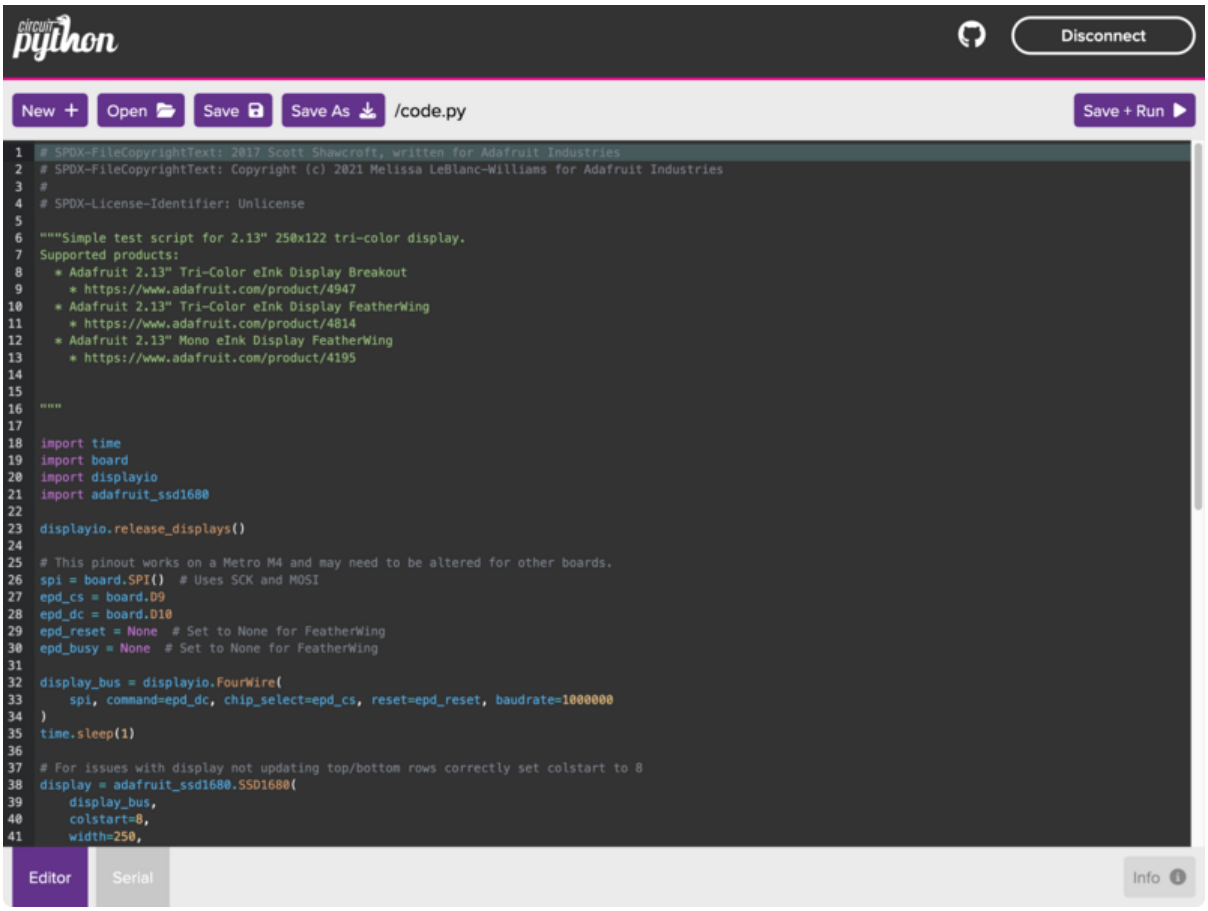
CircuitPython USB Workflow Code Editor Quick Start

Created by Melissa LeBlanc-Williams

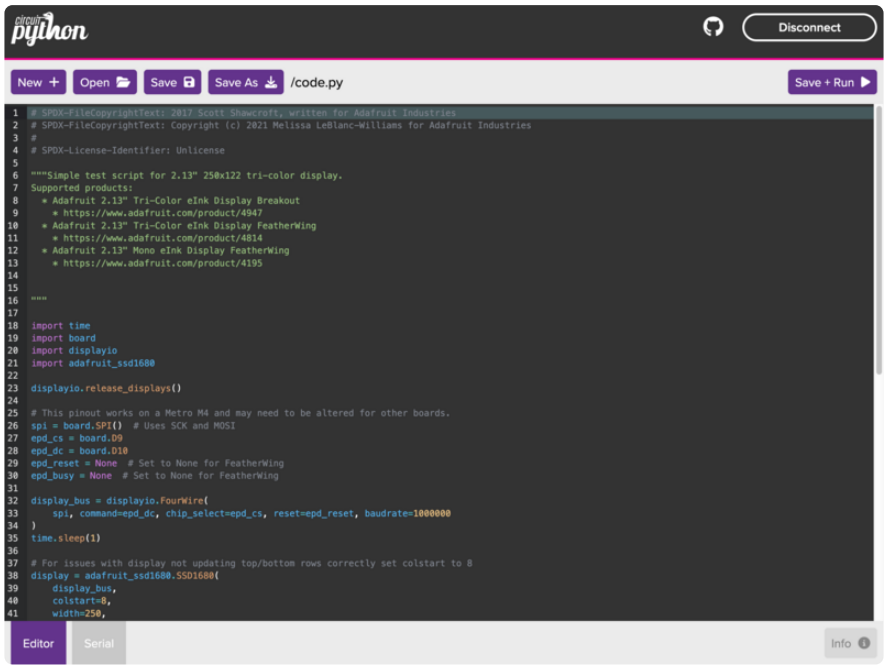


https://learn.adafruit.com/circuitpython-usb-workflow-code-editor-quick-start

Last updated on 2024-08-14 04:15:31 PM EDT

Table of Contents

Overview



The CircuitPython Code Editor (https://adafru.it/10QF)'s USB mode is the most recent mode to be added. It was originally written to use Chrome's File System API to allow direct manipulation of files on a microcontrollers USB CIRCUITPY drive. This worked well for the most part, but there were boards that didn't have native USB ports and thus didn't have a CIRCUITPY drive, such as the Adafruit HUZZAH32 (http://adafru.it/3591), an ESP32-based development board.

Taking a cue from other code editors such as Thonny that allow file manipulation through the REPL, the Code Editor was recently expanded to include similar functionality. For boards that have a CIRCUITPY drive, since the file system is mounted as read-only, the File System API is still used. When the board connects, the board is queried about whether it is mounted as read-only and the appropriate connection method is automatically selected.

For interacting with the REPL, the Code Editor uses the CircuitPython REPL JS library (https://adafru.it/1a5V). This was originally written to interact with the REPL in the same way a user would use it, but was updated to use a special REPL mode called Raw Paste Mode. This allows less chatter and better control of code output and error messages and was intended specifically for code editors to interact.

The REPL library was also given a class of functions that allow File Operations to be performed and is where much of the power of the USB mode update was added.

Device Setup

Make sure you are running version **8.0.0** or later of CircuitPython. You can download it from the downloads page on circuitpython.org (https://adafru.it/Em8).

If you have previously installed it and you're not sure which version of CircuitPython your board is currently running, you can check the **boot_out.txt** file in the root folder of your device. Assuming your mass storage has not been disabled, this should be in the **CIRCUITPY** drive.

Once you have that downloaded, you will want to check out the appropriate learn guide for your specific board for installation instructions. If you would like a more general overview of installing CircuitPython (https://adafru.it/Amd), you can check out the Welcome to CircuitPython (https://adafru.it/cpy-welcome) guide.

Disabling USB Mass Storage

If you would like to use the REPL for file transfer, the USB mass storage device needs to be disabled, otherwise the file editing will need to done via the File System API.

The easiest way to do this is to simply eject the **CIRCUITPY** drive in your OS. However, this will only be temporary until the next time you reset the device or plug it back in. If you'd like it to stay disabled, you can create a **boot.py** file in the root folder of the **CIRCUITPY** drive with the following contents:

```
import storage

storage.disable_usb_drive()
```

Once you have saved the file, go ahead and perform a hard reset on the board by pressing the reset button or by temporarily disconnecting it from power. After it starts back up, the **CIRCUITPY** drive should no longer appear. If you'd like to learn more about editing **boot.py** to customize the device, check out the Customizing USB Devices in CircuitPython (https://adafru.it/SC9) guide.

Re-enabling USB Mass Storage

If you find you want to access the mass storage device after you have disabled it, there are a couple of ways you can do it. The first way to temporarily enable it is to boot it into safe mode. To do this, just reset the device and while it is booting, pay attention to the status NeoPixel. When it starts flashing yellow, press the boot button.

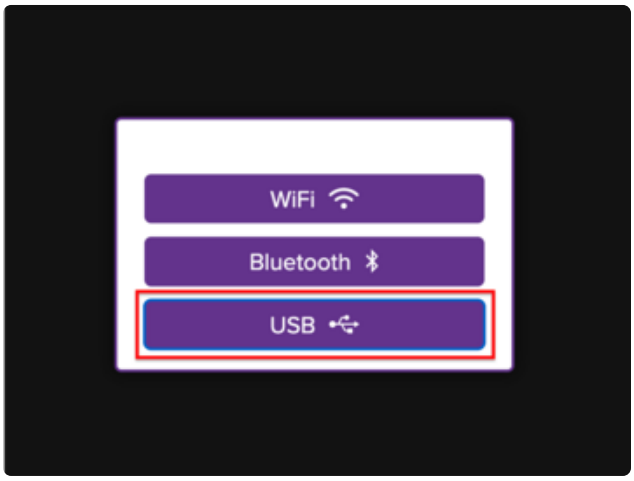
Alternatively, if you'd like to permanently re-enable it, you can run the following code from a serial terminal to remove the **boot.py** file:

```
import storage
import os
storage.remount("/", False)
os.remove("/boot.py")
```

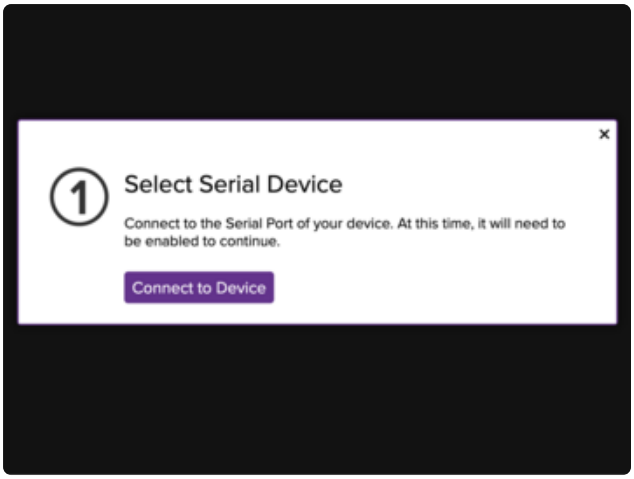
After hard resetting your device, the **CIRCUITPY** drive should reappear, though re-enabling it will cause web workflow to be in read-only mode again.

Connecting

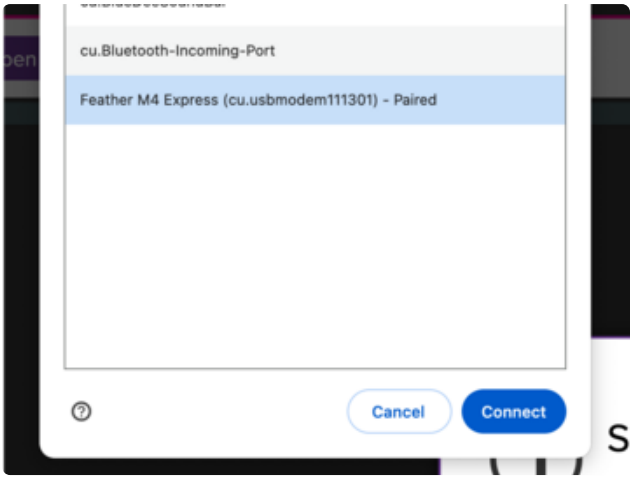
To use the Code Editor, you will need an internet browser such as Google Chrome or Microsoft Edge. It's possible that it may work in other browsers as well, but these ones have been more thoroughly tested.



Open your browser and navigate to https://code.circuitpython.org/ (https://adafru.it/10QF). Select USB on the dialog prompt that comes up.



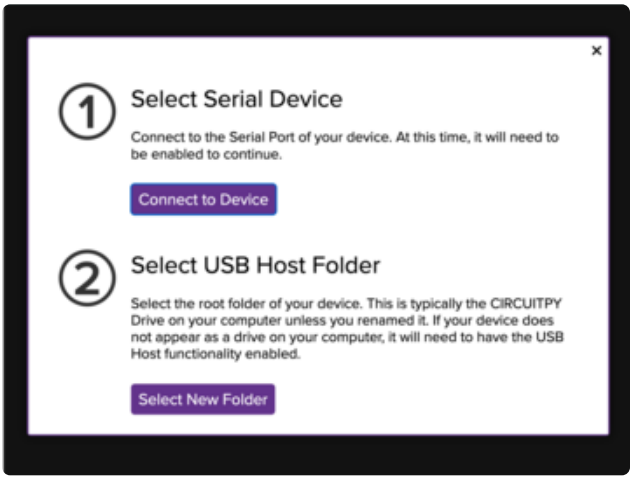
This will display a page of instructions along with a button to bring up a list of devices to connect to.



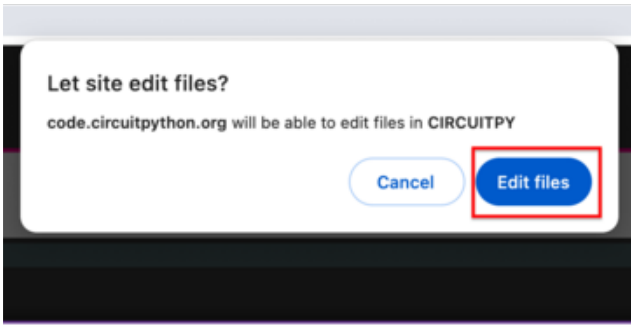
At this point, if your device does not have a CIRCUITPY drive or it is disabled, it should now be connected.

Devices with a CIRCUITPY Drive

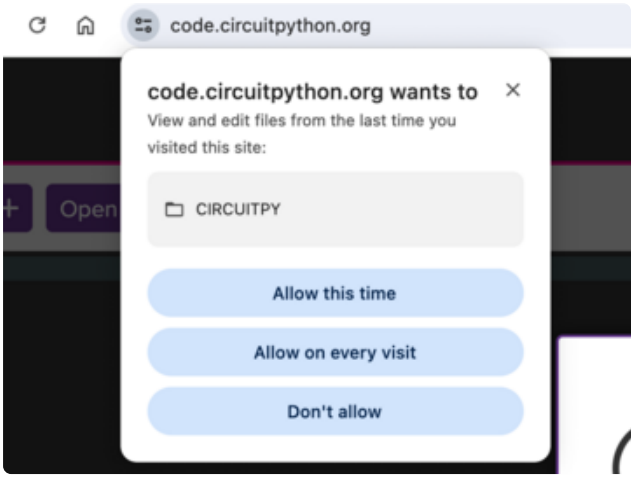
Devices with a CIRCUITPY drive will need a few additional steps because as the File System is mounted read-only for the REPL.



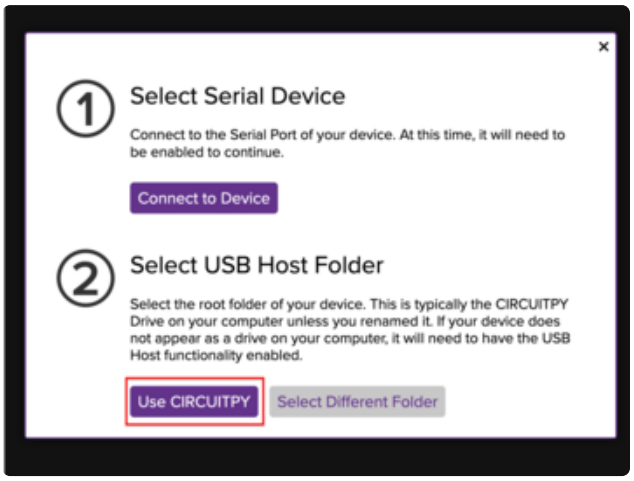
You will then need to select a Host Folder, which should be your CIRCUITPY drive unless you have renamed it.



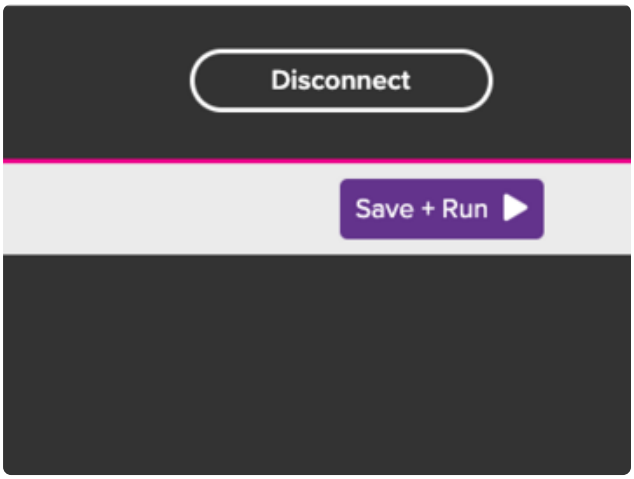
It may ask permission to edit the files in the location that you selected. Choose **Edit files**. This will allow the editor to read and write files.



Alternatively, you may get a dialog that asks if you want to allow CircuitPython to Edit files. You can click **Allow this time** if you'd like to be presented with the dialog each time or **Allow on every visit** if you would rather not be asked.



If you are happy with the location that you selected, choose **Use [Folder Name]** where [Folder Name] is the name of the folder you selected. Otherwise you can choose **Select Different Folder**.

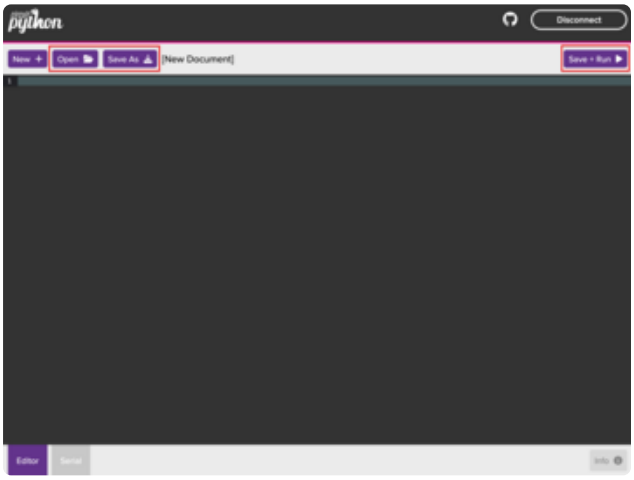


Once you have connected, the Connect button in the upper Right-hand corner should also change to a Disconnect button.

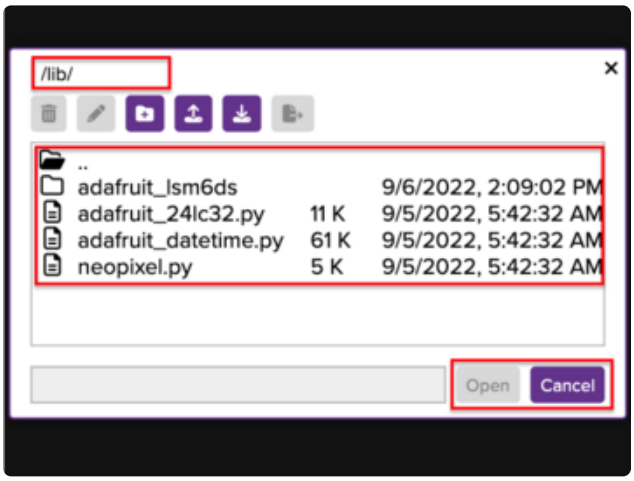
Usage

Opening and Saving Files

Opening and Saving files is designed to be like to most other applications. Just use the buttons along the top of the editor window.

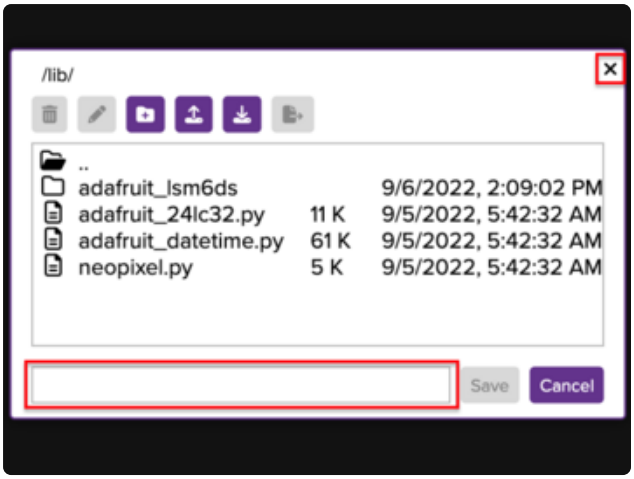


Clicking the **Open** or **Save As** buttons along the top will open the File Dialog. Clicking the **Save** + **Run** button will save your file and run the code. If your file hasn't been saved yet, this will also bring up the file dialog box.



The file dialog that appears is a simplified dialog that displays the current path at the top, allows you to navigate through the file tree to select the file you would like to open, and has buttons on the bottom to open or save the file you would like to use.

Canceling will tell the editor that you do not want to continue with the current operation.



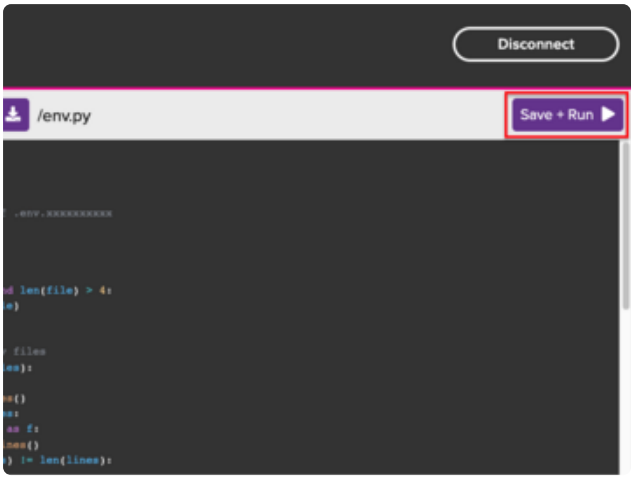
The X at the top performs the same function as the Cancel button as does clicking outside of the dialog.

On the Save As dialog, you can also type in a filename in the field next to the button.

Running Code

As mentioned above, the **Save + Run** button will first save your file, then run the code. The logic to run the code however is currently very simplistic in that it will try a couple of basic strategies to run your code, but doesn't currently do much beyond that.

The way it works is if you are working on **code.py** in the root folder, a soft reset will be performed, which automatically runs **code.py**. If you were working on some code in another file, the editor will attempt to perform an import on this code, which should run it. When you run your code, it will automatically switch over to the serial terminal.



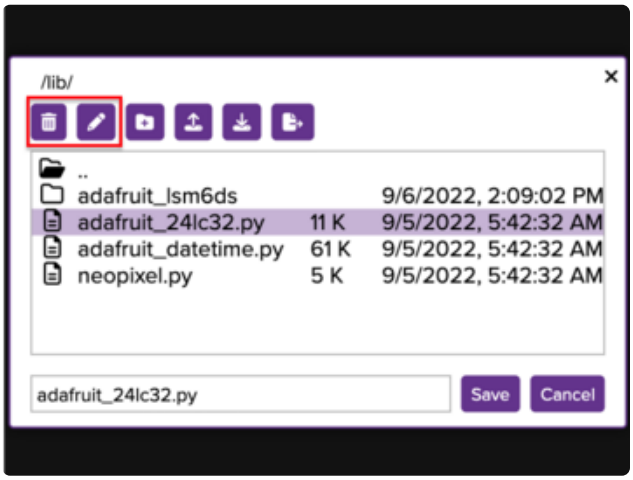
Click the **Save + Run** button to save and run the code current code.

File Dialog Toolbar

The file Dialog toolbar along the top allows you to perform common operations on files and folders regardless of whether you are saving or opening. Clicking the cancel button at the bottom will not undo any operations that were performed with these buttons.

Renaming and Deleting Files and Folders

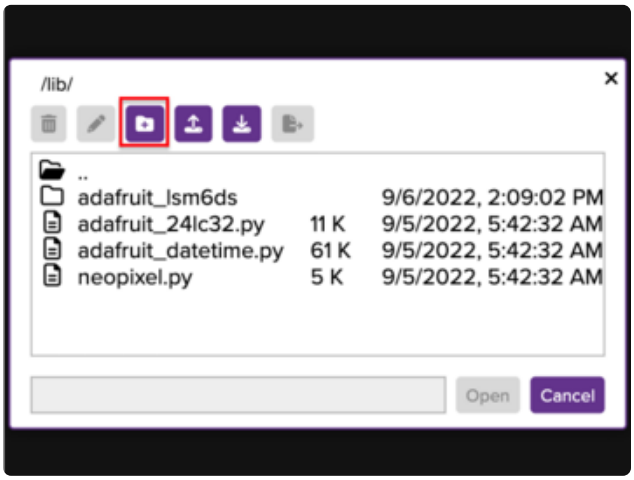
You can rename or delete both files and folders. An item must be selected first for the buttons to become available.

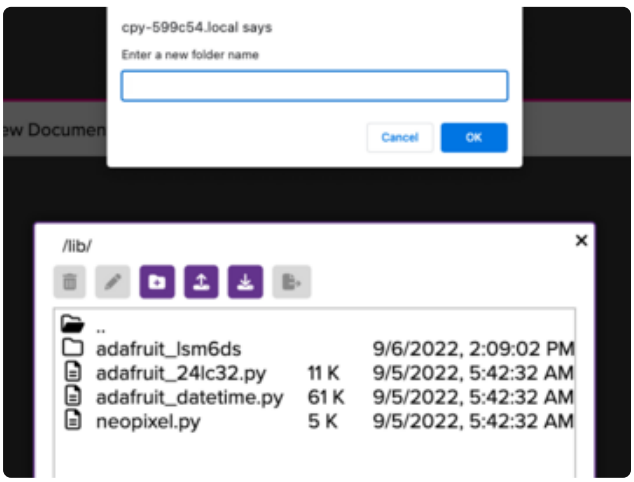


Use the **delete** and **rename** buttons here to perform the corresponding operation on the currently selected file or folder.

Creating New Folders

This feature allows you to create a new folder to store your work inside of.



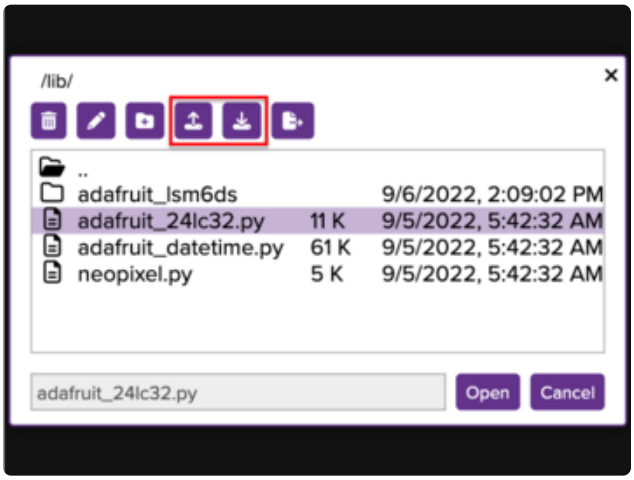


Clicking the **new folder** button at the top will prompt you for a folder name. It will inform you of invalid folder names such as the same name as an existing file or folder or a folder that begins with a period.

Uploading and Downloading Files and Folders

This feature allows you to upload or download files as long as they fit in the available space. If you need to add images or sound files for your project, you can use the upload button to add them. If you need to retrieve a file from your device for whatever reason, the download button will give you access to do that.

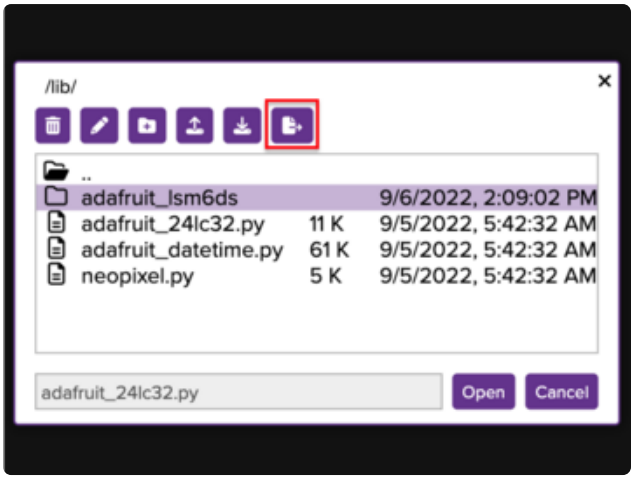
You can also download folders. When you select a folder and click download, the contents of that folder are automatically zipped into a single file. If nothing is selected when you click the download button, the current folder will be used.

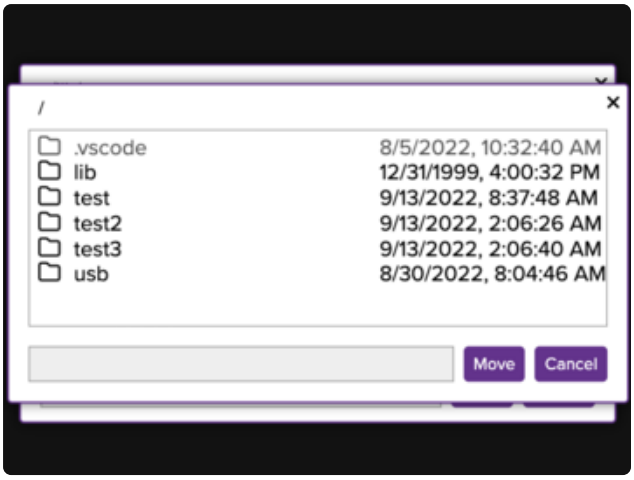


Use the **upload** or **download** buttons to easily add files or retrieve them from your board.

Moving Files and Folders

This feature allows you to move files and folders to a different location on the device. When you click the move button, another prompt will appear on top of the dialog that allows you to navigate to where you would like to move the currently selected item.



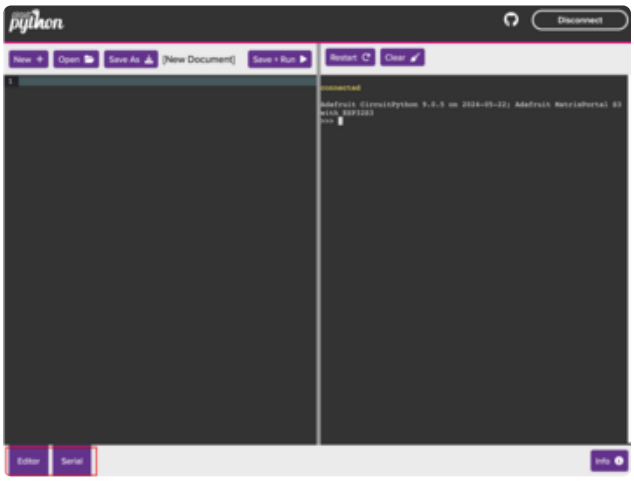


Use the **move** button to move files or folders to a new location on the device.

The second dialog that appears will show only folders and allow you to navigate to where you would like to move the file.

Using the Serial Terminal

The serial terminal allows you to watch the output of your device as well as type inputs just like you can from a separate application like PuTTY, except there's nothing you need to configure. This allows you to access the REPL or view the output of your currently running code.



Use the mode buttons in the bottom left-hand corner to open and close the serial and editor panes.

More Features to Come

The CircuitPython Code Editor is still under development, so expect more features to be added. If you would like to contribute on GitHub (https://adafru.it/10Rc), you can submit any new issues or pull requests for review.