
AdafruitFingerprint Library Documentation

Release 1.0

ladyada

Jul 16, 2020

Contents

1	Dependencies	3
2	Installing from PyPI	5
3	Usage Example	7
4	Contributing	9
5	Documentation	11
6	Table of Contents	13
6.1	Simple test	13
6.2	adafruit_fingerprint	17
6.2.1	Implementation Notes	17
7	Indices and tables	19
	Python Module Index	21
	Index	23

This library will let you use an Adafruit Fingerprint sensor on any UART to get, store, retrieve and query fingerprints!
Great for adding bio-sensing security to your next build.

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

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-fingerprint
```

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

```
sudo pip3 install adafruit-circuitpython-fingerprint
```

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-fingerprint
```


CHAPTER 3

Usage Example

See 'examples' folder for full usage demo!

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/fingerprint_simpletest.py

```
1 import time
2 import board
3 import busio
4 from digitalio import DigitalInOut, Direction
5 import adafruit_fingerprint
6
7 led = DigitalInOut(board.D13)
8 led.direction = Direction.OUTPUT
9
10 uart = busio.UART(board.TX, board.RX, baudrate=57600)
11
12 # If using with a computer such as Linux/RaspberryPi, Mac, Windows with USB/serial_
13 ↪converter:
14 # import serial
15 # uart = serial.Serial("/dev/ttyUSB0", baudrate=57600, timeout=1)
16
17 # If using with Linux/Raspberry Pi and hardware UART:
18 # import serial
19 # uart = serial.Serial("/dev/ttyS0", baudrate=57600, timeout=1)
20
21 finger = adafruit_fingerprint.Adafruit_Fingerprint(uart)
22
23 #####
24
25 def get_fingerprint():
26     """Get a finger print image, template it, and see if it matches!"""
```

(continues on next page)

(continued from previous page)

```

27     print("Waiting for image...")
28     while finger.get_image() != adafruit_fingerprint.OK:
29         pass
30     print("Templating...")
31     if finger.image_2_tz(1) != adafruit_fingerprint.OK:
32         return False
33     print("Searching...")
34     if finger.finger_search() != adafruit_fingerprint.OK:
35         return False
36     return True
37
38
39 # pylint: disable=too-many-branches
40 def get_fingerprint_detail():
41     """Get a finger print image, template it, and see if it matches!
42     This time, print out each error instead of just returning on failure"""
43     print("Getting image...", end="", flush=True)
44     i = finger.get_image()
45     if i == adafruit_fingerprint.OK:
46         print("Image taken")
47     else:
48         if i == adafruit_fingerprint.NO_FINGER:
49             print("No finger detected")
50         elif i == adafruit_fingerprint.IMAGEFAIL:
51             print("Imaging error")
52         else:
53             print("Other error")
54         return False
55
56     print("Templating...", end="", flush=True)
57     i = finger.image_2_tz(1)
58     if i == adafruit_fingerprint.OK:
59         print("Templated")
60     else:
61         if i == adafruit_fingerprint.IMAGEMESS:
62             print("Image too messy")
63         elif i == adafruit_fingerprint.FEATUREFAIL:
64             print("Could not identify features")
65         elif i == adafruit_fingerprint.INVALIDIMAGE:
66             print("Image invalid")
67         else:
68             print("Other error")
69         return False
70
71     print("Searching...", end="", flush=True)
72     i = finger.finger_fast_search()
73     # pylint: disable=no-else-return
74     # This block needs to be refactored when it can be tested.
75     if i == adafruit_fingerprint.OK:
76         print("Found fingerprint!")
77         return True
78     else:
79         if i == adafruit_fingerprint.NOTFOUND:
80             print("No match found")
81         else:
82             print("Other error")
83         return False

```

(continues on next page)

(continued from previous page)

```

84
85
86 # pylint: disable=too-many-statements
87 def enroll_finger(location):
88     """Take a 2 finger images and template it, then store in 'location'"""
89     for fingerimg in range(1, 3):
90         if fingerimg == 1:
91             print("Place finger on sensor...", end="", flush=True)
92         else:
93             print("Place same finger again...", end="", flush=True)
94
95         while True:
96             i = finger.get_image()
97             if i == adafruit_fingerprint.OK:
98                 print("Image taken")
99                 break
100             if i == adafruit_fingerprint.NOFINGER:
101                 print(".", end="", flush=True)
102             elif i == adafruit_fingerprint.IMAGEFAIL:
103                 print("Imaging error")
104                 return False
105             else:
106                 print("Other error")
107                 return False
108
109         print("Templating...", end="", flush=True)
110         i = finger.image_2_tz(fingerimg)
111         if i == adafruit_fingerprint.OK:
112             print("Templated")
113         else:
114             if i == adafruit_fingerprint.IMAGEMESS:
115                 print("Image too messy")
116             elif i == adafruit_fingerprint.FEATUREFAIL:
117                 print("Could not identify features")
118             elif i == adafruit_fingerprint.INVALIDIMAGE:
119                 print("Image invalid")
120             else:
121                 print("Other error")
122             return False
123
124         if fingerimg == 1:
125             print("Remove finger")
126             time.sleep(1)
127             while i != adafruit_fingerprint.NOFINGER:
128                 i = finger.get_image()
129
130         print("Creating model...", end="", flush=True)
131         i = finger.create_model()
132         if i == adafruit_fingerprint.OK:
133             print("Created")
134         else:
135             if i == adafruit_fingerprint.ENROLLMISMATCH:
136                 print("Prints did not match")
137             else:
138                 print("Other error")
139             return False
140

```

(continues on next page)

(continued from previous page)

```

141     print("Storing model #%d..." % location, end="", flush=True)
142     i = finger.store_model(location)
143     if i == adafruit_fingerprint.OK:
144         print("Stored")
145     else:
146         if i == adafruit_fingerprint.BADLOCATION:
147             print("Bad storage location")
148         elif i == adafruit_fingerprint.FLASHERR:
149             print("Flash storage error")
150         else:
151             print("Other error")
152         return False
153
154     return True
155
156 #####
157
158
159
160 def get_num():
161     """Use input() to get a valid number from 1 to 127. Retry till success!"""
162     i = 0
163     while (i > 127) or (i < 1):
164         try:
165             i = int(input("Enter ID # from 1-127: "))
166         except ValueError:
167             pass
168     return i
169
170
171 while True:
172     print("-----")
173     if finger.read_templates() != adafruit_fingerprint.OK:
174         raise RuntimeError("Failed to read templates")
175     print("Fingerprint templates:", finger.templates)
176     print("e) enroll print")
177     print("f) find print")
178     print("d) delete print")
179     print("-----")
180     c = input("> ")
181
182     if c == "e":
183         enroll_finger(get_num())
184     if c == "f":
185         if get_fingerprint():
186             print("Detected #", finger.finger_id, "with confidence", finger.
187 ↪confidence)
188         else:
189             print("Finger not found")
190     if c == "d":
191         if finger.delete_model(get_num()) == adafruit_fingerprint.OK:
192             print("Deleted!")
193         else:
194             print("Failed to delete")

```

6.2 adafruit_fingerprint

This library will let you use an Adafruit Fingerprint sensor on any UART to get, store, retrieve and query fingerprints! Great for adding bio-sensing security to your next build.

- Author(s): ladyada

6.2.1 Implementation Notes

Hardware:

- [Fingerprint sensor](#) (Product ID: 751)
- [Panel Mount Fingerprint sensor](#) (Product ID: 4651)

Software and Dependencies:

- Adafruit CircuitPython firmware (2.2.0+) for the ESP8622 and M0-based boards: <https://github.com/adafruit/circuitpython/releases>

class `adafruit_fingerprint.Adafruit_Fingerprint` (*uart, passwd=(0, 0, 0, 0)*)

UART based fingerprint sensor.

check_module ()

Checks the state of the fingerprint scanner module. Returns OK or error.

count_templates ()

Requests the sensor to count the number of templates and stores it in `self.template_count`. Returns the packet error code or OK success

create_model ()

Requests the sensor take the template data and turn it into a model returns the packet error code or OK success

delete_model (*location*)

Requests the sensor delete a model from flash memory given by the argument location. Returns the packet error code or OK success

empty_library ()

Requests the sensor to delete all models from flash memory. Returns the packet error code or OK success

finger_fast_search ()

Asks the sensor to search for a matching fingerprint template to the last model generated. Stores the location and confidence in `self.finger_id` and `self.confidence`. Returns the packet error code or OK success

finger_search ()

Asks the sensor to search for a matching fingerprint starting at slot 1. Stores the location and confidence in `self.finger_id` and `self.confidence`. Returns the packet error code or OK success

get_fpdata (*sensorbuffer='char', slot=1*)

Requests the sensor to transfer the fingerprint image or template. Returns the data payload only.

get_image ()

Requests the sensor to take an image and store it memory, returns the packet error code or OK success

image_2_tz (*slot=1*)

Requests the sensor convert the image to a template, returns the packet error code or OK success

load_model (*location, slot=1*)

Requests the sensor to load a model from the given memory location to the given slot. Returns the packet error code or success

read_sysparam()

Returns the system parameters on success via attributes.

read_templates()

Requests the sensor to list of all template locations in use and stores them in self.templates. Returns the packet error code or OK success

send_fpdata (*data*, *sensorbuffer='char'*, *slot=1*)

Requests the sensor to receive data, either a fingerprint image or a character/template data. Data is the payload only.

store_model (*location*, *slot=1*)

Requests the sensor store the model into flash memory and assign a location. Returns the packet error code or OK success

verify_password()

Checks if the password/connection is correct, returns True/False

CHAPTER 7

Indices and tables

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

a

`adafruit_fingerprint`, [16](#)

A

Adafruit_Fingerprint (class
 adafruit_fingerprint), 17
adafruit_fingerprint (module), 16

C

check_module() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17
count_templates()
 (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17
create_model() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17

D

delete_model() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17

E

empty_library() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17

F

finger_fast_search()
 (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17
finger_search() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17

G

get_fpdata() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17
get_image() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17

I

image_2_tz() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17

L

in load_model() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17

R

read_sysparam() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 17
read_templates() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 18

S

send_fpdata() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 18
store_model() (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 18

V

verify_password()
 (*adafruit_fingerprint.Adafruit_Fingerprint*
 method), 18