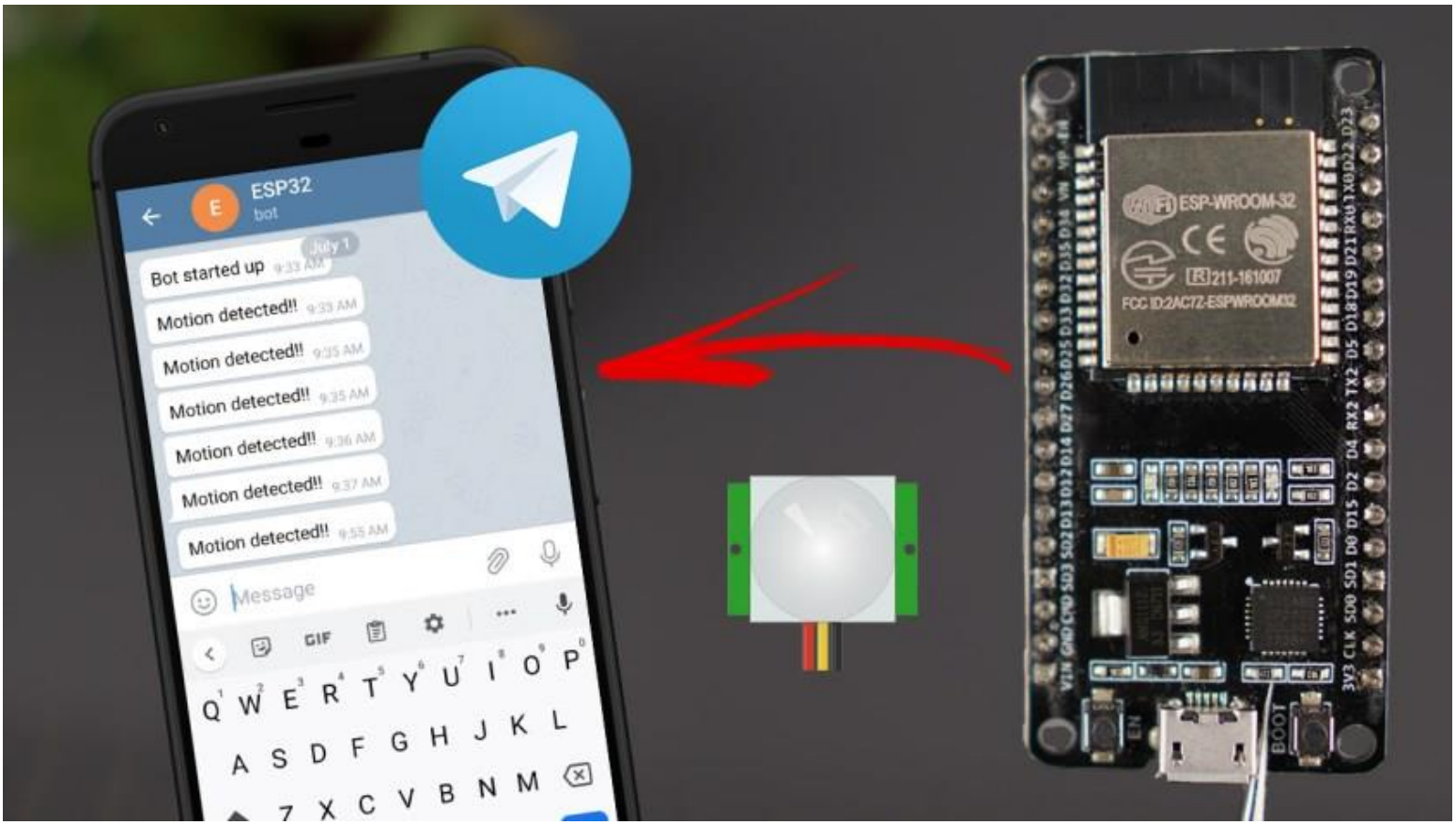
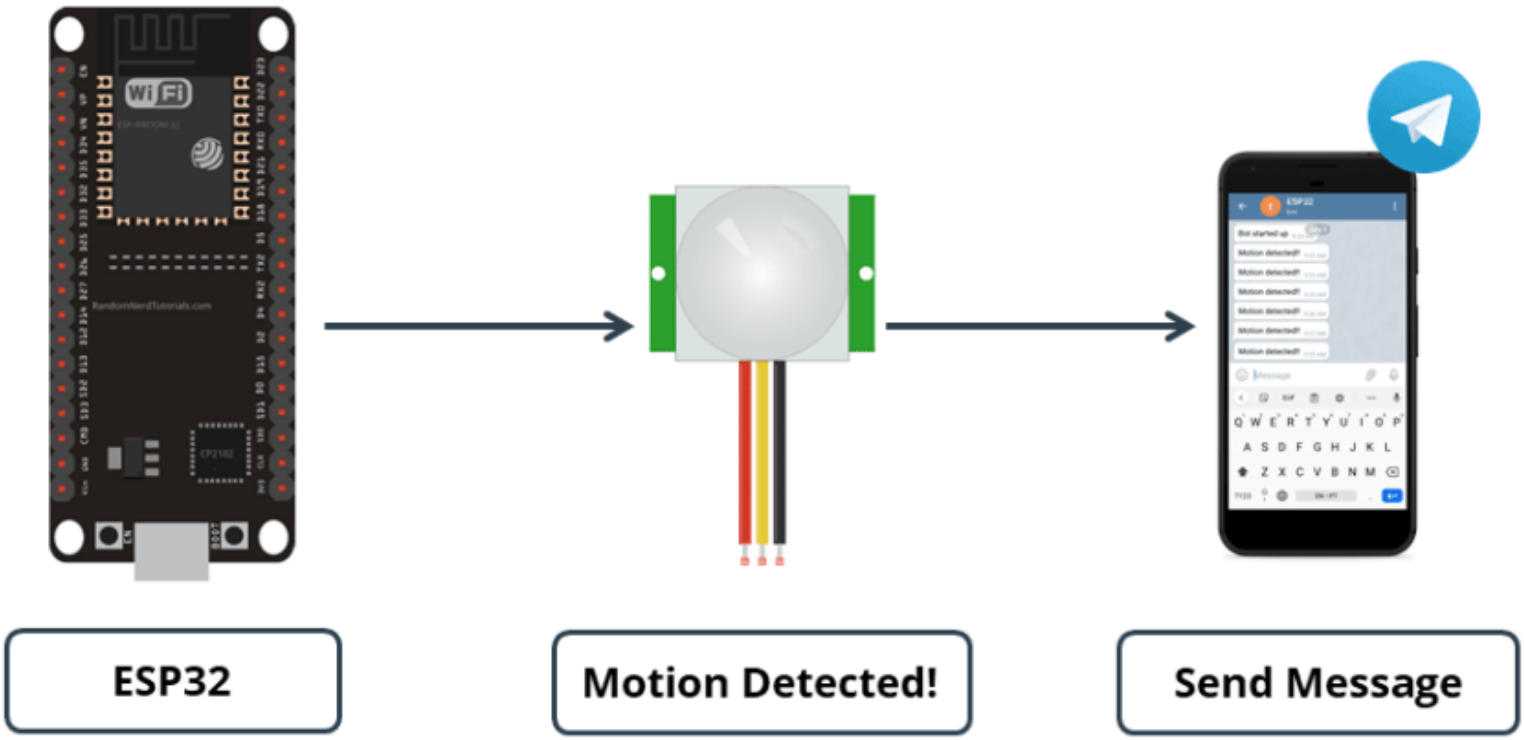
Telegram: ESP32 Motion Detection with Notifications (Arduino IDE)

This tutorial shows how to send notifications to your Telegram account when the ESP32 detects motion. As long as you have access to the internet in your smartphone, you'll be notified no matter where you are. The ESP board will be programmed using Arduino IDE.



Project Overview

This tutorial shows how to get notifications in your Telegram account when the ESP32 detects motion.



Here’s an overview on how the project works:

- You’ll create a Telegram bot for your ESP32.
- The ESP32 is connected to a PIR motion sensor.
- When the sensor detects motion, the ESP32 sends a warning message to your telegram account.
- You’ll be notified in your telegram account whenever motion is detected.

This is a simple project, but shows how you can use Telegram in your IoT and Home Automation projects. The idea is to apply the concepts learned in your own projects.

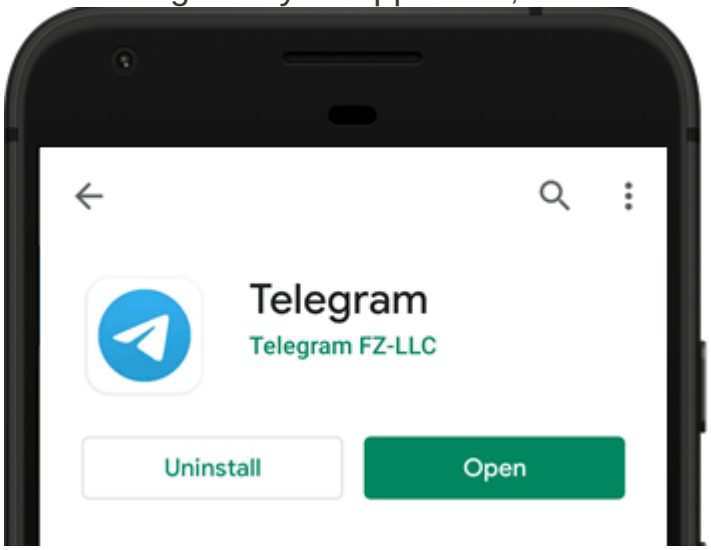
Introducing Telegram

Telegram Messenger is a cloud-based instant messaging and voice over IP service. You can easily install it in your smartphone (Android and iPhone) or computer (PC, Mac and Linux). It is free and without any ads. Telegram allows you to create bots that you can interact with.
“*Bots are third-party applications that run inside Telegram. Users can interact with bots by sending them messages, commands and inline requests. You control your bots using HTTPS requests to Telegram Bot API*“.

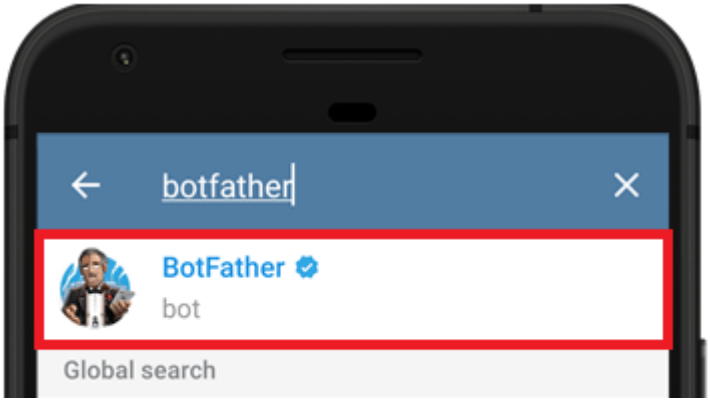
The ESP32 will interact with the Telegram bot to send messages to your telegram account. Whenever motion is detected, you'll receive a notification in your smartphone (as long as you have access to the internet).

Creating a Telegram Bot

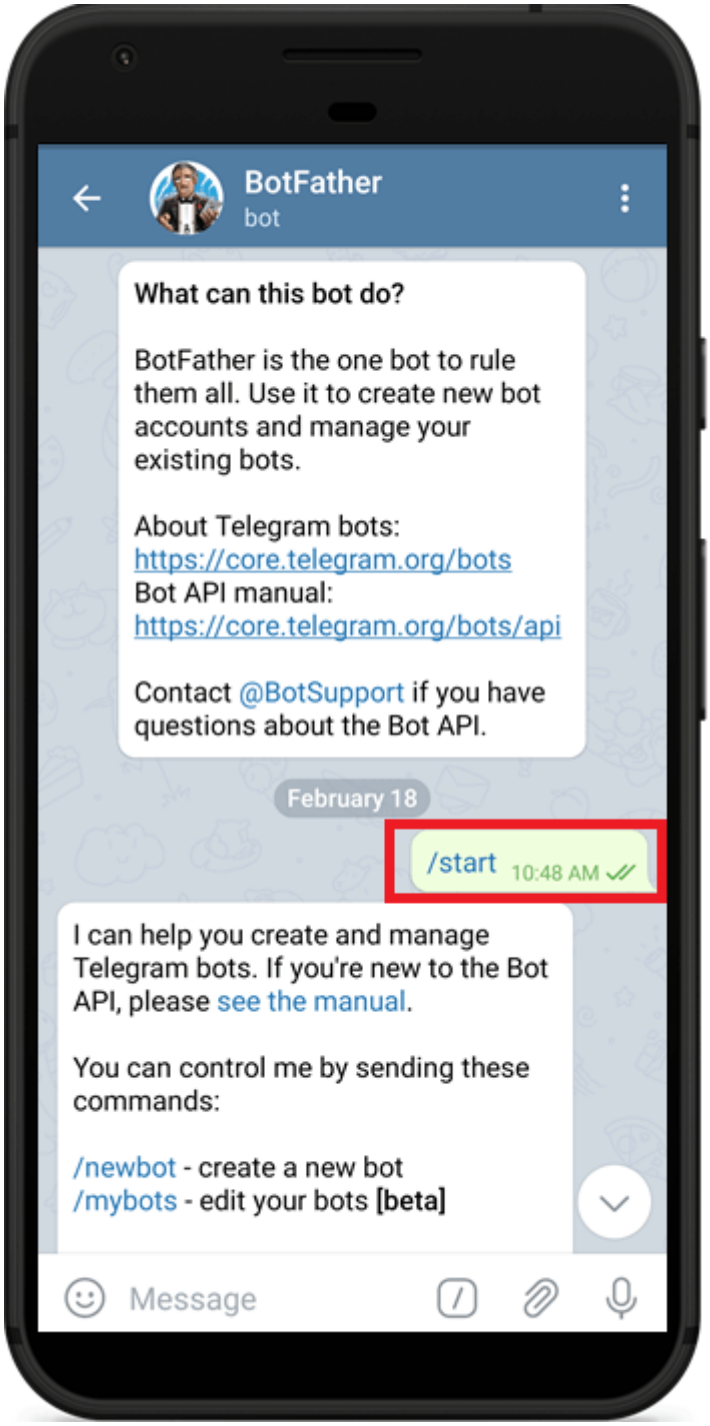
Go to Google Play or App Store, download and install **Telegram**.



Open Telegram and follow the next steps to create a Telegram Bot. First, search for "**botfather**" and click the BotFather as shown below.

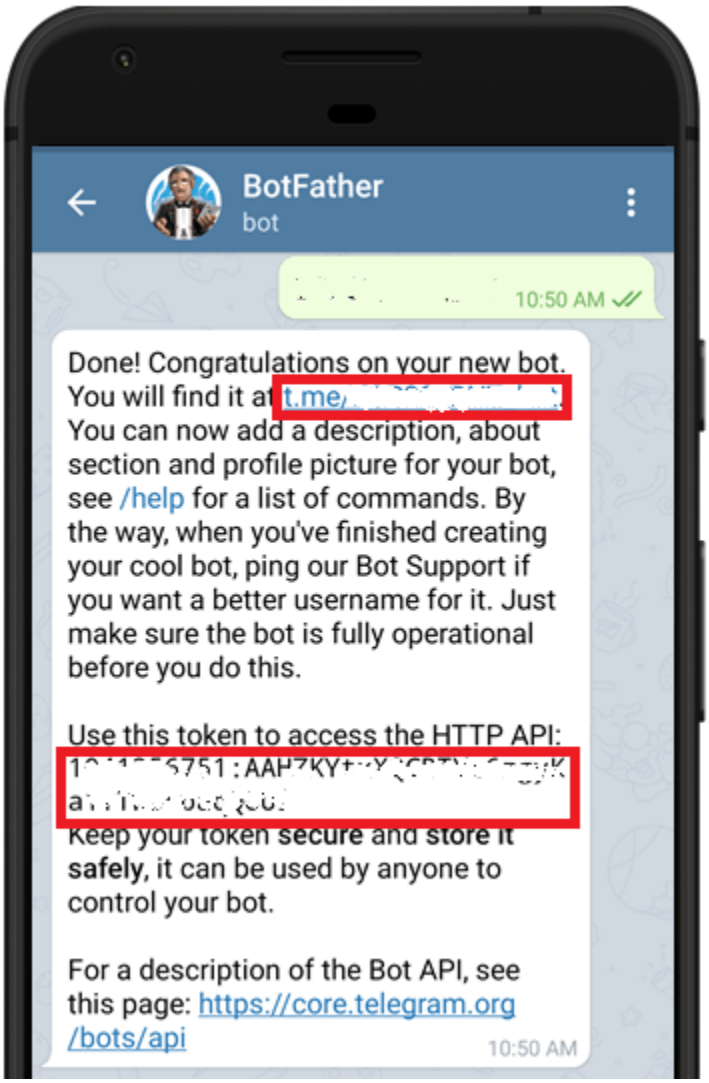


The following window should open and you'll be prompted to click the **start** button.



Type **/newbot** and follow the instructions to create your bot. Give it a name and username.

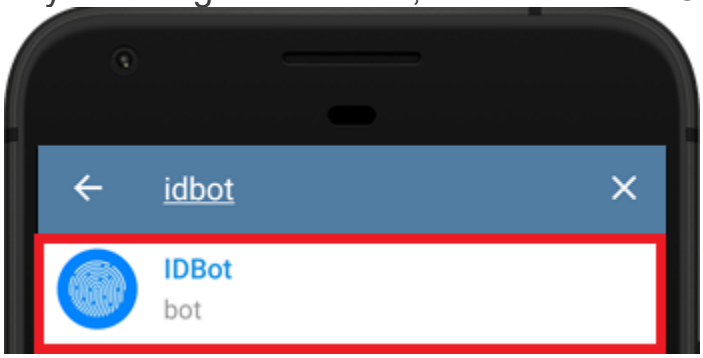
If your bot is successfully created, you'll receive a message with a link to access the bot and the **bot token**. Save the bot token because you'll need it so that the ESP32 can interact with the bot.



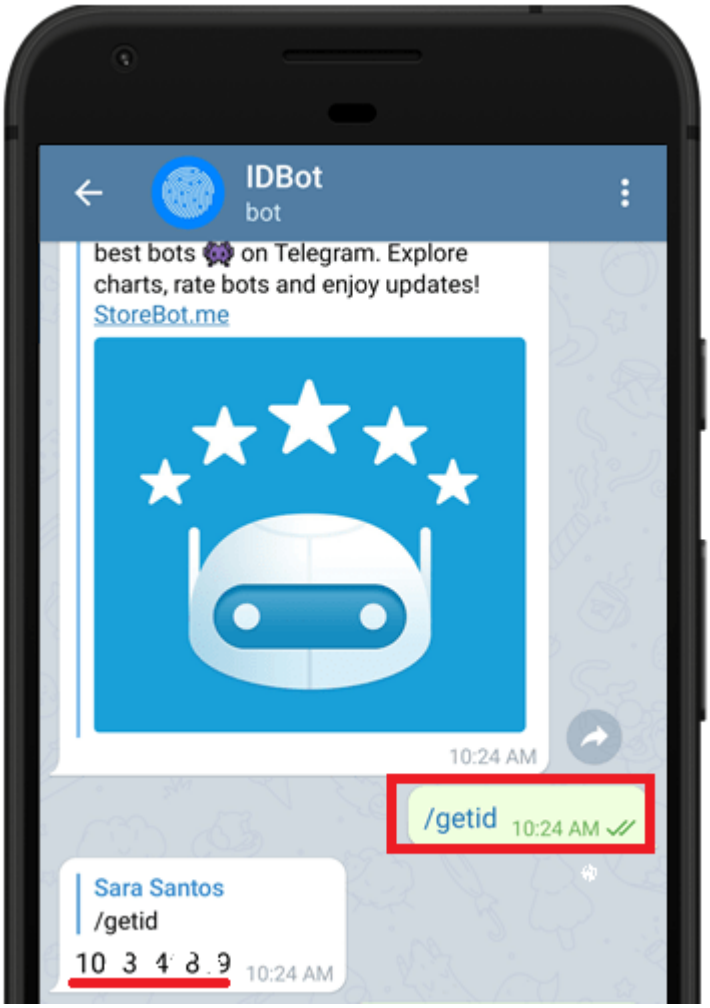
Get Your Telegram User ID

Anyone that knows your bot username can interact with it. To make sure that we ignore messages that are not from our Telegram account (or any authorized users), you can get your Telegram User ID. Then, when your telegram bot receives a message, the ESP can check whether the sender ID corresponds to your User ID and handle the message or ignore it.

In your Telegram account, search for “**IDBot**“



Start a conversation with that bot and type **/getid**. You will get a reply back with your user ID. Save that **user ID**, because you'll need it later in this tutorial.



Preparing Arduino IDE

We'll program the ESP32 board using Arduino IDE, so make sure you have them installed in your Arduino IDE.

Universal Telegram Bot Library

To interact with the Telegram bot, we'll use the Universal Telegram Bot Library created by Brian Lough that provides an easy interface for the Telegram Bot API.
Follow the next steps to install the latest release of the library.

1. Click here to download the Universal Arduino Telegram Bot library.
2. Go to **Sketch** > **Include Library** > **Add.ZIP Library...**.
3. Add the library you've just downloaded.

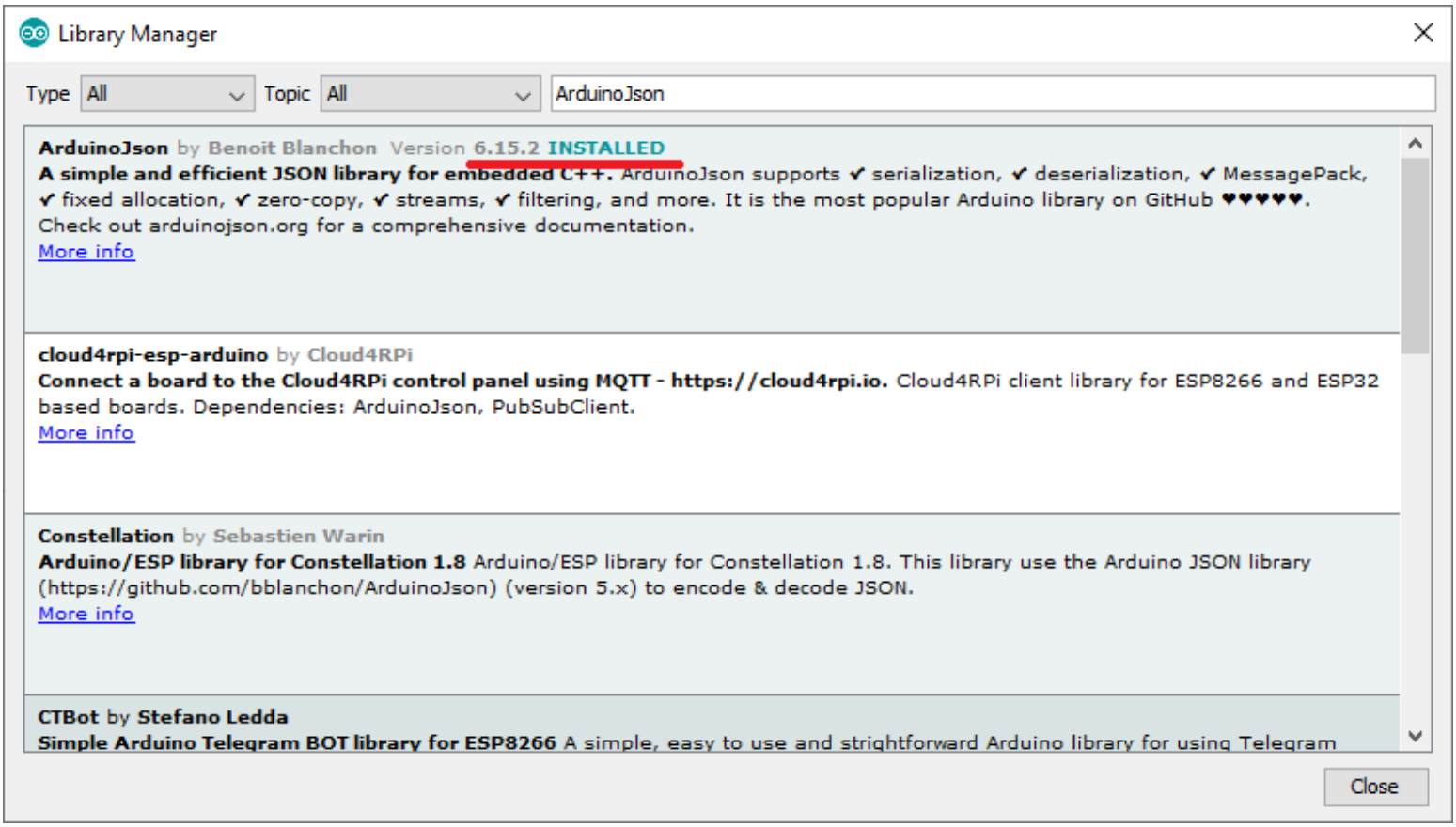
Important: don't install the library through the Arduino Library Manager because it might install a deprecated version.
For all the details about the library, take a look at the Universal Arduino Telegram Bot Library GitHub page.

ArduinoJson Library

You also have to install the ArduinoJson library. Follow the next steps to install the library.

1. Go to **Skech** > **Include Library** > **Manage Libraries**.
2. Search for "ArduinoJson".
3. Install the library.

We're using ArduinoJson library version 6.5.12.



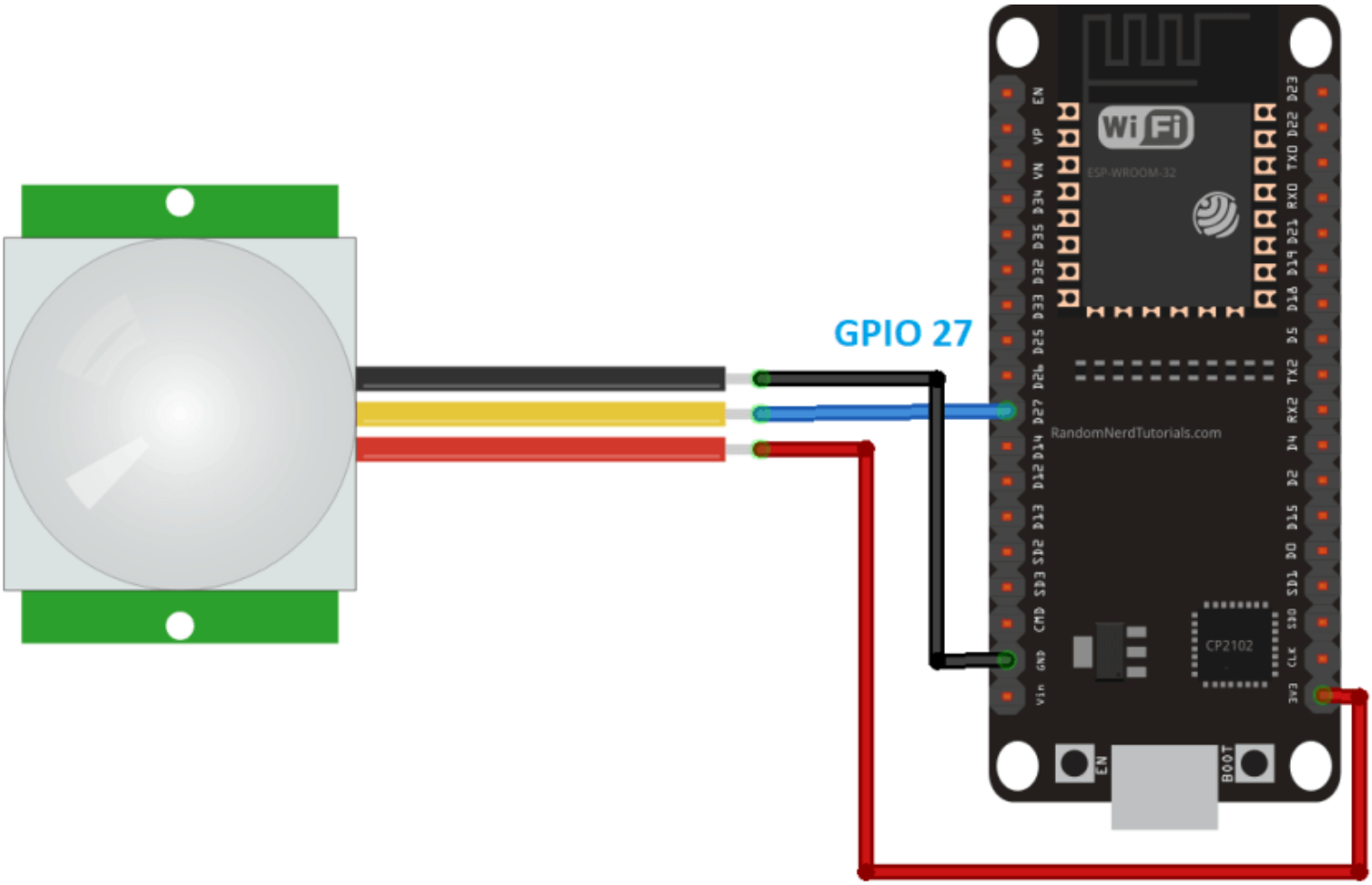
Parts Required

For this project, you need the following parts:

- ESP32 board (read Best ESP32 dev boards)
- Mini PIR motion sensor (AM312) or PIR motion sensor (HC-SR501)
- Jumper wires
- Breadboard

Schematic Diagram

For this project you need to wire a PIR motion sensor to your ESP32 board. Follow the next schematic diagram.



In this example, we're wiring the PIR motion sensor data pin to GPIO 27. You can use any other suitable GPIO.

Telegram Motion Detection with Notifications – ESP32 Sketch

The following code uses your Telegram bot to send a warning message to your telegram account whenever motion is detected. To make this sketch work for you, you need to insert your network credentials (SSID and password), the Telegram Bot token and your Telegram user ID.

```
/*
  Rui Santos



  Project created using Brian Lough's Universal Telegram Bot Library:
https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot
*/

#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

// Replace with your network credentials
const char* ssid = "REPLACE_WITH_YOUR_SSID";
const char* password = "REPLACE_WITH_YOUR_PASSWORD";

// Initialize Telegram BOT
#define BOTtoken "XXXXXXXXXX:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"  //
your Bot Token (Get from Botfather)

// Use @myidbot to find out the chat ID of an individual or a group
// Also note that you need to click "start" on a bot before it can
// message you
#define CHAT_ID "XXXXXXXXXX"

WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);

const int motionSensor = 27; // PIR Motion Sensor
bool motionDetected = false;

// Indicates when motion is detected
void IRAM_ATTR detectsMovement() {
  //Serial.println("MOTION DETECTED!!!");
  motionDetected = true;
}

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

  // PIR Motion Sensor mode INPUT_PULLUP
  pinMode(motionSensor, INPUT_PULLUP);
  // Set motionSensor pin as interrupt, assign interrupt function and
set RISING mode
  attachInterrupt(digitalPinToInterrupt(motionSensor), detectsMovement,
RISING);

  // Attempt to connect to Wifi network:
  Serial.print("Connecting Wifi: ");
  Serial.println(ssid);
```

```
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  bot.sendMessage(CHAT_ID, "Bot started up", "");
}

void loop() {
  if(motionDetected){
    bot.sendMessage(CHAT_ID, "Motion detected!!", "");
    Serial.println("Motion Detected");
    motionDetected = false;
  }
}
```

How the Code Works

This sections explain how the code works.
Start by importing the required libraries.

```
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>
```

Network Credentials

Insert your network credentials in the following variables.

```
const char* ssid = "REPLACE_WITH_YOUR_SSID";
const char* password = "REPLACE_WITH_YOUR_PASSWORD";
```

Telegram Bot Token

Insert your Telegram Bot token you've got from Botfather on the `BOTtoken` variable.

```
#define BOTtoken "XXXXXXXXXX:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"  //
your Bot Token (Get from Botfather)
```

Telegram User ID

Insert your chat ID. The one you've got from the IDBot.

```
#define CHAT_ID "XXXXXXXXXX"
```

Create a new WiFi client with `WiFiClientSecure`.

```
WiFiClientSecure client;
```

Create a `bot` with the token and client defined earlier.

```
UniversalTelegramBot bot(BOTtoken, client);
```

Motion Sensor

Define the GPIO that the motion sensor is connected to.

```
const int motionSensor = 27; // PIR Motion Sensor
```

The `motionDetected` boolean variable is used to indicate whether motion was detected or not. It is set to `false` by default.

```
bool motionDetected = false;
```

detectsMovement()

The `detectsmovement()` function is a callback function that will be executed when motion is detected. In this case, it simply changes the state of the `motionDetected` variable to `true`.

```
void IRAM_ATTR detectsMovement() {
  //Serial.println("MOTION DETECTED!!!");
  motionDetected = true;
}
```

setup()

In the `setup()`, initialize the Serial Monitor.

```
Serial.begin(115200);
```

PIR Motion Sensor Interrupt

Set the PIR motion sensor as an interrupt and set the `detectsMovement()` as the callback function (when motion is detected, that function will be executed):

```
// PIR Motion Sensor mode INPUT_PULLUP
pinMode(motionSensor, INPUT_PULLUP);
// Set motionSensor pin as interrupt, assign interrupt function and set
RISING mode
attachInterrupt(digitalPinToInterrupt(motionSensor), detectsMovement,
RISING);
```

Init Wi-Fi

Initialize Wi-Fi and connect the ESP32 to your local network with the SSID and password defined earlier.

```
WiFi.mode(WIFI_STA);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.println("Connecting to WiFi..");
}
```

Finally, send a message to indicate that the Bot has started up:

```
bot.sendMessage(CHAT_ID, "Bot started up", "");
```

loop()

In the `loop()`, check the state of the `motionDetected` variable.

```
void loop() {
  if(motionDetected){
```

If it's `true`, it means that motion was detected. So, send a message to your Telegram account indicating that motion was detected.

```
bot.sendMessage(CHAT_ID, "Motion detected!!", "");
```

Sending a message to the bot is very simply. You just need to use the `sendMessage()` method on the `bot` object and pass as arguments the recipient's chat ID, the message, and the parse mode.

```
bool sendMessage(String chat_id, String text, String parse_mode = "")
```

Finally, after sending the message, set the `motionDetected` variable to `false`, so it can detect motion again.

```
motionDetected = false;
```

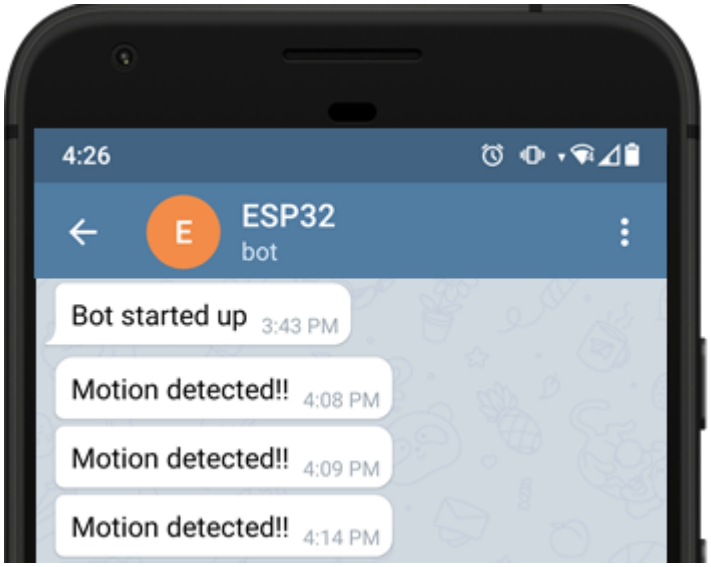
That's pretty much how the code works.

Demonstration

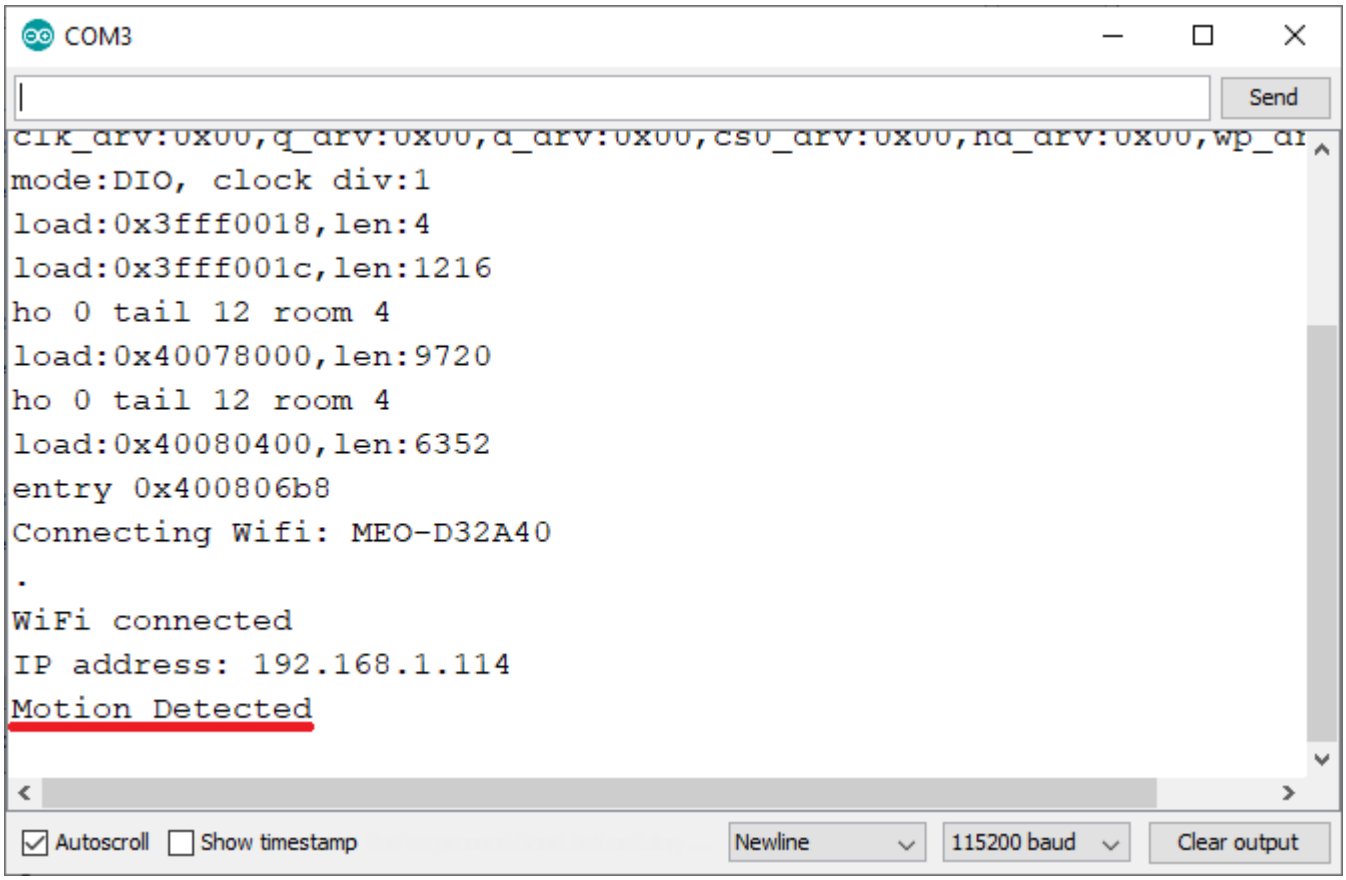
Important: go to your Telegram account and search for your bot. You need to click "**start**" on a bot before it can message you.

Upload the code to your ESP32 board. Don't forget to go to **Tools** > **Board** and select the board you're using. Go to **Tools** > **Port** and select the COM port your board is connected to. After uploading the code, press the ESP32 on-board EN/RST button so that it starts running the code. Then, you can open the Serial Monitor to check what's happening in the background.

When your board first boots, it will send a message to your Telegram account: “Bot started up”. Then, move your hand in front of the PIR motion sensor and check that you’ve received the motion detected notification.



At the same time, this is what you should get on the Serial Monitor.



Wrapping Up

In this tutorial you've learned how to create a Telegram Bot to interact with the ESP32 board. When motion is detected, a message is sent.

With this bot, you can also use your Telegram account to send messages to the ESP32 to control its outputs or request sensor readings, for example. The great thing about using Telegram to control your ESP boards, is that as long as you have an internet connection (and your boards too), you can control and monitor them from anywhere in the world.

We hope you've found this project interesting.

Thanks for reading.