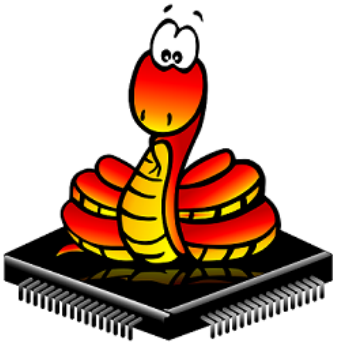
Getting Started with MicroPython on ESP32 and ESP8266

Learn how to get started with MicroPython firmware on the ESP32 and ESP8266. We'll introduce you to MicroPython, show you the differences between MicroPython and regular Python, and how to program your ESP based boards with MicroPython using uPyCraft IDE. After completing this guide, you'll have your first LED blinking using MicroPython.

What is MicroPython?

MicroPython is a re-implementation of Python 3 targeted for microcontrollers and embedded systems. MicroPython is very similar with regular Python. So, if you already know how to program in Python, you also know how to program in MicroPython.



Python vs MicroPython

Apart from a few exceptions, the language features of Python are also available in MicroPython. The biggest difference between Python and MicroPython is that MicroPython was designed to work under constrained conditions.

Because of that, MicroPython does not come with the full standard library. It only includes a small subset of the Python standard library. However, it does include modules to access low-level hardware – this means that there are libraries to easily access and interact with the GPIOs.

Additionally, devices with Wi-Fi capabilities like the ESP8266 and ESP32 include modules to support network connections.

Why MicroPython?

Python is one of the most widely used, simple and easy-to-learn programming languages around. So, the emergence of MicroPython makes it extremely easy and simple to program digital electronics. If you’ve never programmed digital electronics before, MicroPython is a good starting point.

MicroPython’s goal is to make programming digital electronics as simple as possible, so it can be used by anyone. Currently, MicroPython is used by hobbyists, researchers, teachers, educators, and even in commercial products. The code for blinking an LED on a ESP32 or ESP8266 is as simple as follows:

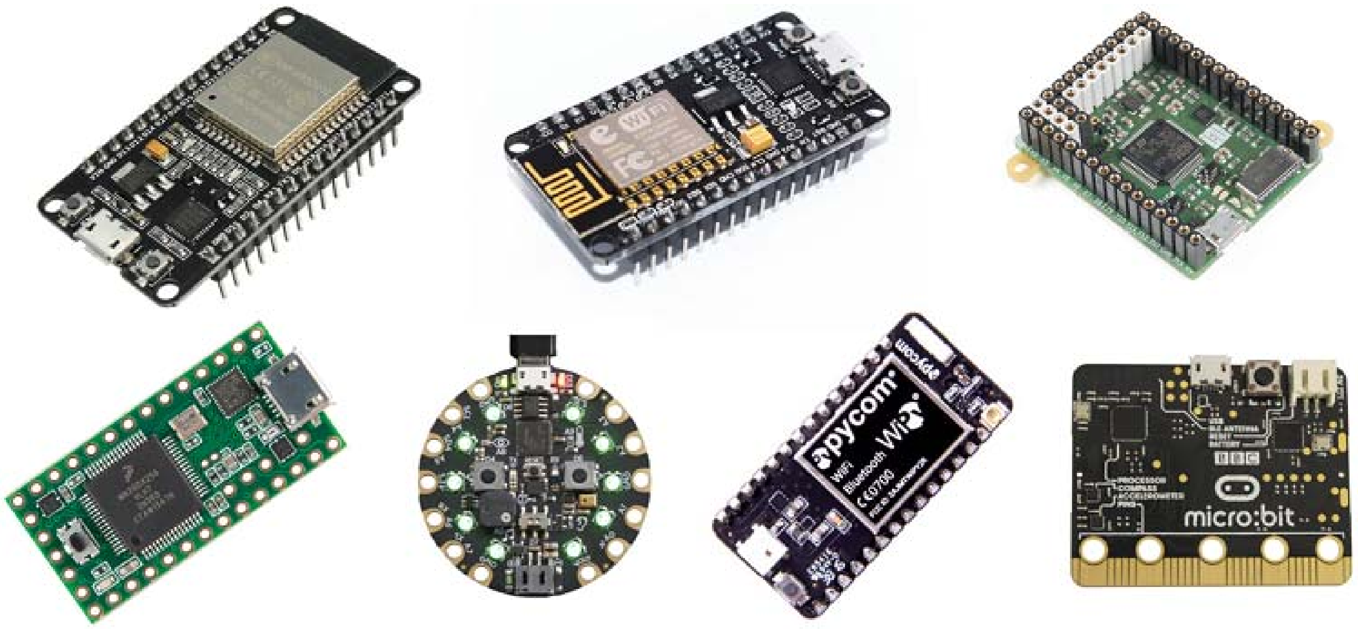
```
# Complete project details at https://RandomNerdTutorials.com

from machine import Pin
from time import sleep

led = Pin(2, Pin.OUT)

while True:
  led.value(not led.value())
  sleep(0.5)
```

View raw code

One great feature of MicroPython is that it comes with an interactive REPL (Read-Evaluate-Print Loop). The REPL allows you to connect to a board and execute code quickly without the need to compile or upload code.

MicroPython – Boards support

MicroPython runs on many different devices and boards, such as:



- ESP32
- ESP8266

- PyBoard
- Micro:Bit
- Teensy 3.X
- WiPy – Pycom
- Adafruit Circuit Playground Express
- Other ESP32/ESP8266 based boards

For more information about other boards that support MicroPython, take a look at the following links:

- Boards running MicroPython – MicroPython Forum
- Boards summary – MicroPython Github

In our projects, we'll use MicroPython with the ESP32 and ESP8266 boards.

ESP32 is the successor of the ESP8266. So, at the moment, not all features are available in MicroPython to take the most out of the ESP32 – it's still an ongoing project. However, it's very usable and you can make a lot of projects with it.

ESP32 and ESP8266 boards are similar, and you won't feel almost any difference programming them using MicroPython. This means that anything you write for the ESP8266 should also run with no changes or minimal changes on the ESP32 (mainly changing the pin assignment).

Installing uPyCraft IDE

Before continuing with this tutorial, you should install uPyCraft IDE in your computer. Follow one of the next tutorials to install uPyCraft IDE:

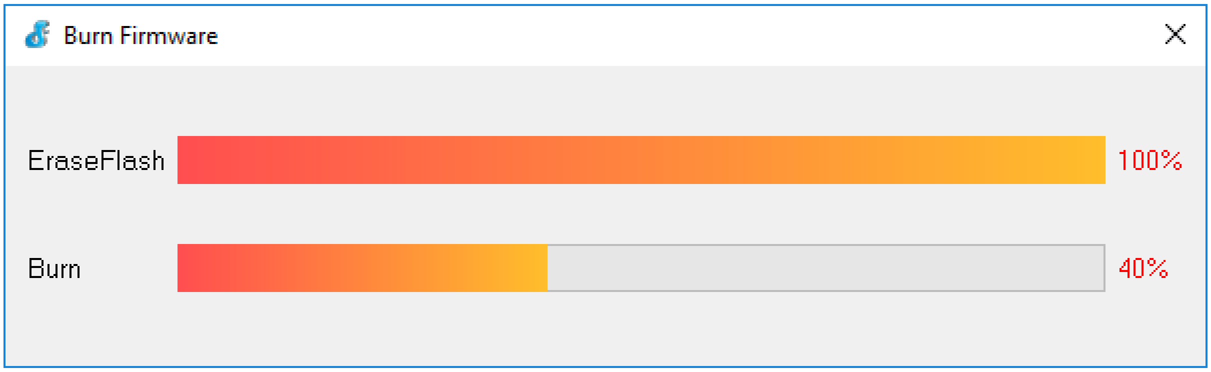
- Install uPyCraft IDE – Windows PC
- Install uPyCraft IDE – Mac OS X
- Install uPyCraft IDE – Linux Ubuntu

Note: if you're having trouble installing or using uPyCraft IDE, we've also created an alternative guide on how to program the ESP32/ESP8266 using Thonny IDE.

Flashing MicroPython Firmware to ESP32/ESP8266

Unlike other boards, MicroPython isn't flashed onto the ESP32 or ESP8266 by default. That's the first thing you need to do to start programming your boards with MicroPython: flash/upload the firmware. Follow the next tutorial to flash MicroPython firmware on your board:

- Flash/Upload MicroPython Firmware to ESP32 and ESP8266



Getting Started with uPyCraft IDE

In this section we'll give you an overview of the uPyCraft IDE software, so that you can start programming the ESP32/ESP8266 with MicroPython.

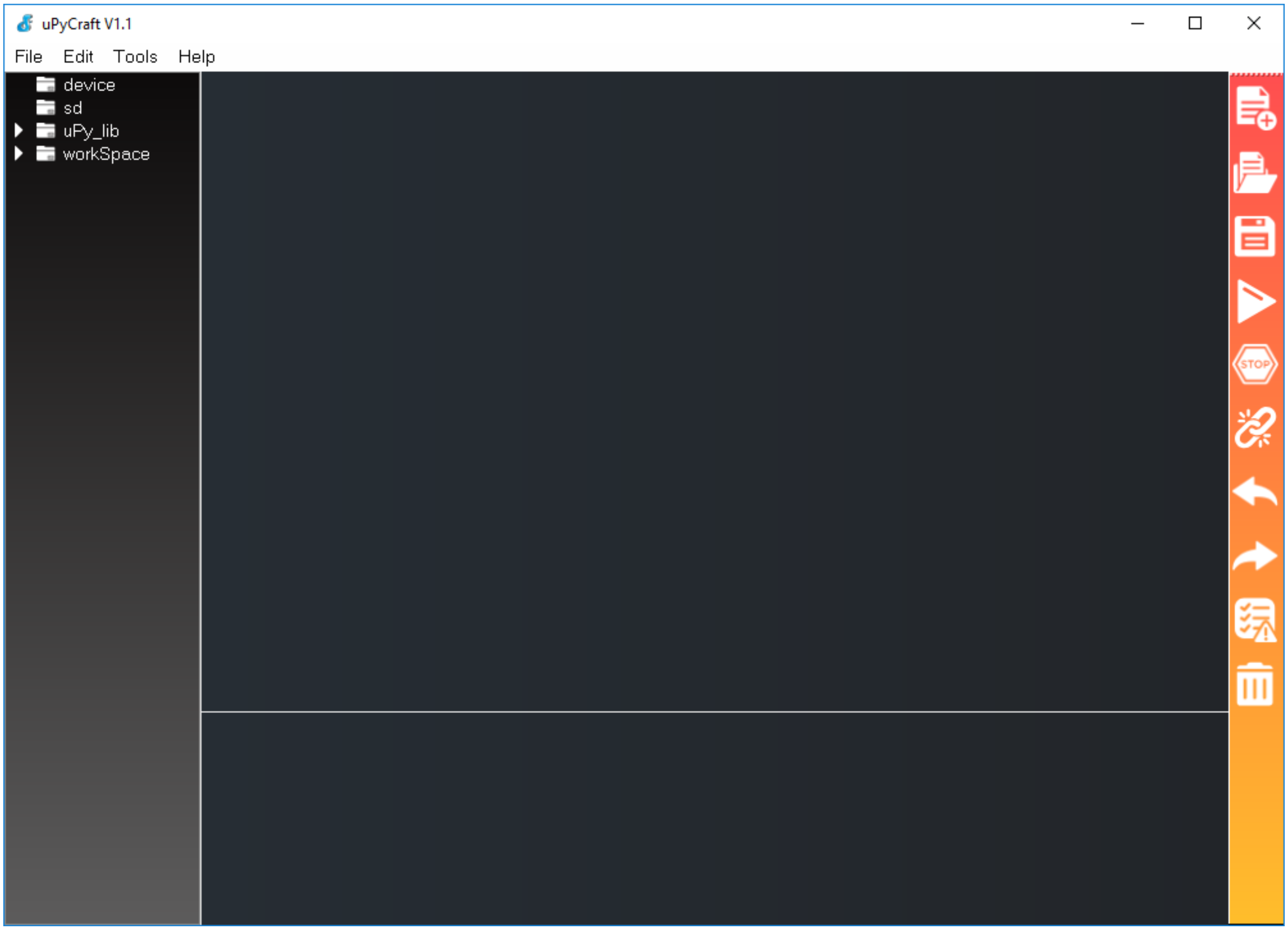
The IDE is a software that contains tools to make the process of development, debugging and upload code easier. There are many ways to program your ESP board with MicroPython. We've chosen uPyCraft IDE because it is simple and intuitive to use and works great with the ESP boards.

At this point, we assumed that you have:

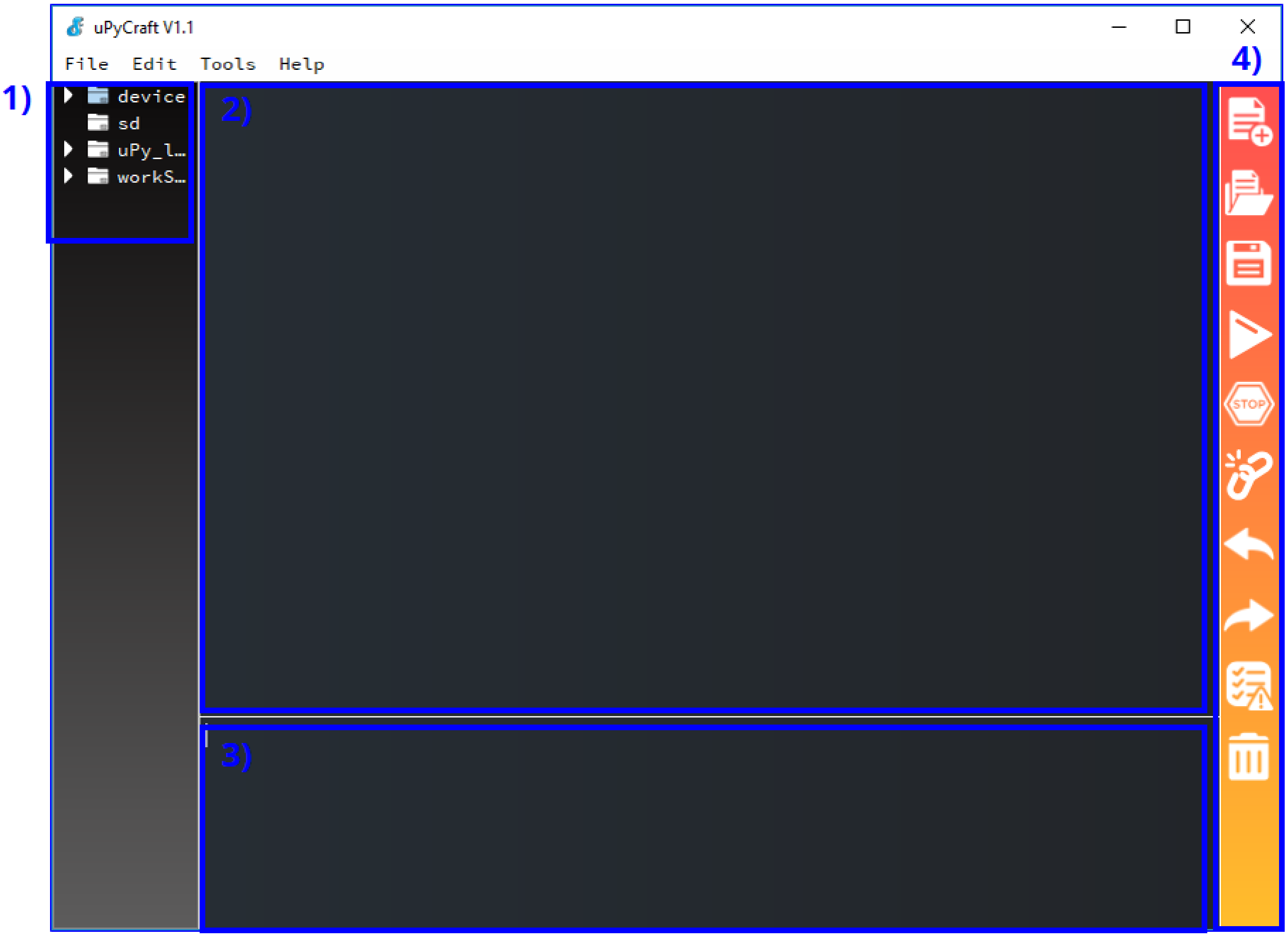
- uPyCraft IDE installed on your computer
- ESP32/ESP8266 flashed with MicroPython firmware

uPyCraft IDE Overview

Open uPyCraft IDE, a new window opens as follows:



Let's take a closer look at each section of uPyCraft IDE:



1. Folder and files
2. Editor
3. MicroPython Shell/Terminal
4. Tools

1. Folder and files

This section shows several folders and files. The **device** folder shows the files that are currently stored on your ESP board. If you have your ESP32 or ESP8266 connected via serial to uPyCraft IDE, when you expand the **device** folder, all files stored should load. By default, you should only have a *boot.py* file. To run your main code, it is recommended to create a *main.py* file.

- **boot.py**: runs when the device starts and sets up several configuration options;
- **main.py**: this is the main script that contains your code. It is executed immediately after the *boot.py*.

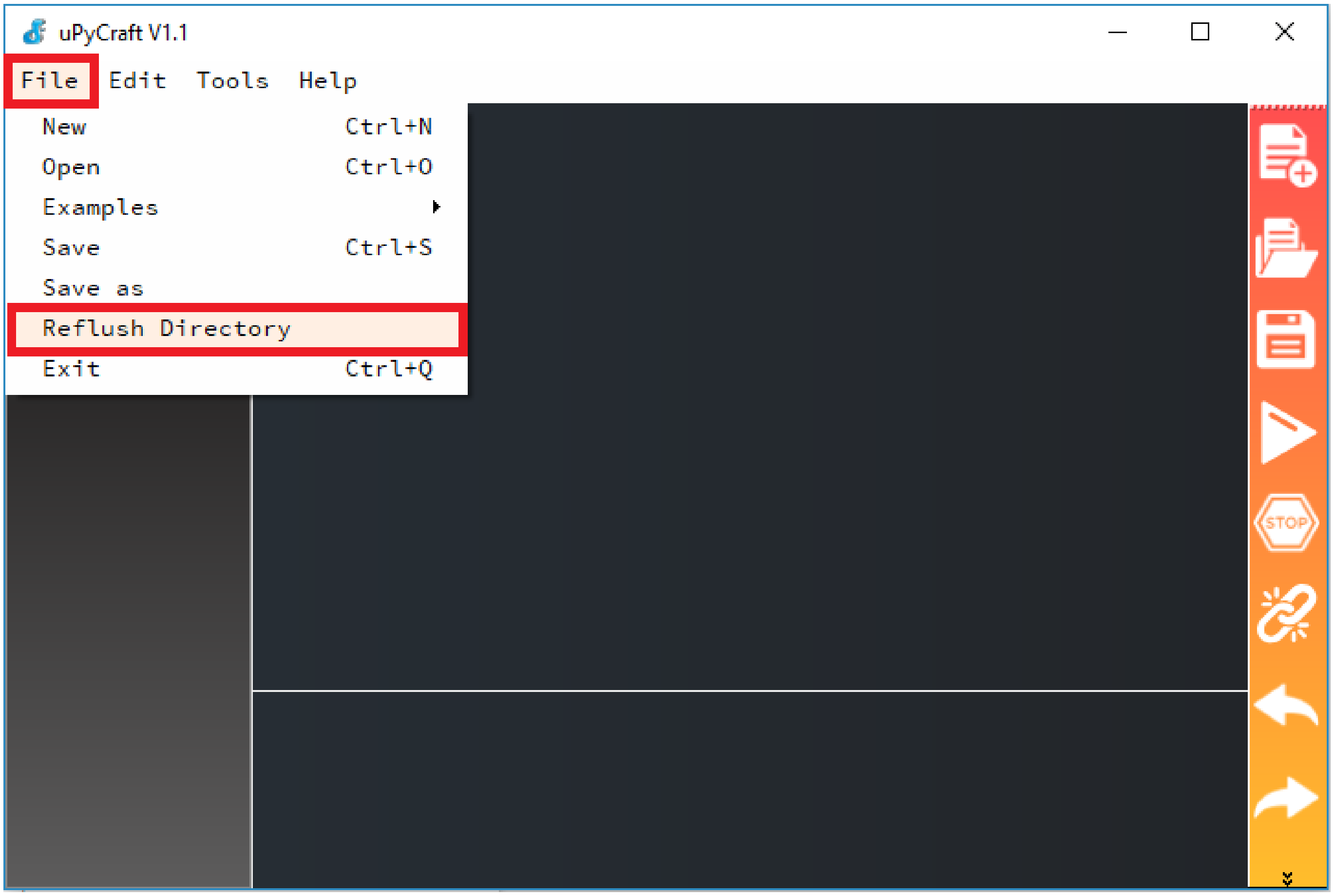
The **sd** folder is meant to access files stored on SD cards – this is only works with boards like the PyBoard that come with an SD card slot.

The **uPy_lib** shows the built-in IDE library files.

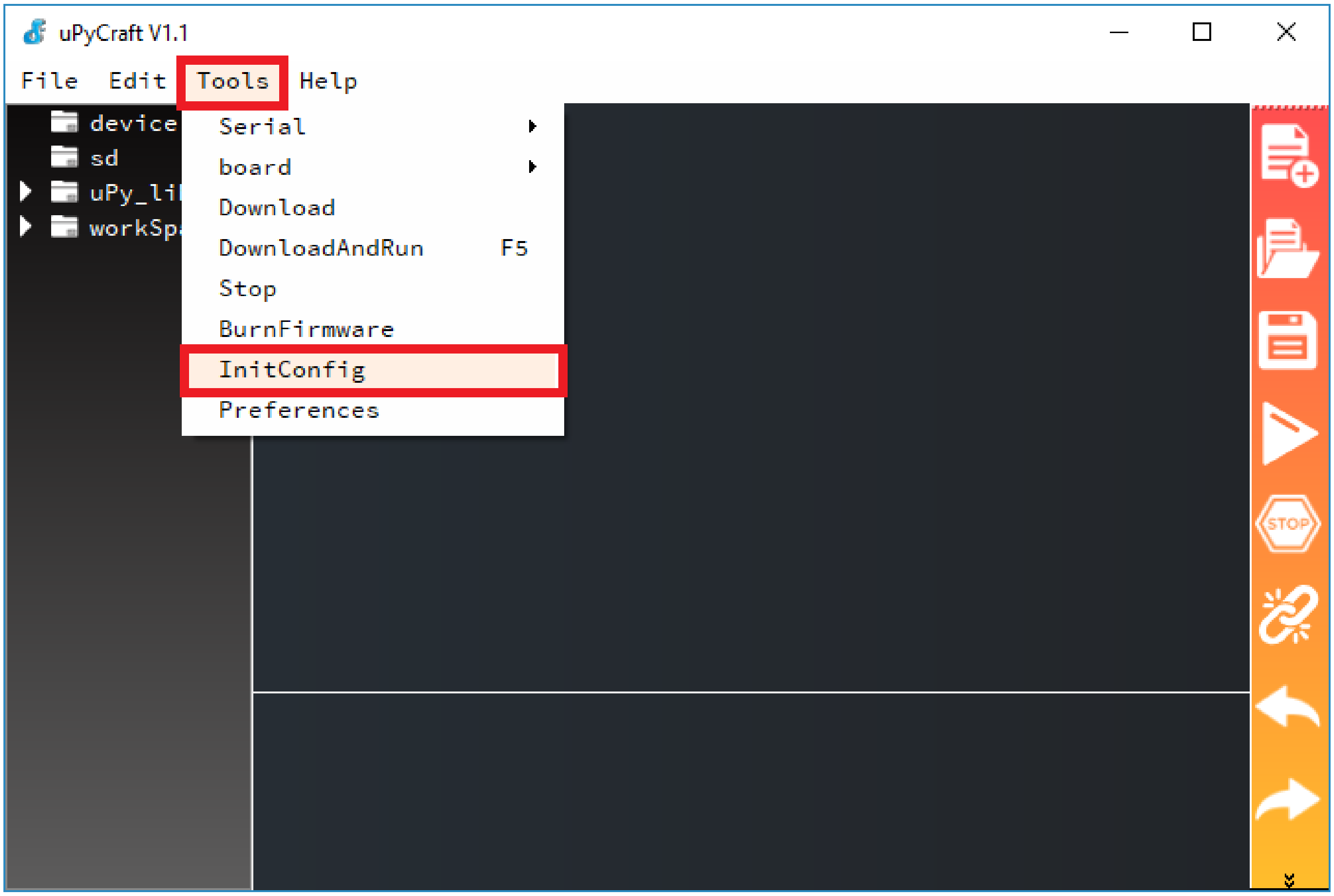
Finally, the **workspace** is a directory to save your files. These files are saved in your computer in a directory defined by you. This is a specially useful to keep all your files organized at hand.

When using uPycraft for the first time, to select your working directory, click the **workspace** folder. A new window pops up for you to chose your **workspace** path. Create a new folder or select an existing folder to be your working directory.

Then, go to **File** > **Reflush Directory** to update the directory.



Note: to change your user directory, simply go to **Tools** >**InitConfig** and click the **workspace** directory folder to chose a different path.



2. Editor

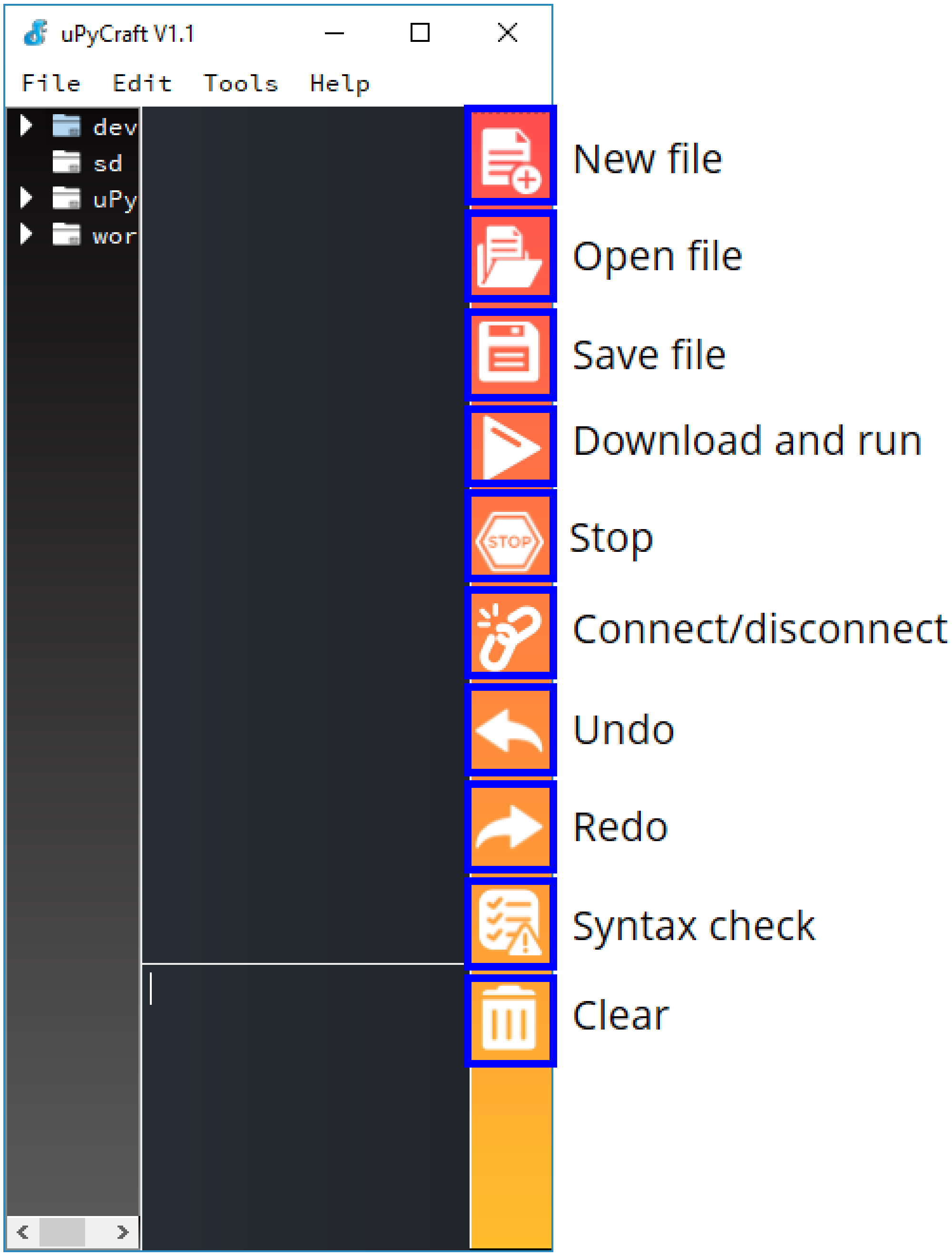
The Editor section is where you write your code and edit your *.py* files. You can open more than one file, and the Editor will open a new tab for each file.

3. MicroPython Shell/terminal

On the MicroPython Shell you can type commands to be executed immediately by your ESP board without the need to upload new files. The terminal also provides information about the state of an executing program, shows errors related with upload, syntax errors, prints messages, etc…

4. Tools

The icons placed at the rightmost side allow you to quickly perform tasks. Each button is labeled in the figure below:



- **New file**: creates a new file on the Editor;
- **Open file**: open a file from your computer;
- **Save file**: saves a file;
- **Download and run**: upload the code to your board and execute the code;
- **Stop**: stop the execution of the code – it's the same as entering CRTL+C on the Shell to stop all scripts from running;
- **Connect/Disconnect**: connect or disconnect to your board via Serial. You must select the serial port first in **Tools** > **Serial**;
- **Undo**: undo last change in the code Editor;
- **Redo**: redo last change in the code Editor;
- **Syntax check**: checks the syntax of your code;

- **Clear**: clear the Shell/terminal window messages.

Running Your First Script

To get you familiar with the process of writing a file and executing code on your ESP32/ESP8266 boards, we'll upload a new script that simply blinks the on-board LED of your ESP32 or ESP8266.

Establishing a communication with the board

After having the MicroPython firmware installed on your board and having the board connected to your computer through an USB cable, follow the next steps:

1. Go to **Tools** > **Board** and select the board you're using.

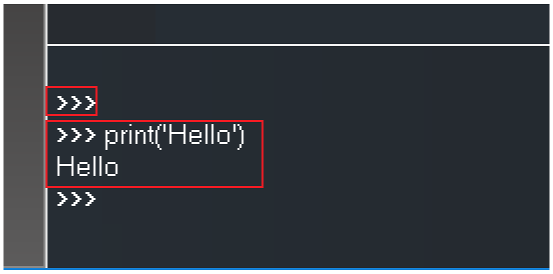
2. Go to **Tools** > **Port** and select the com port your ESP is connected to.

3. Press the **Connect** button to establish a serial communication with your board.

4. The **>>>** should appear in the Shell window after a successful connection with your board. You can type the print command to test if it's working:

```
>>> print('Hello')
Hello
>>>
```

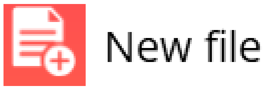
It should print the "Hello" message. Only if you see that message, you can continue with this tutorial. Otherwise, make sure you have established a serial communication with your board or that you've flashed successfully the MicroPython firmware on your board.



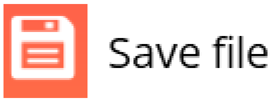
```
>>>
>>> print('Hello')
Hello
>>>
```

Creating the *main.py* file on your board

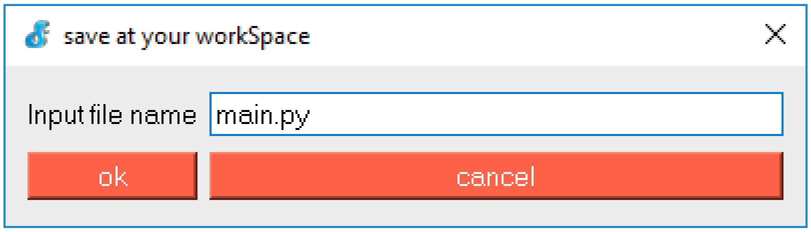
1. Press the “**New file**” button to create a new file.



2. Press the “**Save file**” button to save the file in your computer.



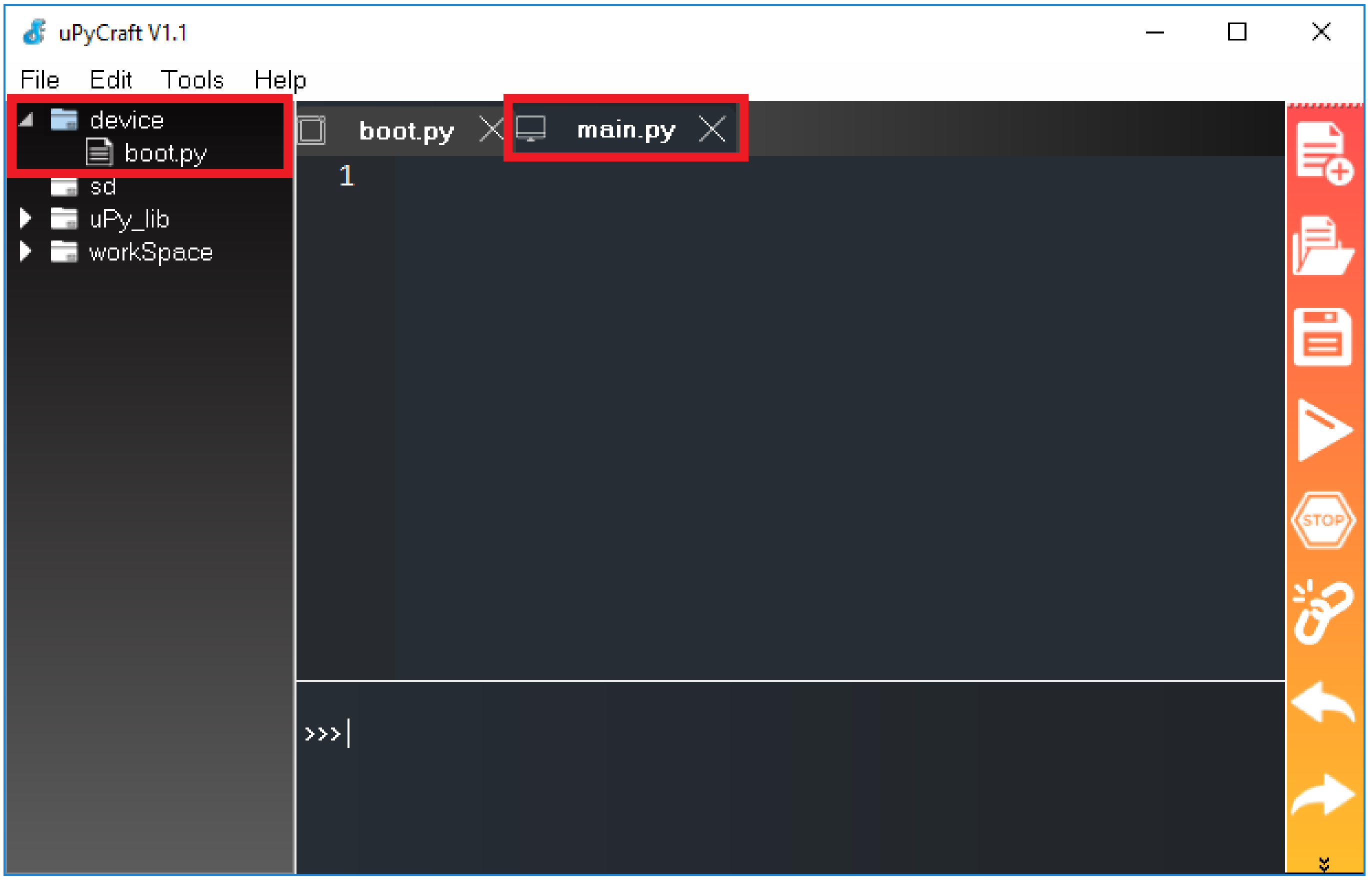
3. A new window opens, name your file *main.py* and save it in your computer:



save at your workSpace

Input file name: main.py

4. After that, you should see the following in your uPyCraft IDE (the *boot.py* file in your device and a new tab with the *main.py* file):



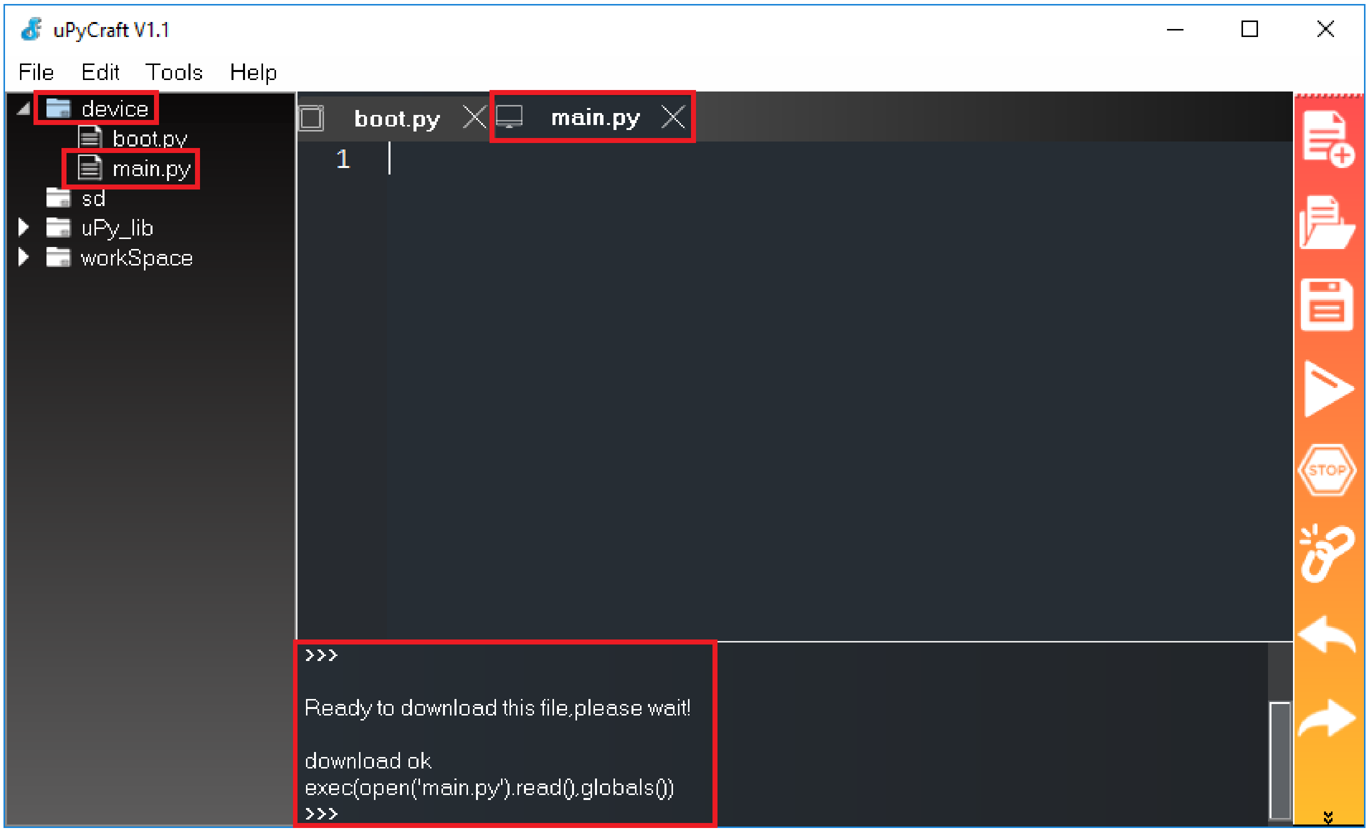
```
1
```

```
>>>
```

5. Click the **“Download and run**” button to upload the file to your ESP board:

Download and run

6. The device directory should now load the *main.py* file. Your ESP has the file *main.py* stored.



```
>>>

Ready to download this file,please wait!

download ok
exec(open('main.py').read(),globals())
>>>
```

Uploading the blink LED script

1. Copy the following code to the Editor on the *main.py* file:

```
# Complete project details at https://RandomNerdTutorials.com

from machine import Pin
from time import sleep

led = Pin(2, Pin.OUT)

while True:
  led.value(not led.value())
  sleep(0.5)
```

View raw code

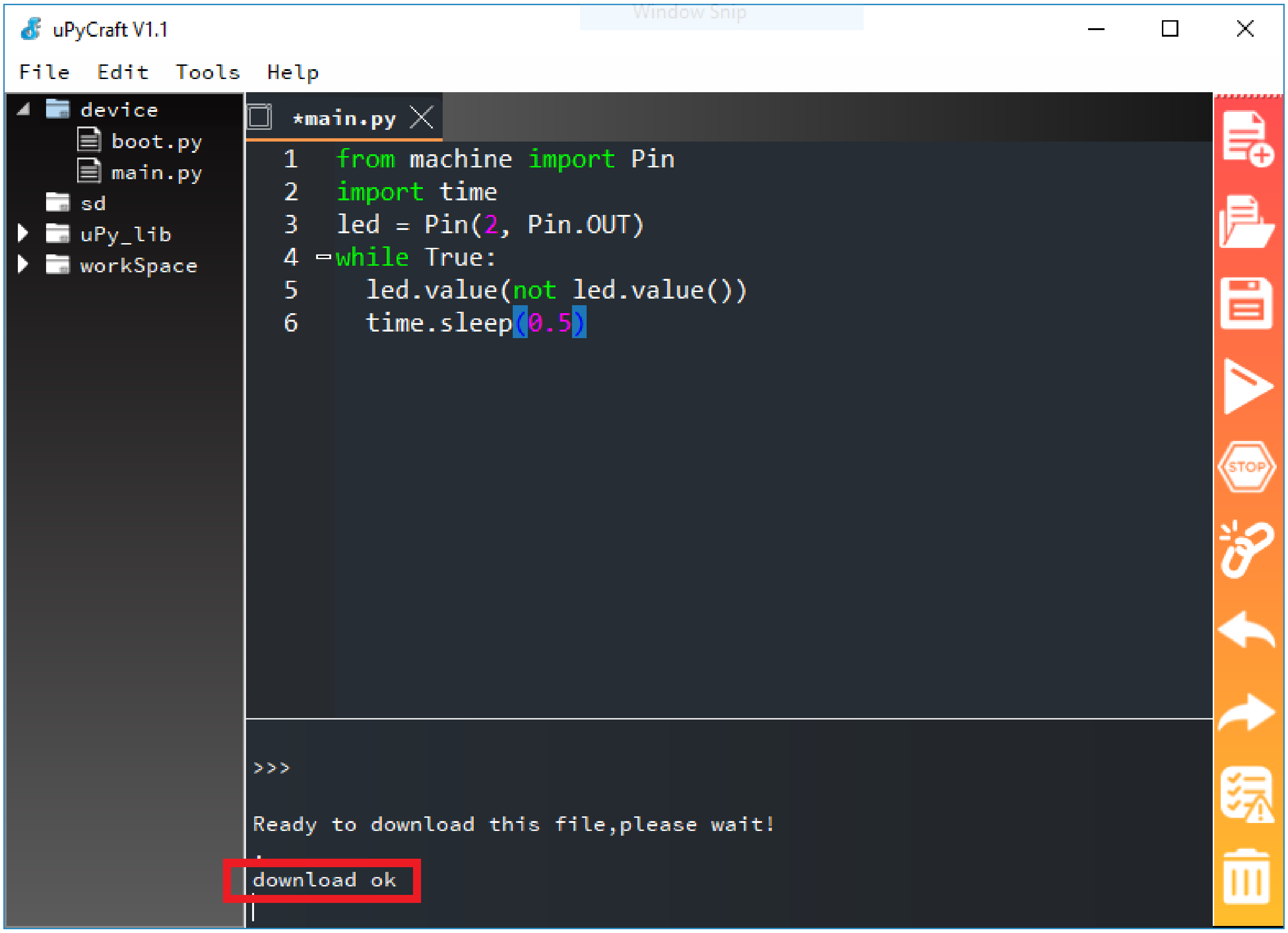
2. Press the "**Stop**" button to stop any script from running in your board:

Stop

3. Click the “**Download and Run button**” to upload the script to the ESP32 or ESP8266:

Download and run

4. You should see a message saying “download ok” in the Shell window.



```
from machine import Pin
import time
led = Pin(2, Pin.OUT)
while True:
  led.value(not led.value())
  time.sleep(0.5)
```

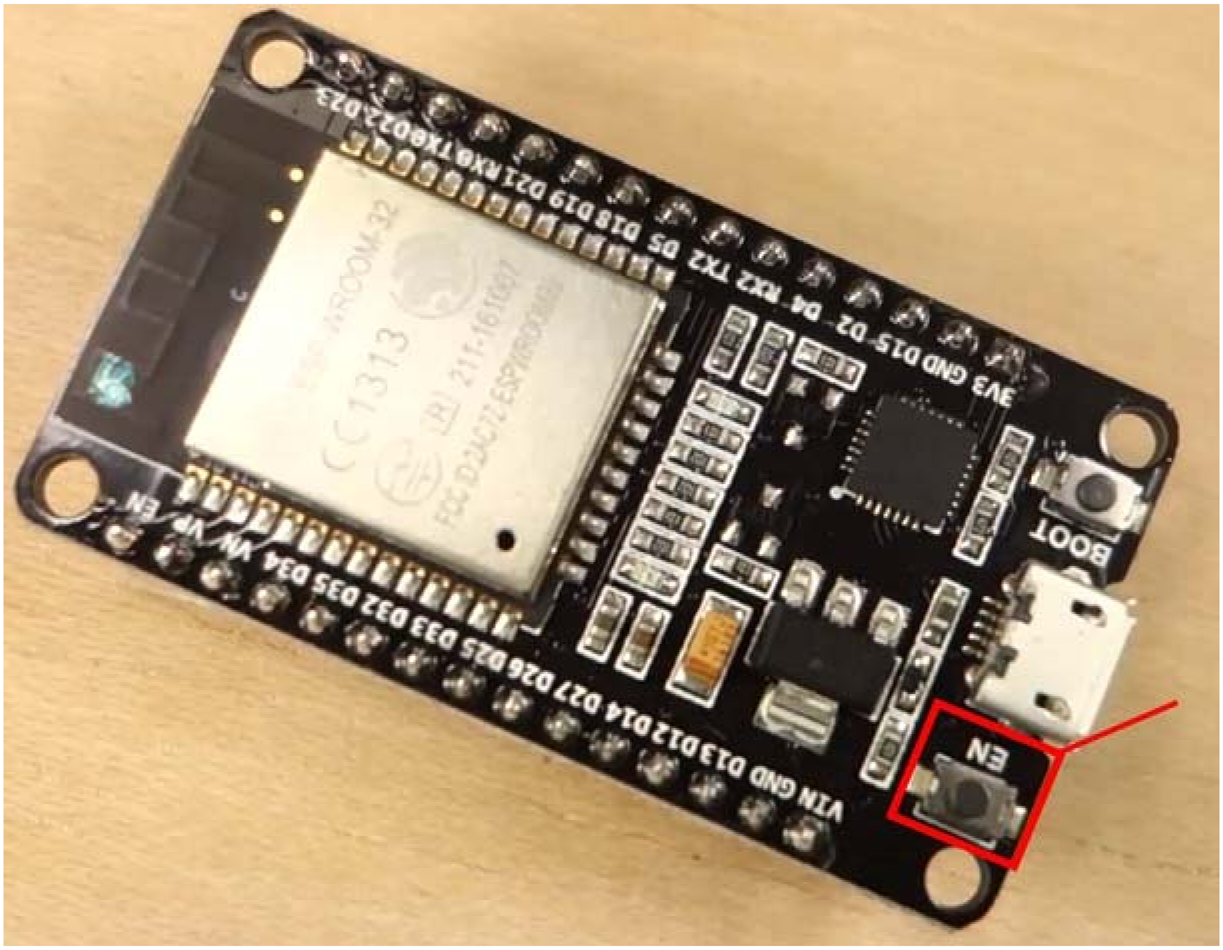
```
>>>

Ready to download this file,please wait!
.
download ok
```

Testing the script

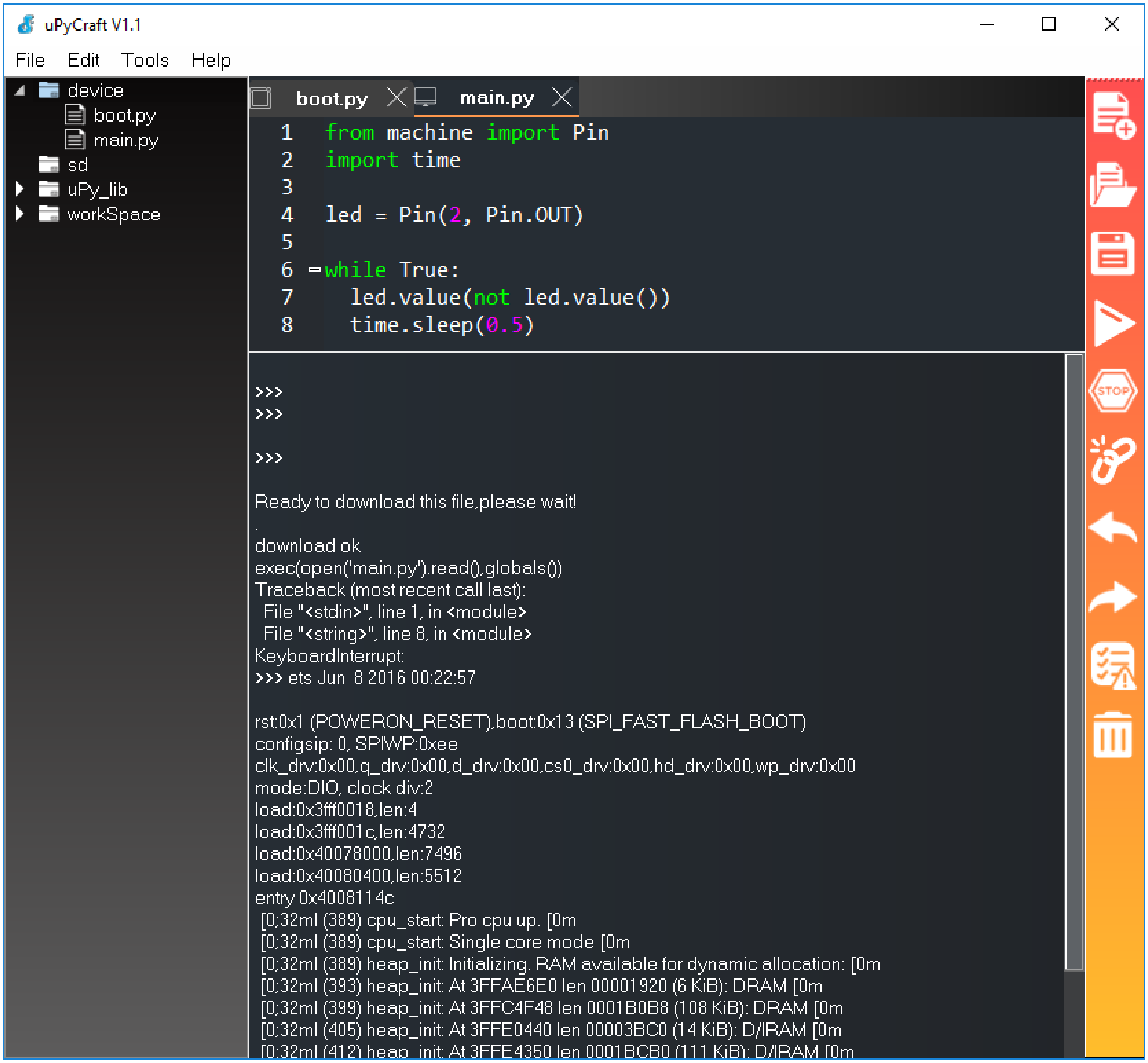
To run the script that was just uploaded to your board, you need to follow these steps:

1. Press the “**Stop**” button

2. Press the on-board ESP32/ESP8266 **EN** (ENABLE) or **RST** (RESET) button to restart your board and run the script from the start:


3V3 GND D15 D2 D4 RX2 TX2 D5 D18 D19 D21 RX0 TX0 D22 D23
EN VP VN D34 D35 D32 D33 D25 D26 D27 D14 D12 D13 GND VIN
ESP-WROOM-32
BOOT
EN

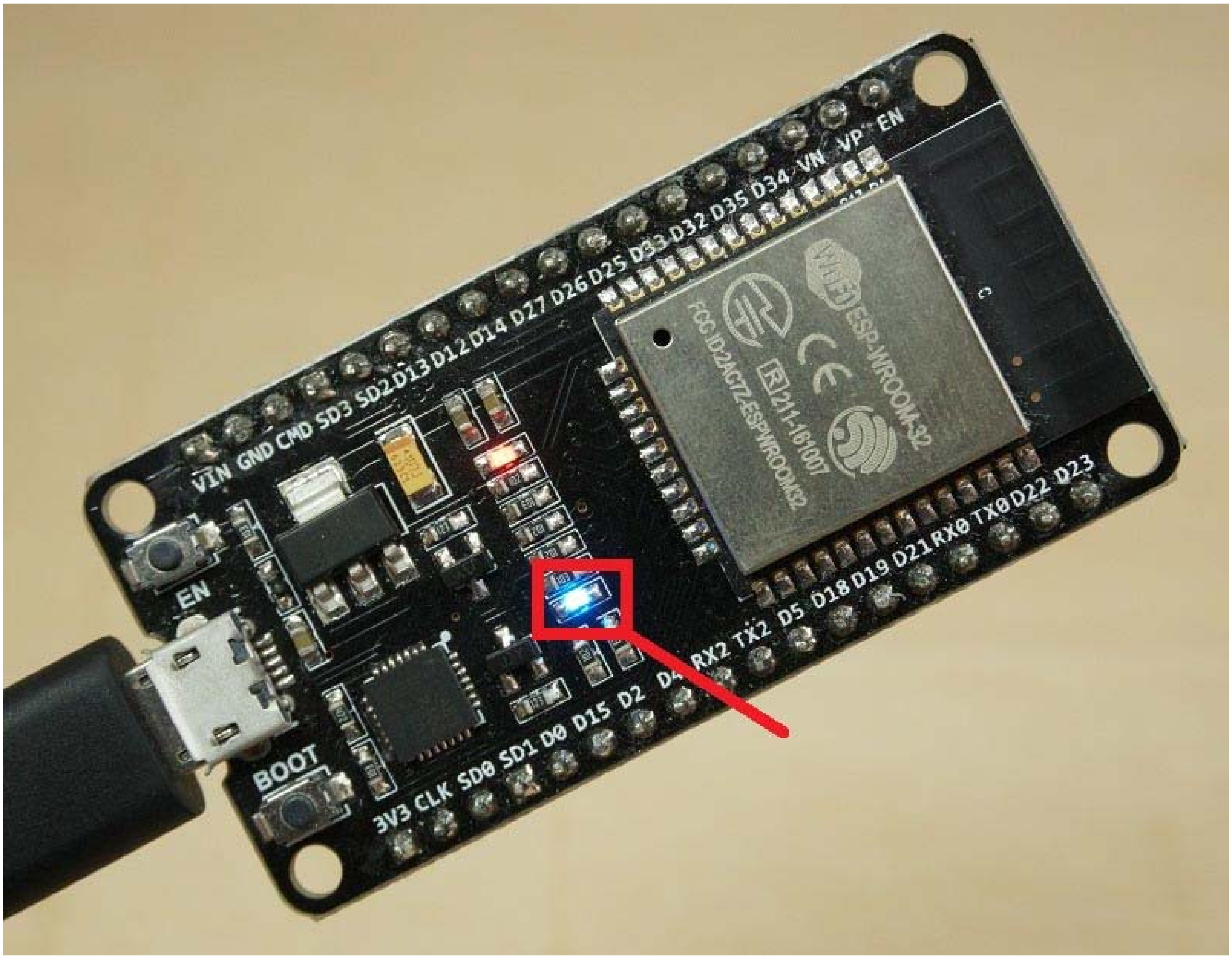
If you're using an ESP32, your Terminal messages should look something as shown in the following figure after a EN/RST button press:



```
from machine import Pin
import time

led = Pin(2, Pin.OUT)

while True:
  led.value(not led.value())
  time.sleep(0.5)
```

```
>>>
>>>

>>>

Ready to download this file,please wait!
.
download ok
exec(open('main.py').read(),globals())
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<string>", line 8, in <module>
KeyboardInterrupt:
>>> ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 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:2
load:0x3fff0018,len:4
load:0x3fff001c,len:4732
load:0x40078000,len:7496
load:0x40080400,len:5512
entry 0x4008114c
 [0;32mI (389) cpu_start: Pro cpu up. [0m
 [0;32mI (389) cpu_start: Single core mode [0m
 [0;32mI (389) heap_init: Initializing. RAM available for dynamic allocation: [0m
 [0;32mI (393) heap_init: At 3FFAE6E0 len 00001920 (6 KiB): DRAM [0m
 [0;32mI (399) heap_init: At 3FFC4F48 len 0001B0B8 (108 KiB): DRAM [0m
 [0;32mI (405) heap_init: At 3FFE0440 len 00003BC0 (14 KiB): D/IRAM [0m
 [0;32mI (412) heap_init: At 3FFE4350 len 0001BCB0 (111 KiB): D/IRAM [0m
```

Your ESP32 or ESP8266 on-board LED should be blinking every 500 milliseconds. Here's where the ESP32's on-board LED is located:


VIN GND CMD SD3 SD2 D13 D12 D14 D27 D26 D25 D33 D32 D35 D34 VN VP EN
EN
BOOT
3V3 CLK SD0 SD1 D0 D15 D2 D4 RX2 TX2 D5 D18 D19 D21 RX0 TX0 D22 D23
ESP-WROOM-32
FCC ID:2AC7Z-ESPWROOM32
R 211-161007

Here’s the ESP8266 on-board LED:

Troubleshooting Tips

We’ve discovered some common problems and error messages that occur with uPyCraft IDE. Usually restarting your ESP with the on-board EN/RST button fixes your problem. Or pressing the uPyCraft IDE “**Stop**” button and repeating your desired action. In case it doesn’t work for you, read these next common errors and discover how to solve them.

Error #1: You get the following message:

```
>>>
Select Serial Port could not open port 'COM4': FileNotFoundError(2,
```

Or an equivalent message:

```
>>>
```

```
could not open port 'COM4': PermissionError(13, 'A device attached
```

Unplug, and plug back your ESP board. Then, double-check that you've selected the right serial port in the **Tools** > **Serial** menu. Then, click the "**Connect/disconnect**" button to establish a serial communication. You should now be able to upload a new script or re-run new code.

This error might also mean that you have your serial port being used in another program (like a serial terminal or in the Arduino IDE). Double-check that you've closed all the programs that might be establishing a serial communication with your ESP board. Then, unplug and plug back your ESP board. Finally, restart the uPyCraft IDE – try to select the serial port in the **Tools** > **Serial** menu.

Error #2: Trouble uploading a new script.

```
>>>
already in download model,please wait.
```

Press the "**Stop**" button in uPyCraft IDE (1 or 2 times) to make sure any code that was running stops. After that, press the "**Download and run**" button to upload the new script to your ESP board.

Error #3: After uploading a new script, if you see the following message:

```
>>>
Ready to download this file,please wait!
...
download ok
os.listdir('.')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
NameError: name 'os' isn't defined
```

Or this message:

```
>>>
Ready to download this file,please wait!
...
download ok
os.listdir('.')
OSError: [Errno 98]
```

It means the new file was uploaded to your board successfully. You can notice that it printed the "**download ok**" message. Press the ESP on-board "**EN/RST**" button to restart your board and re-run the new uploaded script from the beginning.

Error #4: Problem restarting your ESP board, running a new script or opening the serial port:

```
>>>
Brownout detector was triggered
```

The "Brownout detector was triggered" error message means that there's some sort of hardware problem. It's often related to one of the following issues:

- Poor quality USB cable;
- USB cable is too long;
- Board with some defect (bad solder joints);
- Bad computer USB port;
- Or not enough power provided by the computer USB port.

Solution: try a different shorter USB cable (with data wires), try a different computer USB port or use a USB hub with an external power supply.

Important: if you keep having constant problems or weird error messages, we recommend re-flashing your ESP board with the latest version of MicroPython firmware: Flash/Upload MicroPython Firmware to ESP32 and ESP8266.

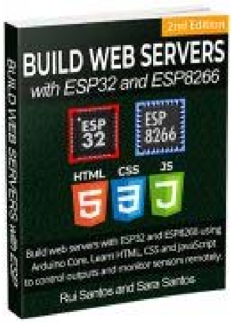
Error #5: When I try to open a serial communication with the ESP32/ESP8266 in uPyCraft IDE, sometimes it prompts the "Burn Firmware" window asking to re-flash the MicroPython firmware.

Basically, we think this is what's happening: when you're running a script in your board, sometimes it's busy running that script and performing the tasks. So, you need to try opening the COM port multiple times or restart the ESP to catch it available to establish the serial communication with uPyCraft IDE.

If you're running a script that uses Wi-Fi, deep sleep, or it's doing multiple tasks, I recommend trying 3 or 4 times to establish the communication. If you can't, I recommend re-flash the ESP with MicroPython firmware.

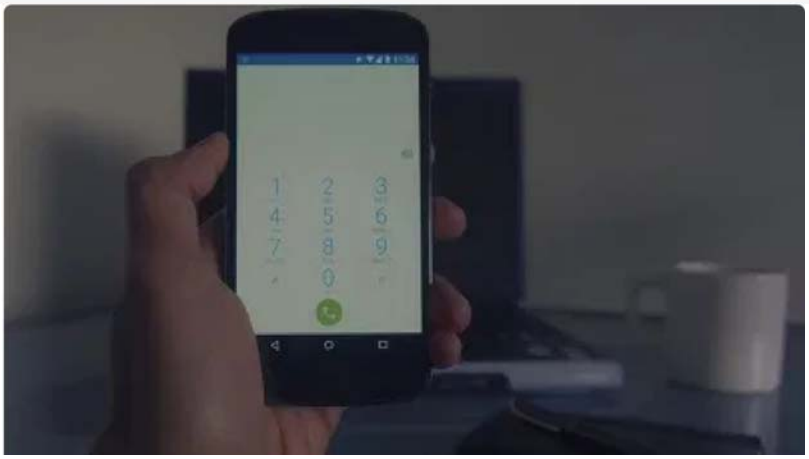
Wrapping Up

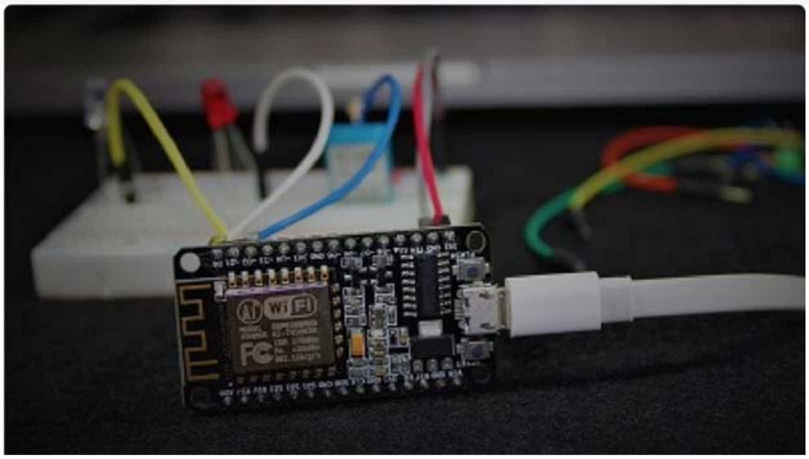
We hope you've enjoyed learning how to program the ESP32 and ESP8266 boards using MicroPython firmware. If you want to learn more about MicroPython, take a look at our eBook: MicroPython Programming with ESP32 and ESP8266.

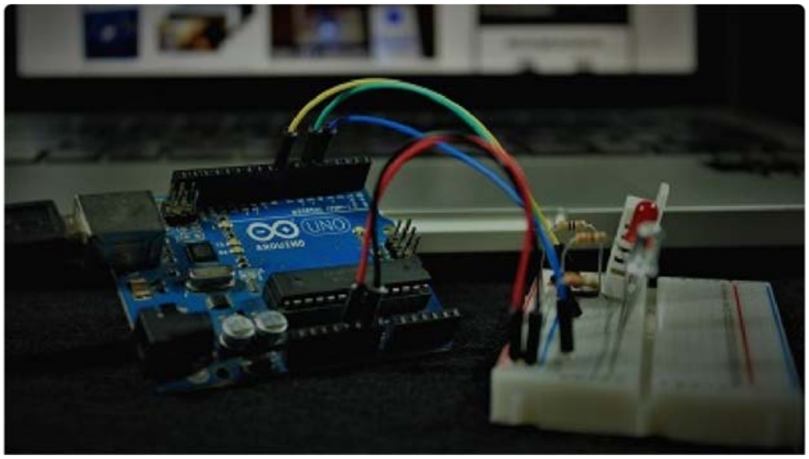


[eBook] Build Web Servers with ESP32 and ESP8266 (2nd Edition)

Recommended Resources







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

What to Read Next…

Guide for TCA9548A I2C Multiplexer: ESP32, ESP8266, Arduino

Getting Started with Thonny MicroPython (Python) IDE for ESP32 and ESP8266

ESP-NOW with ESP8266: Send Data to Multiple Boards (one-to-many)

Enjoyed this project? Stay updated by subscribing our newsletter!

Your Email Address

✉ SUBSCRIBE

34 thoughts on “Getting Started with MicroPython on ESP32 and ESP8266”

Graeme Moffat

October 11, 2018 at 8:18 am

Does the firmware have to be re-flashed if you decide to go back to using the Arduino IDE?

Reply

Sara Santos

October 13, 2018 at 9:54 am

Hi.
To get back using Arduino IDE, so simply need to upload any code using Arduino IDE.
Regards,
Sara 🙂

Reply

Umar Muhammad

October 12, 2018 at 9:17 pm

Thank you so much for this tutorial. After flashing my esp with micropython, is it possible for me to program it with arduino ide again?

Reply

Sara Santos
October 13, 2018 at 9:46 am

Hi.
Yes. You just need to upload code using Arduino IDE and it is ready again to use Arduino IDE.
Regards,
Sara 🙂

Reply

Sastry Dasigi
October 16, 2018 at 3:04 pm

Amazing tutorial! Great stuff. Thanks Rui!

Reply

Sara Santos
October 16, 2018 at 3:49 pm

Thank you!
Regards,
Sara 🙂

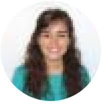
Reply

Paul Gregg

November 2, 2018 at 6:27 pm

I thought I had followed a tutorial here using micro python on ESP32 that set a PWM output on the ESP32. I'm trying to find that tutorial but not having much luck.
Thanks
Paul

Reply



Sara Santos

November 4, 2018 at 9:54 am

Hi Paul.
Here's the tutorial you are looking for:
https://randomnerdtutorials.com/esp32-pwm-arduino-ide/
Regards,
Sara 🙂

Reply

Jayakumar

November 12, 2018 at 8:01 am

My blink LED script doesn't work. I have TTGO MINI 32 V2.0 ESP32 board.
Not sure if the pin 2 is the right LED pin for my board.
Pin layout information:
https://raw.githubusercontent.com/LilyGO/ESP32-MINI-32-V2.0/master/ZZ_Images/image1.jpg

Reply

Sara Santos

November 12, 2018 at 10:40 am

Hi Jayakumar.
I've never used that board. But I was able to find the schematic diagram.
For what I can tell, it seems that the built-in LED for your board is GPIO22.
so, you just need to change the code to assign GPIO22 instead of 2.
Can you try it and see if it solves your problem?
Regards,
Sara 🙂

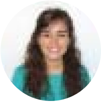
Reply

Bart Valom

February 15, 2019 at 9:19 pm

Thanks for the clear tutorial, my ESP32 is blinking nicely.
I am interested in the course/book, what is it with ‘currently closed for the general public’?
Can you offer some suggestions how to proceed learning micropython?
Best regards,
Bart

Reply



Sara Santos

February 16, 2019 at 5:22 pm

Hi Bart.
At the moment, the MicroPython eBook is only available for those of have already bought one of our courses before (they have access to RNTLAB dashboard).
If you’ve purchased one of our courses before, you can get the MicroPython eBook here: https://rntlab.com/product/micropython-programming-with-esp32-and-esp8266/
Meanwhile, you can read our free resources:
https://randomnerdtutorials.com/?s=micropython
Let me know if you need anything else.
Regards,
Sara

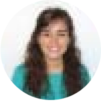
Reply

Miha Horvat

April 3, 2019 at 5:26 pm

hi,
when I want to save my main.py file It doesn't do anything. The "saved" file is not there and It doesn't show me any errors either.

Reply



Sara Santos
April 16, 2019 at 3:11 pm

Hi.
You need to save the file first, before you are able to upload the file to your board.
Fist, save the file by clicking the "Save" icon. Call the file "main.py". Then, click the upload button to upload the code.
If you're having a lot of trouble using uPyCraft IDE, I recommend experimenting with Thonny IDE: https://randomnerdtutorials.com/getting-started-thonny-micropython-python-ide-esp32-esp8266/
I think it is more intuitive to use and less prone to these kind of errors.
Regards,
Sara

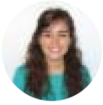
Reply

Didier Rigault
May 24, 2019 at 7:38 am

How to avoid some trouble shooting
– After a first execution (essential) go to your user account \ AppData \ Local \

uPyCraft. In the config.json file replace git.oschina.net by github.com. You will no longer have an alert for a trojan.
– There is a bug concerning SourceCodePro font whose name is incomplete which explains this systematic installation request. It should be SourceCodePro-black.ttf for example and not SourceCodePro.ttf (W10). To be done in a new version.
– I had for several days this error consisting of rebuilding a burnfirmware before connecting to an ESP. This anomaly has disappeared without explanation !!
I am working on the source code to go from Qt4 / PyQt4 to Qt5 / PyQt5
Thank you Sara for your excellent site
Didier

Reply



Sara Santos

May 27, 2019 at 5:58 pm

Hi Didier.
Thank you for sharing your knowledge about that subject.
It will certainly be useful to resolve those annoying bugs in uPyCraft.
Thank you.
Regards.
Sara

Reply

Gerard Virgona

August 17, 2019 at 9:02 pm

Well most people did better than I, I couldn’t even get uPyCraftIDE to load. My Del windows computer kept telling me it couldn’t install as file’s were missing. It suggested to download again & retry. I did this 9 times & each time was told file’s were missing & it wouldn’t install.

Reply

Sara Santos

August 24, 2019 at 10:29 am

Hi Gerard.
I’m sorry you’re having issues with uPyCraft IDE.
Can you try Thonny IDE instead? https://randomnerdtutorials.com/getting-started-thonny-micropython-python-ide-esp32-esp8266/
Usually, our readers don’t have issues with Thonny IDE.
Regards,
Sara

Reply

Kittali Jagadeesh

February 13, 2020 at 9:12 am

Hi
How to control servo motors using micro python and PWM pins

Reply

Sara Santos

February 13, 2020 at 3:23 pm

Hi.
We don’t have any tutorial using servo motor with Micropython.
But we have this PWM tutorial that might help:
https://randomnerdtutorials.com/esp32-esp8266-pwm-micropython/
Regards,
Sara

Reply

Muhammed khalifa

March 30, 2020 at 9:40 am

Hi Sara Santos

Thanks for this lovely tutorial and i have one question that after flasing arduino code to esp32 the micropython firmware will present or we have to reinstall back.

Reply

Sara Santos

March 31, 2020 at 10:33 am

Hi.
If you upload arduino code, you’ll need to install MicroPython again.
Regards,
Sara

Reply

Leandro

April 15, 2020 at 3:33 am

Thanks for the usefull tutorial!!! Great help
I don´t see why, after disconecting and connecting again the ESP32, it doesn’t start blinking, even if I push the EN button.
In fact, the only way to get it blinking, is to upload the program again… As if it wasn’t flash

Reply

Sara Santos

April 15, 2020 at 9:47 am

Hi.
You need to make sure that your script was uploaded to the board with the name main.py otherwise it won’t run.
Regards,
Sara

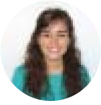
Reply

noaddress

January 27, 2021 at 2:05 pm

Is there a similar program as uPyCraft IDE, but can also connect using wifi?

Reply



Sara Santos

January 27, 2021 at 3:01 pm

Hi.
You can use MicroPython WebREPL: https://micropython.org/webrepl/
Regards,
Sara

Reply

paulo g.

June 2, 2021 at 7:50 am

You can use Thonny. Thonny is much easier now. just a tip:
from machine import Pin
import time

```
led = Pin(16, Pin.OUT)
for i in range(10):
led.on()
time.sleep_ms(500)
led.off()
time.sleep_ms(500)
```

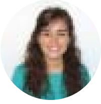
Reply

julio

April 16, 2021 at 7:47 pm

Hello, I'd like to know if wemos d1 mini can be flashed with micropython too.
Thanks in advace
Julio

Reply



Sara Santos

April 16, 2021 at 9:26 pm

Hi.
Yes!
Regards,
Sara

Reply

Alejhandro

August 18, 2021 at 2:16 pm

I can run my program only when I have the ESP32 connect to uPyCraft, if I disconnect and push the EN button to test, it doesn't run!
I want to run my program without be connect to my PC.

Reply

Sara Santos

August 18, 2021 at 2:36 pm

Hi.
Make sure the code is being uploaded to the board with the right name.
Either main.py or boot.py.
Regards,
Sara

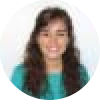
Reply

Dinesh Vyas

September 16, 2021 at 5:17 pm

So after millions of pages you still dont explain how to flush firmware. Why?

Reply



Sara Santos

September 17, 2021 at 9:24 am

Hi.
What do you mean?
Here are the tutorials to flash the firmware:
– https://randomnerdtutorials.com/flash-upload-micropython-firmware-esp32-esp8266/
– https://randomnerdtutorials.com/flashing-micropython-firmware-esptool-py-esp32-esp8266/
Regards,
Sara

Reply

perlchamp

May 2, 2022 at 7:46 pm

comment from may, 2019 :

> > \>
> > *How to avoid some trouble shooting*
> > *– After a first execution (essential) go to your user account \ AppData \ Local \ uPyCraft. In the config.json file replace git.oschina.net by github.com. You will no longer have an alert for a trojan.*
> > *– There is a bug concerning SourceCodePro font whose name is incomplete which explains this systematic installation request. It should be SourceCodePro-black.ttf for example and not SourceCodePro.ttf (W10). To be done in a new version.*

> > *– I had for several days this error consisting of rebuilding a burnfirmware before connecting to an ESP. This anomaly has disappeared without explanation !!*
> > *I am working on the source code to go from Qt4 / PyQt4 to Qt5 / PyQt5*
> > *Thank you Sara for your excellent site*
> > *Didier*
> > <<<

the same problems i have. now we have 2022. THREE years ago, nothing happened. it's a shame, isn't it ?
instead of Pin-Number use LED_BUILTIN .
it's the const implemented by all boards.

for beginners thonny is not the wrong way. then u can use PyCharm with its modules and add-ons/plugins or Atom. uPyCraft is always the wrong way, shure for that !
good luck …
keep corona away !

so long
pc

Reply

Leave a Comment

Name *

Email *

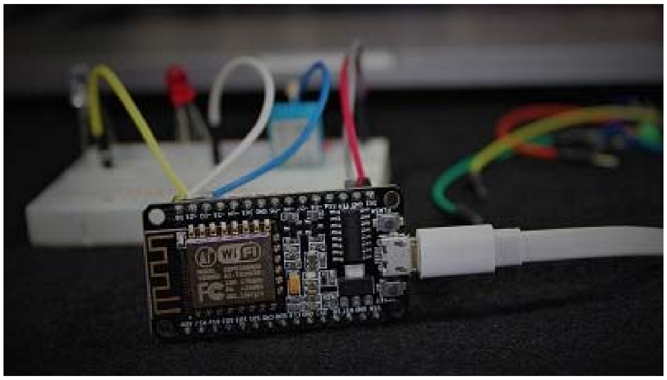
Website

□ Notify me of follow-up comments by email.

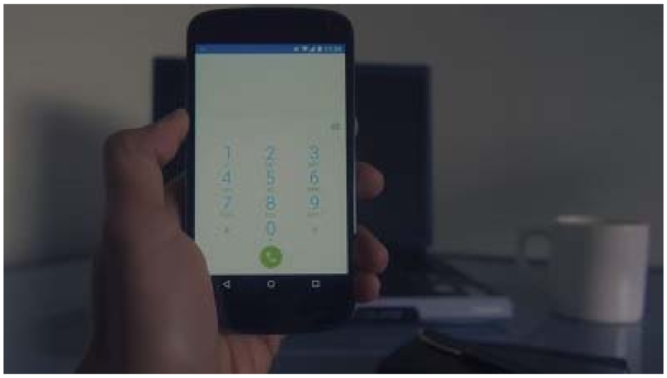
□ Notify me of new posts by email.

Post Comment

Visit Maker Advisor – Tools and Gear for makers, hobbyists and DIYers »



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



Build Web Servers with ESP32 and ESP8266 » boards to control outputs and monitor sensors remotely.

About Support Terms and Conditions Privacy Policy Refunds Complaints' Book

MakerAdvisor.com Join the Lab

Update Privacy Settings