CircuitPython Kernel Documentation

Release 0.3.0.dev

Adafruit Industries

Jun 14, 2018

Contents

Contents:

CHAPTER 1

CircuitPython Kernel

The CircuitPython Kernel is a Jupyter Kernel designed to interact with Adafruit boards running CircuitPython from within a Jupyter Notebook

1.1 Status

This project's status is experimental. It has been tested with CircuitPython (SAMD) boards and the Feather HUZZAH (ESP8266) using CircuitPython 2.x Stable and 3.0.0 Beta 1.

It may break, and if it does, please file an issue on this repository.

1.2 Compatible Boards

1.2.1 Designed for CircuitPython (SAMD21 and SAMD51)

- Adafruit CircuitPlayground Express
- Adafruit Feather M0 Express
- Adafruit Metro M0 Express
- Adafruit Metro M4 Express
- Adafruit Trinket M0
- Adafruit Gemma M0
- Adafruit Trinket M0
- Adafruit ItsyBitsy M0 Express

- Adafruit ItsyBitsy M4 Express

1.2.2 Other Boards

- Adafruit Feather HUZZAH ESP8266

1.3 Download

Official .zip files are available through the latest GitHub releases.

1.4 Documentation

This kernel is fully documented on the Adafruit Learning System (*Guide Coming Soon!*)

There's also documentation for this kernel listed on the ReadTheDocs page for this repo..

CHAPTER 2

Installing Jupyter

Option 1. Installing Jupyter with Anaconda

Don't have a Python installation on your computer? If you're new to all this, the Jupyter Project recommends installing Anaconda , which installs Python, the Jupyter Notebook, and other commonly used packages for scientific computing and data science.

Option 2. Installing Jupyter with PIP

If you have a Python installation already on your computer, you may want to use the Python package manager (pip) instead of Anaconda. You'll need Python 3.3+.

First, ensure that you have the latest version of pip:

```
$ pip3 install --upgrade pip
```

Install the Jupyter Notebook using:

```
$ pip3 install jupyter
```

Ok, now that we have Jupyter installed, let's start the notebook server.

We can launch the server from a command line (either Terminal on macOS/Linux or Command Prompt on Windows) by running:

```
$ jupyter notebook
```

If your installation went well, you'll see information about the notebook server in your command line. Also, your web browser will open to the URL displayed in your command line (http://localhost:8888), displaying the Notebook Dashboard.

CHAPTER 3

Installing the CircuitPython Kernel

First, clone this repository.

```
$ git clone https://github.com/adafruit/circuitpython_kernel.git
```

Navigate into the cloned repository directory. Install this kernel into Jupyter by running:

```
$ pip3 install circuitpython_kernel
```

Then, run

```
$ python3 -m circuitpython_kernel.install
```

- if you encounter errors running this command on macOS/Linux, you'll need to prefix this command with *sudo*

Finally, let's verify the kernel was installed correctly in Jupyter. To do this, run:

```
$ jupyter kernelspec list
```

Your output should show circuitpython as an available kernel:

CHAPTER 4

Launching a CircuitPython Notebook

Launch jupyter by running:

```
$ jupyter notebook
```

Make sure your board is plugged into USB and running CircuitPython by opening a file explorer. It should show up as a removable drive named **CIRCUITPY**.

Then click **new -> circuitpython** to open a new CircuitPython Notebook

A new CircuitPython Notebook should open and you should be able to execute code from within a cell.

CHAPTER 5

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/adafruit/circuitpython_kernel/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

5.1.4 Write Documentation

CircuitPython Kernel could always use more documentation, whether as part of the official CircuitPython Kernel docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/adafruit/circuitpython_kernel/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up *circuitpython_kernel* for local development.

1. Fork the *circuitpython_kernel* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/circuitpython_kernel.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv circuitpython_kernel
$ cd circuitpython_kernel/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 circuitpython_kernel tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/adafruit/circuitpython_kernel/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_circuitpython_kernel
```

CHAPTER 6

Credits

6.1 Development Lead

- Carol Willing <carolcode@willingconsulting.com>

6.2 Contributors

- Brent Rubell <brent@adafruit.com>

CHAPTER 7

History

7.1 0.2.0 (2018-02-08)

- update examples

7.2 0.1.0 (2017-03-23)

- First release on PyPI.

CHAPTER 8

Board Preparation

Before you start using the CircuitPython_Kernel, you'll need a board running CircuitPython. If you're not sure if the board plugged into your computer is running CircuitPython, check your file explorer for a drive named `CIRCUITPY`

8.1 Designed for CircuitPython (SAMD21 and SAMD51)

8.1.1 Boards Supported:

- Circuit Playground Express
- Feather M0
- Trinket M0
- Metro M0 Express
- Gemma M0
- ItsyBitsy M0
- Metro M4
- ItsyBitsy M4

8.1.2 Installing CircuitPython Firmware

- Download the CircuitPython Firmware (.uf2 file) from the CircuitPython Repo
- Plug in board and double click the **reset** button to enter bootloader mode.
- Drag and drop the *.uf2 CircuitPython file to the USB drive.
- If you see the `CIRCUITPY` as the new name of the USB drive, you're ready to go.

8.2 Adafruit Feather Huzzah ESP8266

While they do work with CircuitPython_Kernel, ESP8266-based boards require a different type of installation and configuration from the boards designed for circuitpython.

8.2.1 Installing CircuitPython Firmware

- python3 -m pip install esptool
- – Download the CircuitPython Firmware (.bin file) from the CircuitPython Repo
- Install the SiLabs CP210x driver
- Erase flash `python3 esptool.py --port /path/to/ESP8266 erase_flash`
- Load firmware: `esptool.py --port /path/to/ESP8266 --baud 460800 write_flash --flash_size=detect 0 firmware.bin`
- Press reset or unplug/plug the board.

8.2.2 Access the REPL

Use `screen` program:

```
screen <device> 115200
```

8.3 ampy

- Install ampy `python3 -m pip install adafruit-ampy`
- To get options for listing files and moving files: `ampy --help`

CHAPTER 9

Indices and tables

- genindex
- modindex
- search