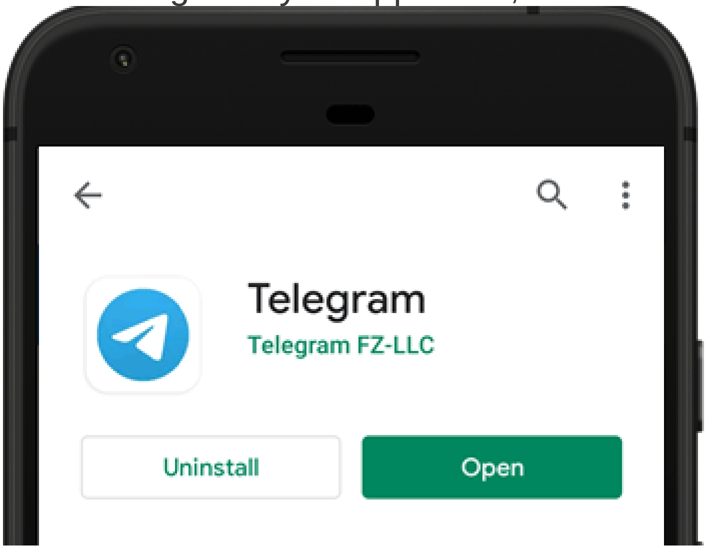
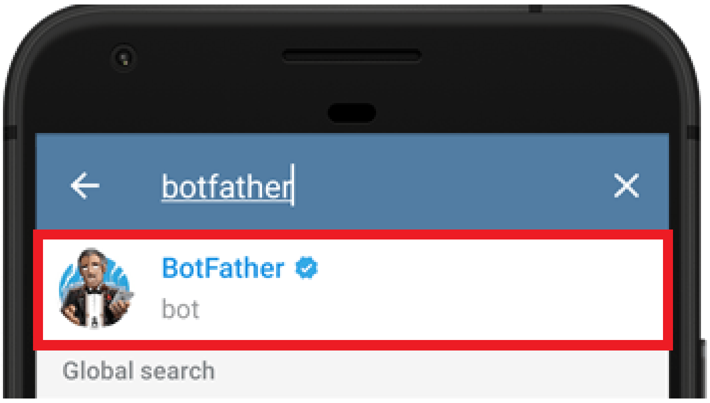
Creating a Telegram Bot

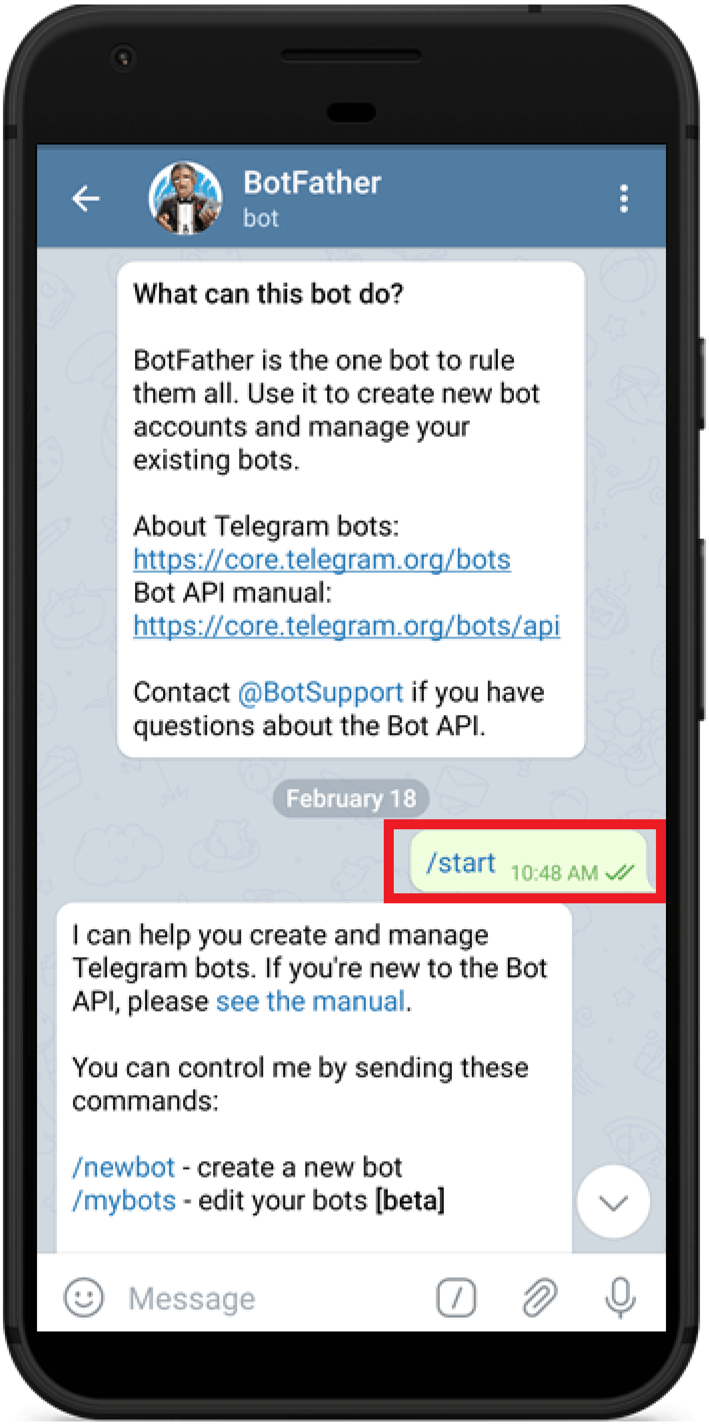
Go to Google Play or App Store, download and install **Telegram**.



Open Telegram and follow the next steps to create a Telegram Bot. First, search for “**botfather**” and click the BotFather as shown below. Or open this link t.me/botfather in your smartphone.



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



BotFather
bot

What can this bot do?

BotFather is the one bot to rule them all. Use it to create new bot accounts and manage your existing bots.

About Telegram bots:
https://core.telegram.org/bots
Bot API manual:
https://core.telegram.org/bots/api

Contact @BotSupport if you have questions about the Bot API.

February 18

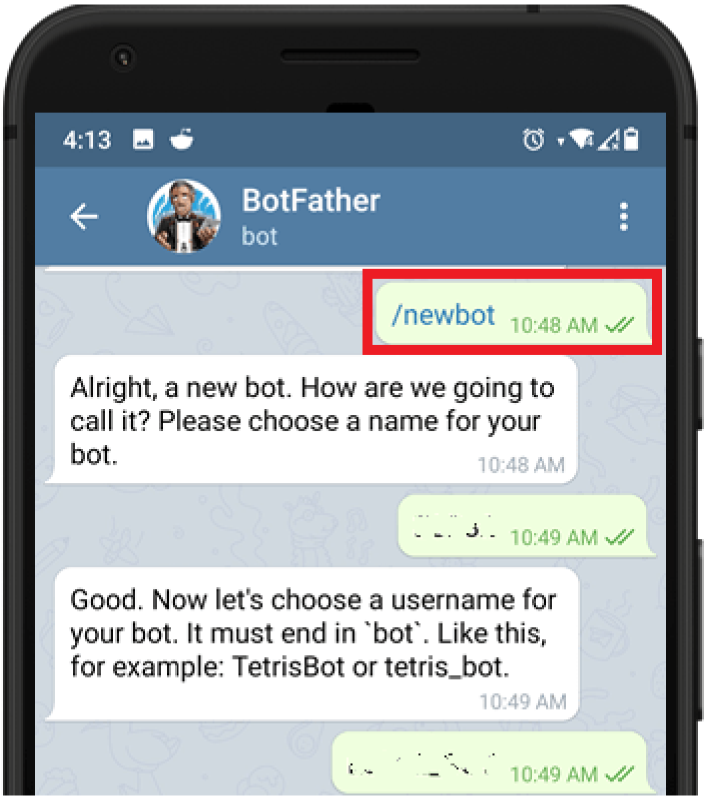
/start 10:48 AM

I can help you create and manage Telegram bots. If you're new to the Bot API, please see the manual.

You can control me by sending these commands:

/newbot - create a new bot
/mybots - edit your bots **[beta]**

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



/newbot 10:48 AM

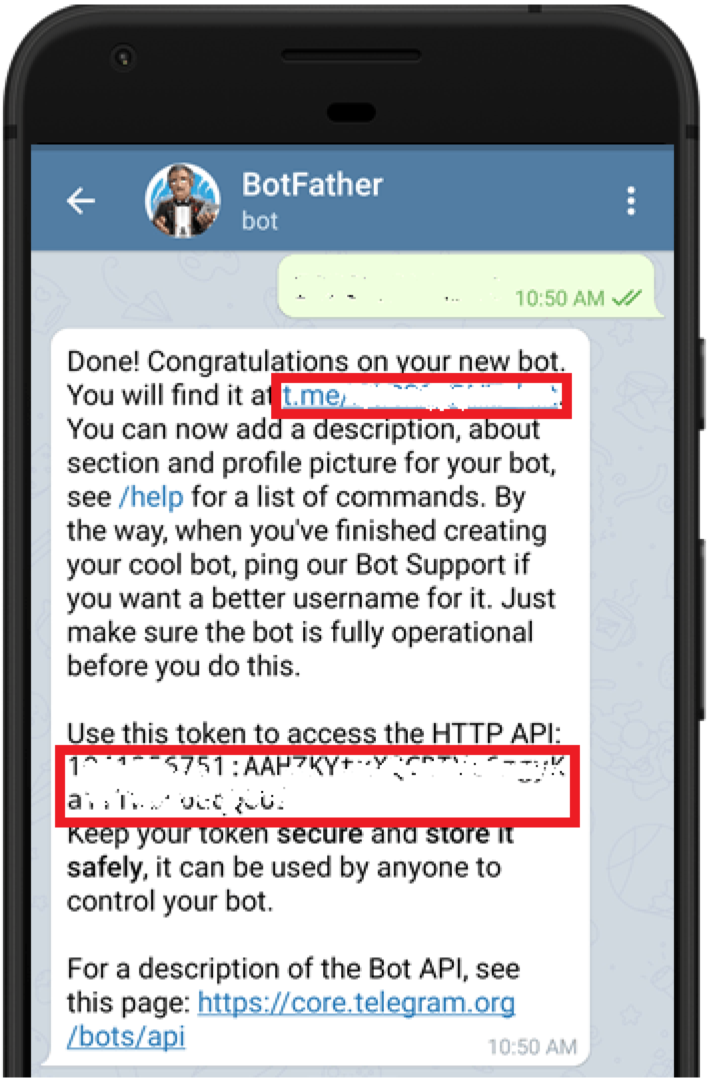
Alright, a new bot. How are we going to call it? Please choose a name for your bot. 10:48 AM

10:49 AM

Good. Now let's choose a username for your bot. It must end in `bot`. Like this, for example: TetrisBot or tetris_bot. 10:49 AM

10:49 AM

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.



[redacted] 10:50 AM

Done! Congratulations on your new bot. You will find it at t.me/[redacted] You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

Use this token to access the HTTP API:
[redacted]
Keep your token **secure** and **store it safely**, it can be used by anyone to control your bot.

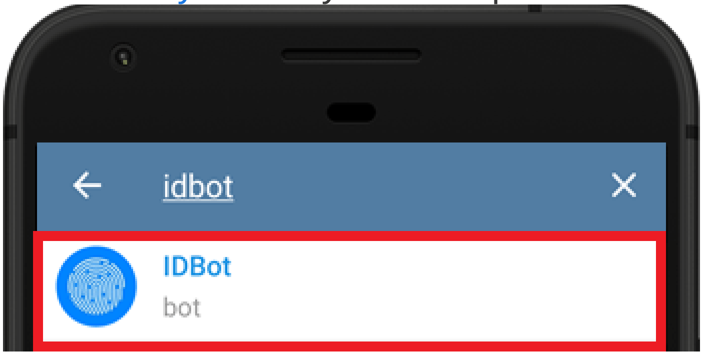
For a description of the Bot API, see this page: https://core.telegram.org/bots/api

10:50 AM

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.

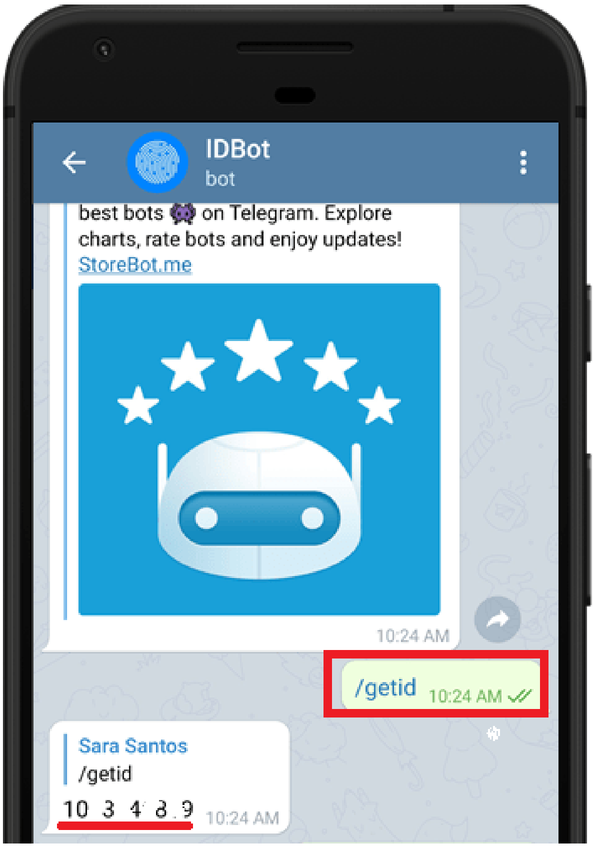
In your Telegram account, search for “IDBot” or open this link t.me/myidbot in your smartphone.



idbot

IDBot
bot

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.



IDBot
bot

best bots 👾 on Telegram. Explore charts, rate bots and enjoy updates!
StoreBot.me
10:24 AM

/getid 10:24 AM

Sara Santos
/getid
10 3 4 8 9 10:24 AM

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

Follow the next steps to install the latest release of the library.

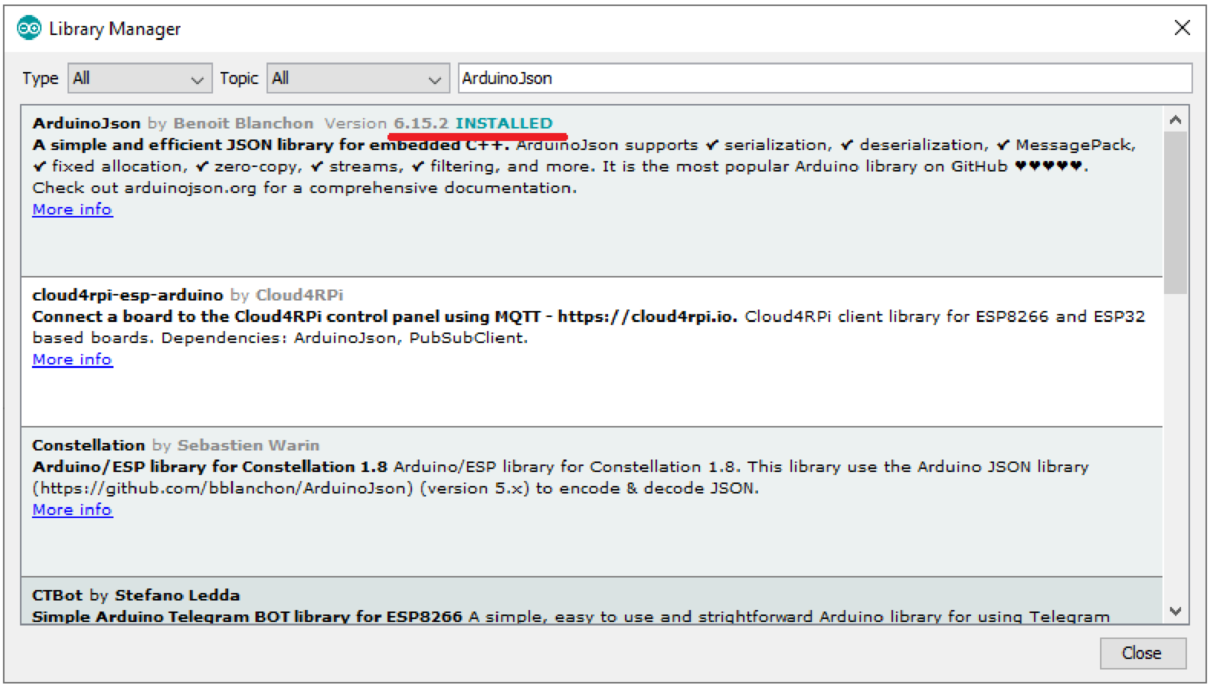
1. In the ZIP folder given there are Few Libraries provided for your project.
2. Go to **Sketch** > **Include Library** > **Add.ZIP Library...**
3. Add all the library in that folder one by one.

Arduino Json Library

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

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

We're using ArduinoJson library version 6.5.12.



ArduinoJson

ArduinoJson by Benoit Blanchon Version 6.15.2 INSTALLED
A simple and efficient JSON library for embedded C++. ArduinoJson supports ✓ serialization, ✓ deserialization, ✓ MessagePack, ✓ fixed allocation, ✓ zero-copy, ✓ streams, ✓ filtering, and more. It is the most popular Arduino library on GitHub ♥♥♥♥♥. Check out arduinojson.org for a comprehensive documentation.
More info

cloud4rpi-esp-arduino by Cloud4RPi
Connect a board to the Cloud4RPi control panel using MQTT - https://cloud4rpi.io. Cloud4RPi client library for ESP8266 and ESP32 based boards. Dependencies: ArduinoJson, PubSubClient.
More info

Constellation by Sebastien Warin
Arduino/ESP library for Constellation 1.8 Arduino/ESP library for Constellation 1.8. This library use the Arduino JSON library (https://github.com/bblanchon/ArduinoJson) (version 5.x) to encode & decode JSON.
More info

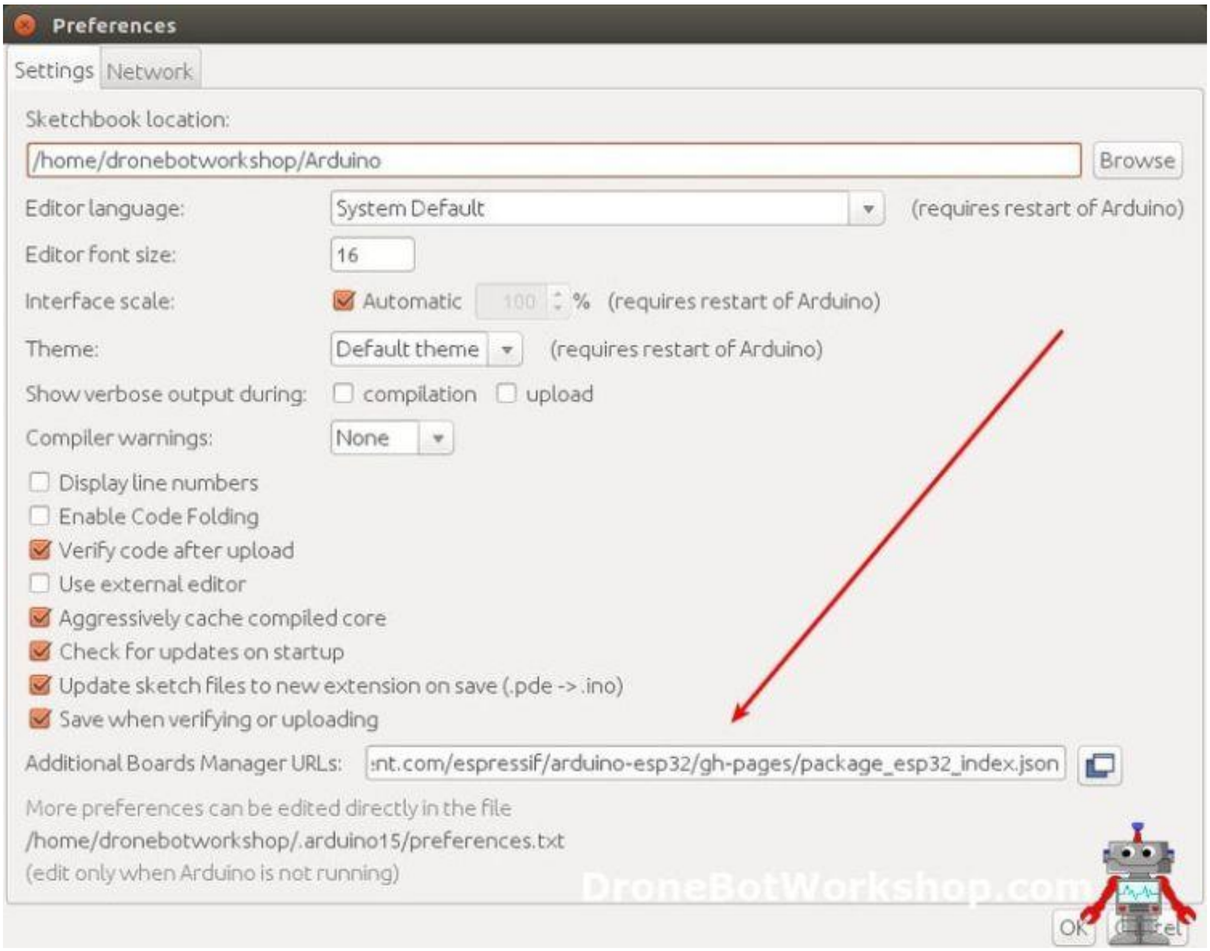
CTBot by **Stefano Ledda**
Simple Arduino Telegram BOT library for ESP8266 A simple, easy to use and strightforward Arduino library for using Telegram

Add Board Management Entry

Here is what you need to do to install the ESP32 boards into the Arduino IDE:

1. Open the Arduino IDE. Make sure that you are at version 1.8 or higher, if not then update your IDE with the latest version.
2. Click on the *File* menu on the top menu bar.
3. Click on the *Preferences* menu item. This will open a Preferences dialog box.
4. You should be on the *Settings* tab in the Preferences dialog box by default.
5. Look for the textbox labeled "*Additional Boards Manager URLs*".
6. If there is already text in this box add a coma at the end of it, then follow the next step.
7. Paste the following link into the text box – ***https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json***
8. Click the OK button to save the setting.

The textbox with the JSON link in it is illustrated here:



Sketchbook location: /home/dronebotworkshop/Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 16

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Theme: Default theme (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings: None

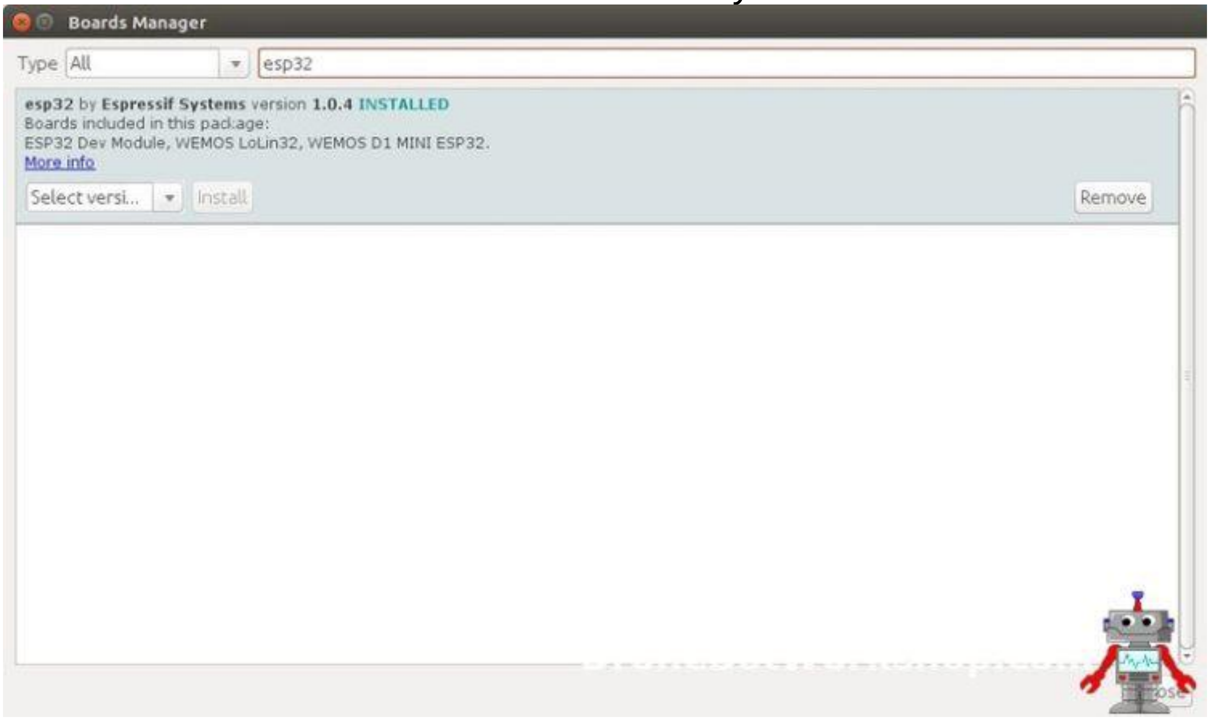
☐ Display line numbers
☐ Enable Code Folding
☑ Verify code after upload
☐ Use external editor
☑ Aggressively cache compiled core
☑ Check for updates on startup
☑ Update sketch files to new extension on save (.pde -> .ino)
☑ Save when verifying or uploading

Additional Boards Manager URLs: :nt.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

More preferences can be edited directly in the file
/home/dronebotworkshop/.arduino15/preferences.txt
(edit only when Arduino is not running)

Next, you will need to use the new entry to actually add the ESPO32 boards to your Arduino IDE. You do that by following this procedure:

1. In the Arduino IDE click on the *Tools* menu on the top menu bar.
2. Scroll down to the *Board:* entry (i.e. *Board: Arduino/Genuino Uno*).
3. A submenu will open when you highlight the *Board:* entry.
4. At the top of the submenu is *Boards Manager*. Click on it to open the Boards Manager dialog box.
5. In the search box in the Boards Manager enter "*esp32*".
6. You should see an entry for "*esp32 by Espressif Systems*". Highlight this entry and click on the *Install* button.
7. This will install the ESP32 boards into your Arduino IDE



Type All esp32

esp32 by **Espressif Systems** version **1.0.4** INSTALLED
Boards included in this package:
ESP32 Dev Module, WEMOS LoLin32, WEMOS D1 MINI ESP32.
More info

If you go back into the *Boards:* submenu you should now see a number of ESP32 boards. You'll need to select the board that matches (or is equivalent to) the ESP32 board you have purchased.
Choose the "Node32s" board in the Boards Manager.

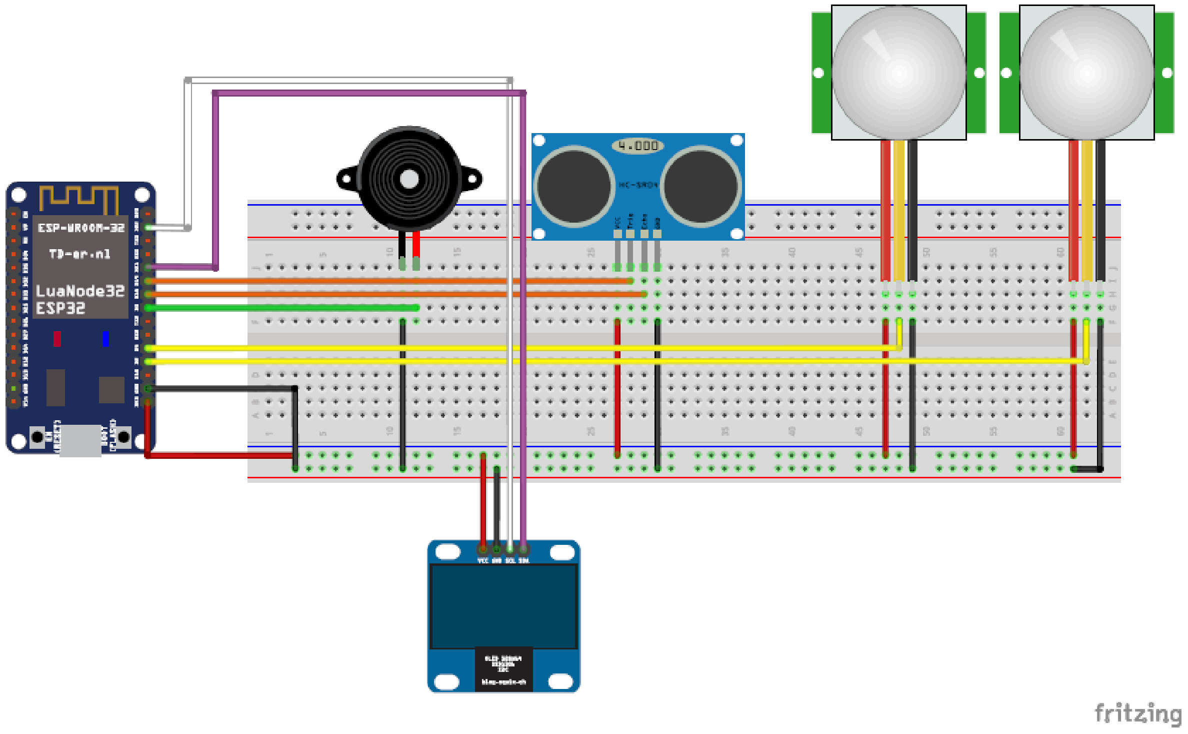
Parts Required

For this project, you need the following parts:

- ESP32 board
- Mini PIR motion sensor
- Ultrasonic motion sensor
- Buzzer
- OLED Display
- Jumper wires
- Breadboard

Schematics

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

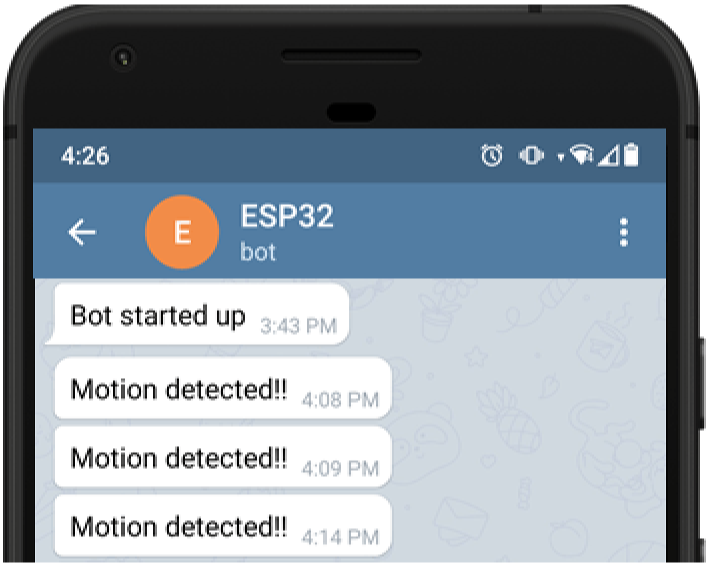


Demonstration

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.



4:26

ESP32
bot

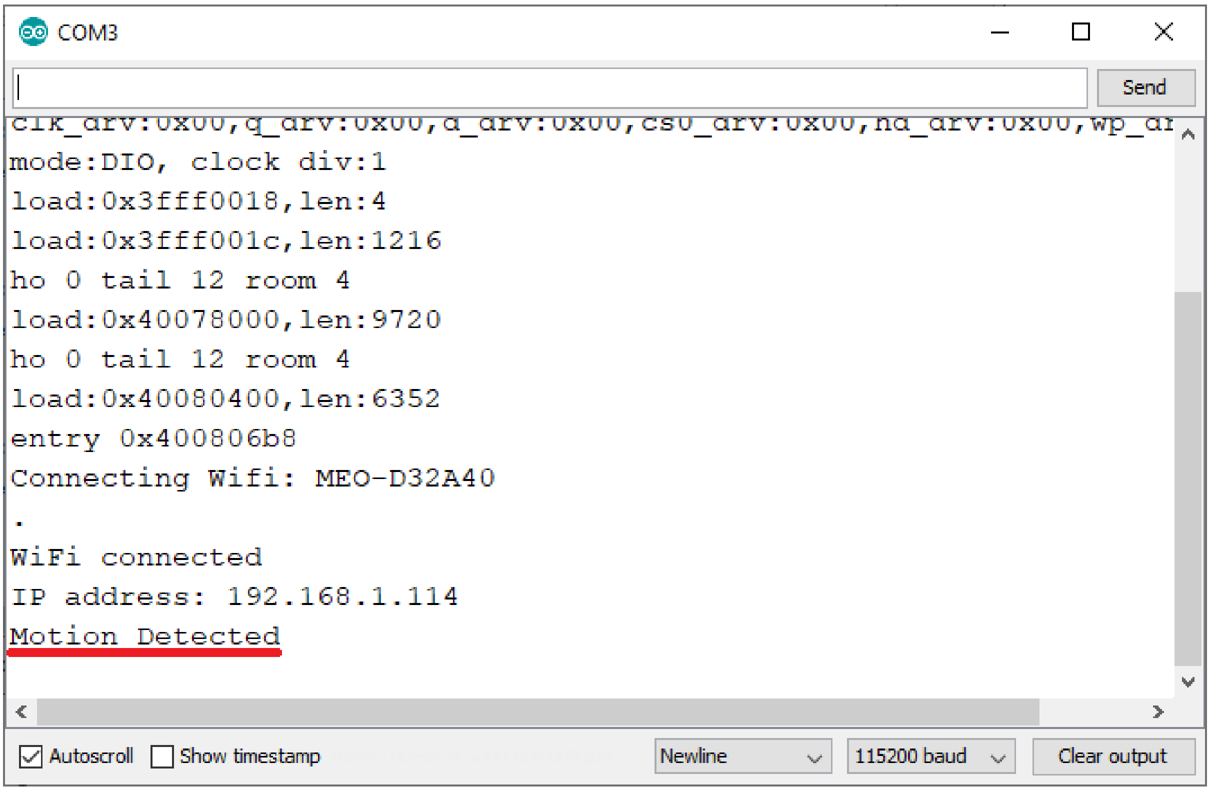
Bot started up 3:43 PM

Motion detected!! 4:08 PM

Motion detected!! 4:09 PM

Motion detected!! 4:14 PM

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



```
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_dr
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:9720
ho 0 tail 12 room 4
load:0x40080400,len:6352
entry 0x400806b8
Connecting Wifi: MEO-D32A40
.
WiFi connected
IP address: 192.168.1.114
Motion Detected
```

☑ Autoscroll □ Show timestamp Newline 115200 baud