

---

# **Adafruit HID Library Documentation**

***Release 1.0***

**Scott Shawcroft**

**Jun 01, 2018**



---

## Contents

---

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Dependencies</b>  | <b>3</b>  |
| <b>2</b> | <b>Usage Example</b>                                       | <b>5</b>  |
| <b>3</b> | <b>Contributing</b>  | <b>9</b>  |
| <b>4</b> | <b>Building locally</b>                                    | <b>11</b> |
| 4.1      | Sphinx documentation . . . . .                             | 11        |
| <b>5</b> | <b>Table of Contents</b>                                   | <b>13</b> |
| 5.1      | Simple test . . . . .                                      | 13        |
| 5.2      | adafruit_hid.keyboard.Keyboard . . . . .                   | 14        |
| 5.3      | adafruit_hid.keycode.Keycode . . . . .                     | 15        |
| 5.4      | adafruit_hid.keyboard_layout_us.KeyboardLayoutUS . . . . . | 21        |
| 5.5      | adafruit_hid.mouse.Mouse . . . . .                         | 22        |
| <b>6</b> | <b>Indices and tables</b>                                  | <b>25</b> |
|          | <b>Python Module Index</b>                                 | <b>27</b> |



This driver simulates USB HID devices. Currently keyboard and mouse are implemented.



# CHAPTER 1

---

## Dependencies

---

This driver depends on:

- [Adafruit CircuitPython](#)

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the [Adafruit library and driver bundle](#).





## CHAPTER 2

---

### Usage Example

---

The Keyboard class sends keypress reports for a USB keyboard device to the host.

The Keycode class defines USB HID keycodes to send using Keyboard.

```
from adafruit_hid.keyboard import Keyboard
from adafruit_hid.keycode import Keycode

# Set up a keyboard device.
kbd = Keyboard()

# Type lowercase 'a'. Presses the 'a' key and releases it.
kbd.send(Keycode.A)

# Type capital 'A'.
kbd.send(Keycode.SHIFT, Keycode.A)

# Type control-x.
kbd.send(Keycode.CONTROL, Keycode.X)

# You can also control press and release actions separately.
kbd.press(Keycode.CONTROL, Keycode.X)
kbd.release_all()

# Press and hold the shifted '1' key to get '!' (exclamation mark).
kbd.press(Keycode.SHIFT, Keycode.ONE)
# Release the ONE key and send another report.
kbd.release(Keycode.ONE)
# Press shifted '2' to get '@'.
kbd.press(Keycode.TWO)
# Release all keys.
kbd.release_all()
```

The KeyboardLayoutUS sends ASCII characters using keypresses. It assumes the host is set to accept keypresses from a US keyboard.

If the host is expecting a non-US keyboard, the character to key mapping provided by KeyboardLayoutUS will

not always be correct. Different keypresses will be needed in some cases. For instance, to type an 'A' on a French keyboard (AZERTY instead of QWERTY), `Keycode.Q` should be pressed.

Currently this package provides only `KeyboardLayoutUS`. More `KeyboardLayout` classes could be added to handle non-US keyboards and the different input methods provided by various operating systems.

```
from adafruit_hid.keyboard import Keyboard
from adafruit_hid.keyboard_layout_us import KeyboardLayoutUS

kbd = Keyboard()
layout = KeyboardLayoutUS(kbd)

# Type 'abc' followed by Enter (a newline).
layout.write('abc\n')

# Get the keycodes needed to type a '$'.
# The method will return (Keycode.SHIFT, Keycode.FOUR).
keycodes = layout.keycodes('$')
```

The `Mouse` class simulates a three-button mouse with a scroll wheel.

```
from adafruit_hid.mouse import Mouse

m = Mouse()

# Click the left mouse button.
m.click(Mouse.LEFT_BUTTON)

# Move the mouse diagonally to the upper left.
m.move(-100, -100, 0)

# Roll the mouse wheel away from the user one unit.
# Amount scrolled depends on the host.
m.move(0, 0, -1)

# Keyword arguments may also be used. Omitted arguments default to 0.
m.move(x=-100, y=-100)
m.move(wheel=-1)

# Move the mouse while holding down the left button. (click-drag).
m.press(Mouse.LEFT_BUTTON)
m.move(x=50, y=20)
m.release_all()          # or m.release(Mouse.LEFT_BUTTON)
```

The `ConsumerControl` class emulates consumer control devices such as remote controls, or the multimedia keys on certain keyboards.

*New in CircuitPython 3.0.*

```
from adafruit_hid.consumer_control import ConsumerControl
from adafruit_hid.consumer_control_code import ConsumerControlCode

cc = ConsumerControl()

# Raise volume.
cc.send(ConsumerControlCode.VOLUME_INCREMENT)

# Pause or resume playback.
cc.send(ConsumerControlCode.PLAY_PAUSE)
```

The Gamepad class emulates a two-joystick gamepad with 16 buttons.

*New in CircuitPython 3.0.*

```
from adafruit_hid.gamepad import Gamepad

gp = Gamepad()

# Click gamepad buttons.
gp.click_buttons(1, 7)

# Move joysticks.
gp.move_joysticks(x=2, y=0, z=-20)
```



## CHAPTER 3

---

### Contributing

---

Contributions are welcome! Please read our [Code of Conduct](#) before contributing to help this project stay welcoming.



## CHAPTER 4

---

### 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-hid --library_
↪location .
```

### 4.1 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.





## 5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/keyboard\_shortcuts.py

```
1 import time
2 from adafruit_hid.keyboard import Keyboard
3 from adafruit_hid.keycode import Keycode
4 import board
5 import digitalio
6
7 kbd = Keyboard()
8
9 # define buttons. these can be any physical switches/buttons, but the values
10 # here work out-of-the-box with a CircuitPlayground Express' A and B buttons.
11 swap = digitalio.DigitalInOut(board.D4)
12 swap.direction = digitalio.Direction.INPUT
13 swap.pull = digitalio.Pull.DOWN
14
15 search = digitalio.DigitalInOut(board.D5)
16 search.direction = digitalio.Direction.INPUT
17 search.pull = digitalio.Pull.DOWN
18
19 while True:
20     # press ALT+TAB to swap windows
21     if swap.value:
22         kbd.send(Keycode.ALT, Keycode.TAB)
23
24     # press CTRL+K, which in a web browser will open the search dialog
25     elif search.value:
26         kbd.send(Keycode.CONTROL, Keycode.K)
27
```

(continues on next page)

(continued from previous page)

```
28 time.sleep(0.1)
```

Listing 2: examples/scroll.py

```

1  import time
2  from adafruit_hid.mouse import Mouse
3  import board
4  import digitalio
5
6  mouse = Mouse()
7
8  # define buttons. these can be any physical switches/buttons, but the values
9  # here work out-of-the-box with a CircuitPlayground Express' A and B buttons.
10 up = digitalio.DigitalInOut(board.D4)
11 up.direction = digitalio.Direction.INPUT
12 up.pull = digitalio.Pull.DOWN
13
14 down = digitalio.DigitalInOut(board.D5)
15 down.direction = digitalio.Direction.INPUT
16 down.pull = digitalio.Pull.DOWN
17
18 while True:
19     # scroll up one unit (varies with host/OS)
20     if up.value:
21         mouse.move(wheel=1)
22
23     # scroll down one unit (varies with host/OS)
24     elif down.value:
25         mouse.move(wheel=-1)
26
27     time.sleep(0.1)
```

## 5.2 adafruit\_hid.keyboard.Keyboard

- Author(s): Scott Shawcroft, Dan Halbert

**class** adafruit\_hid.keyboard.Keyboard

Send HID keyboard reports.

**press** (\*keycodes)

Send a report indicating that the given keys have been pressed.

**Parameters** **keycodes** – Press these keycodes all at once.

**Raises** **ValueError** – if more than six regular keys are pressed.

Keycodes may be modifiers or regular keys. No more than six regular keys may be pressed simultaneously.

Examples:

```

from adafruit_hid.keycode import Keycode

# Press ctrl-x.
kbd.press(Keycode.LEFT_CONTROL, Keycode.X)

# Or, more conveniently, use the CONTROL alias for LEFT_CONTROL:
```

(continues on next page)

(continued from previous page)

```
kbd.press(Keycode.CONTROL, Keycode.X)

# Press a, b, c keys all at once.
kbd.press(Keycode.A, Keycode.B, Keycode.C)
```

**release (\*keycodes)**

Send a USB HID report indicating that the given keys have been released.

**Parameters** **keycodes** – Release these keycodes all at once.

If a keycode to be released was not pressed, it is ignored.

Example:

```
# release SHIFT key
kbd.release(Keycode.SHIFT)
```

**release\_all()**

Release all pressed keys.

**send (\*keycodes)**

Press the given keycodes and then release all pressed keys.

**Parameters** **keycodes** – keycodes to send together

## 5.3 adafruit\_hid.keycode.Keycode

- Author(s): Scott Shawcroft, Dan Halbert

### **class** adafruit\_hid.keycode.Keycode

USB HID Keycode constants.

This list is modeled after the names for USB keycodes defined in [http://www.usb.org/developers/hidpage/Hut1\\_12v2.pdf#page=58](http://www.usb.org/developers/hidpage/Hut1_12v2.pdf#page=58). This list does not include every single code, but does include all the keys on a regular PC or Mac keyboard.

Remember that keycodes are the names for key *positions* on a US keyboard, and may not correspond to the character that you mean to send if you want to emulate non-US keyboard. For instance, on a French keyboard (AZERTY instead of QWERTY), the keycode for ‘q’ is used to indicate an ‘a’. Likewise, ‘y’ represents ‘z’ on a German keyboard. This is historical: the idea was that the keycaps could be changed without changing the keycodes sent, so that different firmware was not needed for different variations of a keyboard.

**A = 4**

a and A

**ALT = 226**

Alias for LEFT\_ALT; Alt is also known as Option (Mac)

**APPLICATION = 101**

Application: also known as the Menu key (Windows)

**B = 5**

b and B

**BACKSLASH = 49**

\ and |

**BACKSPACE = 42**

Delete backward (Backspace)

**C = 6**  
c and C

**CAPS\_LOCK = 57**  
Caps Lock

**COMMA = 54**  
, and <

**CONTROL = 224**  
Alias for LEFT\_CONTROL

**D = 7**  
d and D

**DELETE = 76**  
Delete forward

**DOWN\_ARROW = 81**  
Move the cursor down

**E = 8**  
e and E

**EIGHT = 37**  
8 and \*

**END = 77**  
End (often moves to end of line)

**ENTER = 40**  
Enter (Return)

**EQUALS = 46**  
= and ` +

**ESCAPE = 41**  
Escape

**F = 9**  
f and F

**F1 = 58**  
Function key F1

**F10 = 67**  
Function key F10

**F11 = 68**  
Function key F11

**F12 = 69**  
Function key F12

**F13 = 104**  
Function key F13 (Mac)

**F14 = 105**  
Function key F14 (Mac)

**F15 = 106**  
Function key F15 (Mac)

**F16 = 107**  
Function key F16 (Mac)

**F17 = 108**  
Function key F17 (Mac)

**F18 = 109**  
Function key F18 (Mac)

**F19 = 110**  
Function key F19 (Mac)

**F2 = 59**  
Function key F2

**F3 = 60**  
Function key F3

**F4 = 61**  
Function key F4

**F5 = 62**  
Function key F5

**F6 = 63**  
Function key F6

**F7 = 64**  
Function key F7

**F8 = 65**  
Function key F8

**F9 = 66**  
Function key F9

**FIVE = 34**  
5 and %

**FORWARD\_SLASH = 56**  
/ and ?

**FOUR = 33**  
4 and \$

**G = 10**  
g and G

**GRAVE\_ACCENT = 53**  
` and ~

**GUI = 227**  
Alias for LEFT\_GUI; GUI is also known as the Windows key, Command (Mac), or Meta

**H = 11**  
h and H

**HOME = 74**  
Home (often moves to beginning of line)

**I = 12**  
i and I

**INSERT = 73**  
Insert

**J = 13**  
j and J

**K = 14**  
k and K

**KEYPAD\_ASTERISK = 85**  
Keypad \*

**KEYPAD\_BACKSLASH = 100**  
Keypad \ and | (Non-US)

**KEYPAD\_EIGHT = 96**  
Keypad 8 and Up Arrow

**KEYPAD\_ENTER = 88**  
Keypad Enter

**KEYPAD\_EQUALS = 103**  
Keypad = (Mac)

**KEYPAD\_FIVE = 93**  
Keypad 5

**KEYPAD\_FORWARD\_SLASH = 84**  
Keypad /

**KEYPAD\_FOUR = 92**  
Keypad 4 and Left Arrow

**KEYPAD\_MINUS = 86**  
Keypad -

**KEYPAD\_NINE = 97**  
Keypad 9 and PgUp

**KEYPAD\_NUMLOCK = 83**  
Num Lock (Clear on Mac)

**KEYPAD\_ONE = 89**  
Keypad 1 and End

**KEYPAD\_PERIOD = 99**  
Keypad . and Del

**KEYPAD\_PLUS = 87**  
Keypad +

**KEYPAD\_SEVEN = 95**  
Keypad 7 and Home

**KEYPAD\_SIX = 94**  
Keypad 6 and Right Arrow

**KEYPAD\_THREE = 91**  
Keypad 3 and PgDn

**KEYPAD\_TWO = 90**  
Keypad 2 and Down Arrow

**KEYPAD\_ZERO = 98**  
Keypad 0 and Ins

**L = 15**  
l and L

**LEFT\_ALT = 226**  
Alt modifier left of the spacebar

**LEFT\_ARROW = 80**  
Move the cursor left

**LEFT\_BRACKET = 47**  
[ and {

**LEFT\_CONTROL = 224**  
Control modifier left of the spacebar

**LEFT\_GUI = 227**  
GUI modifier left of the spacebar

**LEFT\_SHIFT = 225**  
Shift modifier left of the spacebar

**M = 16**  
m and M

**MINUS = 45**  
-` and ``\_

**N = 17**  
n and N

**NINE = 38**  
9 and (

**O = 18**  
o and O

**ONE = 30**  
1 and !

**P = 19**  
p and P

**PAGE\_DOWN = 78**  
Go forward one page

**PAGE\_UP = 75**  
Go back one page

**PAUSE = 72**  
Pause (Break)

**PERIOD = 55**  
. and >

**POUND = 50**  
# and ~ (Non-US keyboard)

**POWER = 102**  
Power (Mac)

**PRINT\_SCREEN = 70**  
Print Screen (SysRq)

**Q = 20**  
q and Q

**QUOTE = 52**  
' and "

**R = 21**  
r and R

**RETURN = 40**  
Alias for ENTER

**RIGHT\_ALT = 230**  
Alt modifier right of the spacebar

**RIGHT\_ARROW = 79**  
Move the cursor right

**RIGHT\_BRACKET = 48**  
] and }

**RIGHT\_CONTROL = 228**  
Control modifier right of the spacebar

**RIGHT\_GUI = 231**  
GUI modifier right of the spacebar

**RIGHT\_SHIFT = 229**  
Shift modifier right of the spacebar

**S = 22**  
s and S

**SCROLL\_LOCK = 71**  
Scroll Lock

**SEMICOLON = 51**  
; and :

**SEVEN = 36**  
7 and &

**SHIFT = 225**  
Alias for LEFT\_SHIFT

**SIX = 35**  
6 and ^

**SPACE = 44**  
Alias for SPACEBAR

**SPACEBAR = 44**  
Spacebar

**T = 23**  
t and T

**TAB = 43**  
Tab and Backtab



**THREE = 32**

3 and #

**TWO = 31**

2 and @

**U = 24**

u and U

**UP\_ARROW = 82**

Move the cursor up

**V = 25**

v and V

**W = 26**

w and W

**X = 27**

x and X

**Y = 28**

y and Y

**Z = 29**

z and Z

**ZERO = 39**

0 and )

**classmethod modifier\_bit** (*keycode*)

Return the modifier bit to be set in an HID keycode report if this is a modifier key; otherwise return 0.

## 5.4 adafruit\_hid.keyboard\_layout\_us.KeyboardLayoutUS

- Author(s): Dan Halbert

**class** `adafruit_hid.keyboard_layout_us.KeyboardLayoutUS` (*keyboard*)

Map ASCII characters to appropriate keypresses on a standard US PC keyboard.

Non-ASCII characters and most control characters will raise an exception.

**keycodes** (*char*)

Return a tuple of keycodes needed to type the given character.

**Parameters** **char** (*str of length one.*) – A single ASCII character in a string.

**Returns** tuple of Keycode keycodes.

**Raises** **ValueError** – if *char* is not ASCII or there is no keycode for it.

Examples:

```
# Returns (Keycode.TAB,)
keycodes(' ')
# Returns (Keycode.A,)
keycode('a')
# Returns (Keycode.SHIFT, Keycode.A)
keycode('A')
# Raises ValueError because it's a accented e and is not ASCII
keycode('é')
```

**write** (*string*)

Type the string by pressing and releasing keys on my keyboard.

**Parameters** **string** – A string of ASCII characters.

**Raises** **ValueError** – if any of the characters are not ASCII or have no keycode (such as some control characters).

Example:

```
# Write abc followed by Enter to the keyboard
layout.write('abc\n')
```

## 5.5 adafruit\_hid.mouse.Mouse

- Author(s): Dan Halbert

**class** adafruit\_hid.mouse.**Mouse**

Send USB HID mouse reports.

**LEFT\_BUTTON** = 1

Left mouse button.

**MIDDLE\_BUTTON** = 4

Middle mouse button.

**RIGHT\_BUTTON** = 2

Right mouse button.

**click** (*buttons*)

Press and release the given mouse buttons.

**Parameters** **buttons** – a bitwise-or'd combination of **LEFT\_BUTTON**, **MIDDLE\_BUTTON**, and **RIGHT\_BUTTON**.

Examples:

```
# Click the left button.
m.click(Mouse.LEFT_BUTTON)

# Double-click the left button.
m.click(Mouse.LEFT_BUTTON)
m.click(Mouse.LEFT_BUTTON)
```

**move** (*x=0, y=0, wheel=0*)

Move the mouse and turn the wheel as directed.

**Parameters**

- **x** – Move the mouse along the x axis. Negative is to the left, positive is to the right.
- **y** – Move the mouse along the y axis. Negative is upwards on the display, positive is downwards.
- **wheel** – Rotate the wheel this amount. Negative is toward the user, positive is away from the user. The scrolling effect depends on the host.

Examples:

```
# Move 100 to the left. Do not move up and down. Do not roll the scroll wheel.
m.move(-100, 0, 0)
# Same, with keyword arguments.
m.move(x=-100)

# Move diagonally to the upper right.
m.move(50, 20)
# Same.
m.move(x=50, y=20)

# Roll the mouse wheel away from the user.
m.move(wheel=1)
```

**press (buttons)**

Press the given mouse buttons.

**Parameters buttons** – a bitwise-or'd combination of `LEFT_BUTTON`, `MIDDLE_BUTTON`, and `RIGHT_BUTTON`.

Examples:

```
# Press the left button.
m.press(Mouse.LEFT_BUTTON)

# Press the left and right buttons simultaneously.
m.press(Mouse.LEFT_BUTTON | Mouse.RIGHT_BUTTON)
```

**release (buttons)**

Release the given mouse buttons.

**Parameters buttons** – a bitwise-or'd combination of `LEFT_BUTTON`, `MIDDLE_BUTTON`, and `RIGHT_BUTTON`.

**release\_all ()**

Release all the mouse buttons.



## CHAPTER 6

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### a

`adafruit_hid.keyboard`, [14](#)  
`adafruit_hid.keyboard_layout_us`, [21](#)  
`adafruit_hid.keycode`, [15](#)  
`adafruit_hid.mouse`, [22](#)





## A

A (adafruit\_hid.keycode.Keycode attribute), 15  
adafruit\_hid.keyboard (module), 14  
adafruit\_hid.keyboard\_layout\_us (module), 21  
adafruit\_hid.keycode (module), 15  
adafruit\_hid.mouse (module), 22  
ALT (adafruit\_hid.keycode.Keycode attribute), 15  
APPLICATION (adafruit\_hid.keycode.Keycode attribute), 15

## B

B (adafruit\_hid.keycode.Keycode attribute), 15  
BACKSLASH (adafruit\_hid.keycode.Keycode attribute), 15  
BACKSPACE (adafruit\_hid.keycode.Keycode attribute), 15

## C

C (adafruit\_hid.keycode.Keycode attribute), 16  
CAPS\_LOCK (adafruit\_hid.keycode.Keycode attribute), 16  
click() (adafruit\_hid.mouse.Mouse method), 22  
COMMA (adafruit\_hid.keycode.Keycode attribute), 16  
CONTROL (adafruit\_hid.keycode.Keycode attribute), 16

## D

D (adafruit\_hid.keycode.Keycode attribute), 16  
DELETE (adafruit\_hid.keycode.Keycode attribute), 16  
DOWN\_ARROW (adafruit\_hid.keycode.Keycode attribute), 16

## E

E (adafruit\_hid.keycode.Keycode attribute), 16  
EIGHT (adafruit\_hid.keycode.Keycode attribute), 16  
END (adafruit\_hid.keycode.Keycode attribute), 16  
ENTER (adafruit\_hid.keycode.Keycode attribute), 16  
EQUALS (adafruit\_hid.keycode.Keycode attribute), 16  
ESCAPE (adafruit\_hid.keycode.Keycode attribute), 16

## F

F (adafruit\_hid.keycode.Keycode attribute), 16  
F1 (adafruit\_hid.keycode.Keycode attribute), 16  
F10 (adafruit\_hid.keycode.Keycode attribute), 16  
F11 (adafruit\_hid.keycode.Keycode attribute), 16  
F12 (adafruit\_hid.keycode.Keycode attribute), 16  
F13 (adafruit\_hid.keycode.Keycode attribute), 16  
F14 (adafruit\_hid.keycode.Keycode attribute), 16  
F15 (adafruit\_hid.keycode.Keycode attribute), 16  
F16 (adafruit\_hid.keycode.Keycode attribute), 16  
F17 (adafruit\_hid.keycode.Keycode attribute), 17  
F18 (adafruit\_hid.keycode.Keycode attribute), 17  
F19 (adafruit\_hid.keycode.Keycode attribute), 17  
F2 (adafruit\_hid.keycode.Keycode attribute), 17  
F3 (adafruit\_hid.keycode.Keycode attribute), 17  
F4 (adafruit\_hid.keycode.Keycode attribute), 17  
F5 (adafruit\_hid.keycode.Keycode attribute), 17  
F6 (adafruit\_hid.keycode.Keycode attribute), 17  
F7 (adafruit\_hid.keycode.Keycode attribute), 17  
F8 (adafruit\_hid.keycode.Keycode attribute), 17  
F9 (adafruit\_hid.keycode.Keycode attribute), 17  
FIVE (adafruit\_hid.keycode.Keycode attribute), 17  
FORWARD\_SLASH (adafruit\_hid.keycode.Keycode attribute), 17  
FOUR (adafruit\_hid.keycode.Keycode attribute), 17

## G

G (adafruit\_hid.keycode.Keycode attribute), 17  
GRAVE\_ACCENT (adafruit\_hid.keycode.Keycode attribute), 17  
GUI (adafruit\_hid.keycode.Keycode attribute), 17

## H

H (adafruit\_hid.keycode.Keycode attribute), 17  
HOME (adafruit\_hid.keycode.Keycode attribute), 17

## I

I (adafruit\_hid.keycode.Keycode attribute), 17  
INSERT (adafruit\_hid.keycode.Keycode attribute), 17

## J

J (adafruit\_hid.keycode.Keycode attribute), 18

## K

K (adafruit\_hid.keycode.Keycode attribute), 18

Keyboard (class in adafruit\_hid.keyboard), 14

KeyboardLayoutUS (class in adafruit\_hid.keyboard\_layout\_us), 21

Keycode (class in adafruit\_hid.keycode), 15

keycodes() (adafruit\_hid.keyboard\_layout\_us.KeyboardLayoutUS method), 21

KEYPAD\_ASTERISK (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_BACKSLASH (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_EIGHT (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_ENTER (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_EQUALS (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_FIVE (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_FORWARD\_SLASH (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_FOUR (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_MINUS (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_NINE (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_NUMLOCK (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_ONE (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_PERIOD (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_PLUS (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_SEVEN (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_SIX (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_THREE (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_TWO (adafruit\_hid.keycode.Keycode attribute), 18

KEYPAD\_ZERO (adafruit\_hid.keycode.Keycode attribute), 18

## L

L (adafruit\_hid.keycode.Keycode attribute), 19

LEFT\_ALT (adafruit\_hid.keycode.Keycode attribute), 19

LEFT\_ARROW (adafruit\_hid.keycode.Keycode attribute), 19

LEFT\_BRACKET (adafruit\_hid.keycode.Keycode attribute), 19

LEFT\_BUTTON (adafruit\_hid.mouse.Mouse attribute), 22

LEFT\_CONTROL (adafruit\_hid.keycode.Keycode attribute), 19

LEFT\_GUI (adafruit\_hid.keycode.Keycode attribute), 19

LEFT\_SHIFT (adafruit\_hid.keycode.Keycode attribute), 19

## M

M (adafruit\_hid.keycode.Keycode attribute), 19

MIDDLE\_BUTTON (adafruit\_hid.mouse.Mouse attribute), 22

MINUS (adafruit\_hid.keycode.Keycode attribute), 19

modifier\_bit() (adafruit\_hid.keycode.Keycode class method), 21

Mouse (class in adafruit\_hid.mouse), 22

move() (adafruit\_hid.mouse.Mouse method), 22

## N

N (adafruit\_hid.keycode.Keycode attribute), 19

NINE (adafruit\_hid.keycode.Keycode attribute), 19

## O

O (adafruit\_hid.keycode.Keycode attribute), 19

ONE (adafruit\_hid.keycode.Keycode attribute), 19

## P

P (adafruit\_hid.keycode.Keycode attribute), 19

PAGE\_DOWN (adafruit\_hid.keycode.Keycode attribute), 19

PAGE\_UP (adafruit\_hid.keycode.Keycode attribute), 19

PAUSE (adafruit\_hid.keycode.Keycode attribute), 19

PERIOD (adafruit\_hid.keycode.Keycode attribute), 19

POUND (adafruit\_hid.keycode.Keycode attribute), 19

POWER (adafruit\_hid.keycode.Keycode attribute), 19

press() (adafruit\_hid.keyboard.Keyboard method), 14

press() (adafruit\_hid.mouse.Mouse method), 23

PRINT\_SCREEN (adafruit\_hid.keycode.Keycode attribute), 19

## Q

Q (adafruit\_hid.keycode.Keycode attribute), 20

QUOTE (adafruit\_hid.keycode.Keycode attribute), 20

## R

R (adafruit\_hid.keycode.Keycode attribute), 20

release() (adafruit\_hid.keyboard.Keyboard method), 15

release() (adafruit\_hid.mouse.Mouse method), 23

release\_all() (adafruit\_hid.keyboard.Keyboard method), 15  
 release\_all() (adafruit\_hid.mouse.Mouse method), 23  
 RETURN (adafruit\_hid.keycode.Keycode attribute), 20  
 RIGHT\_ALT (adafruit\_hid.keycode.Keycode attribute), 20  
 RIGHT\_ARROW (adafruit\_hid.keycode.Keycode attribute), 20  
 RIGHT\_BRACKET (adafruit\_hid.keycode.Keycode attribute), 20  
 RIGHT\_BUTTON (adafruit\_hid.mouse.Mouse attribute), 22  
 RIGHT\_CONTROL (adafruit\_hid.keycode.Keycode attribute), 20  
 RIGHT\_GUI (adafruit\_hid.keycode.Keycode attribute), 20  
 RIGHT\_SHIFT (adafruit\_hid.keycode.Keycode attribute), 20

## S

S (adafruit\_hid.keycode.Keycode attribute), 20  
 SCROLL\_LOCK (adafruit\_hid.keycode.Keycode attribute), 20  
 SEMICOLON (adafruit\_hid.keycode.Keycode attribute), 20  
 send() (adafruit\_hid.keyboard.Keyboard method), 15  
 SEVEN (adafruit\_hid.keycode.Keycode attribute), 20  
 SHIFT (adafruit\_hid.keycode.Keycode attribute), 20  
 SIX (adafruit\_hid.keycode.Keycode attribute), 20  
 SPACE (adafruit\_hid.keycode.Keycode attribute), 20  
 SPACEBAR (adafruit\_hid.keycode.Keycode attribute), 20

## T

T (adafruit\_hid.keycode.Keycode attribute), 20  
 TAB (adafruit\_hid.keycode.Keycode attribute), 20  
 THREE (adafruit\_hid.keycode.Keycode attribute), 20  
 TWO (adafruit\_hid.keycode.Keycode attribute), 21

## U

U (adafruit\_hid.keycode.Keycode attribute), 21  
 UP\_ARROW (adafruit\_hid.keycode.Keycode attribute), 21

## V

V (adafruit\_hid.keycode.Keycode attribute), 21

## W

W (adafruit\_hid.keycode.Keycode attribute), 21  
 write() (adafruit\_hid.keyboard\_layout\_us.KeyboardLayoutUS method), 21

## X

X (adafruit\_hid.keycode.Keycode attribute), 21

## Y

Y (adafruit\_hid.keycode.Keycode attribute), 21

## Z

Z (adafruit\_hid.keycode.Keycode attribute), 21  
 ZERO (adafruit\_hid.keycode.Keycode attribute), 21