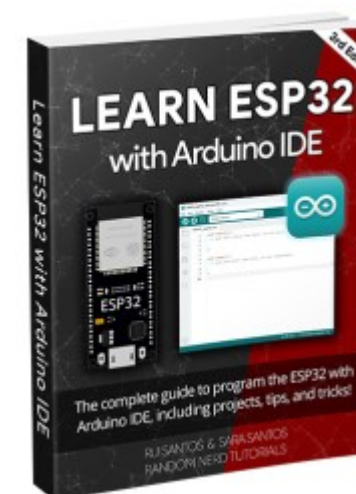


[HOME](#) [ESP32](#) [ESP8266](#) [ESP32-CAM](#) [RASPBERRY PI](#) [MICROPYTHON](#) [RPi PICO](#) [ARDUINO](#)
[REVIEWS](#)

ESP32 OTA (Over-the-Air) Updates – ElegantOTA Library with Arduino IDE

This tutorial shows how to do OTA (over-the-air) updates to your ESP32 boards using the ElegantOTA library (V3 version) with Arduino IDE. This library sets up a web server that lets you update the firmware (a new sketch) on your board wirelessly. This way, you don't need a connection between the ESP32 and your computer to upload a new sketch. This library also allows uploading files to the filesystem (LittleFS or SPIFFS) wirelessly.

Affiliate Disclosure: *Random Nerd Tutorials is a participant in affiliate advertising programs designed to provide a means for us to earn fees by linking to Amazon, eBay, AliExpress, and other sites. We might be compensated for referring traffic and business to these companies.*





By the end of this tutorial, you'll be able to easily add OTA capabilities to your web server projects with the ESP32 to upload new firmware and files to the filesystem wirelessly in the future.

Table Of Contents

Throughout this tutorial, we'll cover:

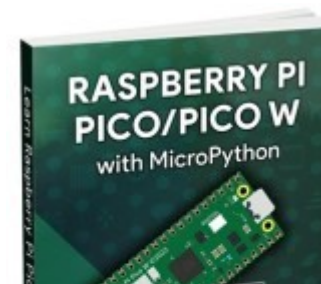
- [Add the ElegantOTA feature to your web server](#)
- [Upload new firmware via OTA to the ESP32 board](#)
- [Upload files to LittleFS via OTA to the ESP32 board](#)

[Learn ESP32 with Arduino IDE eBook »](#)

Complete guide to program the ESP32 with Arduino IDE!



[SMART HOME with Raspberry Pi, ESP32, and ESP8266 »](#) learn how to build a complete home automation system.



We recommend that you follow all the tutorial steps to understand how the [ElegantOTA library](#) works and how you can use it in your projects. To demonstrate how to do this, we'll upload files to build different web server projects.

ESP32 OTA (Over-the-Air) Programming

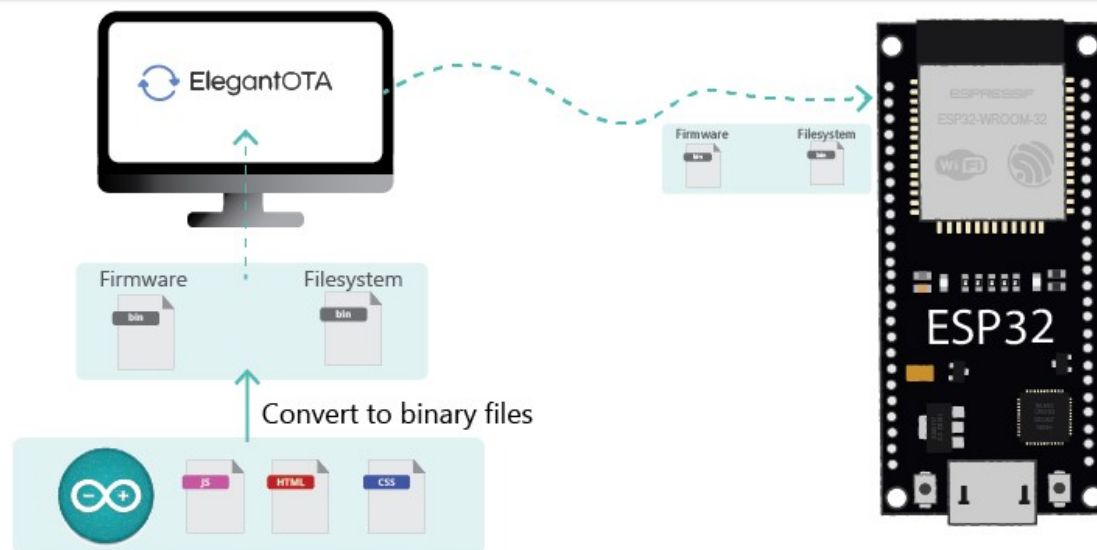
OTA (Over-the-Air) update is the process of loading new firmware to the ESP32 board using a Wi-Fi connection rather than a serial communication. This functionality is extremely useful in case of no physical access to the ESP32 board. You don't need a connection between your computer and the board to upload new code.



[Learn Raspberry Pi Pico/Pico W with MicroPython »](#) The complete getting started guide to get the most out of the the Raspberry Pi Pico/Pico W (RP2040) microcontroller board using MicroPython programming language.



[🔥 Learn LVGL: Build GUIs for ESP32 Projects »](#) Learn how to build Graphical



User Interfaces (GUIs) for ESP32 Projects using LVGL (Light Versatile Graphics Library) with the Arduino IDE.

There are different ways to perform OTA updates. In this tutorial, we'll cover how to do that using the [ElegantOTA library](#) (version V3 — this is the successor of the deprecated [AsyncElegantOTA library](#)). In our opinion, this is one of the best and easiest ways to perform OTA updates.

The ElegantOTA library creates a web server that you can access on your local network to upload new firmware or files to the filesystem (SPIFFS or LittleFS). The files you upload should be in `.bin` format. We'll show you later in this tutorial how to convert your files to `.bin` format.

The only disadvantage of OTA programming is that you need to add the code for OTA in every sketch you upload so that you're able to use OTA in the future. In case of the ElegantOTA library, it consists of just three lines of code.

ElegantOTA Library

The logo for ElegantOTA V3 is displayed on a black rectangular background. The text 'ElegantOTA' is in a white, sans-serif font, with the 'OTA' part in a larger, bold font. Below it, 'V3' is written in a smaller, white, sans-serif font.

Here are some great features of this library:

- It is compatible with the built-in `WebServer.h` library and with several forks of the `ESPAsyncWebServer` library.
- You just need to add three lines of code to add OTA capabilities to your “regular” web server;
- It allows you to update not only new firmware to the board but also files to the ESP32 filesystem (LittleFS or SPIFFS);
- It provides a beautiful and modern web server interface;
- It is available as a pro paid version that adds more features.

OTA Updates with the ElegantOTA Library – Quick Summary

To add OTA capabilities to your projects using the ElegantOTA library, follow these steps:

1) Install the [ElegantOTA](#), [AsyncTCP](#), and [ESPAsyncWebServer](#) libraries;

2) Include ElegantOTA library at the top of the Arduino sketch:

```
#include <ElegantOTA.h>;
```

3) Add the following line in the setup before `server.begin();`

```
ElegantOTA.begin(&server);
```

4) In the `loop()` , add the following line:

```
ElegantOTA.loop();
```

5) Open your browser and go to `http://<IPAddress>/update` , where `<IPAddress>` is your ESP32 IP address to access the web server page for the OTA updates.

Continue reading the tutorial for more detailed steps.

How does OTA Web Updater Work?

The first sketch must be uploaded using a serial connection. This sketch should include the code to set up the OTA Web Updater, allowing you to upload new sketches through a web browser.

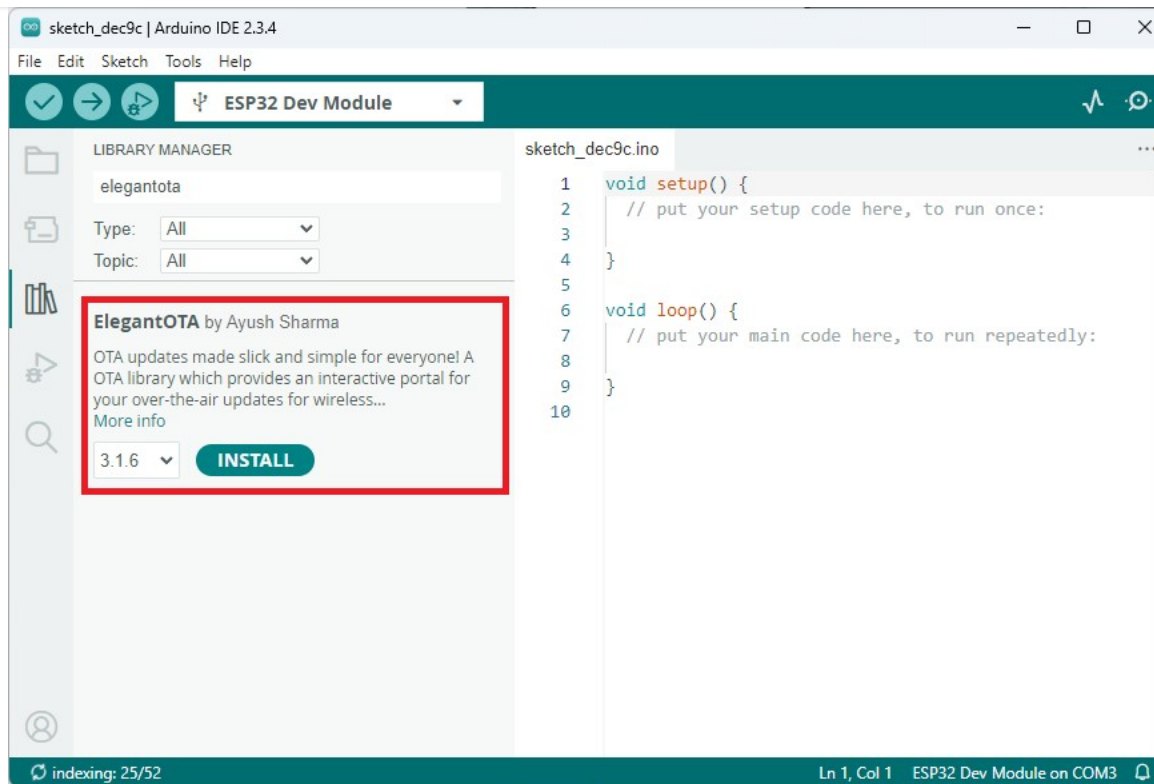
The OTA Web Updater creates a web server where you can upload sketches wirelessly.

If your code doesn't include an OTA routine, you won't be able to use the web server to upload new sketches wirelessly.

After that, make sure every sketch you upload includes OTA routines so you can continue updating the board wirelessly in the future.

Installing the ElegantOTA Library

[Having the ESP32 Board installed in your Arduino IDE](#), you also need to install the ElegantOTA library in the Arduino IDE go to **Sketch > Include Library > Manage Libraries**, search for **ElegantOTA** and install the *ElegantOTA library by Ayush Sharma*.



Enabling Async Mode

For the ElegantOTA library to work in async mode (with the ESPAsyncWebServer library), you need to do the following procedure.

- 1) Go to your Arduino *libraries* directory.
- 2) Open the *ElegantOTA* folder and then open the *src* folder.
- 3) Locate the `ELEGANTOTA_USE_ASYNC_WEBSERVER` macro in the

ElegantOTA.h file, and set it to `1` as follows:

```
#define ELEGANTOTA_USE_ASYNC_WEBSERVER 1
```

```
#ifndef ElegantOTA_h
#define ElegantOTA_h

#include "Arduino.h"
#include "stdlib_noniso.h"
#include "elop.h"

#ifndef ELEGANTOTA_USE_ASYNC_WEBSERVER
#define ELEGANTOTA_USE_ASYNC_WEBSERVER 1
#endif

#ifndef ELEGANTOTA_DEBUG
#define ELEGANTOTA_DEBUG 0
#endif

#ifndef UPDATE_DEBUG
#define UPDATE_DEBUG 0
#endif
```

4) Save the changes to the *ElegantOTA.h* file.

5) Now you can use the ElegantOTA in async mode for your OTA updates and with the ESPAsyncWebServer library.

Install AsyncTCP and ESPAsyncWebServer Libraries

To test the examples in this tutorial, you also need to install the AsyncTCP and the ESPAsyncWebServer libraries.

Note: the ElegantOTA library works with different forked versions of the ESPAsyncWebServer and the AsyncTCP web server libraries, so you don't necessarily need to use the same ones that we're using.

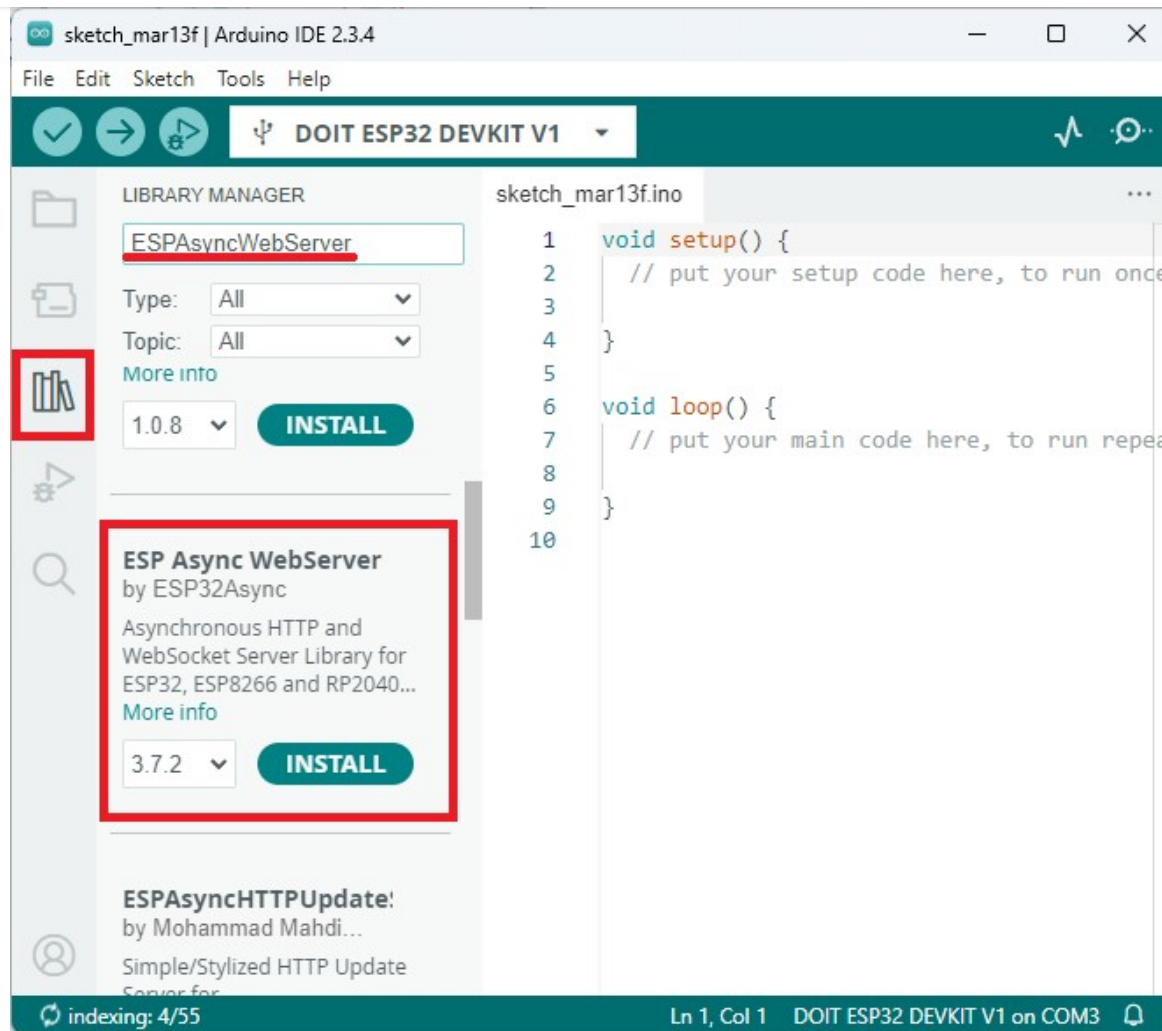
Async Web Server Libraries

We'll build the web server using the following libraries:

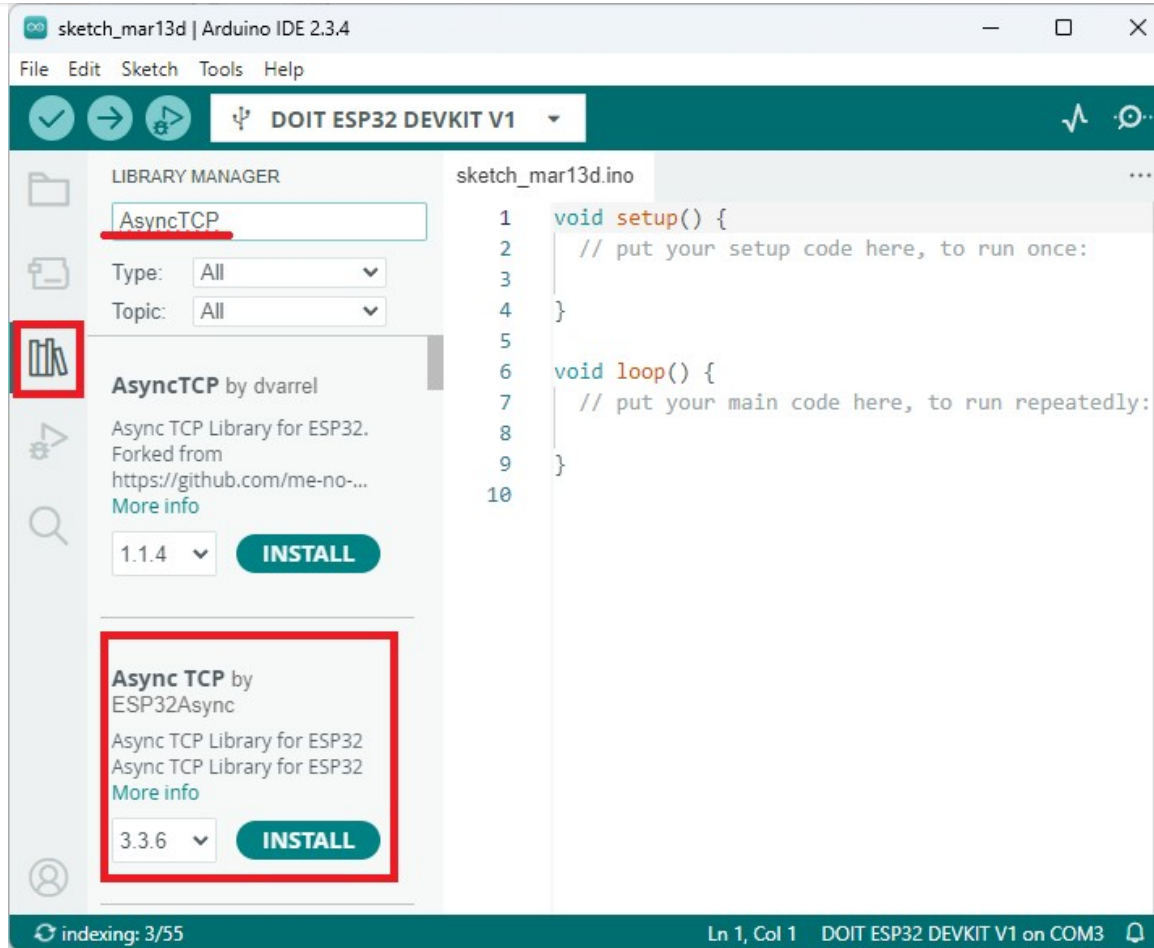
- [ESPAsyncWebServer](#)
- [AsyncTCP](#)

You can install these libraries in the Arduino Library Manager. Open the Library Manager by clicking the Library icon at the left sidebar.

Search for `ESPAsyncWebServer` and install the **ESPAsyncWebServer** by **ESP32Async**.



Then, install the AsyncTCP library. Search for AsyncTCP and install the **AsyncTCP by ESP32Async**.



ElegantOTA ESP32 Basic Example

Let's start with a basic example (based on one of the library's examples). The following code creates a simple web server with the ESP32. The

root URL displays some text, and the **/update** URL displays the interface to update the firmware and the filesystem.

Copy the following code to your Arduino IDE.

```
/*  
  Rui Santos & Sara Santos - Random Nerd Tutorials  
  Complete project details at https://RandomNerdTutorials  
  Permission is hereby granted, free of charge, to any person  
  obtaining a copy of this software and associated documentation  
  files, to use the software for personal or non-commercial  
  purposes, provided the original author and source  
  are credited and copyright notice preserved, as shown  
  within this file and in the LICENSE file  
  in the root directory of this project.  
  */
```

```
#include <Arduino.h>  
#include <WiFi.h>  
#include <AsyncTCP.h>  
#include <ESPAsyncWebServer.h>  
#include <ElegantOTA.h>
```

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

```
AsyncWebServer server(80);
```

```
void setup(void) {  
  Serial.begin(115200);  
  WiFi.mode(WIFI_STA);
```

```
WiFi.begin(ssid, password);  
Serial.println("");  
  
// Wait for connection  
while (WiFi.status() != WL_CONNECTED) {  
    delay(500);  
    Serial.print(".");  
}
```

[View raw code](#)

Insert your network credentials and the code should work straight away.

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

How Does the Code Work?

Start by including the required libraries.

```
#include <Arduino.h>
#include <WiFi.h>
#include <AsyncTCP.h>
#include <ESPAsyncWebServer.h>
#include <ElegantOTA.h>
```

Insert your network credentials in the following variables so that the ESP32 can connect to the Wi-Fi on your local network.

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

Create an `AsyncWebServer` object on port 80:

```
AsyncWebServer server(80);
```

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

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

Initialize Wi-Fi:


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

// Wait for connection
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("");
Serial.print("Connected to ");
Serial.println(ssid);
Serial.print("IP address: ");
Serial.println(WiFi.localIP());
```

Then, handle the client requests. The following lines, send some text
Hi! I am ESP32. when you access the root (/) URL:

```
server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) {
    request->send(200, "text/plain", "Hi! I am ESP32.");
});
```

If your web server needs to handle more requests you can add them

(we'll show you in the next example).

Initialize the server:

```
server.begin();
```

Then, add the next line to start ElegantOTA:

```
ElegantOTA.begin(&server);    // Start ElegantOTA
```

In the `loop()` , add the following line:

```
ElegantOTA.loop();
```

Accessing the Web Server

After uploading code to the board, open the Serial Monitor at a baud rate of 115200. Press the ESP32 on-board RST button. It should display the ESP IP address as follows (yours may be different):

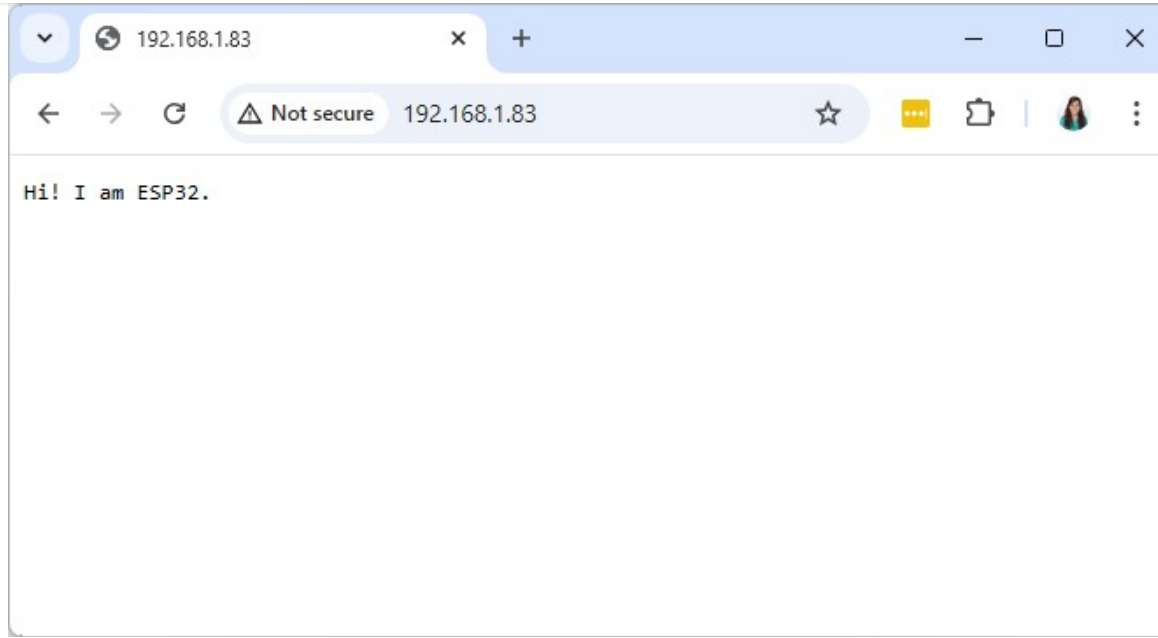


```
Output Serial Monitor x
Message (Enter to send message to 'ESP32 Dev Module' on 'COM4') New Line 115200 baud
...
config: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:4832
load:0x40078000,len:16460
load:0x40080400,len:4
load:0x40080404,len:3504
entry 0x400805cc
..
Connected to ME - A0
IP address: 192.168.1.83
HTTP server started
```

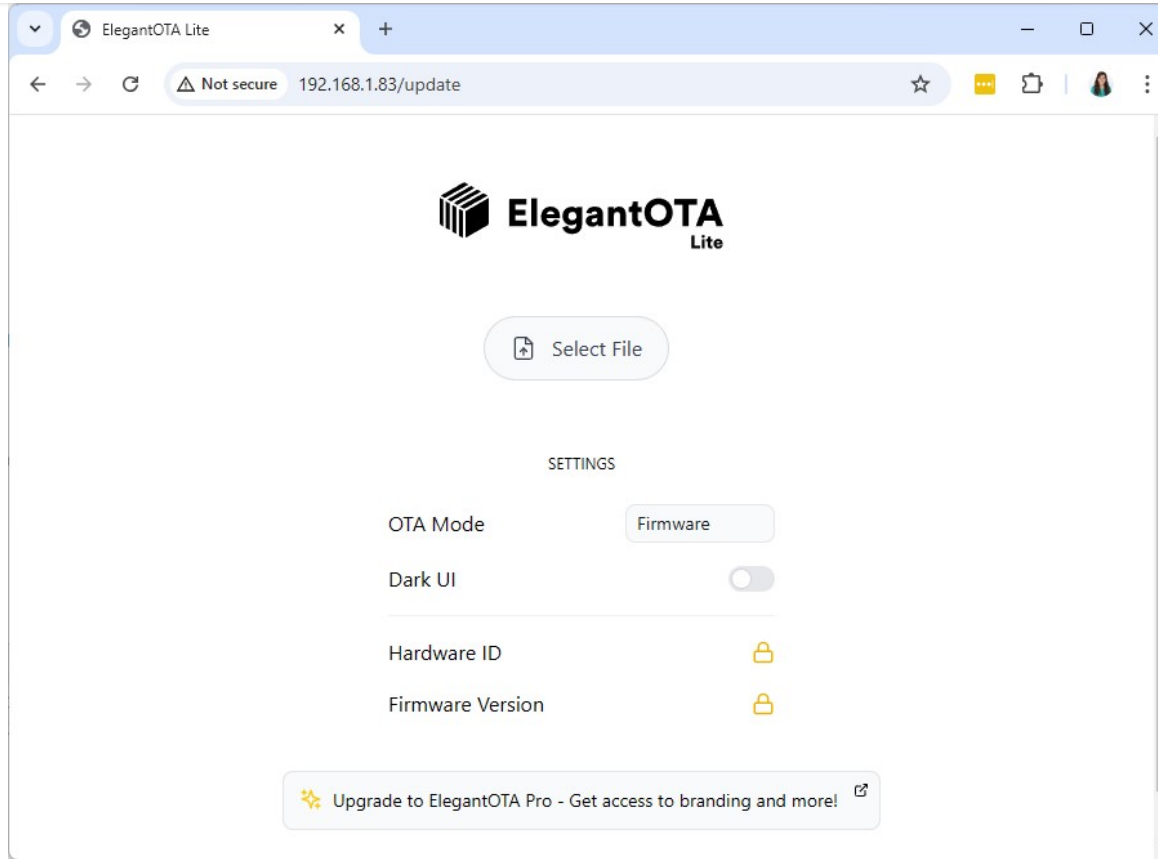
Ln 31, Col 33 ESP32 Dev Module on COM4 1

Note: when uploading the code, if you have the Partition Scheme options for your particular board in the Tools menu, make sure to select a Partition Scheme with space for the filesystem. For example: *Partition Scheme: 16M Flash (2MB APP/12.5MB FATFS)*. If you don't have that option for the particular board you selected, don't worry about it.

In your local network, open your browser and type the ESP32 IP address. You should get access the root (/) web page with some text displayed.



Now, if you want to modify your web server code via OTA, go to the ESP IP address followed by `/update` . The following web page should load.

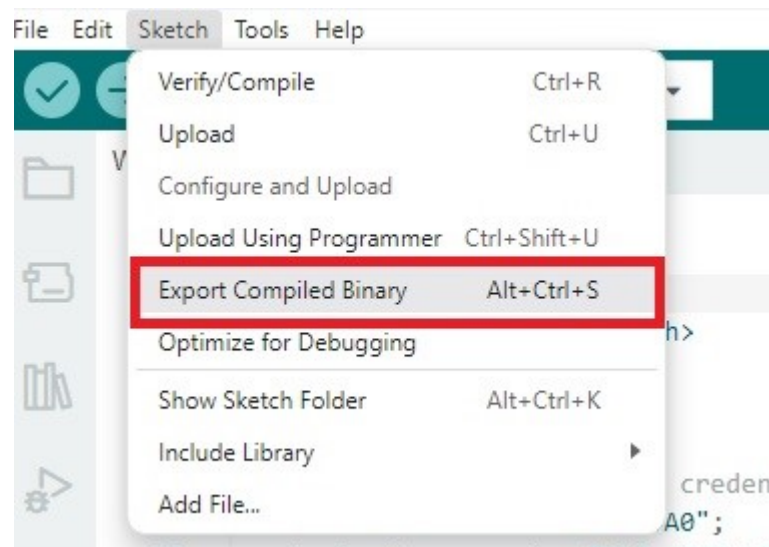


To upload new code to your board or files to the filesystem, you need to upload them in *.bin* format. Follow the next section to learn how to generate a *.bin* file from your sketch in Arduino IDE.

Upload New Code to the ESP32 via OTA

Every file that you upload via OTA should be in *.bin* format. You can generate a *.bin* file from your sketch using the Arduino IDE.

With your sketch opened, you just need to select the ESP32 board you're using in **Tools > Board**, and then go to **Sketch > Export Compiled Binary**. A *.bin* file will be generated from your sketch.



The generated file will be saved under your project folder inside a series of other folders. The file with the *.ino.bin* extension is the one you should upload to your board using the ElegantOTA web page.

Upload a New Web Server Sketch via OTA – Example

Imagine that after uploading the previous sketch, you want to upload a new one that allows you to control an LED via a web interface like [this project](#). Here's the steps you need to follow:

1. Copy the following code to your Arduino IDE. Don't forget to insert your network credentials.

```
/*  
  Rui Santos & Sara Santos - Random Nerd Tutorials  
  Complete project details at https://RandomNerdTutorials  
  Permission is hereby granted, free of charge, to any person  
  obtaining a copy of this software and associated documentation  
  files, to use, copy, modify, merge, publish, distribute, sublicense,  
  and/or sell copies of the software, and/or to permit persons to whom  
  the software is furnished to do so, under the terms of the  
  Creative Commons Attribution license, v1.0. http://creativecommons.org/licenses/by/1.0/  
*/
```

```
// Import required libraries  
#include <Arduino.h>  
#include <WiFi.h>  
#include <AsyncTCP.h>  
#include <ESPAsyncWebServer.h>  
#include <ElegantOTA.h>  
  
// Replace with your network credentials  
const char* ssid = "REPLACE_WITH_YOUR_SSID";  
const char* password = "REPLACE_WITH_YOUR_PASSWORD";  
  
bool ledState = 0;
```



```
const int ledPin = 2;

// Create AsyncWebServer object on port 80
AsyncWebServer server(80);
AsyncWebSocket ws("/ws");

const char index_html[] PROGMEM = R"rawliteral(
<!DOCTYPE HTML><html>
<head>
```

[View raw code](#)

This is the same code used in [this project](#), but it contains the needed lines of code to handle ElegantOTA:

```
#include <ElegantOTA.h>
```

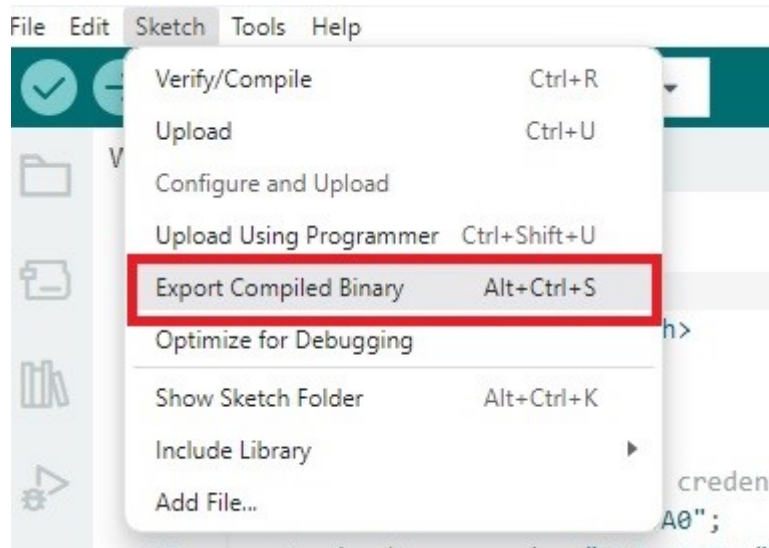
```
ElegantOTA.begin(&server);
```

```
ElegantOTA.loop();
```

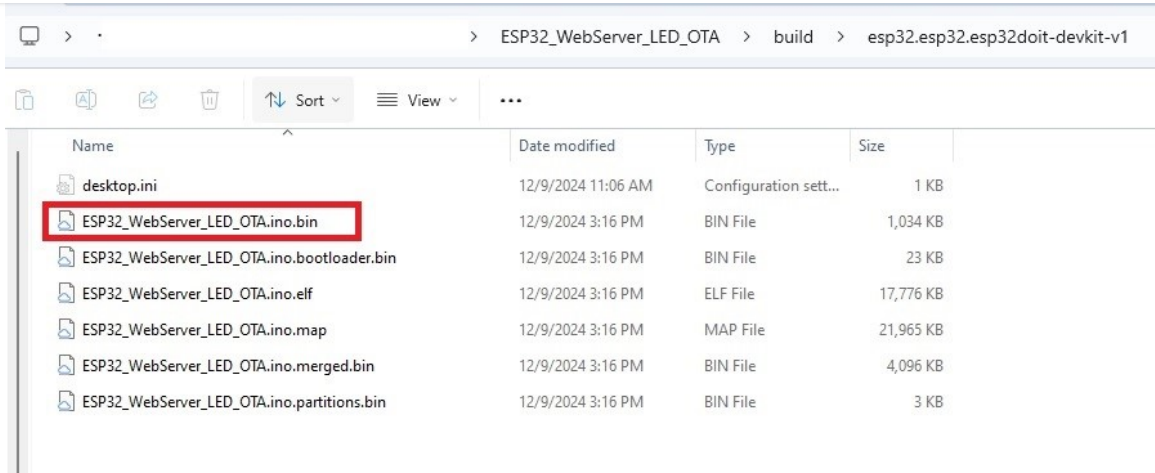
2. Save your sketch: **File > Save** and give it a name. For example:

Web_Server_LED_OTA_ESP32.

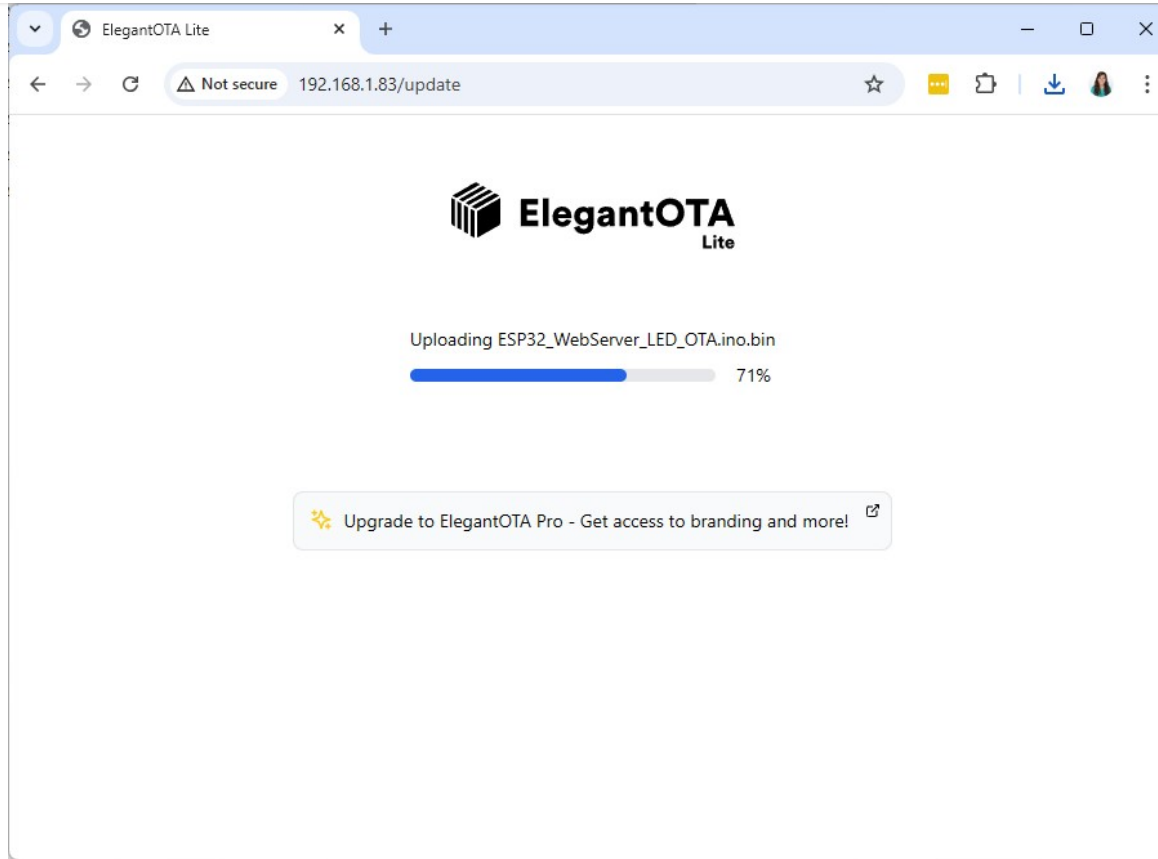
3. Generate a *.bin* file from your sketch. First, select the board model you're using in **Tools > Board**. Then, go to **Sketch > Export Compiled Binary**.



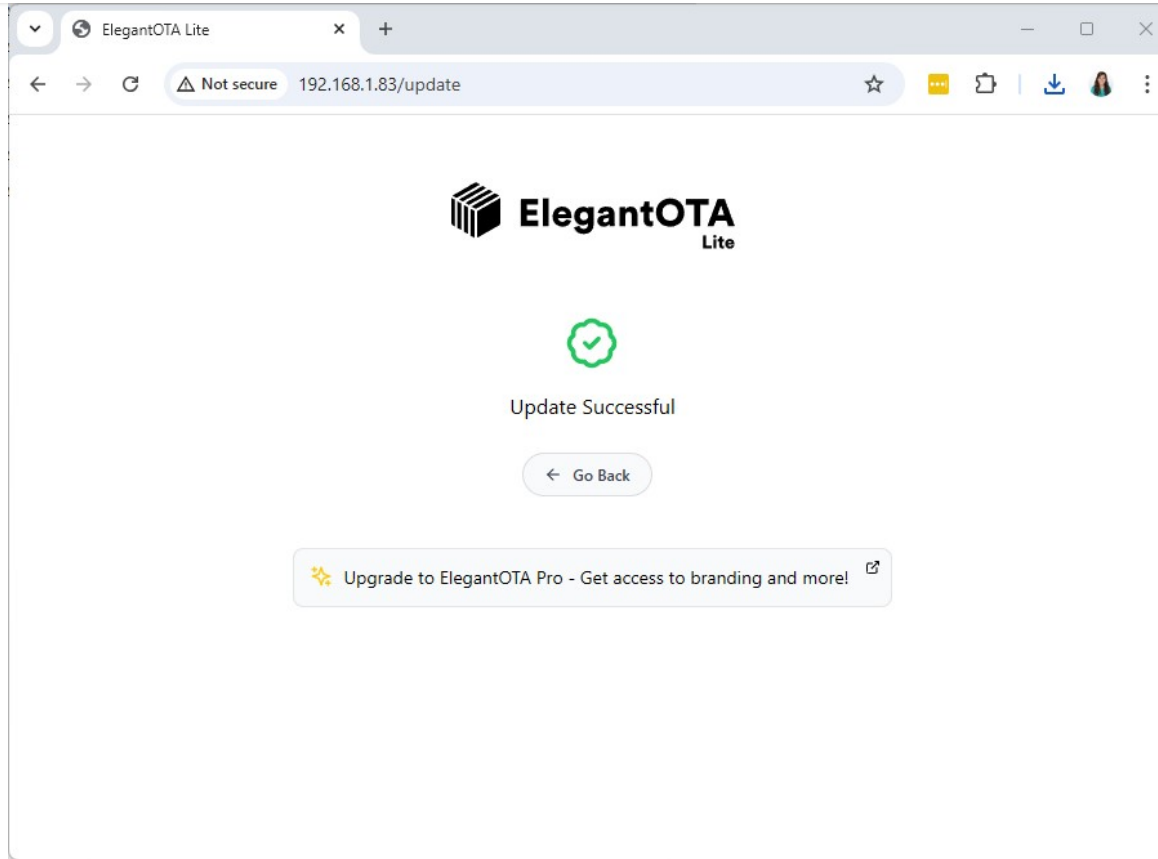
Go to your sketch folder. You should have a *build* folder. Inside that folder, you'll have another folder related to your board model. Open that folder. There'll be several files. You should upload the file with the *ino.bin* extension.



4. Now, you need to upload that file using the ElegantOTA page. Go to your ESP IP address followed by `/update`. Make sure you have the **firmware** option selected. Click on **Choose File** and select the *.ino.bin* file you've just generated.

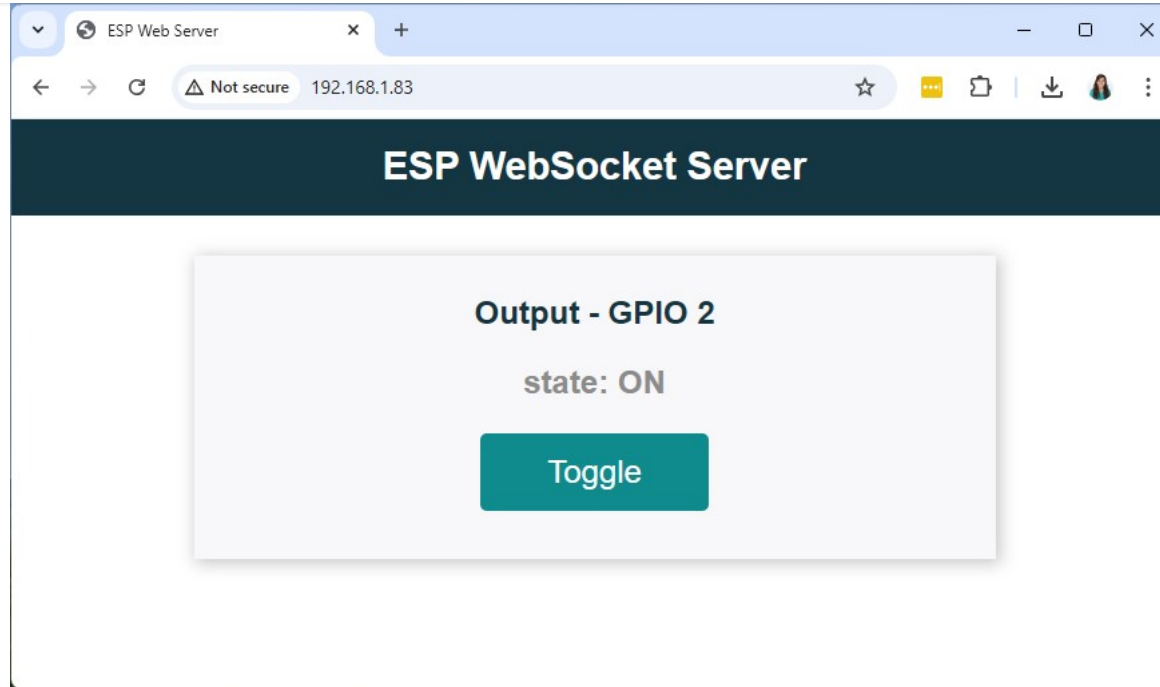


5. After a few seconds, you should get a success message. Then, click on the **Back** button.

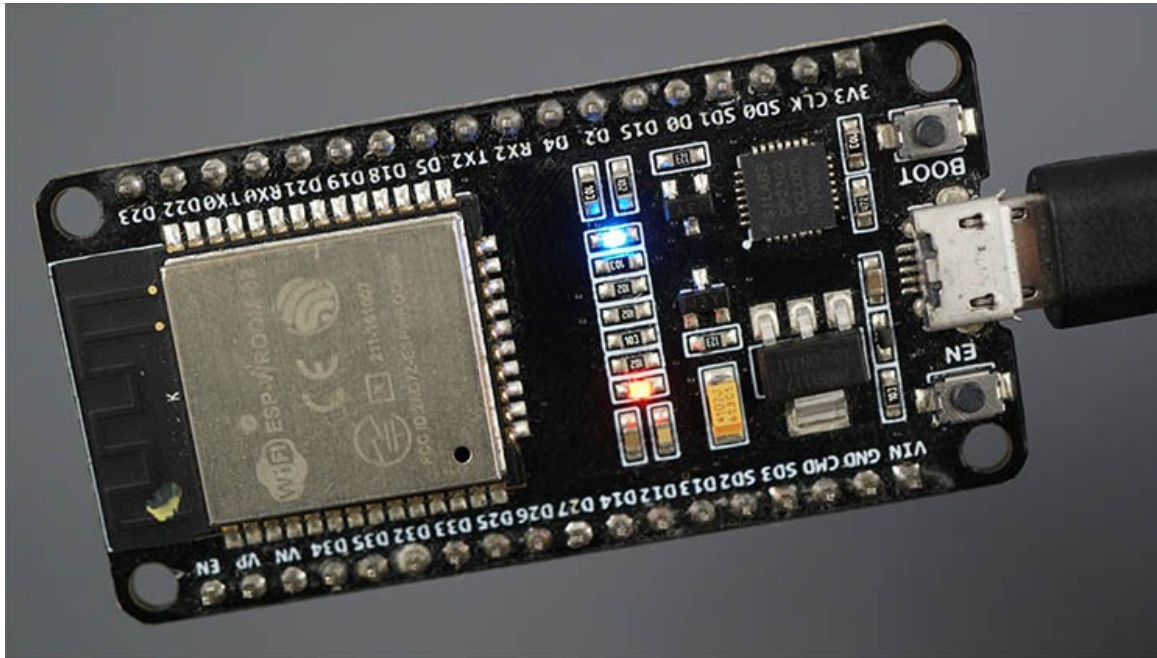


6. Now, you can go to the root (/) URL to access the new web server.

This is the page you should see when you access the ESP IP address on the root (/) URL.



You can click on the button to turn the ESP32 on-board LED on and off.



Because we've also added OTA capabilities to this new web server, we can upload a new sketch in the future if needed. You just need to go to the ESP32 IP address followed by `/update`.

Congratulations, you've uploaded new code to your ESP32 via Wi-Fi using ElegantOTA.

Continue reading if you want to learn how to upload files to the ESP32 filesystem (LittleFS) using the ElegantOTA library.

Upload Files to Filesystem via

OTA to the ESP32

In this section you'll learn to upload files to the ESP32 filesystem (LittleFS) using the ElegantOTA library.

ESP32 Filesystem Uploader Plugin

Before proceeding, you need to have the ESP32 Filesystem Uploader Plugin installed in your Arduino IDE. Follow the next tutorial before proceeding:

- [Arduino IDE 2: Install ESP32 LittleFS Uploader \(Upload Files to the Filesystem\)](#)

Web Server with Files from LittleFS

Imagine the scenario that you need to upload files to the ESP32 filesystem, for example: configuration files; HTML, CSS and JavaScript files to update the web server page; or any other file that you may want to save in the LittleFS filesystem via OTA.

To show you how to do this, we'll create a new web server that serves files from LittleFS: HTML, and CSS files to build a web page that controls the ESP32 built-in LED remotely using a different interface.

Copy the following code to your Arduino IDE.

```
/*  
  Rui Santos & Sara Santos - Random Nerd Tutorials  
  Complete project details at https://RandomNerdTutorials  
  Permission is hereby granted, free of charge, to any person  
  *****/  
  
#include <Arduino.h>  
#include <WiFi.h>  
#include <AsyncTCP.h>  
#include <ESPAsyncWebServer.h>  
#include "LittleFS.h"  
#include <ElegantOTA.h>  
  
// Replace with your network credentials  
const char* ssid = "REPLACE_WITH_YOUR_SSID";  
const char* password = "REPLACE_WITH_YOUR_PASSWORD";  
  
// Create AsyncWebServer object on port 80  
AsyncWebServer server(80);  
  
// Set LED GPIO  
const int ledPin = 2;
```

```
// Stores LED state  
String ledState;
```

```
// Initialize LittleFS
```

[View raw code](#)

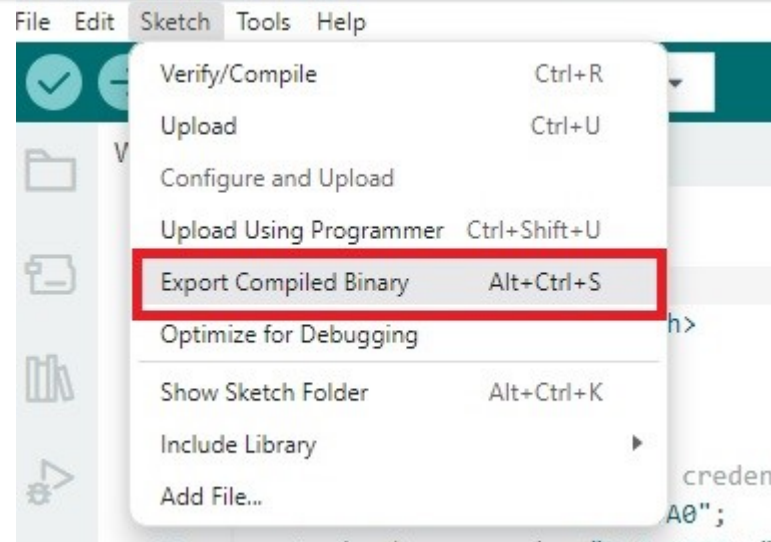
Note: this example is covered and explained in our eBook: [Build Web Servers with the ESP32 and ESP8266](#).

Insert your network credentials in the following variables and save the code.

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

Update Firmware

Create a *.ino.bin* file from this sketch as shown previously (this sketch also includes the needed lines of code to provide OTA capabilities).



Go to the ESP32 IP address followed by `/update` and upload the new firmware.

Next, we'll see how to upload the files.

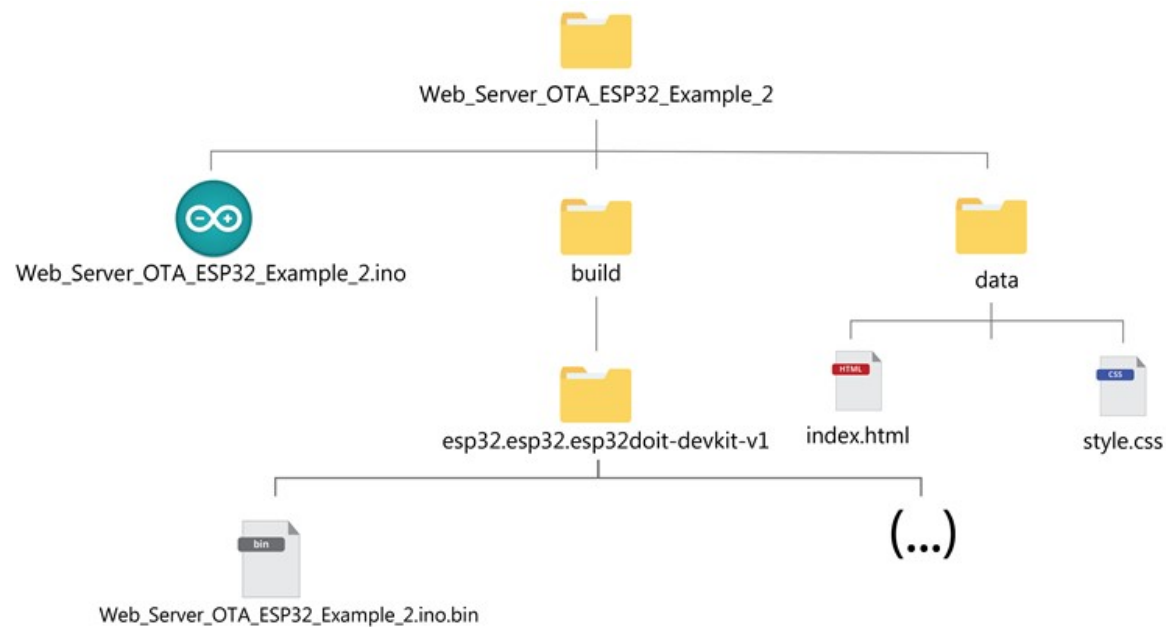
Update the Filesystem

Under the project folder, create a folder called ***data*** and move the following HTML and CSS files (click on the links to download the files):

- [HTML file: *index.html*](#)
- [CSS file: *style.css*](#)
- [Click here to download all project files](#)

To find your project folder, you can simply go to **Sketch > Show Sketch Folder**.

This is the folder structure of your project and where the data folder should be located:



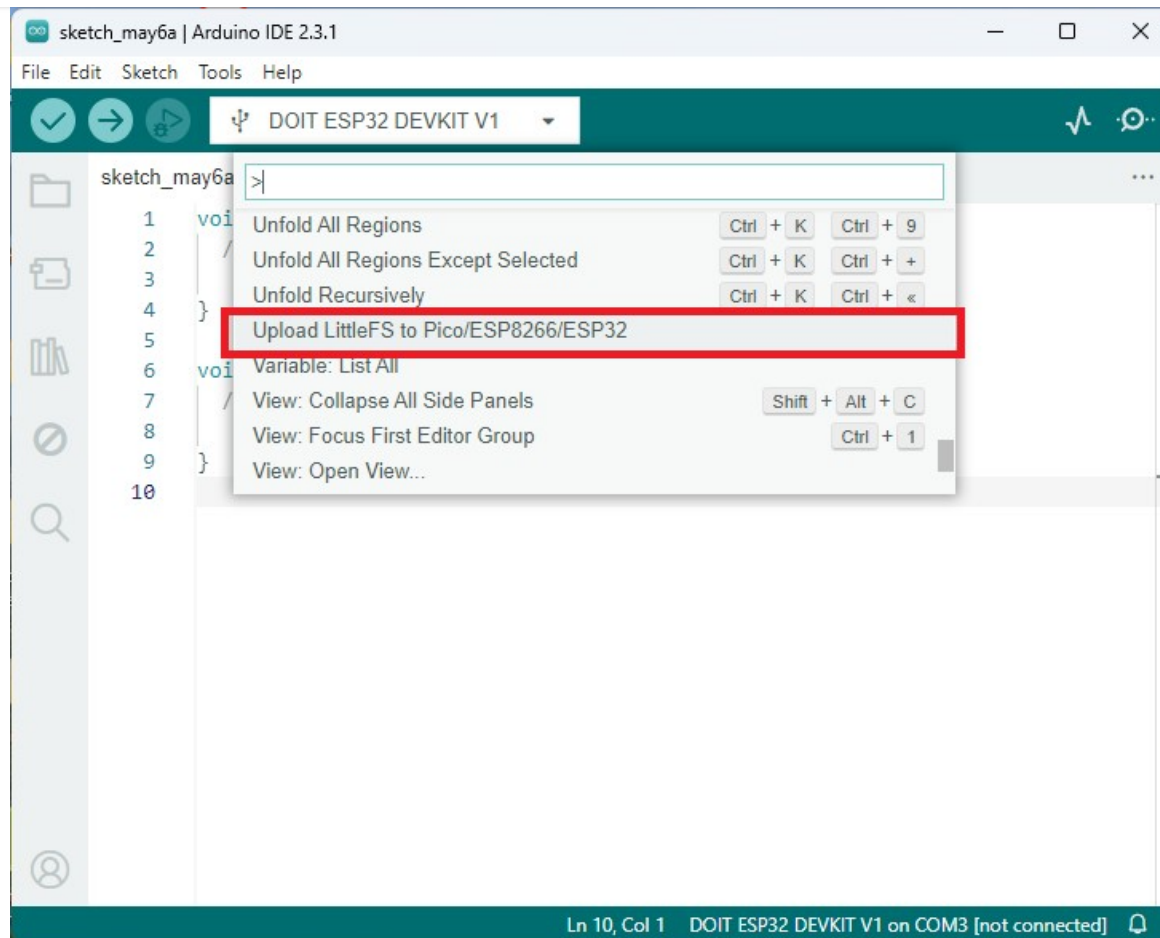
- **Web_Server_OTA_ESP32_Example_2**
 - **data**
 - ***style.css***
 - ***index.html***
 - ***Web_Server_OTA_ESP32_Example_2.ino***
 - **build**
 - **esp32.es32.esp32doit-devkit-v1** (or similar depending on

the selected board)

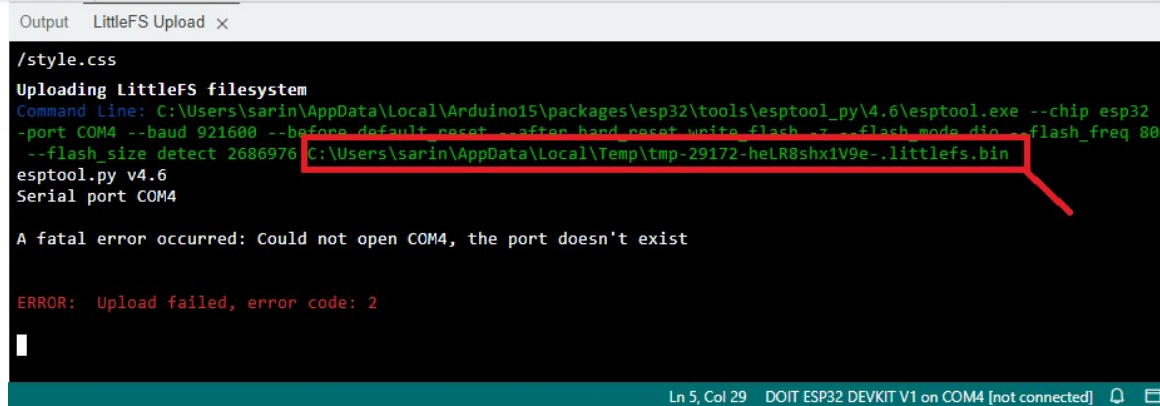
- *Web_Server_OTA_ESP32_Example_2.ino.bin*

After this, with the **ESP32 disconnected from your computer** (that's the whole purpose of OTA), let's pretend we'll upload the files to the filesystem. Press **[Ctrl] + [Shift] + [P]** to open the command palette. An instruction called '**Upload Little FS to Pico/ESP8266/ESP32**' should be there (just scroll down or search for the name of the instruction).

If you don't have that option, it means you don't have the Filesystem Uploader Plugin installed in your Arduino IDE. Follow the next tutorial to install it: [Arduino IDE 2: Install ESP32 LittleFS Uploader \(Upload Files to the Filesystem\)](#).



You'll get an error because there isn't any ESP32 board connected to your computer – don't worry. It will still create a *.bin* file from the *data* folder.



```
Output LittleFS Upload x
/style.css
Uploading LittleFS filesystem
Command Line: C:\Users\sarin\AppData\Local\Arduino15\packages\esp32\tools\esptool_py\4.6\esptool.exe --chip esp32 -
-port COM4 --baud 921600 --before default_reset --after hard_reset write_flash -z --flash_mode dio --flash_freq 80m
--flash_size detect 2686976 C:\Users\sarin\AppData\Local\Temp\tmp-29172-heLR8shx1V9e-.littlefs.bin
esptool.py v4.6
Serial port COM4

A fatal error occurred: Could not open COM4, the port doesn't exist

ERROR: Upload failed, error code: 2

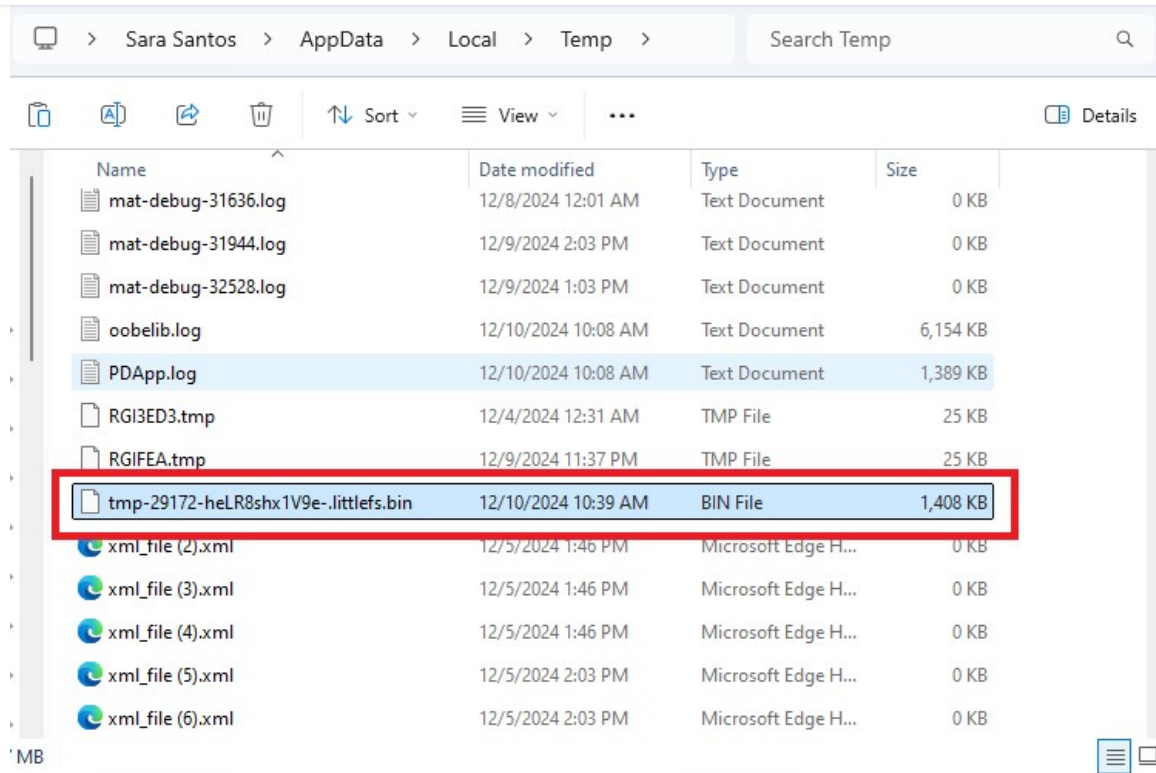
Ln 5, Col 29 DOIT ESP32 DEVKIT V1 on COM4 [not connected]
```

On the debugging window you'll see the *.littlefs.bin* file location. That's that file that you should upload (in our case the file is called *tmp-29172-heLR8chx1V9e-.littlefs.bin*).

In our case, this is the path where that file is located:

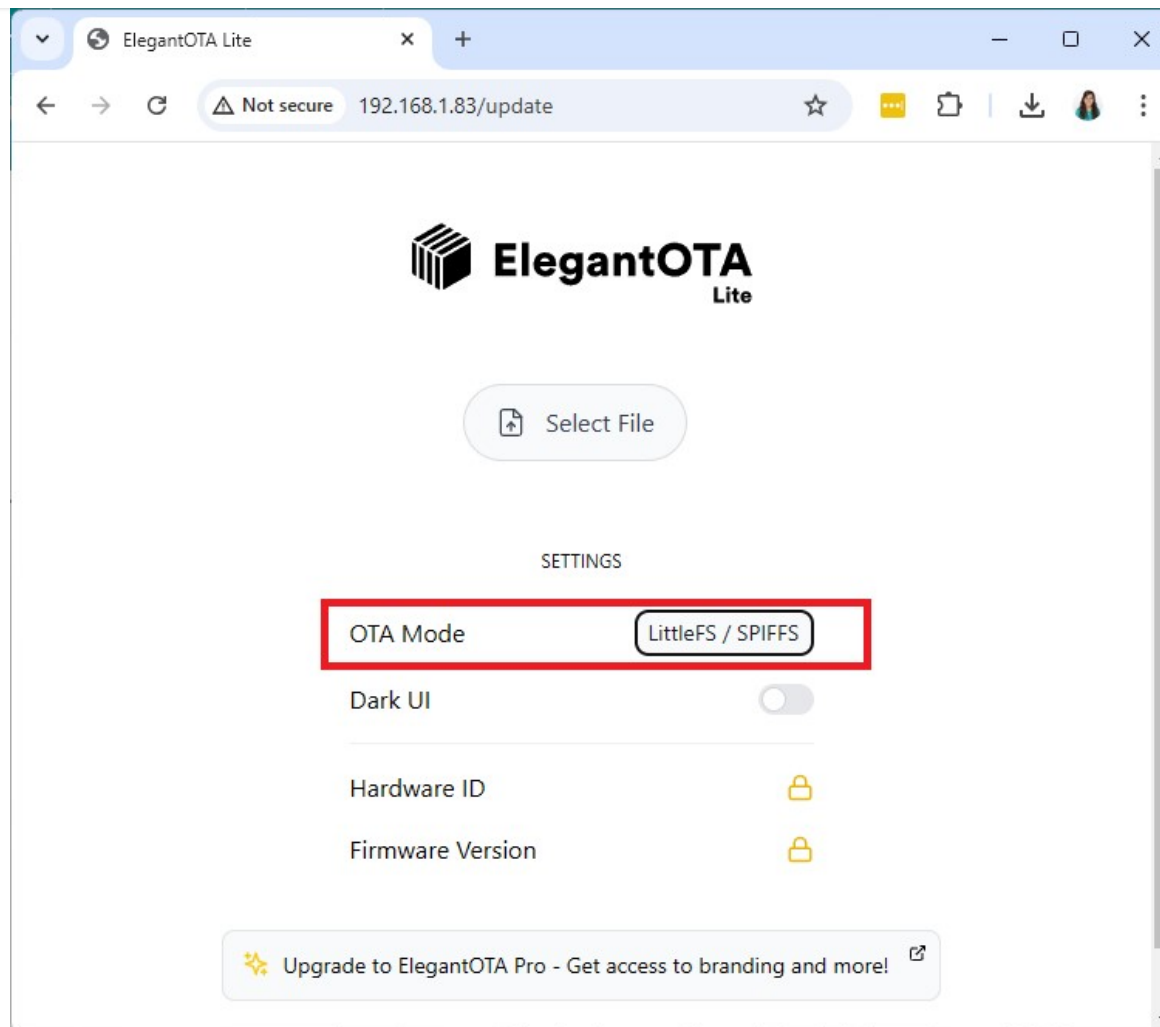
```
C:\Users\sarin\AppData\Local\Temp\tmp-29172-
heLR8chx1V9e-.littlefs.bin.
```

Find that file in your computer.



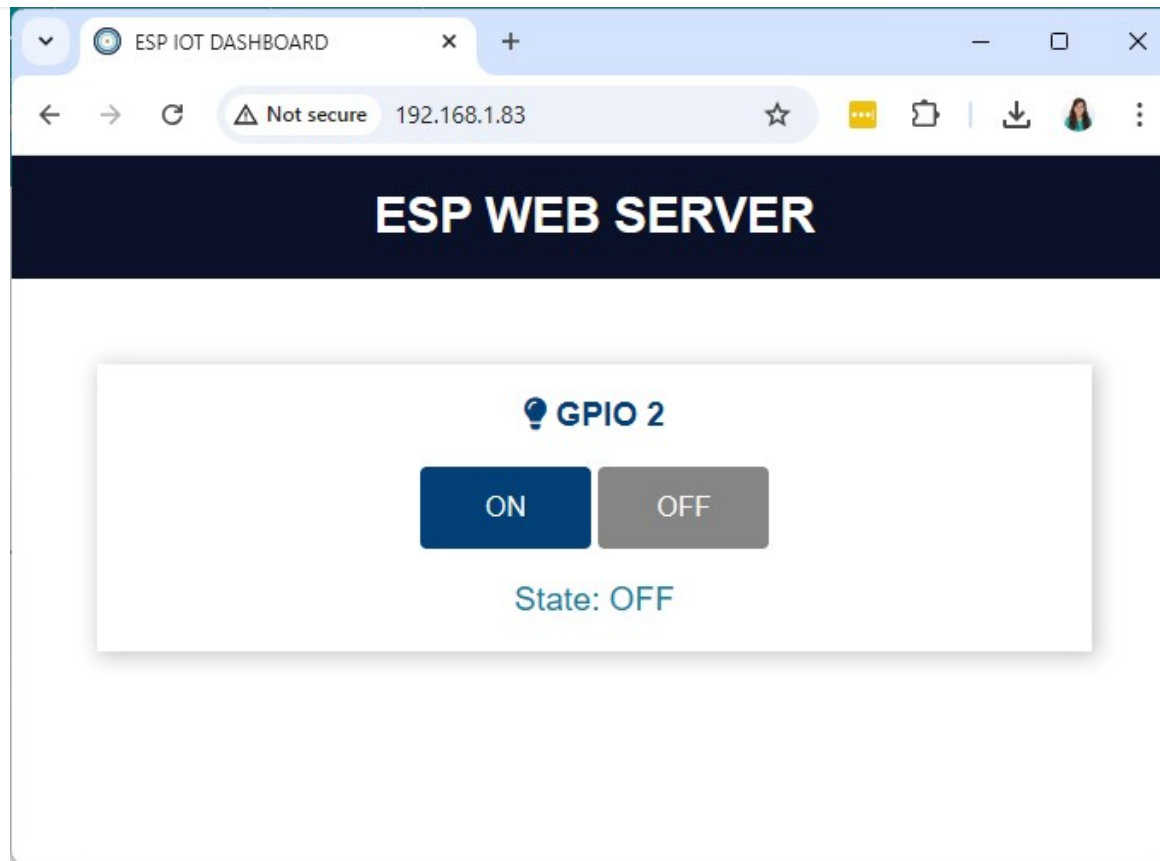
To make things easier, you can copy that file to your project folder.

Now that we have a *.littlefs.bin* file from the *data* folder, we can upload that file. Go to your ESP32 IP address followed by `/update`. Make sure you have the **Filesystem** option selected in the OTA Mode option.



Then, select the file with the *.littlefs.bin* extension.

After successfully uploading, click the **Back** button. And go to the root (/) URL again. You should get access to the following web page that controls the ESP32 built-in LED with two buttons.



If you need to update something on your project, you just need to go to your ESP32 IP address followed by `/update`.

Congratulations! You've successfully uploaded files to the ESP32 filesystem using the ElegantOTA library.

Wrapping Up

In this tutorial you've learned how to add OTA capabilities to your Async Web Server projects using the ElegantOTA library. This library is straightforward to use—you just need to add three lines of code to your project.

Additionally, this library also allows you to upload new firmware or files to the LittleFS filesystem effortlessly using a web page.

We hope you've found this tutorial useful.

Learn more about the ESP32 with our resources:

- [Learn ESP32 with Arduino IDE](#) (eBook)
- [Build ESP32 Web Servers with Arduino IDE](#) (eBook)
- [More ESP32 Projects and Tutorials...](#)

Thanks for reading.

PCBWay PCB Fabrication & Assembly

ONLY \$5 for 10 PCBs

✓ 24-hour Build Time ✓ Quality Guaranteed

✓ Most Soldermask Colors:

■ ■ ■ ■ ■ ■ ■ ■ ■ ■

Order now



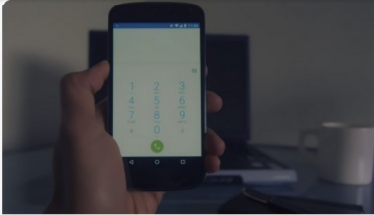
www.pcbway.com



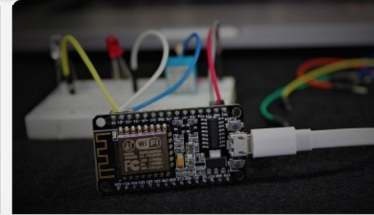
SMART HOME with Raspberry Pi, ESP32, ESP8266 [eBook]

Learn how to build a home automation system and we'll cover the following main subjects: Node-RED, Node-RED Dashboard, Raspberry Pi, ESP32, ESP8266, MQTT, and InfluxDB database [DOWNLOAD »](#)

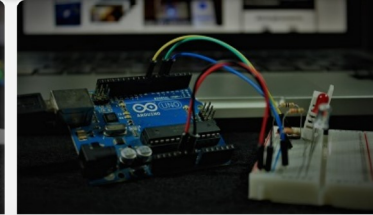
Recommended Resources



Build a Home Automation System from Scratch » With Raspberry Pi, ESP8266, Arduino, and Node-RED.



Home Automation using ESP8266 eBook and video course » Build IoT and home automation projects.



Arduino Step-by-Step Projects » Build 25 Arduino projects with our course, even with no prior experience!

What to Read Next...

ESP8266
NodeMCU
Static/Fixed IP
Address
(Arduino IDE)

ESP32 with
RCWL-0516
Microwave
Radar
Proximity
Sensor
(Arduino IDE)

ESP8266
NodeMCU
Web Server:
Display Sensor
Readings in
Gauges

Enjoyed this project? Stay updated by
subscribing our newsletter!

☐ SUBSCRIBE

83 thoughts on “ESP32 OTA (Over-the-Air) Updates – ElegantOTA Library with Arduino IDE”



Alex

December 18, 2024 at 10:20 am

Thank you all for publishing this Very interesting subject.

Ofcourse I will try it.

The only remark I have is that a third party like “ElegantOta” will be involved in our, sometimes privacy related, ESP- projects. In my case I prefer to setup an Ftp- server/client for remotely update

sketches or scripts to my ESP- boards.
Anyhow thank you very much for ALL of your effort in publishing these interesting projects!!

Alex

[Reply](#)



Dan

March 11, 2025 at 12:52 pm

Hi Alex,

Is it possible for you to share your FTP server/client (the ESP part, supposed to be the server) ?

Regards

[Reply](#)

**Ugur**

December 18, 2024 at 2:03 pm

Hi , in order to integrate this code to mine , should I change this `define ELEGANTOTA_USE_ASYNC_WEBSERVER 1` again ? Or I might move src file to my project file. I also want to define the functions that initialize the sensors in a separate source file to improve the readability of the code. Then I can include that file like `#include "file.h"`. What can you suggest for this?

Thanks for your efforts.

– Ugur

[Reply](#)**john smith**

December 19, 2024 at 5:51 pm

I would be interested about, why this code does not run on esp32-s3 ?

[Reply](#)**Sara Santos**

December 20, 2024 at 11:12 am

Hi.
What's the issue you get on your board?
Regards,
Sara

[Reply](#)**rfef**

December 20, 2024 at 6:13 pm

```
assert failed: tcp_alloc /IDF/components/lwip/lwip/  
src/core/tcp.c:1851 (Required to lock TCPIP core  
functionality!)
```

```
Backtrace: 0x403764ce:0x3fca9720  
0x4037cc2d:0x3fca9740 0x40383621:0x3fca9760
```

```
0x42022a1b:0x3fca9890 0x42022b85:0x3fca98b0
0x420059c4:0x3fca98d0 0x4200b712:0x3fca9920
0x42002c86:0x3fca9940 0x4200f75f:0x3fca9980
0x4037d91a:0x3fca99a0
```

ELF file SHA256: bdf4fae64

Rebooting...

ESP-ROM:esp32s3-20210327

Build:Mar 27 2021

rst:0xc (RTC_SW_CPU_RST),boot:0x2b
(SPI_FAST_FLASH_BOOT)

Saved PC:0x4037a8f6

SPIWP:0xee

mode:DIO, clock div:1

load:0x3fce2820,len:0x1188

load:0x403c8700,len:0x4

load:0x403c8704,len:0xbf0

load:0x403cb700,len:0x30e4

entry 0x403c88ac

[Reply](#)

**Sara Santos**

December 21, 2024 at 12:26 am

There is an issue with current version of
ESP32 core 3.1.0.
Please downgrade to 3.0.7.
That should solve the issue.
Regards,

Sara

[Reply](#)**An**

December 25, 2024 at 6:02 pm

If its any help, I have had this same
'Assert' failure using the Arduino
IDE 1.8.19, and spent many hours
trying to find a suitable solution.
Eventually, I used the Arduino 2

IDE, and the same code compiled with no problems.



Sara Santos

January 27, 2025 at 6:41 pm

Hi.

If you're still getting issues, you can get it permanently fixed.

See the solution here: https://rntlab.com/question/solvedassert-failed-tcp_alloc-idf-components-lwip-lwip-src-core-tcp-c1851-required-to-lock-tcpip-core-functionality/

Regards,
Sara

[Reply](#)

**Rodolfo**

January 6, 2025 at 8:06 am

Hi !! Did all changes, it keeps rebooting once and again. ESP32WROOM DA module. Any idea ?
Thanks !

[Reply](#)**Sara Santos**

January 27, 2025 at 6:42 pm

Hi.

I believe your issue is related to this: https://rntlab.com/question/solvedassert-failed-tcp_alloc-idf-components-lwip-lwip-src-core-tcp-c1851-required-to-lock-tcpip-core-functionality/

Regards,
Sara

[Reply](#)



larryesp32

December 19, 2024 at 10:44 pm

should be very helpful.

[Reply](#)

**Jeff Addleman**

December 19, 2024 at 11:36 pm

Will this work with ESPNow running on two ESP32s?
I am using ESPNow to send and receive data to and from the two
micros.

[Reply](#)**Janis**

December 20, 2024 at 11:13 am

Thank you

[Reply](#)**Hans**

December 21, 2024 at 12:24 pm

Hi Sara,
thanks for your nice OTA code. I have been following your instructions including downgrading to the esp 3.0.7 version. But i get this awful message while compiling:
cannot convert 'AsyncWebServer' to 'WebServer'
This message refers to the line
ElegantOTA.begin(&server); // Start ElegantOTA

Any idea which bloody mistake I'm making?
Thank for any help
Hans

[Reply](#)



Sara Santos

December 22, 2024 at 10:55 am

Hi.
Please read all the instructions.
You missed this section of the tutorial: **Enabling Async Mode**

Regards,
Sara

[Reply](#)



Hans

December 22, 2024 at 3:07 pm

Hello Sara,
please accept my deep thanks. Indeed, I missed to change the library setting. And now everything works fine.

Since long I have an ESP32 in my garden as wheater station, transmitting the data to my portable phone. Now it's much easier to make any changes.

By the way: The ESP actually operates in deep sleep mode waking up every 30 min. If I add for instance a 30 sec waiting time after data transmission, may I update during that period?

Have a wonderful Christmas time

Best regards from Germany

Hans

[Reply](#)



Sara Santos

December 23, 2024 at 11:13 am

Hi.

Yes. If you can send the new sketch before the board goes to sleep again.

Regards,
Sara

[Reply](#)



An

December 25, 2024 at 6:05 pm

If you include the ElegantOTA.h file, it has code to include `<AsyncTCP.h>` and `<ESPAsyncWebServer.h>`, so there is no need to `#include` these at the start of your own code.

[Reply](#)



Heikki Währn

December 22, 2024 at 12:21 pm

Hello !

I integrated my sketch with ElegantOTA and started the web server, it gives IP 192.168.1.133. I can access the server using my Android tablet (192.168.1.133/update) shows the OTA update screen. However using my PC (where my bins are) browser just cannot connect to the server. I have tested Firefox and Brave, and also typing <http://192.168.1.133/update>, no success. What is the problem ?

Is it some browser setting or should I modify my sketch ?

[Reply](#)



Sara Santos

December 22, 2024 at 2:35 pm

Hi.

What do you get when you try to access the web server on your browser?

Have you tried Google Chrome?

Is your computer on the same network?

Regards,

Sara

[Reply](#)



Heikki Währn

December 22, 2024 at 4:51 pm

Thanks Sara,
my Android devices and ESP32 are connected to
Wifi-router (same network) but the PC was
connected vai Ethernet cable (for speed), this is
not same network, (I didn't know this). When PC is
connected wirelessy,
it works as excepted !
Thanks and Merry Christmas for you in RNT !!
Keep going for your excellent tutorials

[Reply](#)



Sara Santos

December 23, 2024 at 11:13 am

Great.
I'm glad we found the issue,

Thank you and Merry Christmas for you too.

Regards,
Sara

[Reply](#)



Hans

December 22, 2024 at 3:13 pm

Hello Heikki,
sometimes I experience the same problem.
My hint:
make sure that your browser OPENS the site and is not
searching the site.
Rgds Hans

[Reply](#)



Juergen

December 28, 2024 at 12:51 pm

Hello,
I have two problems with running the two examples shown.
1. In the first example “ElegantOTA ESP32 Basic Example”
I cannot connect to my WLAN (name and password → OK)
2. In the second example “Upload a New Web Server Sketch via
OTA – Example”
I get the following error messages when compiling:
E:
\\MeineProgramme\\Arduino_Uno\\ESP32_OTA\\ESP32_OTA_02\\E

SP32_OTA_02.ino: In function 'void onEvent(AsyncWebSocket*, AsyncWebSocketClient*, AwsEventType, void*, uint8_t*, size_t)':
E:

\MeineProgramme\Arduino_Uno\ESP32_OTA\ESP32_OTA_02\E

SP32_OTA_02.ino:171:21: warning: format '%u' expects
argument of type 'unsigned int', but argument 3 has type
'uint32_t' {aka 'long unsigned int'} [-Wformat=]

171 | Serial.printf("WebSocket client #%u connected from %s\n",
client->id(), client->remoteIP().toString().c_str());

| ^~~~~~
~~~~~

||

| uint32\_t {aka long unsigned int}

E:

\MeineProgramme\Arduino\_Uno\ESP32\_OTA\ESP32\_OTA\_02\E

SP32\_OTA\_02.ino:174:21: warning: format '%u' expects  
argument of type 'unsigned int', but argument 3 has type  
'uint32\_t' {aka 'long unsigned int'} [-Wformat=]

174 | Serial.printf("WebSocket client #%u disconnected\n", client-  
>id());

| ^~~~~~  
~~~~~

||

| uint32_t {aka long unsigned int}

E:

\MeineProgramme\Arduino_Uno\ESP32_OTA\ESP32_OTA_02\ESP32_OTA_02.ino: In lambda function:

E:

\MeineProgramme\Arduino_Uno\ESP32_OTA\ESP32_OTA_02\ESP32_OTA_02.ino:224:18: error: no matching function for call to 'AsyncWebServerRequest::send(int, const char [10], const char [2845], String (&)(const String&))'

224 | request->send(200, "text/html", index_html, processor);

|

~~~~~^~~~~~

~~~~~

In file included from E:

\MeineProgramme\Arduino_Uno\ESP32_OTA\ESP32_OTA_02\ESP32_OTA_02.ino:14:

e:

\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\src/ESPAsyncWebServer.h:241:10: note: candidate: 'void AsyncWebServerRequest::send(const String&, size_t, AwsResponseFiller, AwsTemplateProcessor)'

241 | void send(const String& contentType, size_t len,

AwsResponseFiller callback, AwsTemplateProcessor

templateCallback=nullptr);

| ^~~~

e:

\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\src

rc/ESPAsyncWebServer.h:241:72: note: no known conversion for argument 3 from 'const char [2845]' to 'AwsResponseFiller' {aka 'std::function'}

241 | void send(const String& contentType, size_t len,
 AwsResponseFiller callback, AwsTemplateProcessor
 templateCallback=nullptr);

| ~~~~~~^~~~~~

e:

\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
 rc/ESPAsyncWebServer.h:240:10: note: candidate: 'void
 AsyncWebServerRequest::send(Stream&, const String&, size_t,
 AwsTemplateProcessor)'

240 | void send(Stream &stream, const String& contentType,
 size_t len, AwsTemplateProcessor callback=nullptr);

| ^~~~

e:

\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
 rc/ESPAsyncWebServer.h:240:23: note: no known conversion for argument 1 from 'int' to 'Stream&'

240 | void send(Stream &stream, const String& contentType,
 size_t len, AwsTemplateProcessor callback=nullptr);

| ~~~~~~^~~~~~

e:

\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
 rc/ESPAsyncWebServer.h:238:10: note: candidate: 'void

```
AsyncWebServerRequest::send(FS&, const String&, const
String&, bool, AwsTemplateProcessor)'
238 | void send(FS &fs, const String& path, const String&
contentType=String(), bool download=false,
AwsTemplateProcessor callback=nullptr);
| ^~~~
e:
\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
rc/ESPAsyncWebServer.h:238:19: note: no known conversion for
argument 1 from 'int' to 'AsyncWebServerRequest::FS&' {aka
'fs::FS&'}
238 | void send(FS &fs, const String& path, const String&
contentType=String(), bool download=false,
AwsTemplateProcessor callback=nullptr);
| ~~~~^~
e:
\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
rc/ESPAsyncWebServer.h:236:10: note: candidate: 'void
AsyncWebServerRequest::send(AsyncWebServerResponse*)'
236 | void send(AsyncWebServerResponse *response);
| ^~~~
e:
\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
rc/ESPAsyncWebServer.h:236:10: note: candidate expects 1
argument, 4 provided
```

```
e:
\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
rc/ESPAsyncWebServer.h:237:10: note: candidate: 'void
AsyncWebServerRequest::send(int, const String&, const
String&)'
237 | void send(int code, const String& contentType=String(),
const String& content=String());
| ^~~~
e:
\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
rc/ESPAsyncWebServer.h:237:10: note: candidate expects 3
arguments, 4 provided
e:
\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
rc/ESPAsyncWebServer.h:239:10: note: candidate: 'void
AsyncWebServerRequest::send(File, const String&, const
String&, bool, AwsTemplateProcessor)'
239 | void send(File content, const String& path, const String&
contentType=String(), bool download=false,
AwsTemplateProcessor callback=nullptr);
| ^~~~
e:
\MeineProgramme\Arduino_Uno\libraries\ESPAsyncWebServer\s
rc/ESPAsyncWebServer.h:239:20: note: no known conversion for
argument 1 from 'int' to 'AsyncWebServerRequest::File' {aka
```

```
'fs::File'}  
239 | void send(File content, const String& path, const String&  
    | contentType=String(), bool download=false,  
    | AwsTemplateProcessor callback=nullptr);  
    | ~~~~~^~~~~~
```

Compilation error: no matching function for call to
'AsyncWebServerRequest::send(int, const char [10], const char
[2845], String (&)(const String&))'
Kind Regards
Juergen

[Reply](#)**Juergen**

December 28, 2024 at 1:00 pm

Hello,
I use:
Arduino IDE 2.3.4
ESP32 V3.1.0 von Espressif
Device: ESP32 Dev Module

Juergen

[Reply](#)



Sara Santos

December 28, 2024 at 4:37 pm

Hi.

Downgrade the ESP32 boards version to 3.0.7 and try again.

Let me know if this fixes the issue.

Regards,

Sara

[Reply](#)



Milan Lomoz

December 29, 2024 at 1:00 am

Hi,

During the compilation of the code included in: “Upload a New Web Server Sketch via OTA – Example”, I got following message:

“no matching function for call to
'AsyncWebServerRequest::send(int, const char [10], const char [2845], String (&)(const String&))' “

What is wrong, please? My version of Arduino IDE is 1.8.12.

Thank you

Milan

[Reply](#)



Sara Santos

December 30, 2024 at 10:21 pm

Hi.

Check the version of your libraries and the ESP32 core.

I also recommend doing the upgrade for Arduino IDE 2.

Regards,

Sara

[Reply](#)



Jose Serena

January 8, 2025 at 11:38 am

To avoid the error change this line

```
request->send(200, "text/html", index_html, processor);
```

to

```
request->send(200, "text/html", index_html);
```

To see the status as soon as you load the page add the

line "notifyClients();" to the case WS_EVT_CONNECT.

```
case WS_EVT_CONNECT:
```

```
Serial.printf("WebSocket client #%u connected from
```

```
%s\n", client->id(), client->remoteIP().toString().c_str());  
notifyClients();  
break;
```

[Reply](#)**Juergen**

December 29, 2024 at 7:54 am

Hello,
I have downgrade the ESP32 boards to the version 3.07 but with this I have the same problems.
Kind Regards
Juergen

[Reply](#)



Juergen

January 2, 2025 at 2:59 pm

Hello,

In the second example “Upload a New Web Server Sketch via OTA – Example“

I get the following error message when compiling:

With the program code:

```
// Route for root / web page
```

```
server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){  
  request->send(200, "text/html", index_html, processor);  
});
```

in the function “void setup()“

I get the message:

Compilation error: no matching function for call to

```
'AsyncWebServerRequest::send(int, const char [10], const char  
[2845], String (&)(const String&))'
```

I use.

OS = WINDOWS 11

Arduino IDE v 2.3.2

LIB's: AsyncTCP, ESPAsyncWebServer from the website

ESP32 boards v3.07

Juergen

[Reply](#)

**Jose Serena**

January 8, 2025 at 11:39 am

To avoid the error change this line

```
request->send(200, "text/html", index_html, processor);  
to  
request->send(200, "text/html", index_html);
```

To see the status as soon as you load the page add the line "notifyClients();" to the case WS_EVT_CONNECT.

```
case WS_EVT_CONNECT:
```

```
Serial.printf("WebSocket client #%u connected from  
%s\n", client->id(), client->remoteIP().toString().c_str());  
notifyClients();  
break;
```

[Reply](#)**Juergen**

January 9, 2025 at 2:04 pm

Hello,
thank you very much for the changes shown, so I
can compile the sketch and perform an update.
Kind Regards
Juergen

[Reply](#)



melt

January 4, 2025 at 11:32 am

i was thinking that version 2 works fine and version 3 includes
upgrade messages to pro which doesn't look very appealing
though its understandable. What would be the concrete
advantage to go to version 3 of the library?

[Reply](#)



Ayush

January 10, 2025 at 1:46 pm

Hi melt, Ayush here – author of ElegantOTA. It does have a lot of improvements and I'm happy to share the following key points:

Version 3 apart from new UI has bug fixes related to stability of OTA updates as reported by a few users. It has built-in upload MAC validation so any kind of corruption while uploading your OTA file will be detected which prevents bricking your devices.

You also now have callbacks: onStart, onProgress and onEnd, which can be used to perform certain tasks based on your application. For example, updating the LCD etc. (docs.elegantota.pro/features/callbacks)

[Reply](#)



melt

January 10, 2025 at 5:48 pm

thank you for the info! and thank you for ur great work ! out of curiosity, how much time ur computer needs to compile the library? is it also for you

about 2 minutes?

[Reply](#)



Ayush

January 10, 2025 at 6:23 pm

You're welcome! It's about 1 minute when compiling with Arduino IDE and 30 seconds on PlatformIO. This is totally dependent on how powerful is your PC.

[Reply](#)

**melt**

January 11, 2025 at 8:08 am

oh didn't know platformIO builds
faster thank you very much! have a
nice weekend!

**ade**

June 20, 2025 at 8:06 pm

Good work.

I came across the library, and I tried to test it on wemos d1 mini.

but it's showing MD5 mismatch error when uploading reach 100%.

what can be the issue and how to solve it.

thanks.

[Reply](#)**Czeslaw**

January 11, 2025 at 4:12 pm

I run OTA projects under Arduino IDE2. And it works, but there is a problem.

When I try to remotely upload subsequent file corrections in the 'DATA' directory to the ESP, it can only be done once.

After compiling the corrected project with the export function,
When I call the 'CTRL+SHIFT+P' function again,
No new file appears in C:
\\Users\\PC\\AppData\\Local\\Temp\\xxx.littlefs.bin
I only get the message:

"LittleFS Filesystem Uploader v1.5.3 — github.com/earlephilhower/arduino-littlefs-upload
Sketch Path: C:
\\Users\\PC\\Documents\\Arduino_ESP32_folder\\ESP32_controllery
_input\\edit____ESP32__inp_mil_email_ota_181\\controllerEmail
Data Path: C:
\\Users\\PC\\Documents\\Arduino_ESP32_folder\\ESP32_controllery
_input\\edit____ESP32__inp_mil_email_ota_181\\controllerEmail\\
data
Device: ESP32 series, model esp32
Using partition: default
Partitions: C:
\\Users\\PC\\AppData\\Local\\Arduino15\\packages\\esp32\\hardware\\
esp32\\2.0.17\\tools\\partitions\\default.csv
Start: 0x290000
End: 0x3f0000
ERROR: No port specified, check IDE menus."

My question is why the file can't be generated again. Does this
version of ElegantOTA work like that? Where could the error be?

[Reply](#)**Czeslaw**

January 12, 2025 at 2:53 pm

I spent many hours, made dozens of attempts to compile 'data' files. My conclusion is as follows.

As long as the Arduino platform is open, you can modify files in the data directory, compile them and send them remotely to the ESP board. However, if the Arduino is closed and reopened, the remote data file transfer function is canceled. And then you have to reconnect the ESP to the computer with a cable and repeat the steps again.

[Reply](#)**Volker Löbe**

January 19, 2025 at 4:33 pm

Hello and thank you for the description.
My project worked until today.
Suddenly, after downloading to the ESP32 dev module, I get the following message in the serial monitor:

```
assert failed: tcp_alloc /IDF/components/lwip/lwip/src/core/  
tcp.c:1851 (Required to lock TCPIP core functionality!)
```

```
Backtrace: 0x400826e9:0x3ffb1fc0 0x4008d23d:0x3ffb1fe0  
0x4009359e:0x3ffb2000 0x400fa4ff:0x3ffb2130  
0x400fa679:0x3ffb2150 0x400daa60:0x3ffb2170  
0x400e08fe:0x3ffb21c0 0x400d3c6a:0x3ffb21e0  
0x400e592f:0x3ffb2270 0x4008e372:0x3ffb2290
```

So far I haven't had any problems.
The program hasn't been changed!

I use Arduino IDE 2.3.3
AsyncTCP from dvarrel 1.1.1
ESP Async WebServer from Me-No-Dev 1.2.4
Best regards
Volker

[Reply](#)

**Sara Santos**

January 20, 2025 at 9:44 am

Hi.

That's currently an issue with the recent version of the ESP32 core.

Downgrade to version 3.0.7. Go to Tools > Boards > Boards Manager > ESP32 and downgrade to version 3.0.7.

It should solve that issue.

Regards,

Sara

[Reply](#)

**Sara Santos**

January 27, 2025 at 6:40 pm

If you're still getting issues, there's a way to solve this permanently.

Please see my solution here: https://rntlab.com/question/solvedassert-failed-tcp_alloc-idf-components-lwip-lwip-src-core-tcp-c1851-required-to-lock-tcpip-core-functionality/

Regards,
Sara

[Reply](#)



Ian

January 28, 2025 at 4:39 am

Sorry Sara with your solution above installed my ESP32 is still continuously rebooting.Any other thoughts.

Regards Ian

[Reply](#)



Ian

January 28, 2025 at 4:41 am

I'm using Arduino IDE Arduino IDE 2.3.4

[Reply](#)



Sara Santos

January 28, 2025 at 10:45 am

Can you tell me...
Version of the libraries?
Version of the ESP32 core?



Sara Santos

January 28, 2025 at 10:46 am

Please note that we're using
different libraries now.

**Rob**

February 21, 2025 at 11:45 am

With ArduinoOTA you can directly push an OTA update from your IDE. That looks a lot simpler to me compared to: exporting the file, browse to the webserver and finding the file again. Why would you use ElegantOTA?

[Reply](#)**Asghar**

March 10, 2025 at 8:27 pm

How can we use OTA to update bin file multiple times .I updated the bin file one time but I want to change it and update another file by OTA using Wifi, but I don't know how to do the job.

[Reply](#)

**Dan**

March 11, 2025 at 12:39 pm

Hi

I'm trying your tuto

About libraries, you wrote

Click the links below to download the libraries' .zip folders.

ESPAsyncWebServer

AsyncTCP

*These libraries aren't available to install through the Arduino
Library Manager, so you need >to copy the library .zip files*

.....

Actually, there are 2 versions of each library already included il
Arduino's 2.3.4 Libraries list.

(wondering why), are yours part of them ?

It's a bit confusing

Regards

[Reply](#)**Sara Santos**

March 13, 2025 at 10:25 am

Hi.

They are now available.

We'll update all our guides that use those libraries.

Regards,

Sara

[Reply](#)**Dan**

March 11, 2025 at 12:43 pm

After the last comment, I started to receive emails to confirm subscription, but it don't work, I have this messae each time

Your Subscription Confirmation link is expired or invalid

Click on this button and we will send you a new link

[Reply](#)



Bill Webb

March 18, 2025 at 11:08 pm

I tried compiling your example and got the following errors:

```
c:
\Users\17147\Documents\Arduino\libraries\ESP_Async_WebServ
er\src\ESPAsyncWebServer.h: In member function 'tcp_state
AsyncWebServer::state() const':
c:
\Users\17147\Documents\Arduino\libraries\ESP_Async_WebServ
er\src\ESPAsyncWebServer.h:1086:49: error: passing 'const
AsyncServer' as 'this' argument discards qualifiers [-fpermissive]
1086 | return static_cast(_server.status());
    | ~~~~~^~
In file included from C:
\Users\17147\AppData\Local\Temp.arduinoIDE-
unsaved2025218-6392-
```

```
c9tnzp.5xyn6\sketch_mar18a\sketch_mar18a.ino:9:
c:\Users\17147\Documents\Arduino\libraries\AsyncTCP\src/
AsyncTCP.h:198:13: note: in call to 'uint8_t
AsyncServer::status()'
198 | uint8_t status();
    | ^~~~~~
```

Compilation error: exit status 1

I am using the 2.3.4 IDE and the Espressif Library ESP32 3.1.1.

Your thoughts? The first error probably caused the second error.
The first error looks like something that should be a warning, not an error.

[Reply](#)



Sara Santos

March 19, 2025 at 3:15 pm

Hi.

Maybe you need to install the new ESPAsyncWebServer and AsyncTCP libraries by ESP32Async. These are

available to install in the ARduino library Manager.

Regards,

Sara

[Reply](#)



Bill Webb

March 19, 2025 at 7:14 pm

Well, that was interesting. The error noted in my original post went away. I was using the me-no-dev ESPAsync Web server library. I deleted it and its dependencies and the reloaded from the IDE manager. Now I have new errors.

```
c:
\Users\17147\Documents\Arduino\libraries\ESPAs
yncWebServer\src\WebAuthentication.cpp: In
function 'bool getMD5(uint8_t*, uint16_t, char*)':
```

```
c:
\Users\17147\Documents\Arduino\libraries\ESPAs
yncWebServer\src\WebAuthentication.cpp:74:3:
error: 'mbedtls_md5_starts_ret' was not declared
```



```
in this scope; did you mean 'mbedtls_md5_starts'?
74 | mbedtls_md5_starts_ret(&_ctx);
    | ^~~~~~~~~~~~~~~~~~~~~~
    | mbedtls_md5_starts
c:
/Users/17147/Documents/Arduino/libraries/ESPAs
yncWebServer/src/WebAuthentication.cpp:75:3:
error: 'mbedtls_md5_update_ret' was not declared
in this scope; did you mean
'mbedtls_md5_update'?
75 | mbedtls_md5_update_ret(&_ctx, data, len);
    | ^~~~~~~~~~~~~~~~~~~~~~
    | mbedtls_md5_update
c:
/Users/17147/Documents/Arduino/libraries/ESPAs
yncWebServer/src/WebAuthentication.cpp:76:3:
error: 'mbedtls_md5_finish_ret' was not declared
in this scope; did you mean 'mbedtls_md5_finish'?
76 | mbedtls_md5_finish_ret(&_ctx, _buf);
    | ^~~~~~~~~~~~~~~~~~~~~~
    | mbedtls_md5_finish
Multiple libraries were found for "WiFi.h"
Used:
Compilation error: exit status 1
```

This error is documented in <https://github.com/me-no-dev/ESPAsyncWebServer/issues/1419>

Although this issue is closed, I don't see a released solution.

Any ideas?

[Reply](#)



Sara Santos

March 19, 2025 at 10:58 pm

Are you installing the libraries by
ESP32Async?

Regards,
Sara

[Reply](#)

**Dan**

March 19, 2025 at 8:35 pm

Hi Bill

Let me check some items

Did you copy/paste the code from this page ? (I suppose YES)

Which Arduino IDE are you using ? (1.xx or 2.xx)

Which board have you declared ?

Version of ESP32 boards library ?

Have you tried to download necessary libraries using the
hyperlinks in this page and install their ZIP files manually ?

I recently compiled this example without problems, you should
too

[Reply](#)**Bill Webb**

March 19, 2025 at 10:33 pm

Yes, copy and paste.

IDE 2.3.4

ESP32 board manager 3.1.1

ESP32 Dev Module

Deleted libraries and reloaded using the zips.

And..... Same error

[Reply](#)



Sara Santos

March 19, 2025 at 10:56 pm

Hi.

Don't use the zip files.

Use the newest fork of the libraries available on the Arduino library manager.

Regards,

Sara

[Reply](#)

**Bill Webb**

March 19, 2025 at 11:02 pm

Got it! There ere several copies of some of the libraries. I had not deleted all of the old ones.

[Reply](#)**Dan**

March 19, 2025 at 11:11 pm

Hi Bill
the only difference with my setup is the board...
have you tried another target (ESP32 C or S series) ?
ESP32 v3 is quite recent, you can easily try to downgrade to v2

[Reply](#)

**Forrest Erickson**

March 21, 2025 at 1:05 pm

Update this page with a “favicon.png” or refer us to another tutorial

Hello RNTs,
The example index.html code calls out a favicon in the header.

Could you add that file to this project tutorial or refer us to another of your tutorials (or make one) what explains favicon?

[Reply](#)**Tapio**

March 28, 2025 at 10:39 am

Greetings from Finland! 🇫🇮

This is a fantastic example – it worked flawlessly right out of the box! ✨

I've extended it by adding a remote serial debugging tool (together with helpfull AI tools (DeepSeek and GitHub Copilot). Would you be open to me sharing this improved version in the Random Nerd Tutorials Facebook community when it's ready? I'll of course give full credit to your original implementation.

Thank you for sharing such a well-designed and practical example – it's been incredibly helpful!"

[Reply](#)

**Sara Santos**

March 28, 2025 at 5:37 pm

Hi.

Yes.

You can share it in our group.

Also, don't forget the merit to the library's creator.

Regards,

Sara

[Reply](#)

**Leo Duivenvoorde**

April 5, 2025 at 1:21 pm

Hello,

I have made a sketch with the OTA -things in there as described in your article. It worked great.

Today I opened the same sketch again. Compiled it OK in

Arduino IDE 2.3.5.

Then I just added the line " String fileName = "Test-OTA-Blink"; ".
And tried to compile it. But now I get the message that the WiFi.h library is not there. (Just 1 minute before this it compiled OK.)
WiFi.h is present in my sketch-folder: myDocs/Arduino/Libraries/WiFi/src/

Can you explain this?

With best regards,
Leo

[Reply](#)



Sara Santos

April 5, 2025 at 2:18 pm

Hi.

Do you have an ESP32 board selected in Tools > Board?

Regards,

Sara

[Reply](#)



Dan

April 5, 2025 at 4:58 pm

hi, it sounds like a syntax error (a {, }, ; missing) or your new line is not at the right place, Is it a global value ?

[Reply](#)



Leo Duivenvoorde

April 6, 2025 at 9:01 am

Initially, it was indeed at the wrong place: above

<Arduino.h>, which I normally do not use. I commented the line out, but same problem (no WiFi.h present) kept coming back. Today, I restarted the computer and problem did not come back.

But, I noticed that now the IDE used the WiFi.h from :
Multiple libraries were found for "WiFi.h"

Used: C:

\Users\leodu\AppData\Local\Arduino15\packages\esp32\hardware\esp32\3.2.0\libraries\WiFi

Not used: D:

\LEO\OneDrive\Documenten\Arduino\libraries\WiFi

So, the problem is solved.

But, at hindsight, I do sometimes notice that I have to reboot the computer after compilations-errors in order to clear a problem; just like something remains in memory.

Also: Is there a way I can store all the libraries and packages on one single drive/location? This is much easier for back-ups etc.

But, any way, thanks for your help.

With best regards,
Leo

[Reply](#)**Rrr**

May 14, 2025 at 3:33 am

Hello, this post is about ESP32 flashing its own flash memory.
How to make one ESP32[1] flash a second ESP32[2]?

I believe that in the case that an ESP32[1] can program another ESP32[2] (perhaps using the serial port), it is possible to send the firmware without the OTA part, so that an ESP32[1] will always have its firmware preserved as an “ESP32 programmer”

[Reply](#)**Cristian**

June 22, 2025 at 10:23 pm

Hi, i havent been able to set the authentication, i think i have tried

everything, i am doing the setAuth... But was never able to make it work, any thought?

[Reply](#)



Austin

September 28, 2025 at 12:54 am

It's unclear from the documentation, but setAuth must be called AFTER .begin(), because ElegantOTA.begin(&server) actually takes 3 arguments (server, username, password) and will overwrite what you did in .setAuth(). Or you can just not call .setAuth() and pass your username and password to the .begin()

```
(void  
ElegantOTAClass::begin(ELEGANTOTA_WEBSERVER  
*server, const char * username, const char * password){}
```

[Reply](#)

**Solo**

July 25, 2025 at 4:33 pm

Hi, the ElegantOTA upload fails consistently at about the 40% mark with "Upload failed \n Server returned status code 0". Sketch uses 1365335 bytes (43%) of program storage space. Maximum is 3145728 bytes. Global variables use 56228 bytes (17%) of dynamic memory, leaving 271452 bytes for local variables. Maximum is 327680 bytes.

I am using a ESP32S3 with 16MB on the Arduino IDE. Partition Scheme is: 16M Flash (3MB App/9.9 MB FATFS).

Can you help suggest what I could do to fix this? Thanks.

[Reply](#)**Sara Santos**

July 26, 2025 at 11:05 am

Hi.

Without more information, it is very difficult to understand what might be wrong.

Maybe it is a good idea to try to open an issue in the library Github page:<https://github.com/ayushsharma82/ElegantOTA/issues>

Regards,
Sara

[Reply](#)



Solo

August 16, 2025 at 5:11 pm

Thanks for your response. The thing now works fine with “Partition Scheme: 16M Flash (2MB APP/12.5MB FATFS)”. I have no idea as to why !!

Your web guides are great. And thanks to Ayush Sharma as well.

Perhaps you could mention the partitioning requirements in this article. Some readers may not even know the fundamentals and that OTA requires partitioning. Perhaps it is described in a different article.

[Reply](#)



Sara Santos

August 17, 2025 at 10:36 am

Hi.
I think I have a partition scheme like that
set by default.
That's why I wasn't aware we needed to
select that.

Thanks for letting me know.

Regards,
Sara

[Reply](#)



Cheetara

August 4, 2025 at 10:12 am

I launched the Arduino sketch incorporating all your suggestions,
but the ESP32 never connected to the Wi-Fi. It just keeps
showing the ‘.’ symbol without interruption.
The ElegantOTA version is 3.17 by Ayush Sharma; the ESP
Async WebServer by ESP32Asyn version is 3.7.10; and the

Async TCP by ESP32Asyn version is 3.4.6.
I wrote the credentials of the Wi-Fi used to connect all devices to the network in the SSID and Password fields.

[Reply](#)



Sara Santos

August 4, 2025 at 3:51 pm

The dots usually mean that the board is not able to connect to Wi-Fi.

Or the credentials are wrong or the router is not close enough.

Double-check the SSID and password. Uppercase, lowercase and special chracters, and spaces matter.

Regards,

Sara

[Reply](#)

**Dan**

August 11, 2025 at 2:35 pm

Hi Sara,
you both always make good job
is there a way to make OTA work with an ESP in WIFI_AP
mode?

[Reply](#)**Dan**

August 16, 2025 at 6:15 pm

te answer is yes

[Reply](#)**Clovis Fritzen**

December 9, 2025 at 11:48 pm

Hey, got it all working, thank you for the always excellent tutorials. Obrigado aqui do Brasil. I always link your tutorials when writing my own.

[Reply](#)



Sara Santos

December 11, 2025 at 2:58 pm

That's great!
Thank you.
Regards,
Sara

[Reply](#)

Leave a Comment

☐ Notify me of follow-up comments by email.

☐ Notify me of new posts by email.

Post Comment

[About](#) [Support](#) [Terms and Conditions](#) [Privacy Policy](#) [Refunds](#) [Complaints' Book](#) [MakerAdvisor.com](#) [Join the Lab](#)

Copyright © 2013-2026 · [RandomNerdTutorials.com](#) · All Rights Reserved

Mettre à jour les paramètres de confidentialité