Getting started with PlatformIO IDE extension on VSCodium (Linux)

PlatformIO

Install PlatformIO IDE Extension on VS Code, following the guide https://randomnerdtutorials.com/vs-code-platformio-ide-esp32-esp8266-arduino/

Troubleshooting – Cannot install Platformio on VScodium (not VSCode), no such extension exists for VScodium

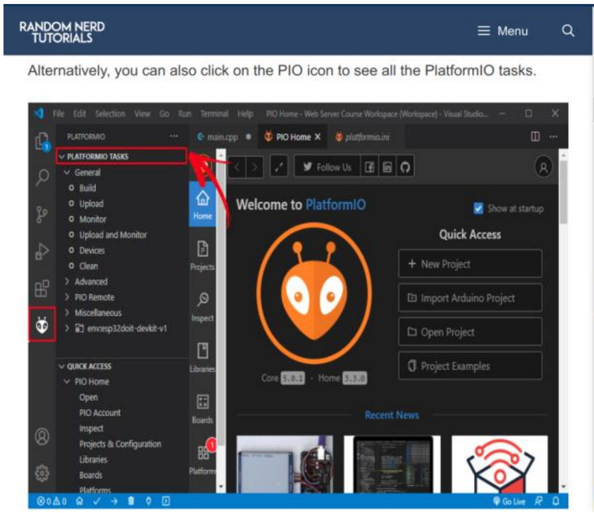
Resolved, by manual installation of ms-vscode.cpptools and platformio.platformio-ide, according to https://community.platformio.org/t/platformio-on-vscodium/31136

Troubleshooting – Cannot finish platformio installation, fail with “PlatformIO: Can not find working Python 3.6+ Interpreter. Please install the latest Python 3 and restart VSCode”.

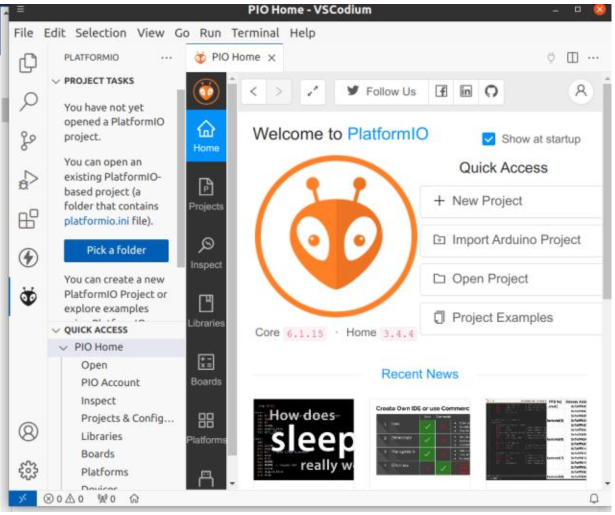
Resolved, using the command “**sudo apt install -y python3-venv**”, following the forum https://community.platformio.org/t/ubuntu-vscode-pio-extension-install-platformio-can-not-find-working-python-3-6-interpreter/27853

Create new Project

- PlatformIO tasks – absent, refer the guide https://randomnerdtutorials.com/vs-code-platformio-ide-esp32-esp8266-arduino/
- I have “Project Tasks” instead. However, not the same picture I expected to get.
- On the left – the guide, on the right – my environment.



Alternatively, you can also click on the PIO icon to see all the PlatformIO tasks.

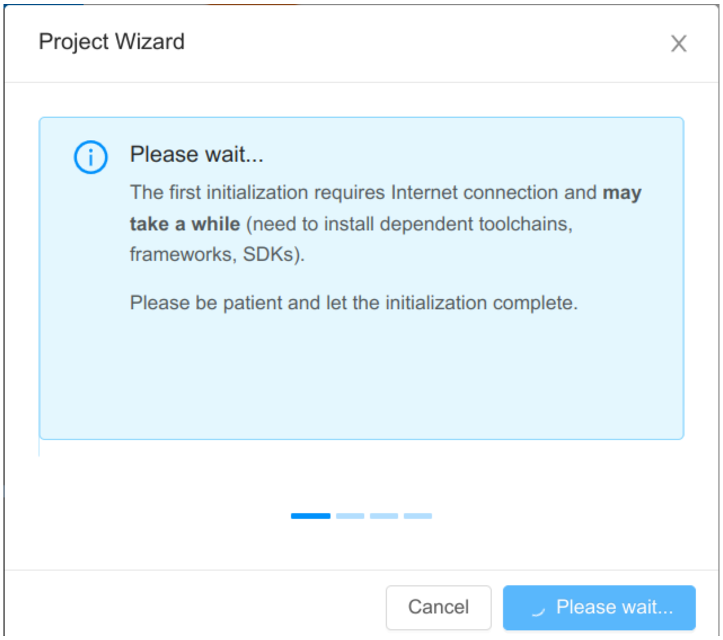


You have not yet opened a PlatformIO project.

You can open an existing PlatformIO-based project (a folder that contains platformio.ini file).

You can create a new PlatformIO Project or explore examples

- Let’s keep going with the guide and create a new project first.
- Default location is /home/Documents/PlatformIO/Projects.
- The wizard appears:

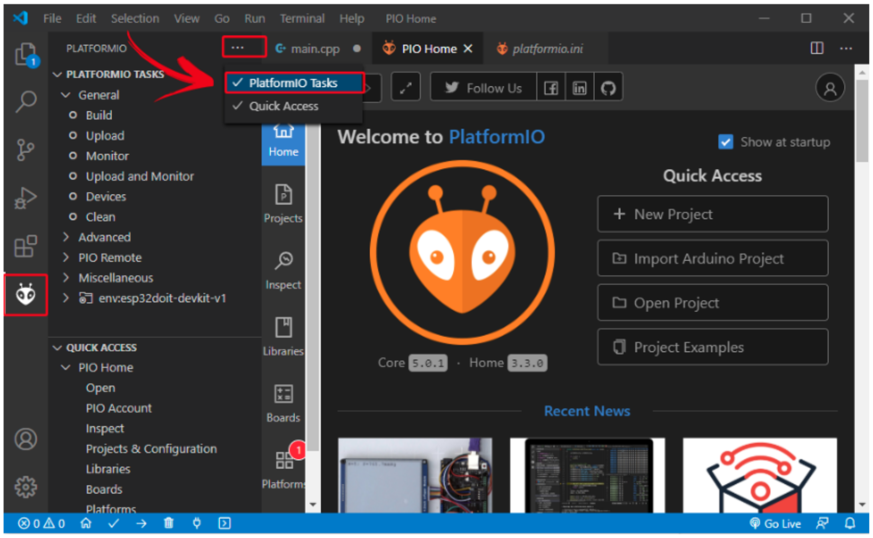


Please wait...

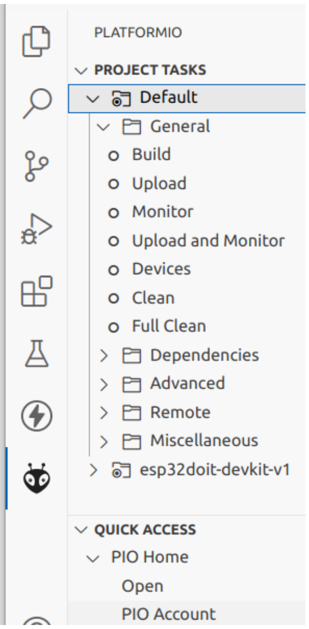
The first initialization requires Internet connection and **may take a while** (need to install dependent toolchains, frameworks, SDKs).

Please be patient and let the initialization complete.

- Now I get almost the same picture as expected.



PlatformIO IDE Overview



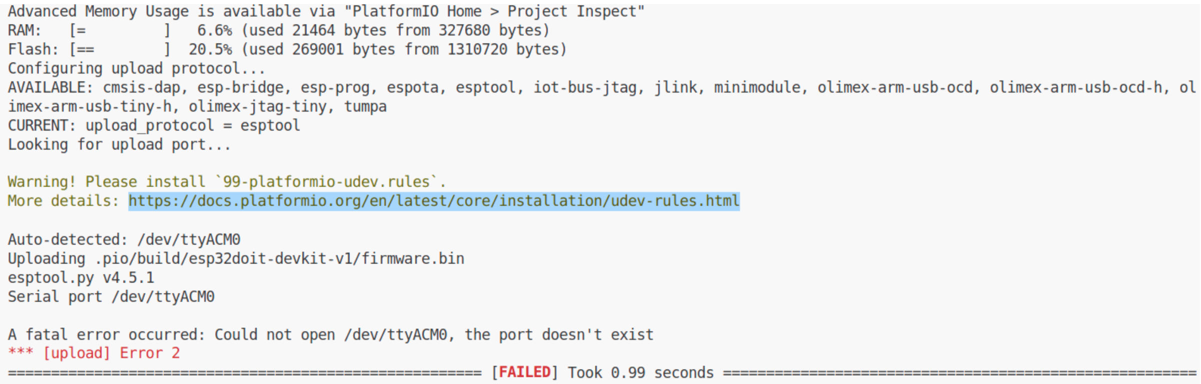
Upload code from PlatformIO to ESP32

Uploading Code using PlatformIO IDE, from the guide https://randomnerdtutorials.com/vs-code-platformio-ide-esp32-esp8266-arduino/:

- Copy the code into src/main.cpp file and upload it.

Troubleshooting – Failed to upload: "A fatal error occurred: Could not open /dev/ttyACM0, the port doesn't exist".

- First try: Close the Serial Monitor in Arduino IDE / Platform IO
- Second try: Follow the instructions on output:

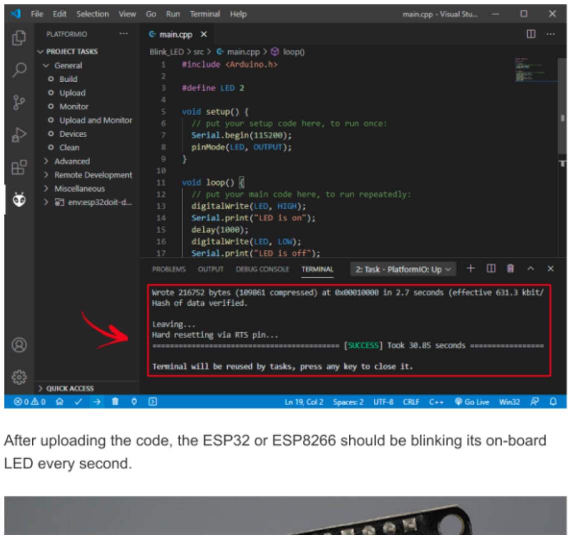


```
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM:   [=         ]   6.6% (used 21464 bytes from 327680 bytes)
Flash: [==        ]  20.5% (used 269001 bytes from 1310720 bytes)
Configuring upload protocol...
AVAILABLE: cmsis-dap, esp-bridge, esp-prog, espota, esptool, iot-bus-jtag, jlink, minimodule, olimex-arm-usb-ocd, olimex-arm-usb-ocd-h, ol
imex-arm-usb-tiny-h, olimex-jtag-tiny, tumpa
CURRENT: upload_protocol = esptool
Looking for upload port...

Warning! Please install `99-platformio-udev.rules`.
More details: https://docs.platformio.org/en/latest/core/installation/udev-rules.html

Auto-detected: /dev/ttyACM0
Uploading .pio/build/esp32doit-devkit-v1/firmware.bin
esptool.py v4.5.1
Serial port /dev/ttyACM0

A fatal error occurred: Could not open /dev/ttyACM0, the port doesn't exist
*** [upload] Error 2
====================================================== [FAILED] Took 0.99 seconds ======================================================
```

-
- https://docs.platformio.org/en/latest/core/installation/udev-rules.html
- Open the Terminal and run:
  curl -fsSL https://raw.githubusercontent.com/platformio/platformio-

core/develop/platformio/assets/system/99-platformio-udev.rules | sudo tee /etc/udev/rules.d/99-platformio-udev.rules

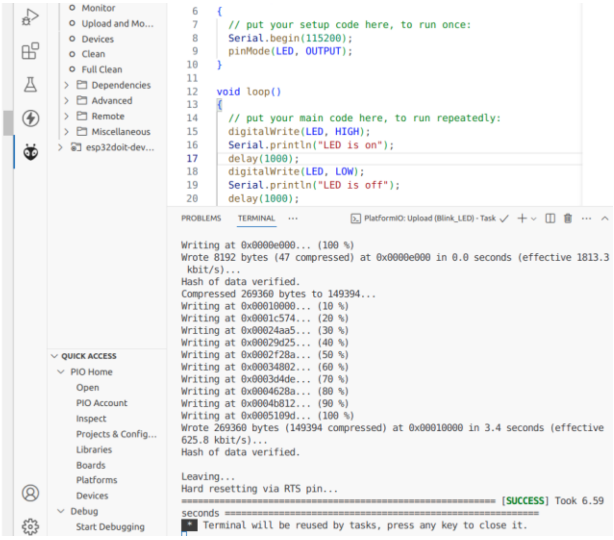
- Then: sudo service udev restart
- This workaround RESOLVED the upload issue, and the built-in led blinks as expected:



```
#include <Arduino.h>

#define LED 2

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  pinMode(LED, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(LED, HIGH);
  Serial.print("LED is on");
  delay(1000);
  digitalWrite(LED, LOW);
  Serial.print("LED is off");
```

```
Wrote 216752 bytes (109861 compressed) at 0x00010000 in 2.7 seconds (effective 631.3 kbit/
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
========================================== [SUCCESS] Took 30.85 seconds ================

Terminal will be reused by tasks, press any key to close it.
```

After uploading the code, the ESP32 or ESP8266 should be blinking its on-board LED every second.

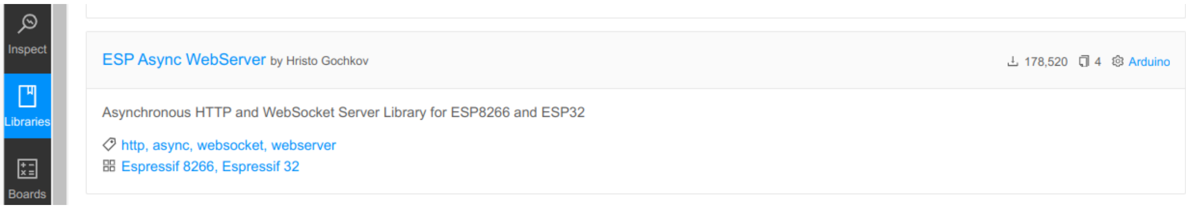


```
{
  // put your setup code here, to run once:
  Serial.begin(115200);
  pinMode(LED, OUTPUT);
}

void loop()
{
  // put your main code here, to run repeatedly:
  digitalWrite(LED, HIGH);
  Serial.println("LED is on");
  delay(1000);
  digitalWrite(LED, LOW);
  Serial.println("LED is off");
  delay(1000);
```

```
Writing at 0x0000e000... (100 %)
Wrote 8192 bytes (47 compressed) at 0x0000e000 in 0.0 seconds (effective 1813.3
 kbit/s)...
Hash of data verified.
Compressed 269360 bytes to 149394...
Writing at 0x00010000... (10 %)
Writing at 0x0001c574... (20 %)
Writing at 0x00024aa5... (30 %)
Writing at 0x00029d25... (40 %)
Writing at 0x0002f28a... (50 %)
Writing at 0x00034802... (60 %)
Writing at 0x0003d4de... (70 %)
Writing at 0x0004628a... (80 %)
Writing at 0x0004b812... (90 %)
Writing at 0x0005109d... (100 %)
Wrote 269360 bytes (149394 compressed) at 0x00010000 in 3.4 seconds (effective
625.8 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
=========================================================== [SUCCESS] Took 6.59
seconds ==========================================================
 *  Terminal will be reused by tasks, press any key to close it.
```

Add libraries

- PIO Home-> Libraries. For example, search for “ESP Async WebServer”:



ESP Async WebServer by Hristo Gochkov

Asynchronous HTTP and WebSocket Server Library for ESP8266 and ESP32

http, async, websocket, webserver

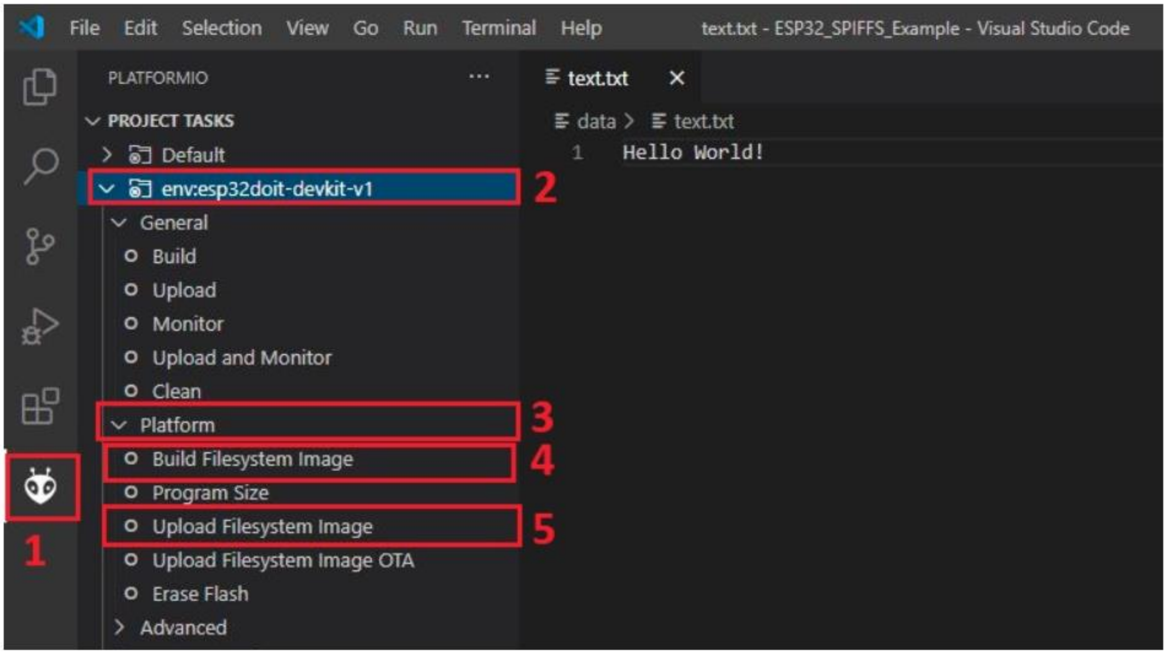
Espressif 8266, Espressif 32

Choose the one owned by Hristo Gochkov with is also maintained by him and press the ”Add to Project” button. This will also resolve the dependencies and install them, hence no need to separately install those.

File System – SPIFFS

Read the Data Storage guide.

- How to upload FS to ESP32 using PlatformIO and VSCode, refer https://randomnerdtutorials.com/esp32-vs-code-platformio-spiffs/
- SPIFFS tool doesn't support directories, so everything is saved on a flat structure.
- The "data" folder should be under the project folder and the files you want to upload should be inside the data folder. Otherwise, it won't work.
- Uploading Filesystem Image:
  1. Click the PIO icon at the left side bar. The project tasks should open.
  2. Select **env:esp32doit-devkit-v1** (it may be slightly different depending on the board you're using).
  3. Expand the **Platform** menu.
  4. Select **Build Filesystem** Image.
  5. Finally, click **Upload Filesystem Image**.

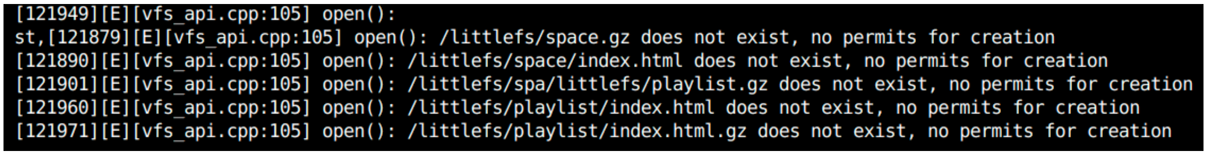


```
Hello World!
```

File System - LittleFS

- On PlatformIO ini file, set the key: board_build.filesystem = littlefs
- Note the name is "littlefs" and not "LittleFS" like the library.
- Restart VSCodium, erase flash and rebuild the image.

- Troubleshooting – when browsing the site, get these errors:

  ```
  [121949][E][vfs_api.cpp:105] open():
  st,[121879][E][vfs_api.cpp:105] open(): /littlefs/space.gz does not exist, no permits for creation
  [121890][E][vfs_api.cpp:105] open(): /littlefs/space/index.html does not exist, no permits for creation
  [121901][E][vfs_api.cpp:105] open(): /littlefs/spa/littlefs/playlist.gz does not exist, no permits for creation
  [121960][E][vfs_api.cpp:105] open(): /littlefs/playlist/index.html does not exist, no permits for creation
  [121971][E][vfs_api.cpp:105] open(): /littlefs/playlist/index.html.gz does not exist, no permits for creation
  ```

  Giving up, revert to use built-in SPIFFS…