura o pino do LED como
saida
  delay(10);
  digitalWrite(ledPin, HIGH);               // inicializa o LED como apagado

  // interrupcao no Sensor pino D5 dispara Alarme
  attachInterrupt(TELEGRAM_SENSOR_PIN, telegramSensorPressed, RISING);

  WiFi.mode(WIFI_STA);      // Configura o WIFI do NodeMCU para modo estação
  WiFi.disconnect();        // desconecta o WIFI
  delay(100);               // atraso de 100 milisegundos

  Serial.print("Conectando no Wifi: ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED)    // aguardando a conexão WEB
  {
    Serial.print(".");
    delay(500);                            // atraso de 0,5 segundos
  }
  Serial.println("");
  Serial.println("WiFi conectado !");        // WIFI conectado
  Serial.print("IP address: ");
  IPAddress ip = WiFi.localIP();
  Serial.println(ip);                       // imprime Endereco IP
  ipAddress = ip.toString();

}

void telegramSensorPressed()
{
  if  (alarmeAtive)          // Se o alarme estiver ativado
  {
    Serial.println("Alarme disparado!");
    int sensor = digitalRead(TELEGRAM_SENSOR_PIN);  // verifica estado do sensor
    if (sensor == HIGH)
    {
      telegramSensorPressedFlag = true;     // Sensor do Alarme foi acionado
      digitalWrite(ledPin, LOW);            // acende LED azul do Alarme
      alarmeTriggered = 1;                  // Alarme disparado
    }
  }
  return;
}

void sendTelegramMessage() {
```

```
  String message = "Alarme Residencial disparado !";
  message.concat("\n");
  message.concat("Zona 1 - portão da garagem");
  message.concat("\n");
  message.concat("SSID: ");
  message.concat(ssid);
  message.concat(" IP: ");
  message.concat(ipAddress);
  message.concat("\n");
  if (bot.sendMessage(CHAT_ID, message, "Markdown"))
  {
    Serial.println("Mensagem Telegram enviada com sucesso");
  }
  telegramSensorPressedFlag = false;
}

void handleNewMessages(int numNewMessages)
{
  Serial.print("Mensagem recebida = ");
  Serial.println(String(numNewMessages));

  for (int i = 0; i < numNewMessages; i++)
  {
    String chat_id = String(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 == "/alarmeon")                                  //  comando Ativa
Alarme
    {
      alarmeAtive = 1;                                        // ativa alarme
      alarmeTriggered = 0;                                    // desliga disparo
do Alarme
      bot.sendMessage(chat_id, "Alarme foi Ativado !", "");     // envia
mensagem
    }

    if (text == "/alarmeoff")                                 //  comando Desativa
Alarme
    {
      alarmeAtive = 0;                                        // desativa alarme
      alarmeTriggered = 0;                                    // desliga disparo
do Alarme
      digitalWrite(ledPin, HIGH);                             // apaga LED azul
Alarme
      bot.sendMessage(chat_id, "Alarme foi Desativado !", "");  // envia
mensagem
    }

    if (text == "/status")                                    // comando estado do
Alarme
    {
      if (alarmeAtive)                                        // Se o alarme
estiver ativado
      {
        bot.sendMessage(chat_id, "Alarme está Ativado", "");
      }
      else
      {
        bot.sendMessage(chat_id, "Alarme está Desativado", "");
      }
```

```
      if (alarmeTriggered) bot.sendMessage(chat_id, "Alarme está Disparado !",
"");
    }

    if (text == "/start")                                  // comando comeca
    {
      alarmeAtive = 1;                                      // ativa alarme
      String welcome = "Alarme Residencial IOT operacional, " + from_name +
".\n";
      welcome += "Alarme foi Ativado!\n\n";
      welcome += "/alarmeon : para ativar Alarme\n";
      welcome += "/alarmeoff : para desativar Alarme\n";
      welcome += "/status : mostra o estado do Alarme\n";
      bot.sendMessage(chat_id, welcome, "Markdown");
    }
  }
}

void loop()
{
  if ( telegramSensorPressedFlag )
  {
    sendTelegramMessage();
  }
  if (millis() > Bot_lasttime + Bot_mtbs)    // controlando as mensagens
  {
    int numNewMessages = bot.getUpdates(bot.last_message_received + 1);

    while (numNewMessages)                  // numero de novas mensagens
    {
      Serial.println("Mensagem recebida do Telegram");
      handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    }
    Bot_lasttime = millis();
  }
}
```

Copie o Sketch acima na **Arduino IDE**. Conecte a placa NodeMCU no seu PC. Certifique-se que ela foi reconhecida. Após compilar e carregar o programa no **NodeMCU-ESP12** (clique no botão **Carregar**), abra a janela da console (Serial Monitor) da IDE. Clique no botão **Monitor serial** e altere a velocidade para **115200 Bps** na barra inferior da janela.

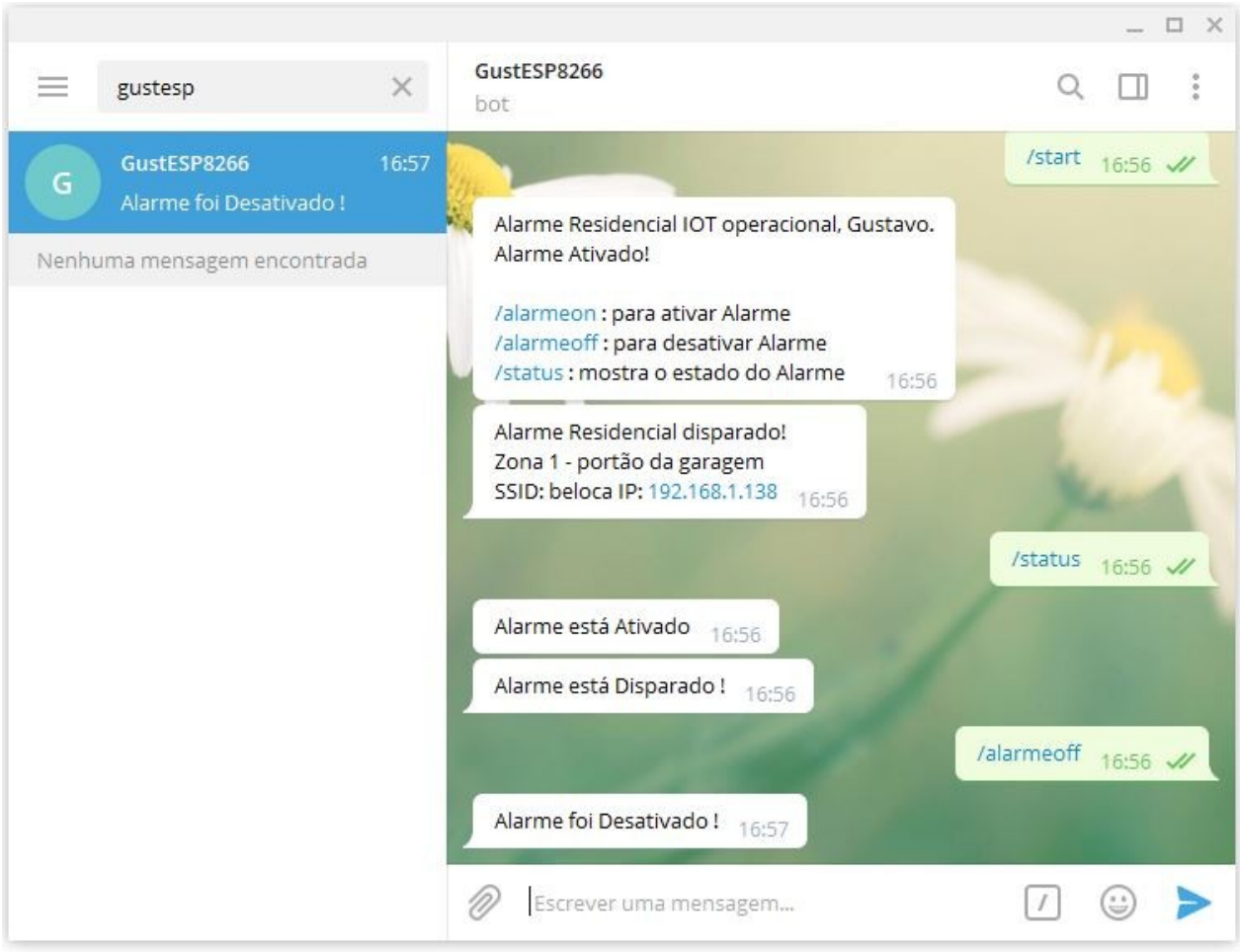
No aplicativo **TELEGRAM**, procure pelo nome do seu usuário BOT . No meu caso **GustESP8266**. Clique no nome para abrir a janela de comunicação. Clique em **Começar** para inicializar o Alarme. Sempre que executar algum comando, aguarde alguns segundos para obter a resposta !

Comandos do Alarme Residencial IOT (pode clicar em cima dos comandos já digitados) :

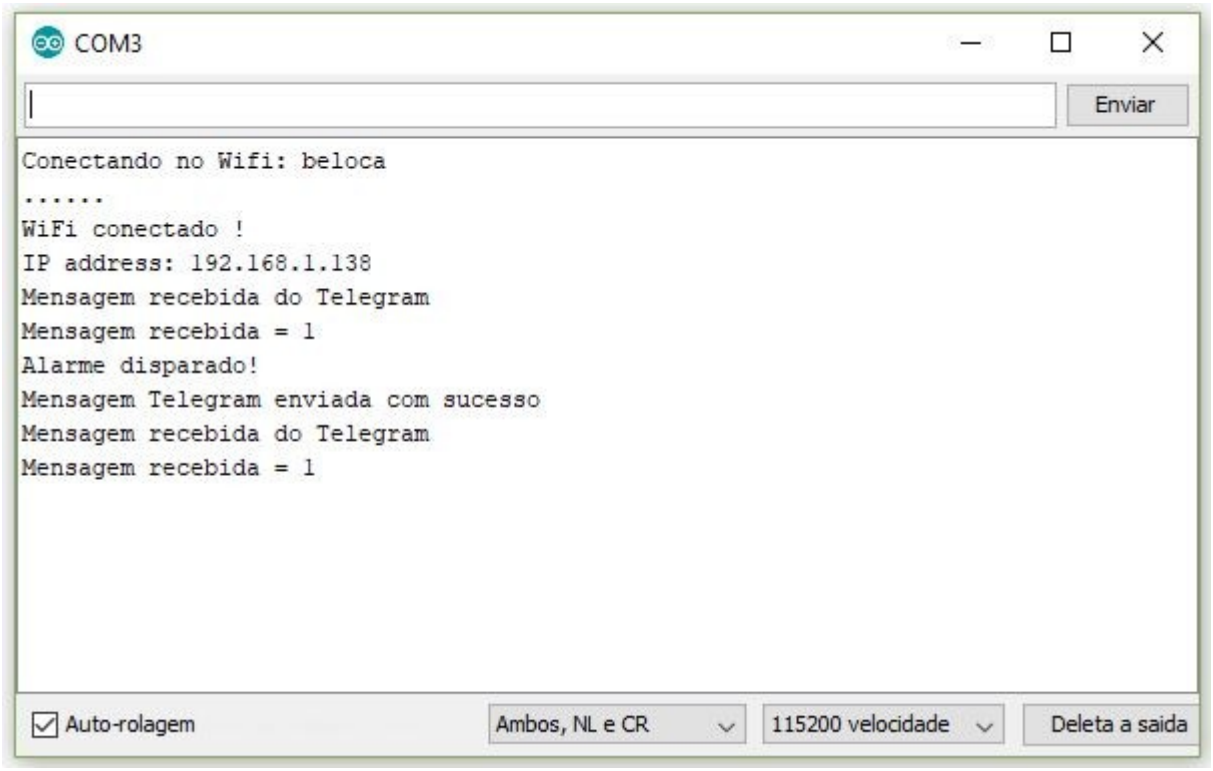
/alarmeon : para ativar Alarme
/alarmeoff : para desativar Alarme
/status : mostra o estado do Alarme
/start : para reiniciar o Alarme

Essa é uma tela do TELEGRAM para exemplificar o uso do Alarme. Ativei o Alarme clicando em **Começar**. Apertei o botão do sensor – **Alarme foi disparado** e o LED azul acendeu. Dei o comando de **/status** e apareceu as mensagens : **Alarme Ativado e Alarme Disparado !** Dei o

comando **/alarmeoff** para desativar o alarme. Para ativar novamente o alarme, dê o comando **/alarmeon**.



Através da Console da **Arduino IDE (Monitor Serial),** poderá monitorar todas as atividades do Alarme Residencial IOT (fase de testes).



Se tiver alguma dúvida, deixe um comentário!

Tutoriais sobre NodeMCU – ESP12 :

NodeMCU – ESP12 – Introdução (1)

NodeMCU – ESP12 – Usando Arduino IDE (2)

Avaliações: **5.0**. de 1 voto.