Using Circuit Playground Express, MakeCode and CircuitPython on a Chromebook

Created by Anne Barela

https://learn.adafruit.com/using-circuit-playground-express-makecode-circuitpython-on-a-chromebook

Last updated on 2021-11-15 07:25:11 PM EST
# Table of Contents

## Overview
- Which Chromebook should I use? 4

## Chromebook Setup
- Initial Setup 5
- The Chrome Browser and Chrome OS 5
- Saving Files 6

## Using MakeCode
- Loading MakeCode 8
- New Projects 9
- Load UF2 files (existing project files) 9
- Load an existing MakeCode page and Download to CircuitPlayground 9
- Using MakeCode 11
- Loading the MakeCode Program to Your Device 11
- Saving Your Work: Files 13

## Using CircuitPython
- Running Caret 15
- Downloading Code from the Adafruit Learning System 16
- Editing Text 17
- Copying CircuitPython Code to Your Board 18

## Beagle Term 18
Overview

In the past, programming electronics has taken proprietary software on a PC or Mac.

With Microsoft MakeCode and CircuitPython, users have the choice of programming their projects with the computer of their choice: PC, Mac, Linux, and Chromebooks. Yep!

Chromebooks have been introduced to homes and schools as a low cost alternative to more expensive computers or tablets.

Historically it's been challenging to program on a Chromebook because you can't install IDEs or compilers without unlocking them. But thanks to MakeCode and CircuitPython - its a lot easier!

Chromebooks run a small operating system called Chrome OS. A Chromebook uses Google applications for web browsing in Chrome, word processing, spreadsheets and other basic applications via cloud-based Google services. Chromebooks can also download applications via the Chrome Web Store and the Google Play store.

MakeCode uses the Chrome browser and does not require any app support.

CircuitPython will use two apps to edit files and to interact with the board.
Which Chromebook should I use?

There are a good number of vendors selling Chromebooks. You may be limited to the Chromebook at a specific institution or you may be looking to purchase one.

If you are looking to purchase a Chromebook, the typical caveats for buying any computer apply:

- More recent releases generally have better capabilities and processors, making your experience faster and better.
- Consider support and warranty issues, plus consider the return policy if you do not like the unit.
- Be sure the device has a USB port for connecting microcontroller boards. A USB2 port is fine as is USB3 (which is USB2 backward compatible). USB C ports may need an adapter to work.
- If buying a used device, also consider how old the unit is, how much use it received, battery status, and functionality in addition to the above considerations. Very early generation devices may have less memory or limited processing compared to newer units but may be bargains. Ensure used units have chargers and that the charger is the correct one for that device.

In general, Chromebooks appear to come with either 2GB or 4GB or RAM memory and generally 16GB or more solid state disk space. If you can get a unit with 4GB memory, it may allow you to do more than 2GB memory units. A good return policy may help if you feel less than satisfied.

Chromebook Setup

If you are at a school or already have your Chromebook, skip down to the Saving Files part of this page.
Initial Setup

Follow the instructions which came with your Chromebook for unpackaging, charging the battery, etc.

1. You will need the log in information to your local WiFi, as nearly all Chromebooks use WiFi as their internet connection. You will need connectivity to set up Google services.
2. When you power on the device, it will ask you for your Google account name and password or to create a new one. If you already have a Google account and want this device to use that account, that's fine.
3. If the device is yours and it asks to update the Chrome OS, please do so to ensure you have the latest features and security updates.
4. Schools and institutions may have specifics for logging in to Google services. Follow their instructions. Often these groups do not want you to change device settings, update software, etc.

The Chrome Browser and Chrome OS

The Web Tabs area allows you to have multiple browser windows open at the same time. To create a new tab, press the + icon.

The Web Address Bar is where the web page address is typed in. Alternatively you can use the Google Search area to search for a web site.

If you are looking for other Google services, they are in the upper right. The 9 dots icon provides a full list. There you can find GMail, Youtube, Google Drive, etc.
If you need to change browser settings, add browser extensions, etc. you can find them in the cryptic three vertical dots icon in the far upper right corner.

Running applications on the Chromebook are depicted as Application icons in the bottom of the display.

**Saving Files**

Besides web browsing, you will need to know how to manipulate files on the Chromebook. The latest version of Chrome OS has a Files icon at the bottom. Click that.

If you don't have the blue Files icon, click the circle at the bottom left, then the ^ icon above the search bar then select the blue Files icon.

The Recent files area is useful for finding files that you have just used.

The My Files area is where you might be copying and saving files while you work on your Chromebook - the Downloads subdirectory is the space is on the flash drive memory of the Chromebook itself.

Remember when you are done using the Chromebook, it is best to copy your work onto Google Drive, an external thumb/flash drive, or somewhere else to ensure you have a backup of your files. If in a class, ask your instructor where they suggest saving files for permanent storage of your project.
If you are saving items to and from your Google Drive for more permanent storage or sharing, there is a Google Drive link on the left. You can drag files in the Files app to the Google Drive area and the file will be copied to your Google Drive. Google Drive space can be accessed anywhere you have a Google log in.

If you have a thumb/flash drive that you are saving files on, the drive name will appear on the left, below the Google Drive entry with a USB symbol that looks like a trident.

Boards like the Adafruit Circuit Playground Express and other programmable boards also show up as flash drives. The names may vary. CPLAYBOOT is the flash drive for firmware and MakeCode for the Adafruit Circuit Playground Express. Other boards will have xxxxBOOT names.

If the drive is named CIRCUITPY, that is for loading files for CircuitPython. If you have this drive and really want the CPLAYBOOT/xxxxBOOT drive, press the Reset button on the microcontroller board twice and the boot drive should appear instead of the CIRCUITPY drive.

If you have a file listing of any of these areas, you can click a file name, press the lower left of the Chromebook trackpad while dragging the file via the left mouse button to another file area to initiate copying the file. There appears to not be a copy/paste mechanism.
Using MakeCode

Microsoft MakeCode provides an exceptional environment for programming microcontrollers. The block-based coding includes capabilities that take advantage of board features.

MakeCode runs well with the Google Chrome browser in Chromebooks in addition to Microsoft Edge on PCs and other modern browsers.

Loading MakeCode

Start your Chromebook and load the Chrome web browser (the Chrome icon looks like a four-color beach ball). Put the following into the address bar:

https://www.microsoft.com/makecode (https://adafru.it/D1K)

Microsoft will change the address slightly, no worries. You can click the star icon in your address bar to make this a favorite.

Select the board you are going to program.

The rest of this guide will generally assume you are programming an Adafruit Circuit Playground Express which is both MakeCode and CircuitPython compatible.

Clicking the board will take you to a screen specific to the board you chose. Choosing Adafruit Circuit Playground Express takes you to https://makecode.adafruit.com/ (https://adafru.it/wmd).
You have several choices at this point:

New Projects

You can start a new project with the New Project button. If you have any previous projects, like mine titled "Untitled" from 7 months ago, they will appear next to New Project.

Load UF2 files (existing project files)

If you have a project on a disk, say on a flash drive or on your Google Drive in a file that ends in .UF2, then you can use the Import button circled above to load that file in for editing.

Load an existing MakeCode page and Download to CircuitPlayground

If you are working with an Adafruit tutorial and there is a link or button to the Microsoft MakeCode for the project, you can click that and it should automatically open in the editor in Chrome. As an example, the Adafruit tutorial MakeCode for Circuit Playground Express (https://adafruit.it/wWd) has an example near the bottom of this page (https://adafruit.it/D1R) that will download a UF2 code file of the color wheel example.

Another example is shown below - Soil Moisture with Circuit Playground Express (https://adafruit.it/C7z) has code on this page (https://adafruit.it/D1S).
You will get a browser page such as the following:

You can work with the code in this window but it is best to load the code into the full MakeCode editor. Select the Edit button in the upper right and then you'll get a new browser tab with the full MakeCode program.

Note the large pink Download button in the lower left - we'll use it soon!
Using MakeCode

You use MakeCode on Chrome the same as on other operating systems at this point. See the following tutorials for information or https://www.microsoft.com/makecode (https://adafru.it/D1K) for their tutorials:

- MakeCode for Circuit Playground Express (https://adafru.it/wWd)
- NeoPixels with MakeCode (https://adafru.it/D1L)
- Sensors in MakeCode (https://adafru.it/D1M)
- Adafruit Circuit Playground Express (https://adafru.it/adafruit-cpx)
- Other Adafruit Learning System Projects using MakeCode (https://adafru.it/D1N)

Loading the MakeCode Program to Your Device

When you have the code ready to load to your device, plug your device (such as an Adafruit Circuit Playground Express) to the Chromebook via a USB Male A to USB micro-B cable.

Be sure to use a USB cable with both power and data wires, preferably a good quality cable. If your program does not load, you might have a bad cable, try another and never use the tiny ones that come with cell phone backup batteries, they are power only.

When you have your program ready to load, press the Reset button on the board once. For the Circuit Playground Express the circle of LED lights turns red then green. This means it is ready. The device should now be recognized by the Chromebook as a flash drive named CPLAYBOOT in the Files app.

If you have previously installed Arduino or CircuitPython, you may need to double press the reset button to get your board into bootloader mode.
Click the pink Download button in the MakeCode window. A File Save box pops up.

Unfortunately you cannot drag the file to the CPLAYBOOT drive in this window, you must save the file to a drive first. This is a current MakeCode restriction.

Save the file on whatever drive you wish. Using My Files -> Downloads is ok unless your teacher says otherwise, other choices could be your own connected flash/thumb drive.
Open the Files app and click on the drive you saved your file on (like My Files -> Downloads). Now you can drag that file name to the CPLAYBOOT drive. If you don't see a drive named CPLAYBOOT (or similar for other boards), then ensure that:

1. You have the board connected via a good USB cable to the USB port of the Chromebook
2. You have pressed the Reset button and the lights are green and you have a CPLAYBOOT (or similar for other boards) drive in the Files app.

Your program should now be running!

Whenever you make changes, you'll need to do the same steps of saving the file to a disk then dragging the uf2 file to the CPLAYBOOT drive, after it appears by pressing Reset on the board.

Saving Your Work: Files

When you are done with your work, you should save your progress back to a file. Click the big pink Download button in the lower left. You'll get a screen like the one below:
Use the left panel to set where the file should be saved to. If you select My Files -> Downloads, they will be saved on the Chromebook. If you share the Chromebook, put a USB flash drive in and select it. You can consider using Google Drive also in this area.

The file name is circled in pink above. You can type any file name you want, but note you should use dashes between words instead of spaces. The end of the file must be .uf2 so it can be loaded on your project or in MakeCode later. When you have the save location and filename done, click the Save button.

You can use the Import button discussed previously to reload a .uf2 MakeCode file to use the code again.

---

**Using CircuitPython**

At the time of this guide, there is no version of the Mu editor (Adafruit's preferred CircuitPython editor/REPL) for Chromebook.

Instead of Mu, two programs will be used. One is a text editor and the other is a terminal program to see program output and error messages.

You will need one of each type of app from the Chrome web store. Apps can cost money or they can be free. We have selected two free, compatible programs, you are free to choose other programs:

- Caret Editor extension for Chrome - [Chrome Page](https://adafruit.it/D1O), [Author Page](https://adafruit.it/D1P)
- Beagle Term - [Chrome Page](https://adafruit.it/D1Q)
Once installed, you can launch the programs from their Chrome page or via the Chromebook Start circle.

See the next couple pages on using these apps for CircuitPython work on your Chromebook.

The pages refer to the following other Adafruit CircuitPython tutorial:

- [CircuitPython Made Easy on Circuit Playground Express](https://adafruit.it/cp-on-cpx-made-easy)

---

**Caret Editor**

**Running Caret**

Click the Start circle and click Caret. If you don't see it, click the ^ icon at the top of the opened screen and it will list icons for all the apps available. Scroll through and find Caret. Click the Caret icon to run.
Downloading Code from the Adafruit Learning System

Nearly all code of any length in the Adafruit Learning System (https://adafruit.it/id3) is stored on GitHub (https://adafruit.it/Clx). The Learning System interface includes a Download file link on embedded GitHub code listings.

The following page is from CircuitPython Made Easy on Circuit Playground Express (https://adafruit.it/D1U) - the Download file link is to this code on GitHub (https://adafruit.it/D1V).

When you click the Download file link, Chrome OS will open a File dialog asking where to save the file. You can save it on the Chromebook flash drive My Files -> Downloads or your own thumb drive.

Nearly all Python/CircuitPython files end in the file extension .py. When saving the downloaded file, you can select a meaningful name like servo.py or you can name it the same as you will name it on your board: code.py.
Editing Text

Run Caret by pressing the Chrome Start circle and selecting Caret.

You'll get the following screen:

![Caret interface](image)

You can start typing code now.

If you have existing code or if you downloaded code, you can open that file now. Select File -> Open and you get a File dialog box where you can select your code file (usually ends in .py).

It is possible you want to edit the code already on your board. Navigate to the CIRCU ITPY drive and click code.py to open.

```python
import time
import digitalio
import board
import crickit

# Two onboard GPX buttons for input (low level saves memory)
button_a = digitalio.DigitalInOut(board.BUTTON_A)
button_a.direction = digitalio.Direction.INPUT
button_a.pull = digitalio.Pull.DOWN

button_b = digitalio.DigitalInOut(board.BUTTON_B)
button_b.direction = digitalio.Direction.INPUT
button_b.pull = digitalio.Pull.DOWN

# Create one servo on Crickit's servo port M1
servo = crickit.servo_1

while True:
    if button_a.value:
        print("Button A pressed!")
        servo.angle = 0
        time.sleep(0.1)
    if button_b.value:
        print("Button B pressed!")
        servo.angle = 180
        time.sleep(0.1)
```
Edit away. Caret will color code your code as it knows it is Python (see lower right of screen).

Save your work via File -> Save File As... and name your code appropriately. We suggest naming the file you intend to immediately use code.py as that is what CircuitPython will expect.

You probably want to save the code also on Google Drive or your own flash drive (if you do not own the Chromebook to ensure you have a copy). The code will be saved on the flash of your microcontroller board but that is not the safest place, copies are good!

**Copying CircuitPython Code to Your Board**

Your code is edited and saved to a flash disk. You now want to put it on your device and run it.

Your board must have CircuitPython installed, but we have a detailed guide on how to do that!  

[https://adafru.it/Amd](https://adafru.it/Amd)

Run the Files app. Be sure your find your code. Your program should be named code.py to be able to be found by CircuitPython (not just anyfile.py, actually named code.py).

Plug your board (like an Adafruit Circuit Playground Express or other CircuitPython compatible board) into your ChromeBook via a known good USB A to micro-B cable. In the Files app, a new drive appears named CIRCUITPY. If you get a CPLAYBOOT (or other BOOT) drive or no drive, you probably need to install CircuitPython on the board.  
Follow this guide to do the install  
[https://adafru.it/Amd](https://adafru.it/Amd).

Now drag your CircuitPython source file (code.py) to the board. The code should immediately run.

---

**Beagle Term**

You will need an application to view the output from the CircuitPython program. Without Mu, we turn to Beagle Term. Install the application by going to this  

[Chrome Page](https://adafru.it/D1Q)  
and clicking the Install button.
Click the Start circle. If you don't see Beagle Term on that screen, go to the apps icons by clicking the ^ symbol. Search for the Beagle Term icon.

The Beagle Term opening screen is shown below. the device Port may show up as /dev/ttyACM0 or /dev/ttyACM1 on my Samsung Chromebook corresponding to the two USB ports it has, one on each side of the unit. The port names may vary among different Chromebook devices.

The Bitrate should not matter, you can set the Bitrate as 9600 which we recommend for consistency.

If your code outputs text messages using the `print` statement, they can be viewed in Beagle Term.
Pressing the ctrl-c key combination should (should) get you to the CircuitPython REPL. Beagle Term sometimes has failed me here, so I'll not guarantee your Chromebook with Beagle Term might do this. I've looked for a Beagle Term replacement with little success.

If it works, you'll get the REPL as in the screen below.

Serial output is great for debugging. CircuitPython will output error messages to the terminal as well as send output via print statements you put in your code.

If you are in the REPL and want to restart your code.py program, press **CTRL-d**.

There is no code plot capability or save to Chromebook file capability. There are rough cut and paste functions built in (see the scissors icon above). This makes Beagle Term inferior to Mu on larger PCs/Macs. If Mu is ported to Chrome OS, we'll look to update this page!