

---

# **AdafruitLIDARLite Library Documentation**

***Release 1.0***

**ladyada**

**Mar 20, 2020**



---

## Contents

---

<b>1</b>	<b>Dependencies</b>	<b>3</b>
<b>2</b>	<b>Installing from PyPI</b>	<b>5</b>
<b>3</b>	<b>Usage Example</b>	<b>7</b>
<b>4</b>	<b>Contributing</b>	<b>9</b>
<b>5</b>	<b>Documentation</b>	<b>11</b>
<b>6</b>	<b>Table of Contents</b>	<b>13</b>
6.1	Simple test . . . . .	13
6.2	adafruit_lidarlite . . . . .	14
6.2.1	Implementation Notes . . . . .	14
<b>7</b>	<b>Indices and tables</b>	<b>15</b>
	<b>Python Module Index</b>	<b>17</b>
	<b>Index</b>	<b>19</b>



A CircuitPython & Python library for Garmin LIDAR Lite sensors over I2C



# CHAPTER 1

---

## Dependencies

---

This driver depends on:

- [Adafruit CircuitPython](#)
- [Bus Device](#)

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





## CHAPTER 2

---

### 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-lidarlite
```

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

```
sudo pip3 install adafruit-circuitpython-lidarlite
```

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-lidarlite
```



## CHAPTER 3

---

### Usage Example

---

```
import time
import board
import busio
import adafruit_lidarlite

# Create library object using our Bus I2C port
i2c = busio.I2C(board.SCL, board.SDA)

# Default configuration, with only i2c wires
sensor = adafruit_lidarlite.LIDARLite(i2c)

while True:
    try:
        # We print tuples so you can plot with Mu Plotter
        print((sensor.distance,))
    except RuntimeError as e:
        # If we get a reading error, just print it and keep truckin'
        print(e)
    time.sleep(0.01) # you can remove this for ultra-fast measurements!
```



## CHAPTER 4

---

### Contributing

---

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



## CHAPTER 5

---

### Documentation

---

For information on building library documentation, please check out [this guide](#).





## 6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/lidarlite\_simpletest.py

```
1 import time
2 import board
3 import busio
4 import adafruit_lidarlite
5
6
7 # Create library object using our Bus I2C port
8 i2c = busio.I2C(board.SCL, board.SDA)
9
10 # Default configuration, with only i2c wires
11 sensor = adafruit_lidarlite.LIDARLite(i2c)
12
13 # Optionally, we can pass in a hardware reset pin, or custom config
14 # import digitalio
15 # reset = digitalio.DigitalInOut(board.D5)
16 # sensor = adafruit_lidarlite.LIDARLite(i2c, reset_pin=reset,
17 #     configuration=adafruit_lidarlite.CONFIG_MAXRANGE)
18
19 # If you want to reset, you can do so, note that it can take 10-20 seconds
20 # for the data to 'normalize' after a reset (and this isnt documented at all)
21 # sensor.reset()
22
23 while True:
24     try:
25         # We print tuples so you can plot with Mu Plotter
26         print((sensor.distance,))
27     except RuntimeError as e:
```

(continues on next page)

(continued from previous page)

```
28     # If we get a reading error, just print it and keep truckin'
29     print(e)
30     time.sleep(0.01) # you can remove this for ultra-fast measurements!
```

## 6.2 adafruit\_lidarlite

A CircuitPython & Python library for Garmin LIDAR Lite sensors over I2C

- Author(s): ladyada

### 6.2.1 Implementation Notes

**Hardware:**

**Software and Dependencies:**

- Adafruit CircuitPython firmware for the supported boards: <https://github.com/adafruit/circuitpython/releases>
- Adafruit's Bus Device library: [https://github.com/adafruit/Adafruit\\_CircuitPython\\_BusDevice](https://github.com/adafruit/Adafruit_CircuitPython_BusDevice)

**class** `adafruit_lidarlite.LIDARLite` (*i2c\_bus*, \*, *reset\_pin=None*, *configuration=0*, *address=98*)

A driver for the Garmin LIDAR Lite laser distance sensor. :param `i2c_bus`: The `busio.I2C` object to use. This is the only required parameter. :param `int address`: (optional) The I2C address of the device to set after initialization.

**configure** (*config*)

Set the LIDAR desired style of measurement. There are a few common configurations Garmin suggests: `CONFIG_DEFAULT`, `CONFIG_SHORTFAST`, `CONFIG_DEFAULTFAST`, `CONFIG_MAXRANGE`, `CONFIG_HIGHSENSITIVE`, and `CONFIG_LOWSENSITIVE`.

**distance**

The measured distance in cm. Will take a bias reading every 100 calls

**read\_distance** (*bias=False*)

Perform a distance reading with or without 'bias'. It's recommended to take a bias measurement every 100 non-bias readings (they're slower)

**reset** ()

Hardware reset (if pin passed into init) or software reset. Will take 100 readings in order to 'flush' measurement unit, otherwise data is off.

**status**

The status byte, check datasheet for bitmask

## CHAPTER 7

---

### Indices and tables

---

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



### a

`adafruit_lidarlite`, [14](#)



### A

`adafruit_lidarlite` (*module*), [14](#)

### C

`configure()` (*adafruit\_lidarlite.LIDARLite method*), [14](#)

### D

`distance` (*adafruit\_lidarlite.LIDARLite attribute*), [14](#)

### L

`LIDARLite` (*class in adafruit\_lidarlite*), [14](#)

### R

`read_distance()` (*adafruit\_lidarlite.LIDARLite method*), [14](#)

`reset()` (*adafruit\_lidarlite.LIDARLite method*), [14](#)

### S

`status` (*adafruit\_lidarlite.LIDARLite attribute*), [14](#)