MP3 Playback in CircuitPython with Lars the Sloth Puppet

Created by John Park

Last updated on 2021-11-04 05:19:38 PM EDT
# Guide Contents

Guide Contents ............................................. 2

Overview .................................................. 3
  Parts .................................................... 3
  Plush Doll ............................................. 5
  Who is Lars? ........................................... 5

Build the MP3 Player .................................... 7
  Add Header Pins ....................................... 8
  Enable Switch ......................................... 8
  Battery Power ......................................... 10
  Fasten with Hardware ................................ 11
  Speaker ............................................... 12

CircuitPython on Feather M4 Express ............. 13
  Set up CircuitPython Quick Start! .................. 13
  Further Information .................................. 14

CircuitPython Libraries ............................... 15
  The Adafruit CircuitPython Library Bundle ...... 16
  Downloading the Adafruit CircuitPython Library Bundle ........................................... 16
  The CircuitPython Community Library Bundle ......................................................... 16
  Downloading the CircuitPython Community Library Bundle ........................................ 17
  Understanding the Bundle .......................... 17
  Example Files ......................................... 18
  Copying Libraries to Your Board ................. 18
  Understanding Which Libraries to Install ...... 18
  Example: ImportError Due to Missing Library .................................................. 21
  Library Install on Non-Express Boards .......... 22
  Updating CircuitPython Libraries and Examples ..................................................... 22

Code the MP3 Player .................................... 23
  Text Editor ........................................... 23
  Code.py ............................................. 23
  Audio Samples ........................................ 24
  Make Your Own ....................................... 25
  How It Works ........................................ 25
  Test It .............................................. 25

Implant the MP3 Player in Puppet ................. 27
  Seam Ripper ......................................... 28
  Feed in the Speaker ................................ 29
  Bag It ............................................... 30
  USB Jack ............................................. 31
Overview

You can now play MP3 files directly in CircuitPython with no dedicated decoding hardware! You'll be able to use this new capability in all sorts of projects, including this creepy Lars the Sloth voice box project!

The benefit of MP3 over other non-compressed file formats, such as WAV, has always been their small file size, meaning you can pack a lot of songs or samples into limited memory. However, decompressing those small files for playback was too taxing for older microcontrollers, so we had to use dedicated decoder chips at an additional cost, including licensing fees.

Thanks to the raw speed of M4, the days of requiring a dedicated MP3 audio decoder chip are behind us. And, since the patents have expired, we can now distribute MP3 playback libraries free of charge (https://adafru.it/Hey)!

Parts

Adafruit Feather M4 Express - Featuring ATSAMD51
It's what you've been waiting for, the Feather M4 Express featuring ATSAMD51. This Feather is fast like a swift, smart like an owl, strong like an ox-bird (it's half ox,...
$22.95
In Stock
Add to Cart

Adafruit Prop-Maker FeatherWing
The Adafruit Feather series gives you lots of options for a small, portable, rechargeable microcontroller board. Perfect for fitting into your next prop build! This FeatherWing will...
Out of Stock

Mini Oval Speaker - 8 Ohm 1 Watt
Hear the good news! This wee speaker is a great addition to any audio project where you need 8 ohm impedance and 1W or less of power. We particularly like...
$1.95
In Stock
Add to Cart

Lithium Ion Polymer Battery Ideal For Feathers - 3.7V 400mAh
Lithium-ion polymer (also known as 'lipo' or 'lipoly') batteries are thin, light, and powerful. The output ranges from 4.2V when completely charged to 3.7V. This...
$6.95
In Stock
Tactile On/Off Switch with Leads
Squeeze once to turn on, squeeze again to turn off! This clicky switch makes a great power switch or mode toggler. We like this switch because it's easy to embed in a seam for...

Out of Stock

Micro B Round Panel Mount Extension Cable - 30cm
If you need to add a panel-mount connection for USB Micro B, but don't have the time or ability to cut a custom oval or square hole, this Round Panel Mount...

$4.95

In Stock

20-pin 0.1" Female Header - Red - 5 pack
Female header is like the duct tape of electronics. It's great for connecting things together, soldering to perf-boards, sockets for wires or break-away header, etc. We go through...

$2.50

In Stock

Break-away 0.1" 36-pin strip male header - Red - 10 pack
In this world nothing can be said to be certain, except we need headers, headers, and more headers!Each pack contains ten red 36-pin 0.1" pitch...

Out of Stock

Black Nylon Machine Screw and Stand-off Set – M2.5 Thread
Totaling 380 pieces, this M2.5 Screw Set is a must-have for your workstation. You’ll have enough screws, nuts, and hex standoffs to fuel your maker...

$16.95

In Stock
Plush Doll

The choice is yours, but if you want to embed creepy, semi-threatening voice samples into a murder puppet, you could do worse than a Sickening Sloth Fuggler (https://adafru.it/LgB).

The teeth. Oh the teeth.

Who is Lars?

Lars (https://adafru.it/LgF) is the sworn enemy of CVT Joseph (https://adafru.it/Lha), the ice cream truck godfather of Los Angeles, and owner of CVT Soft Serve (https://adafru.it/Lhb).

Lars is angry that Joe won't reveal the location of his Wonka-style Golden ice cream gift cards. This video linked below will make everything clear:

https://www.instagram.com/p/B--_P8JJWWy/ (https://adafru.it/Lhc)

Actor Matt Biedel (https://adafru.it/Lhd) (Umbrella Academy, Narcos) has kindly given us permission to use Lars's voice. At least that's what Joe claims.
This project is not sponsored by CVT Soft Serve. John is just really obsessed with Lars and great ice cream.
Build the MP3 Player

You can play .mp3 files with just a Feather M4 plus an amplifier and speaker. However, for the puppet sound playback, we'll make a nice, compact player with some extra features.

The Propmaker FeatherWing gives us a very convenient built-in amplifier and speaker output port, as well as an accelerometer with tap-detection we will use to trigger the sample playback.

With a small battery, USB port extender, and toggle switch for powering the board on and off via the Enable pin, we'll have a great little player that can be embedded inside the puppet, while still allowing it to be triggered from the outside, recharged and even coded without requiring disassembly.
Add Header Pins
Solder a set of female and male header pins to the Feather and FeatherWing respectively as shown here. This guide (https://adafruit.it/Lhe) shows how to solder in the headers.

Enable Switch
The Enable pin on the Feather can be used to turn the board on and off.

Push the wires through the Propmaker FeatherWing's hold as shown (for strain relief) and then solder it to the G and En pads (either wire in either hole is fine).
Battery Power
Connect the battery to the JST battery jack and add a little piece of double-stick foam tape to keep it secure between the two boards.
Fasten with Hardware

Use four M2.5 standoff and eight short M2.5 screws to fasten the boards together.
Speaker
Next, plug in the speaker and you'll be ready for programming and testing your .mp3 player!
CircuitPython on Feather M4 Express

CircuitPython (https://adafru.it/tB7) is a derivative of MicroPython (https://adafru.it/BeZ) designed to simplify experimentation and education on low-cost microcontrollers. It makes it easier than ever to get prototyping by requiring no upfront desktop software downloads. Simply copy and edit files on the CIRCUITPY drive to iterate.

The following instructions will show you how to install CircuitPython. If you've already installed CircuitPython but are looking to update it or reinstall it, the same steps work for that as well!

Set up CircuitPython Quick Start!

Follow this quick step-by-step for super-fast Python power :) 

Click the link above and download the latest UF2 file.

Download and save it to your desktop (or wherever is handy).

Plug your Feather M4 into your computer using a known-good USB cable.

A lot of people end up using charge-only USB cables and it is very frustrating! So make sure you have a USB cable you know is good for data sync.

Double-click the Reset button next to the USB connector on your board, and you will see the NeoPixel RGB LED turn green. If it turns red, check the USB cable, try another USB port, etc. Note: The little red LED next to the USB connector will pulse red. That's ok!

If double-clicking doesn't work the first time, try again. Sometimes it can take a few tries to get the rhythm right!
You will see a new disk drive appear called FEATHERBOOT.

Drag the `adafruit_circuitpython_etc.uf2` file to FEATHERBOOT.

The LED will flash. Then, the FEATHERBOOT drive will disappear and a new disk drive called CIRCUITPY will appear.

That's it, you're done! :)

Further Information

For more detailed info on installing CircuitPython, check out Installing CircuitPython (https://adafruit.it/Amd).
CircuitPython Libraries

As CircuitPython development continues and there are new releases, Adafruit will stop supporting older releases. Visit https://circuitpython.org/downloads to download the latest version of CircuitPython for your board. You must download the CircuitPython Library Bundle that matches your version of CircuitPython. Please update CircuitPython and then visit https://circuitpython.org/libraries to download the latest Library Bundle.

Each CircuitPython program you run needs to have a lot of information to work. The reason CircuitPython is so simple to use is that most of that information is stored in other files and works in the background. These files are called libraries. Some of them are built into CircuitPython. Others are stored on your CIRCUITPY drive in a folder called lib. Part of what makes CircuitPython so great is its ability to store code separately from the firmware itself. Storing code separately from the firmware makes it easier to update both the code you write and the libraries you depend.

Your board may ship with a lib folder already, it's in the base directory of the drive. If not, simply create the folder yourself. When you first install CircuitPython, an empty lib directory will be created for you.

CircuitPython libraries work in the same way as regular Python modules so the Python docs (https://adafruit.it/rar) are an excellent reference for how it all should work. In Python terms, you can place our library files in the lib directory because it's part of the Python path by default.

One downside of this approach of separate libraries is that they are not built in. To use them, one needs to copy them to the CIRCUITPY drive before they can be used. Fortunately, there is a library bundle.

The bundle and the library releases on GitHub also feature optimized versions of the libraries with the .mpy file extension. These files take less space on the drive and have a smaller memory footprint as they are loaded.

Due to the regular updates and space constraints, Adafruit does not ship boards with the entire bundle. Therefore, you will need to load the libraries you need when you begin working with your board. You can find example code in the guides for your board that depends on external libraries.
Either way, as you start to explore CircuitPython, you'll want to know how to get libraries on board.

The Adafruit CircuitPython Library Bundle

Adafruit provides CircuitPython libraries for much of the hardware they provide, including sensors, breakouts and more. To eliminate the need for searching for each library individually, the libraries are available together in the Adafruit CircuitPython Library Bundle. The bundle contains all the files needed to use each library.

Downloading the Adafruit CircuitPython Library Bundle

You can download the latest Adafruit CircuitPython Library Bundle release by clicking the button below. The libraries are being constantly updated and improved, so you'll always want to download the latest bundle.

**Match up the bundle version with the version of CircuitPython you are running.** For example, you would download the 6.x library bundle if you're running any version of CircuitPython 6, or the 7.x library bundle if you're running any version of CircuitPython 7, etc. If you mix libraries with major CircuitPython versions, you will get incompatible mpy errors due to changes in library interfaces possible during major version changes.

Download the bundle version that matches your CircuitPython firmware version. If you don't know the version, check the version info in *boot_out.txt* file on the CIRCUITPY drive, or the initial prompt in the CircuitPython REPL. For example, if you're running v7.0.0, download the 7.x library bundle.

There's also a *py* bundle which contains the uncompressed python files, you probably *don't* want that unless you are doing advanced work on libraries.

The CircuitPython Community Library Bundle

The CircuitPython Community Library Bundle is made up of libraries written and provided by members of the CircuitPython community. These libraries are often written when community members encountered hardware not supported in the Adafruit Bundle, or to support a personal project. The authors all chose to submit these libraries to the Community Bundle make them available to the community.

**These libraries are maintained by their authors and are not supported by Adafruit.** As you would with any library, if you run into problems, feel free to file an issue on the GitHub repo for the library. Bear in mind, though, that most of these libraries are supported by a single person and you should be patient about receiving a response. Remember, these folks are not paid by Adafruit, and are volunteering their personal time when possible to provide support.
Downloading the CircuitPython Community Library Bundle

You can download the latest CircuitPython Community Library Bundle release by clicking the button below. The libraries are being constantly updated and improved, so you'll always want to download the latest bundle.

https://adafru.it/VCn

The link takes you to the latest release of the CircuitPython Community Library Bundle on GitHub. There are multiple versions of the bundle available. **Download the bundle version that matches your CircuitPython firmware version.** If you don't know the version, check the version info in `boot_out.txt` file on the CIRCUITPY drive, or the initial prompt in the CircuitPython REPL. For example, if you're running v7.0.0, download the 7.x library bundle.

Understanding the Bundle

After downloading the zip, extract its contents. This is usually done by double clicking on the zip. On Mac OSX, it places the file in the same directory as the zip.

Open the bundle folder. Inside you'll find two information files, and two folders. One folder is the lib bundle, and the other folder is the examples bundle.

Now open the lib folder. When you open the folder, you'll see a large number of `.mpy` files, and folders.
Example Files

All example files from each library are now included in the bundles in an examples directory (as seen above), as well as an examples-only bundle. These are included for two main reasons:

- Allow for quick testing of devices.
- Provide an example base of code, that is easily built upon for individualized purposes.

Copying Libraries to Your Board

First open the lib folder on your CIRCUITPY drive. Then, open the lib folder you extracted from the downloaded zip. Inside you’ll find a number of folders and .mpy files. Find the library you’d like to use, and copy it to the lib folder on CIRCUITPY.

If the library is a directory with multiple .mpy files in it, be sure to copy the entire folder to CIRCUITPY/lib.

This also applies to example files. Open the examples folder you extracted from the downloaded zip, and copy the applicable file to your CIRCUITPY drive. Then, rename it to code.py to run it.

Understanding Which Libraries to Install

You now know how to load libraries on to your CircuitPython-compatible microcontroller board. You may now be wondering, how do you know which libraries you need to install? Unfortunately, it’s not always
straightforward. Fortunately, there is an obvious place to start, and a relatively simple way to figure out the rest. First up: the best place to start.

When you look at most CircuitPython examples, you'll see they begin with one or more `import` statements. These typically look like the following:

- `import library_or_module`

However, `import` statements can also sometimes look like the following:

- `from library_or_module import name`
- `from library_or_module.subpackage import name`
- `from library_or_module import name as local_name`

They can also have more complicated formats, such as including a `try` / `except` block, etc.

The important thing to know is that an `import` statement will always include the name of the module or library that you're importing.

Therefore, the best place to start is by reading through the `import` statements.

Here is an example import list for you to work with in this section. There is no setup or other code shown here, as the purpose of this section involves only the import list.

```python
import time
import board
import neopixel
import adafruit_lis3dh
import usb_hid
from adafruit_hid.consumer_control import ConsumerControl
from adafruit_hid.consumer_control_code import ConsumerControlCode
```

Keep in mind, not all imported items are libraries. Some of them are almost always built-in CircuitPython modules. How do you know the difference? Time to visit the REPL.

In the Interacting with the REPL section (https://adafruit.it/Awz) on The REPL page (https://adafruit.it/Awz) in this guide, the `help("modules")` command is discussed. This command provides a list of all of the built-in modules available in CircuitPython for your board. So, if you connect to the serial console on your board, and enter the REPL, you can run `help("modules")` to see what modules are available for your board. Then, as you read through the `import` statements, you can, for the purposes of figuring out which libraries to load, ignore the statement that import modules.

The following is the list of modules built into CircuitPython for the Feather RP2040. Your list may look similar or be anything down to a significant subset of this list for smaller boards.
Now that you know what you're looking for, it's time to read through the import statements. The first two, `time` and `board`, are on the modules list above, so they're built-in.

The next one, `neopixel`, is not on the module list. That means it's your first library! So, you would head over to the bundle zip you downloaded, and search for `neopixel`. There is a `neopixel.mpy` file in the bundle zip. Copy it over to the `lib` folder on your `CIRCUITPY` drive. The following one, `adafruit_lis3dh`, is also not on the module list. Follow the same process for `adafruit_lis3dh`, where you'll find `adafruit_lis3dh.mpy`, and copy that over.

The fifth one is `usb_hid`, and it is in the modules list, so it is built in. Often all of the built-in modules come first in the import list, but sometimes they don't! Don't assume that everything after the first library is also a library, and verify each import with the modules list to be sure. Otherwise, you'll search the bundle and come up empty!

The final two imports are not as clear. Remember, when `import` statements are formatted like this, the first thing after the `from` is the library name. In this case, the library name is `adafruit_hid`. A search of the bundle will find an `adafruit_hid` folder. When a library is a folder, you must copy the `entire folder and its contents as it is in the bundle` to the `lib` folder on your `CIRCUITPY` drive. In this case, you would copy the entire `adafruit_hid` folder to your `CIRCUITPY/lib` folder.

Notice that there are two imports that begin with `adafruit_hid`. Sometimes you will need to import more than one thing from the same library. Regardless of how many times you import the same library, you only need to load the library by copying over the `adafruit_hid` folder once.

That is how you can use your example code to figure out what libraries to load on your CircuitPython-compatible board!

There are cases, however, where libraries require other libraries internally. The internally required library is called a `dependency`. In the event of library dependencies, the easiest way to figure out what other libraries are required is to connect to the serial console and follow along with the `ImportError` printed there. The following is a very simple example of an `ImportError`, but the concept is the same for any missing library.
Example: **ImportError** Due to Missing Library

If you choose to load libraries as you need them, or you're starting fresh with an existing example, you may end up with code that tries to use a library you haven't yet loaded. This section will demonstrate what happens when you try to utilise a library that you don't have loaded on your board, and cover the steps required to resolve the issue.

This demonstration will only return an error if you do not have the required library loaded into the `lib` folder on your **CIRCUITPY** drive.

Let's use a modified version of the Blink example.

```python
import board
import time
import simpleio

led = simpleio.DigitalOut(board.LED)

while True:
    led.value = True
    time.sleep(0.5)
    led.value = False
    time.sleep(0.5)
```

Save this file. Nothing happens to your board. Let's check the serial console to see what's going on.

You have an **ImportError**. It says there is **no module named 'simpleio'**. That's the one you just included in your code!

Click the link above to download the correct bundle. Extract the lib folder from the downloaded bundle file. Scroll down to find **simpleio.mpy**. This is the library file you're looking for! Follow the steps above to load an individual library file.

The LED starts blinking again! Let's check the serial console.
No errors! Excellent. You've successfully resolved an ImportError!

If you run into this error in the future, follow along with the steps above and choose the library that matches the one you're missing.

Library Install on Non-Express Boards

If you have an M0 non-Express board such as Trinket M0, Gemma M0, QT Py M0, or one of the M0 Trinkeys, you'll want to follow the same steps in the example above to install libraries as you need them. Remember, you don't need to wait for an ImportError if you know what library you added to your code. Open the library bundle you downloaded, find the library you need, and drag it to the lib folder on your CIRCUITPY drive.

You can still end up running out of space on your M0 non-Express board even if you only load libraries as you need them. There are a number of steps you can use to try to resolve this issue. You'll find suggestions on the Troubleshooting page (https://adafruit.it/Den).

Updating CircuitPython Libraries and Examples

Libraries and examples are updated from time to time, and it's important to update the files you have on your CIRCUITPY drive.

To update a single library or example, follow the same steps above. When you drag the library file to your lib folder, it will ask if you want to replace it. Say yes. That's it!

A new library bundle is released every time there's an update to a library. Updates include things like bug fixes and new features. It's important to check in every so often to see if the libraries you're using have been updated.
Code the MP3 Player

Once your Feather is set up with CircuitPython 5.3.0 or greater, you'll also need to add some libraries. Follow this page (https://adafruit.it/ABU) for information on how to download and add libraries to your Feather.

From the library bundle you downloaded in that guide page, transfer the following libraries onto the Feather's /lib directory:

- adafruit_bus_device
- adafruit_lis3dh
- neopixel

Text Editor

Adafruit recommends using the Mu editor for using your CircuitPython code with the Feather boards. You can get more info in this guide (https://adafruit.it/ANO).

Alternatively, you can use any text editor that saves files.

Code.py

Copy the code below and paste it into Mu. Then, save it to your Feather as code.py.
import time
import board
import busio
import digitalio
import audioio
import audiomp3
import adafruit_lis3dh

start_stop_0 = False  # set to True to play all samples once on startup

# Set up accelerometer on I2C bus
i2c = busio.I2C(board.SCL, board.SDA)
int1 = digitalio.DigitalInOut(board.D6)
accel = adafruit_lis3dh.LIS3DH_I2C(i2c, int1=int1)
accel.set_tap(1, 100)  # single or double-tap, threshold

# Set up speaker enable pin
enable = digitalio.DigitalInOut(board.D10)
enable.direction = digitalio.Direction.OUTPUT
enable.value = True

speaker = audioio.AudioOut(board.A0)
sample_number = 0

print("Lars says, 'Hello, CVT Joseph. Tap to play.'")

if startup_play:  # Play all on startup
    for i in range(10):
        sample = "/lars/lars_0{}.mp3".format(i)
        print("Now playing: '{}'".format(sample))
        mp3stream = audiomp3.MP3Decoder(open(sample, "rb"))
        speaker.play(mp3stream)

        while speaker.playing:
            time.sleep(0.1)
        enable.value = speaker.playing

while True:
    if accel.tapped and speaker.playing is False:
        sample = "/lars/lars_0{}.mp3".format(sample_number)
        print("Now playing: '{}".format(sample))
        mp3stream = audiomp3.MP3Decoder(open(sample, "rb"))
        speaker.play(mp3stream)
        sample_number = (sample_number + 1) % 10
        enable.value = speaker.playing

Audio Samples
You can get started by using these .mp3 files. (Of course, you're free to use any files that you like, see
below for more info on that.)

Download the .zip file linked below and then uncompress the file. Drag the /lars directory onto the root level of your Feather’s CIRCUITPY drive.

https://adafruit.it/LgD

Make Your Own

To make your own .mp3, use audio software such as Audacity (https://adafruit.it/Lee) to save them with these settings:

- bit rate: anywhere from 16 to 320 kb/s (lower will be smaller, higher is better quality)
- sample rate: 22050Hz or 44100Hz are recommended, although 16kHz, 24kHz, and 48kHz should also work
- channels: mono or stereo
- must be DRM free

How It Works

Here’s what the code does:

- Imports the necessary libraries for audiomp3 playback and accelerometer tap detection with the adafruit_lis3dh
- Sets up the accelerometer
- Specifies the speaker enable pin (this allows the speaker to be disabled when not in use)
- If startup_play is set to True, each sample in the /lars directory with the name format of /lars_0*.mp3 will be played using the mp3stream object
- In the main loop of the program, the accelerometer is checked for tap detection. When the tap occurs, the next sample is played. When the tenth one is played, the list loops around back to the first one again.

Test It

With the enable switch clicked on, try tapping the Feather/Propmaker to trigger a sample. When a sample has finished you can tap again to trigger the next one.

Note, there is a small volume adjustment screw on the Propmaker FeatherWing that can be turned with a small screwdriver to adjust the maximum volume.

Now, let’s add the sound player to our puppet friend Lars!
Implant the MP3 Player in Puppet

Before making any incisions, we'll lay out the parts on the puppet to get a feel for where things will go. This is a good time to add the USB panel-mount port extender to the micro USB jack on the Feather.

We want to get the speaker up near Lars’s inexplicably human-toothed maw, with the Feather board near the center of his back so it is easy to tap it to trigger the sound sample playback.

I decided to place the on/off switch for the enable pin in Lars's right paw-claw-hand thing, and the USB port jack in his left foot.
Seam Ripper

Use a seam ripper or small scissors to open up the seam on the bottom of Lars’s left foot.
Feed in the Speaker
Feed the speaker up through the foot hole and work it up toward the head area.
Bag It
To prevent the board from snagging on the doll’s batting on the way in, you can wrap the feather in a small bag with a zip tie or rubber band to close off the neck as shown here.
USB Jack

The last thing to push into the plush toy will be the USB panel-mount jack.

Use a medium sized zip tie to secure it to the end of the foot as shown, then trim off the excess zip tie end.
You can now plug a USB cable into Lars's foot for battery charging, swapping out .mp3 files, or even adjusting the CircuitPython code!

Test out your finished puppet now by making sure the Enable switch is clicked on, then tap his back to play!

You may need to wiggle the speaker around to get it to an optimal placement. Also ensure the battery is charged or the USB cable plugged in to get sufficient power.
Note, in the video below I was using a double-tap to trigger.