# SciOx Project Notes

## Setting up the ESP32

### 1 Connect the ESP32 to your computer

This is physically simple – use a USB-USB micro data cable. The only potential problem is having the right USB drivers. Generally they are ok, but some brands of USB UART (the serial-USB converter on the ESP board), notably those from WCH, require updates – Windows is a bit rubbish at this. Check out <https://www.wch-ic.com/downloads/CH341SER_ZIP.html>, which is Windows-specific ad might not be right for you situation or actual UART, but has links to all the other WCH types and OS versions (Linux, MacOS).

### 2 Start Thonny, point it at the ESP32, and install Micropython on it

1. Click on “Thonny” in the menu bar
2. Select “Preferences…”
3. Click on the “Interpreter” tab
4. You’ll see a dropdown “Which kind of interpreter…”; click on it, and select “Micropython ESP32”. This tells Thonny that you want to use Micropython running on an ESP32 instead of Python running on your computer
5. Below that you will see Port or WebREPL – drop the list and select the correct port – we’ll tell you which one to use
6. Click on “Install or update Micropython (esptool)”
7. Wait until it’s done, then click “OK” to close the config window
8. The ESP32 will restart and you should see a prompt in the lower Shell window, something like

MicroPython v1.23.0 on 2024-06-02; Generic ESP32 module with ESP32

Type "help()" for more information.

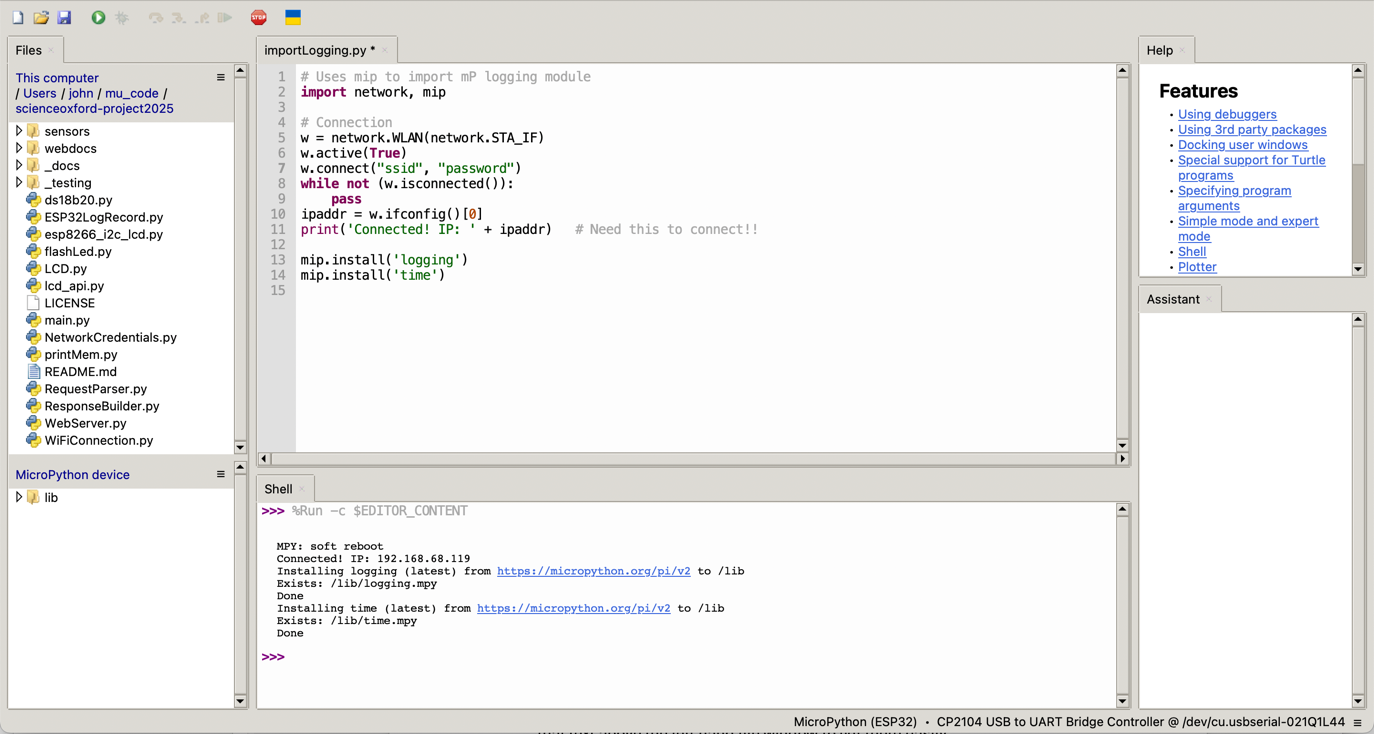
>>>

1. You’re done with this step – try typing print(“hello!”) at the >>> prompt

### 3 Loading the libraries onto the ESP32

The ESP32 is a full-blown “proper” computer, with storage and files etc. on it. To make the code work, it needs all the Python code that you want to use on it, or the Python interpreter will not be able to run it. This is a Bad Thing…

1. You will need the logging and time modules installed on the ESP32, which are part of MicroPython but have to be loaded separately. In the Thonny file window on the left, you’ll see the project files and directories. Double-click on the \_testing folder to open it, you’ll see the file importLogging.py. Double-click on that, which will open it in your Thonny editor window. Click on the green “Run” button towards the left of the menu bar at the top – this should run the script on the ESP32, which will connect to wifi, import the right libraries, and output a message in the bottom window to say that was successful. The illustration below shows what this all looks like, except my ESP32 already had them, so mip says that. You can also see the lib folder on the ESP32, which is where logging and time were saved.



1. Now you need to load all the modules that are part of this project, which are in the folder above - scienceoxford-project2025. Just click on the link with that text above the left hand file window to get there easily. Select everything in the computer file windows **except**

\_docs

\_testing

LICENSE

README.md

Now right-click on the selected files, and select “Upload to /” from the pop-up menu. Thonny should say it’s uploading 22 things, and you’ll see a progress bar. When it’s done it should show all the new files/directories on the ESP32, like this:

A screenshot of a computer

Description automatically generated

1. Now restart your ESP32 by using the reset button on the ESP32 itself. It should reboot and start running the main.py script, and display some information in the Thonny Shell window, like this:

A screenshot of a computer program

Description automatically generated

You can see messages from the ESP32 microcode, and then timed messages from our Python code – the date & time are not correct until the ESP32 connects to the network and sets them from the NTP servers on the Internet. If none of this happened, then let someone know.

The LCD should also be displaying information, like this (depending on what sensors are available):

A blue digital display with white numbers

Description automatically generated