Adafruitfeatherwing Library Documentation

Release 1.0

Scott Shawcroft

Jan 29, 2019

Contents

This library provides FeatherWing specific classes for those that require a significant amount of initialization.

CHAPTER 1

Dependencies

These drivers depends on:

- Adafruit CircuitPython
- INA219
- Seesaw

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the Adafruit library and driver bundle and highly recommended over installing each one.

1.1 Installing from PyPI

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver locally from PyPI. To install for current user:

```
pip3 install adafruit-circuitpython-featherwing
```

To install system-wide (this may be required in some cases):

```
sudo pip3 install adafruit-circuitpython-featherwing
```

To install in a virtual environment in your current project:

```
mkdir project-name && cd project-name
python3 -m venv .env
source .env/bin/activate
pip3 install adafruit-circuitpython-featherwing
```

CHAPTER 2

Contributing

Contributions are welcome! Please read our Code of Conduct before contributing to help this project stay welcoming.

CHAPTER 3

Building locally

To build this library locally you'll need to install the circuitpython-build-tools package.

```
python3 -m venv .env
source .env/bin/activate
pip install circuitpython-build-tools
```

Once installed, make sure you are in the virtual environment:

```
source .env/bin/activate
```

Then run the build:

```
circuitpython-build-bundles --filename_prefix adafruit-circuitpython-featherwing --
↪library_location .
```

3.1 Sphinx documentation

Sphinx is used to build the documentation based on rST files and comments in the code. First, install dependencies (feel free to reuse the virtual environment from above):

```
python3 -m venv .env
source .env/bin/activate
pip install Sphinx sphinx-rtd-theme
```

Now, once you have the virtual environment activated:

```
cd docs
sphinx-build -E -W -b html . _build/html
```

This will output the documentation to `docs/_build/html`. Open the index.html in your browser to view them. It will also (due to -W) error out on any warning like Travis will. This is a good way to locally verify it will pass.

CHAPTER 4

Table of Contents

4.1 Simple tests

Ensure your device works with this simple test.

Listing 1: examples/featherwing_ina219_simpletest.py

```
""" Example to print out the voltage and current using the INA219 """
import time
from adafruit_featherwing import ina219_featherwing

INA219 = ina219_featherwing.INA219FeatherWing()

while True:
    print("Bus Voltage:   {} V".format(INA219.bus_voltage))
    print("Shunt Voltage: {} V".format(INA219.shunt_voltage))
    print("Voltage:       {} V".format(INA219.voltage))
    print("Current:       {} mA".format(INA219.current))
    print("")
    time.sleep(0.5)
```

Listing 2: examples/featherwing_joy_simpletest.py

```
"""This example zeros the joystick, and prints when the joystick moves
   or the buttons are pressed."""
import time
from adafruit_featherwing import joy_featherwing

wing = joy_featherwing.JoyFeatherWing()
last_x = 0
last_y = 0

while True:
    x, y = wing.joystick
```

(continues on next page)

(continued from previous page)

```
    if (abs(x - last_x) > 3) or (abs(y - last_y) > 3):
        last_x = x
        last_y = y
        print(x, y)
    if wing.button_a:
        print("Button A!")
    if wing.button_b:
        print("Button B!")
    if wing.button_x:
        print("Button X!")
    if wing.button_y:
        print("Button Y!")
    if wing.button_select:
        print("Button SELECT!")
    time.sleep(.01)
```

4.2 adafruit_featherwing.ina219_featherwing

Helper for using the INA219 FeatherWing.

- Author(s): Kattni Rembor

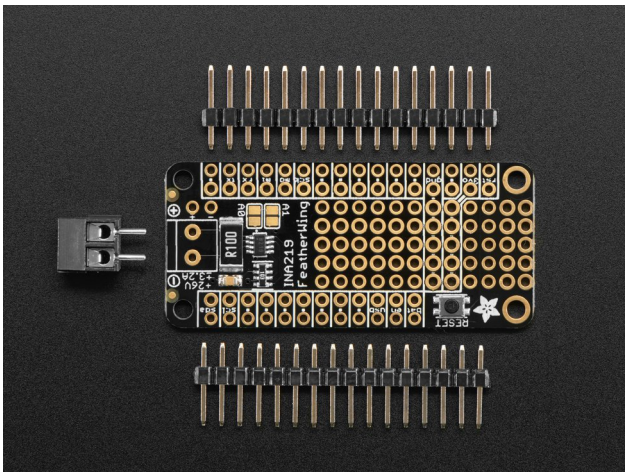
class adafruit_featherwing.ina219_featherwing.**INA219FeatherWing**

Class representing an Adafruit INA219 FeatherWing.

Automatically uses the feather's I2C bus.

bus_voltage

Bus voltage returns volts.



This example prints the bus voltage with the appropriate units.

```
from adafruit_featherwing import ina219_featherwing
import time

ina219 = ina219_featherwing.INA219FeatherWing()

while True:
```

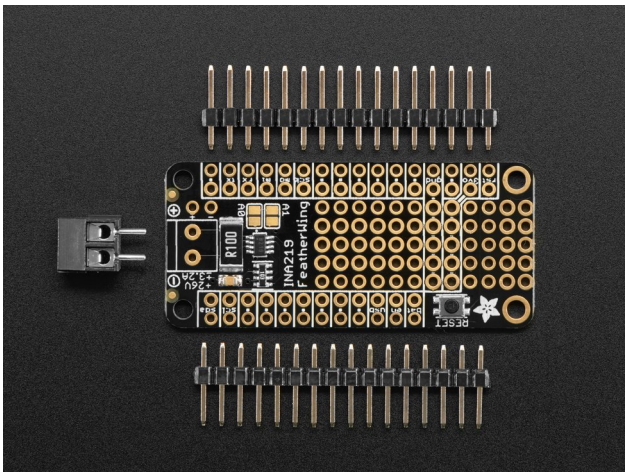
(continues on next page)

(continued from previous page)

```
    print("Bus Voltage: {} V".format(ina219.bus_voltage))
    time.sleep(0.5)
```

current

Current returns mA.



This example prints the current with the appropriate units.

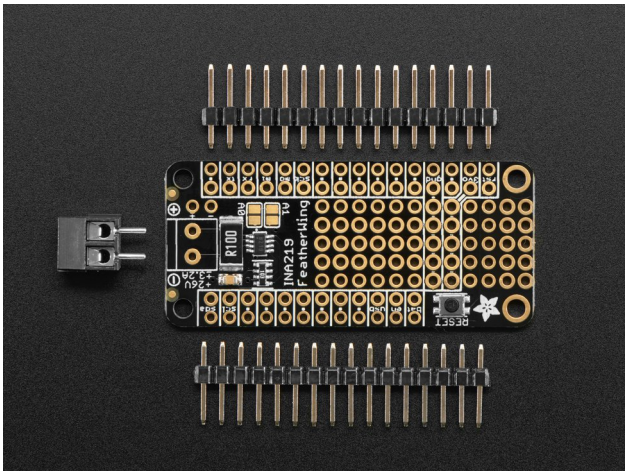
```
from adafruit_featherwing import ina219_featherwing
import time

ina219 = ina219_featherwing.INA219FeatherWing()

while True:
    print("Current: {} mA".format(ina219.current))
    time.sleep(0.5)
```

shunt_voltage

Shunt voltage returns volts.



This example prints the shunt voltage with the appropriate units.

```
from adafruit_featherwing import ina219_featherwing
import time
```

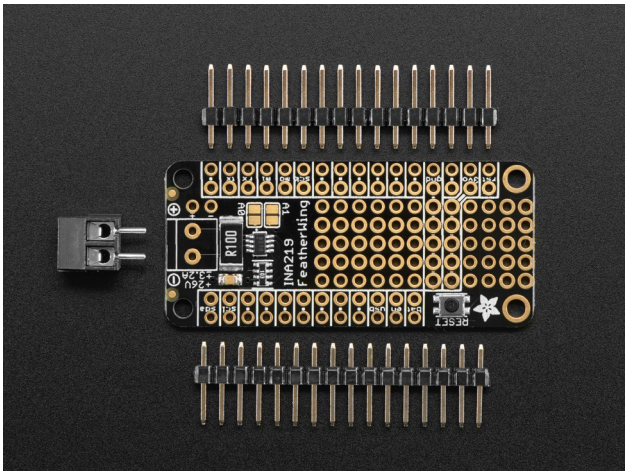
(continues on next page)

(continued from previous page)

```
ina219 = ina219_featherwing.INA219FeatherWing()

while True:
    print("Shunt Voltage: {} V".format(ina219.shunt_voltage))
    time.sleep(0.5)
```

voltage
Voltage, known as load voltage, is bus voltage plus shunt voltage. Returns volts.



This example prints the voltage with the appropriate units.

```
from adafruit_featherwing import ina219_featherwing
import time

ina219 = ina219_featherwing.INA219FeatherWing()

while True:
    print("Voltage: {} V".format(ina219.voltage))
    time.sleep(0.5)
```

4.3 adafruit_featherwing.joy_featherwing

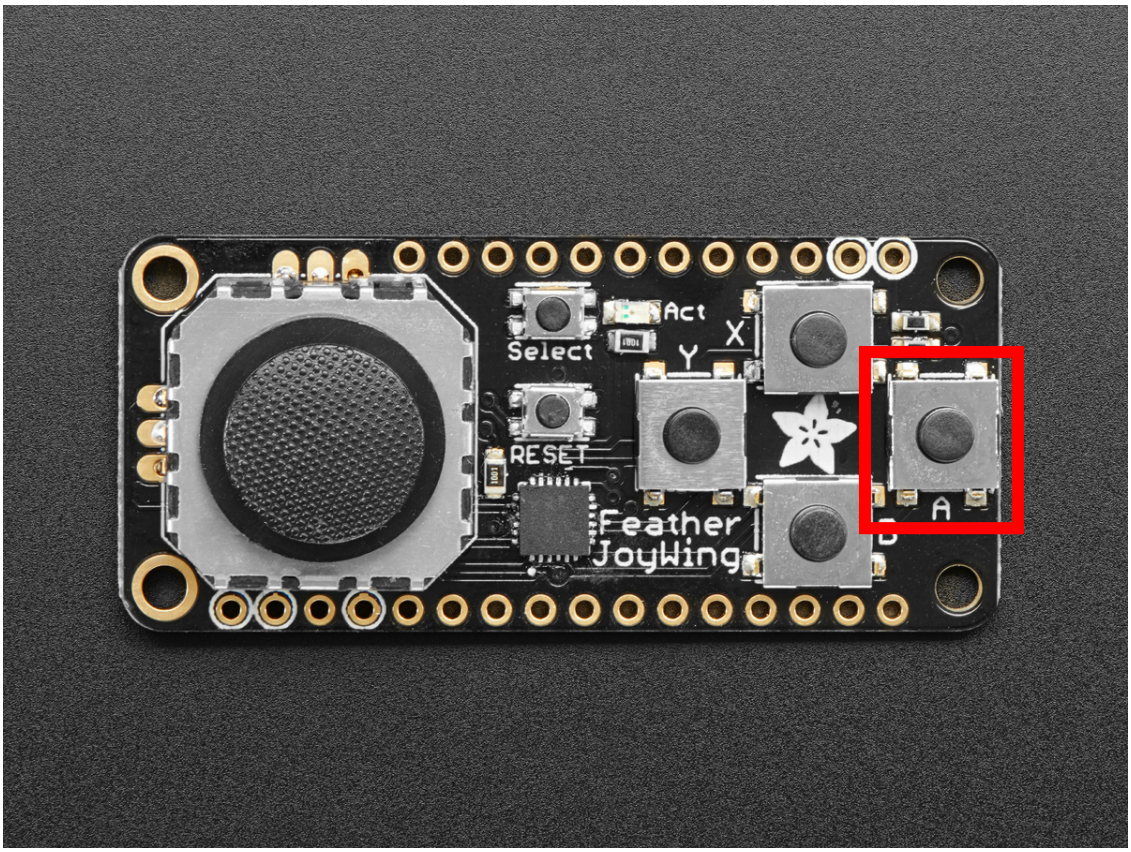
Helper for using the Joy FeatherWing.

- Author(s): Kattni Rembor

class adafruit_featherwing.joy_featherwing.**JoyFeatherWing**
Class representing an Adafruit Joy FeatherWing.

Automatically uses the feather's I2C bus.

button_a
Joy featherwing button A.



This example prints when button A is pressed.

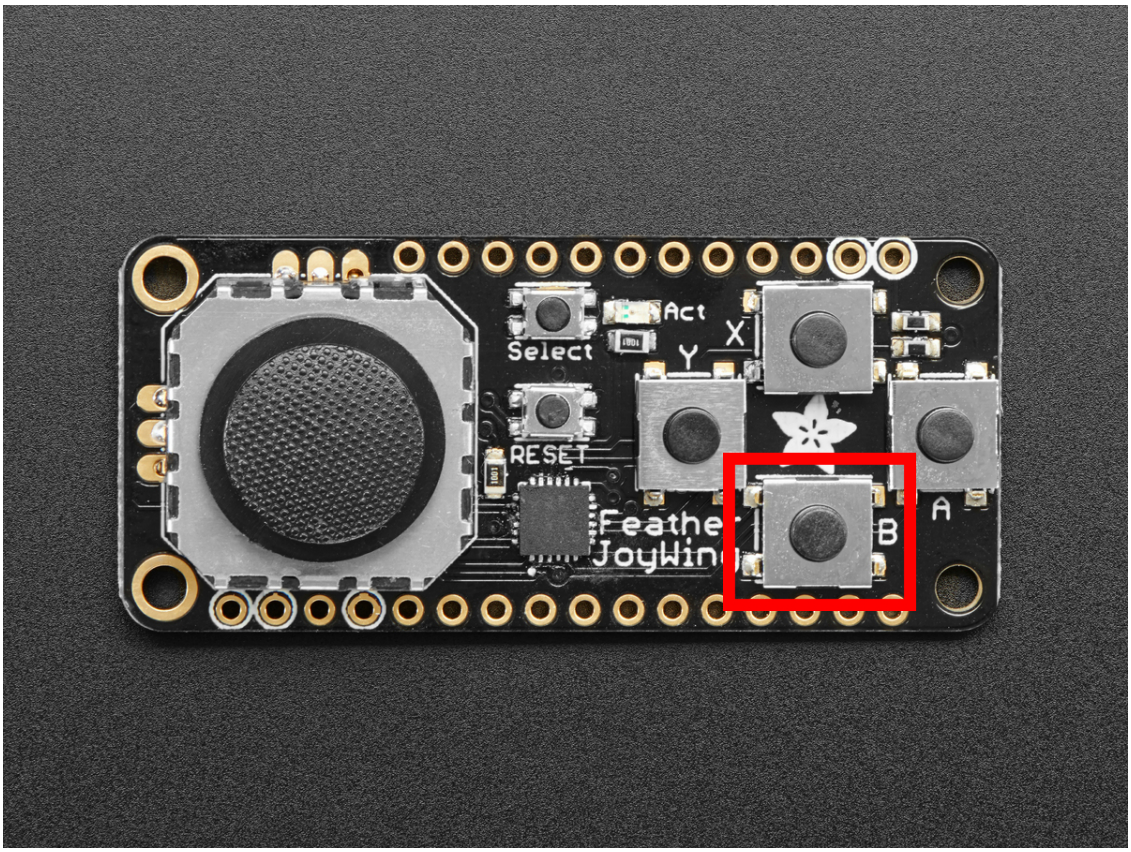
```
from adafruit_featherwing import joy_featherwing
import time

wing = joy_featherwing.JoyFeatherWing()

while True:
    if wing.button_a:
    print("Button A pressed!")
```

button_b

Joy featherwing button B.



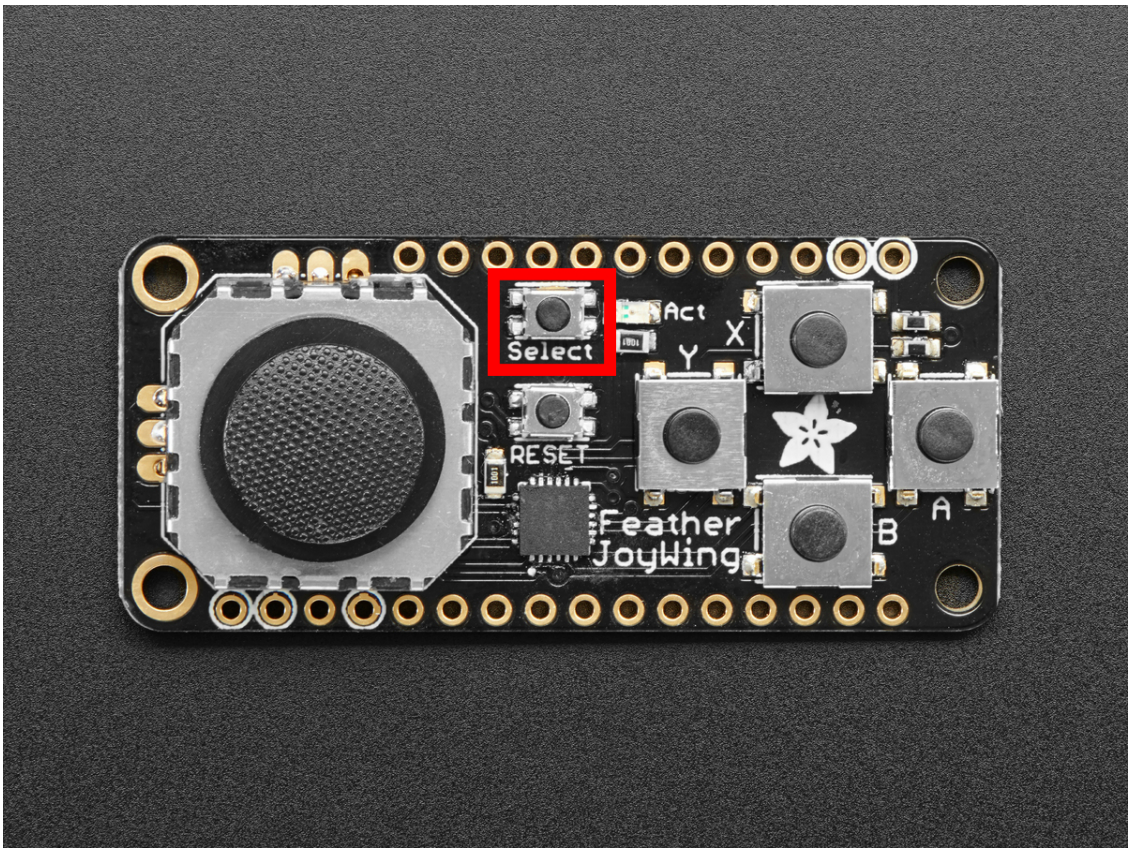
This example prints when button B is pressed.

```
from adafruit_featherwing import joy_featherwing
import time

wing = joy_featherwing.JoyFeatherWing()

while True:
    if wing.button_b:
    print("Button B pressed!")
```

button_select
Joy featherwing button SELECT.



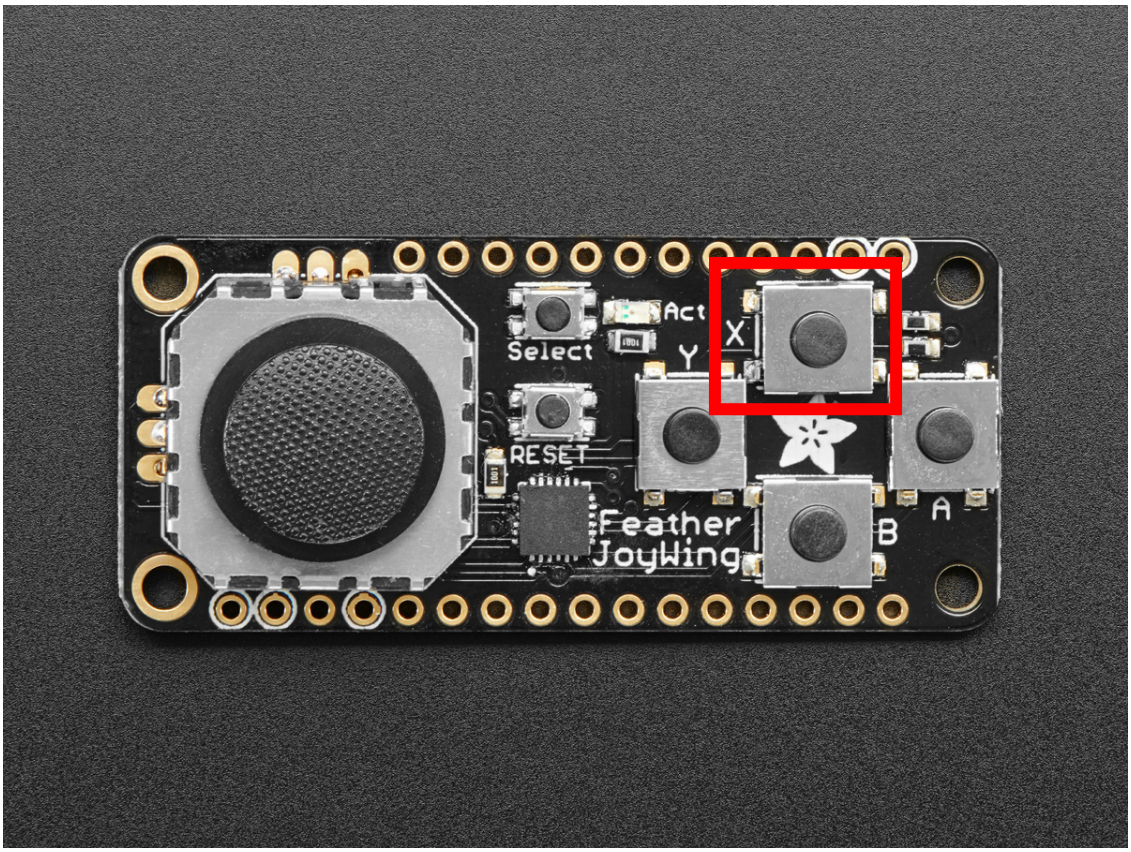
This example prints when button SELECT is pressed.

```
from adafruit_featherwing import joy_featherwing
import time

wing = joy_featherwing.JoyFeatherWing()

while True:
    if wing.button_select:
    print("Button SELECT pressed!")
```

button_x
Joy featherwing button X.



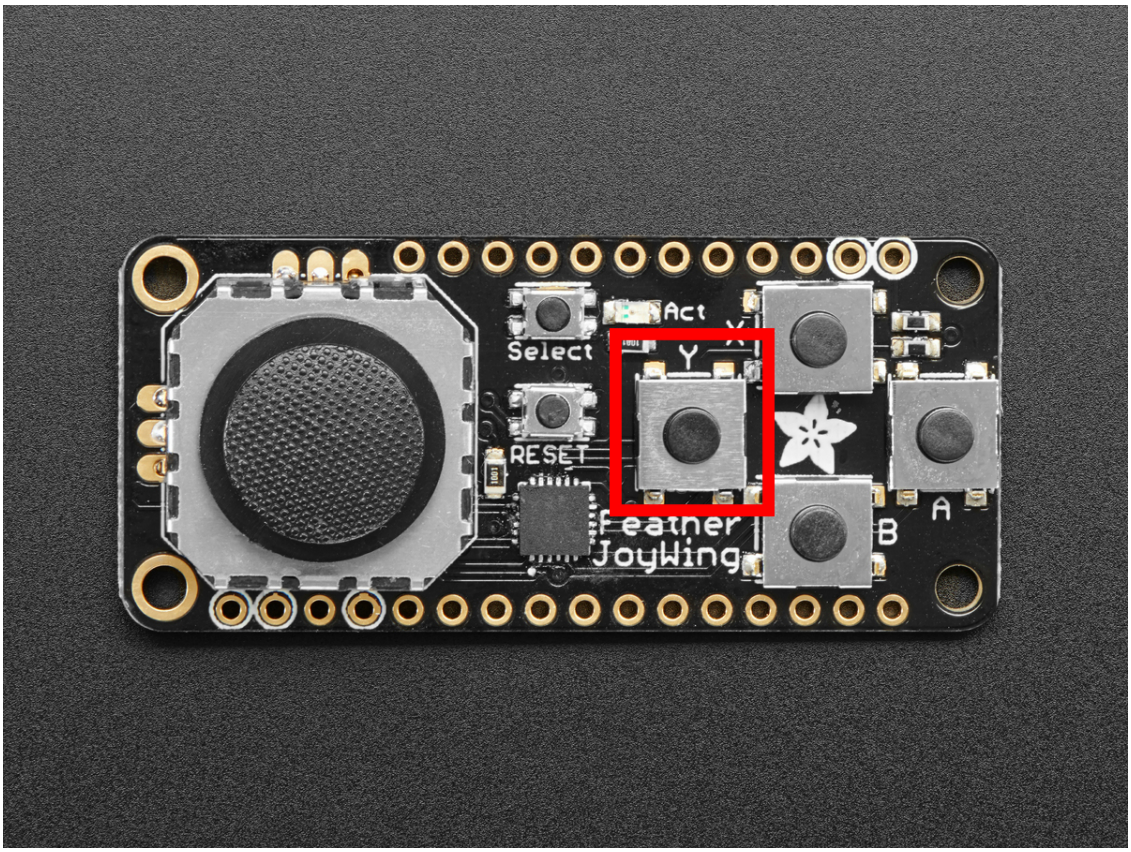
This example prints when button X is pressed.

```
from adafruit_featherwing import joy_featherwing
import time

wing = joy_featherwing.JoyFeatherWing()

while True:
    if wing.button_x:
    print("Button X pressed!")
```

button_y
Joy featherwing button Y.



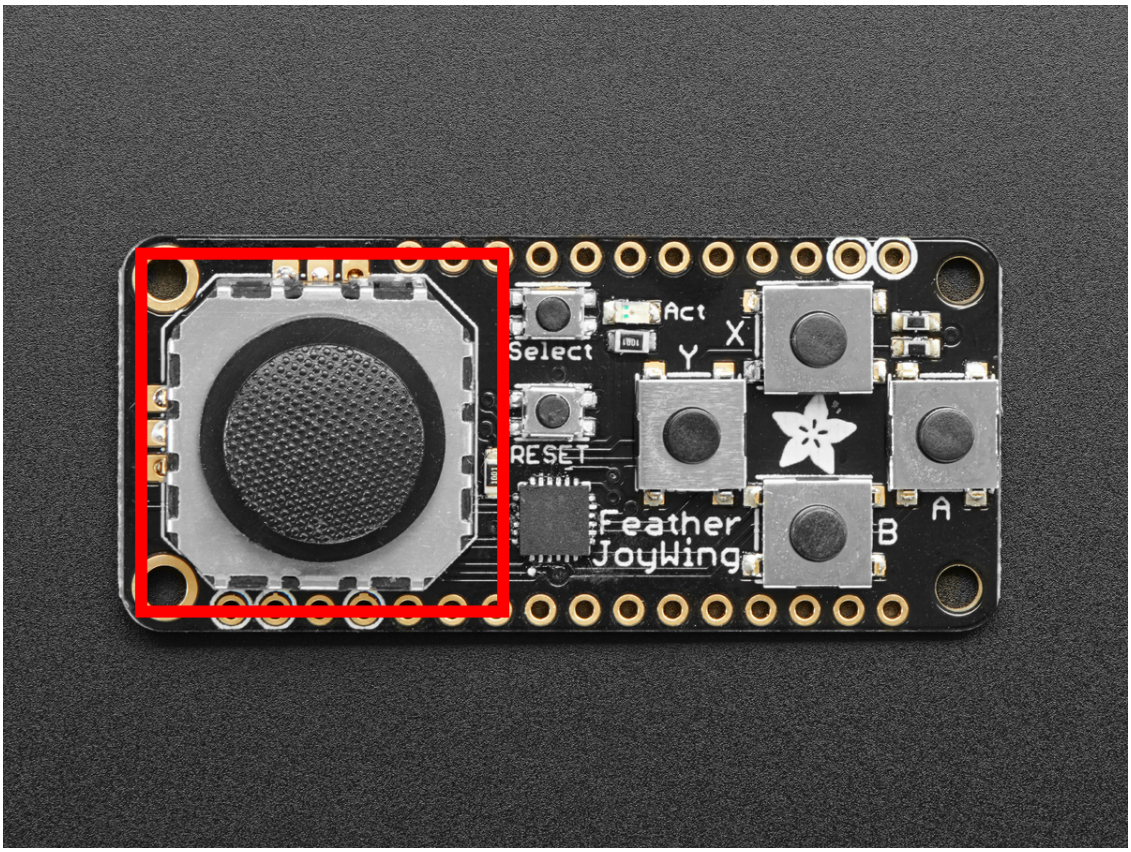
This example prints when button Y is pressed.

```
from adafruit_featherwing import joy_featherwing
import time

wing = joy_featherwing.JoyFeatherWing()

while True:
    if wing.button_y:
    print("Button Y pressed!")
```

joystick
Joy FeatherWing joystick.



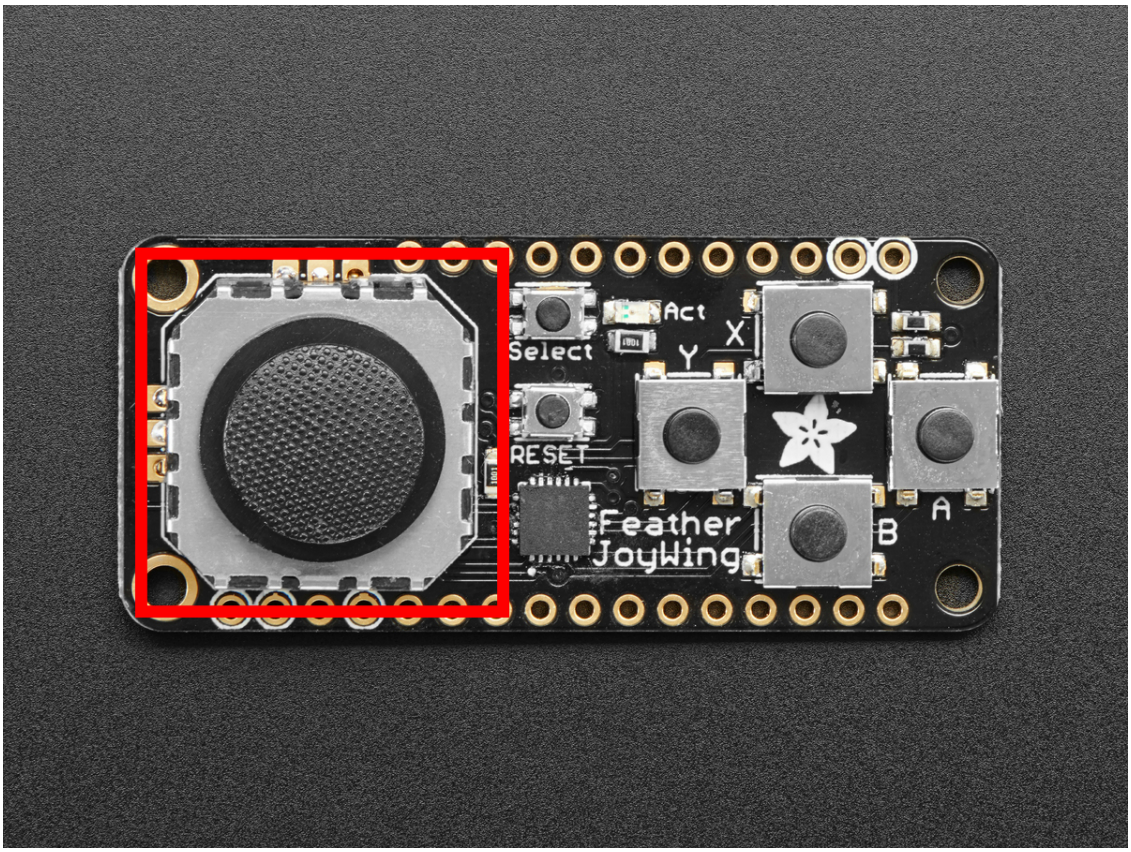
This example zeros the joystick, and prints the coordinates of joystick when it is moved.

```
from adafruit_featherwing import joy_featherwing
import time

wing = joy_featherwing.JoyFeatherWing()
last_x = 0
last_y = 0
wing.zero_joystick()

while True:
    x, y = wing.joystick
    if (abs(x - last_x) > 3) or (abs(y - last_y) > 3):
        last_x = x
        last_y = y
        print(x, y)
    time.sleep(0.01)
```

joystick_offset
Offset used to correctly report (0, 0) when the joystick is centered.



Provide a tuple of (x, y) to set your joystick center to (0, 0). The offset you provide is subtracted from the current reading. For example, if your joystick reads as (-4, 0), you would enter (-4, 0) as the offset. The code will subtract -4 from -4, and 0 from 0, returning (0, 0).

This example supplies an offset for zeroing, and prints the coordinates of the joystick when it is moved.

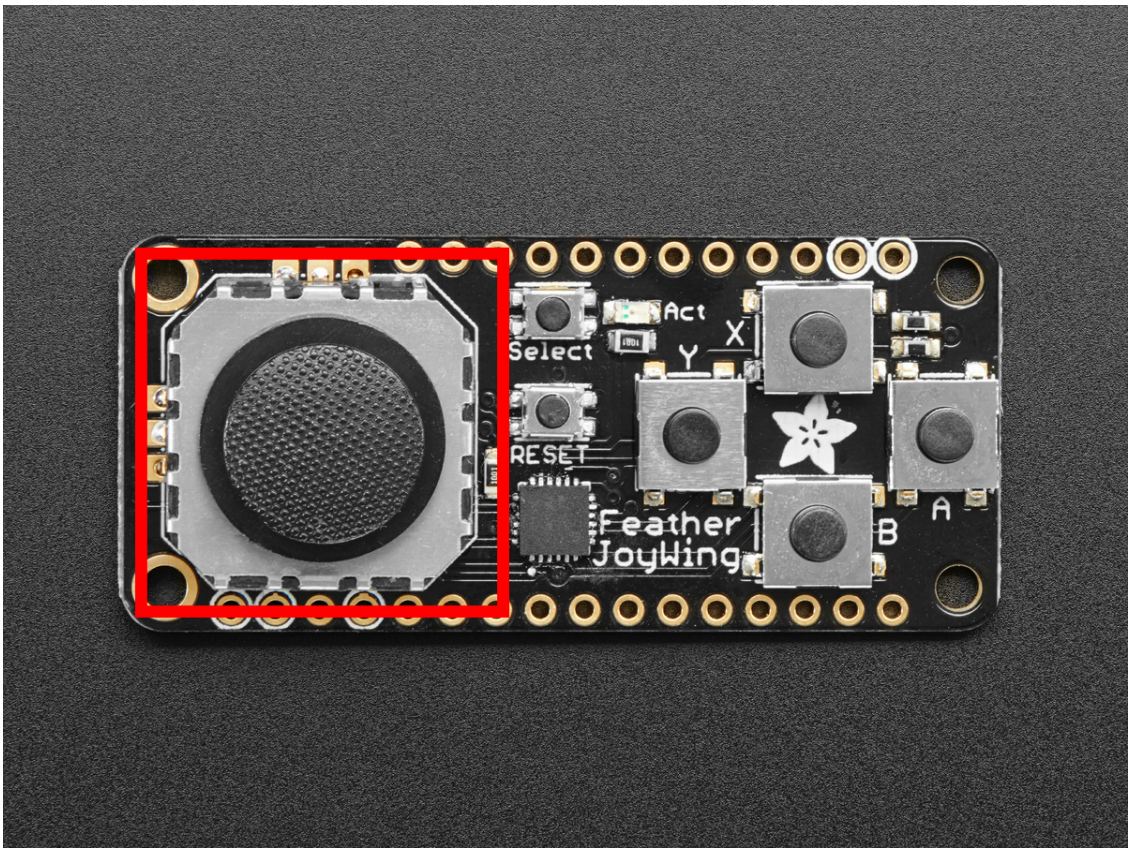
```
from adafruit_featherwing import joy_featherwing
import time

wing = joy_featherwing.JoyFeatherWing()
last_x = 0
last_y = 0

while True:
    wing.joystick_offset = (-4, 0)
    x, y = wing.joystick
    if (abs(x - last_x) > 3) or (abs(y - last_y) > 3):
        last_x = x
        last_y = y
        print(x, y)
    time.sleep(0.01)
```

zero_joystick()

Zeros the joystick by using current reading as (0, 0). Note: You must not be touching the joystick at the time of zeroing for it to be accurate.



This example zeros the joystick, and prints the coordinates of joystick when it is moved.

```
from adafruit_featherwing import joy_featherwing
import time

wing = joy_featherwing.JoyFeatherWing()
last_x = 0
last_y = 0
wing.zero_joystick()

while True:
    x, y = wing.joystick
    if (abs(x - last_x) > 3) or (abs(y - last_y) > 3):
        last_x = x
        last_y = y
        print(x, y)
    time.sleep(0.01)
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index