

---

# **Adafruit HT16K33 Library Documentation**

***Release 1.0***

**Radomir Dopieralski**

**Mar 11, 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>7</b>
<b>4</b>	<b>Building locally</b>	<b>9</b>
4.1	Sphinx documentation . . . . .	9
<b>5</b>	<b>Table of Contents</b>	<b>11</b>
5.1	Simple test . . . . .	11
5.2	adafruit_ht16k33.ht16k33 . . . . .	13
5.3	Matrix Displays . . . . .	13
5.4	Segment Displays . . . . .	13
<b>6</b>	<b>Indices and tables</b>	<b>15</b>
	<b>Python Module Index</b>	<b>17</b>



This is a library for using the I<sup>2</sup>C-based LED matrices with the HT16K33 chip. It supports both 16x8 and 8x8 matrices, as well as 7- and 14-segment displays.

- **Notes**

1. This library is intended for Adafruit CircuitPython's API. For a library compatible with MicroPython machine API see this [library](#).
2. This library does not work with the Trellis 4x4 LED+Keypad board. For that product use: [CircuitPython Trellis Library](#)



# 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

---

### Usage Example

---

```
# Import all board pins.
from board import *
import busio

# Import the HT16K33 LED matrix module.
from adafruit_ht16k33 import matrix

# Create the I2C interface.
i2c = busio.I2C(SCL, SDA)

# Create the matrix class.
# This creates a 16x8 matrix:
matrix = matrix.Matrix16x8(i2c)
# Or this creates a 8x8 matrix:
#matrix = matrix.Matrix8x8(i2c)
# Or this creates a 8x8 bicolor matrix:
#matrix = matrix.Matrix8x8x2
# Finally you can optionally specify a custom I2C address of the HT16k33 like:
#matrix = matrix.Matrix16x8(i2c, address=0x70)

# Clear the matrix. Always call show after changing pixels to make the display
# update visible!
matrix.fill(0)
matrix.show()

# Set a pixel in the origin 0,0 position.
matrix.pixel(0, 0, 1)
# Set a pixel in the middle 8, 4 position.
matrix.pixel(8, 4, 1)
# Set a pixel in the opposite 15, 7 position.
matrix.pixel(15, 7, 1)
matrix.show()
```



## 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-ht16k33 --
↳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 5.1: examples/matrix.py

```
1  # Basic example of clearing and drawing a pixel on a LED matrix display.
2  # This example and library is meant to work with Adafruit CircuitPython API.
3  # Author: Tony DiCola
4  # License: Public Domain
5
6  # Import all board pins.
7  from board import *
8  import busio
9
10 # Import the HT16K33 LED matrix module.
11 from adafruit_ht16k33 import matrix
12
13
14 # Create the I2C interface.
15 i2c = busio.I2C(SCL, SDA)
16
17 # Create the matrix class.
18 # This creates a 16x8 matrix:
19 matrix = matrix.Matrix16x8(i2c)
20 # Or this creates a 8x8 matrix:
21 #matrix = matrix.Matrix8x8(i2c)
22 # Or this creates a 8x8 bicolor matrix:
23 #matrix = matrix.Matrix8x8x2
24 # Finally you can optionally specify a custom I2C address of the HT16k33 like:
25 #matrix = matrix.Matrix16x8(i2c, address=0x70)
26
27 # Clear the matrix. Always call show after changing pixels to make the display
28 # update visible!
```

```
29 matrix.fill(0)
30 matrix.show()
31
32 # Set a pixel in the origin 0,0 position.
33 matrix.pixel(0, 0, 1)
34 # Set a pixel in the middle 8, 4 position.
35 matrix.pixel(8, 4, 1)
36 # Set a pixel in the opposite 15, 7 position.
37 matrix.pixel(15, 7, 1)
38 matrix.show()
```

Listing 5.2: examples/segments.py

```
1  # Basic example of setting digits on a LED segment display.
2  # This example and library is meant to work with Adafruit CircuitPython API.
3  # Author: Tony DiCola
4  # License: Public Domain
5
6  # Import all board pins.
7  from board import *
8  import busio
9
10 # Import the HT16K33 LED segment module.
11 from adafruit_ht16k33 import segments
12
13
14 # Create the I2C interface.
15 i2c = busio.I2C(SCL, SDA)
16
17 # Create the LED segment class.
18 # This creates a 7 segment 4 character display:
19 display = segments.Seg7x4(i2c)
20 # Or this creates a 14 segment alphanumeric 4 character display:
21 #display = segments.Seg14x4(i2c)
22 # Finally you can optionally specify a custom I2C address of the HT16k33 like:
23 #display = segments.Seg7x4(i2c, address=0x70)
24
25 # Clear the display. Always call show after changing the display to make the
26 # update visible!
27 display.fill(0)
28 display.show()
29
30 # Set the first character to '1':
31 display.put('1', 0)
32 # Set the second character to '2':
33 display.put('2', 1)
34 # Set the third character to 'A':
35 display.put('A', 2)
36 # Set the forth character to 'B':
37 display.put('B', 3)
38 # Make sure to call show to see the changes above on the display!
39 display.show()
```



## 5.2 adafruit\_ht16k33.ht16k33

- Authors: Radomir Dopieralski & Tony DiCola for Adafruit Industries

**class** adafruit\_ht16k33.ht16k33.**HT16K33** (*i2c, address=112*)

The base class for all displays. Contains common methods.

**blink\_rate** (*rate=None*)

The blink rate. Range 0-3.

**brightness** (*brightness*)

The brightness. Range 0-15.

**fill** (*color*)

Fill the whole display with the given color.

**show** ()

Refresh the display and show the changes.

## 5.3 Matrix Displays

**class** adafruit\_ht16k33.matrix.**Matrix16x8** (*i2c, address=112*)

A double matrix or the matrix wing.

**pixel** (*x, y, color=None*)

Get or set the color of a given pixel.

**class** adafruit\_ht16k33.matrix.**Matrix8x8** (*i2c, address=112*)

A single matrix.

**pixel** (*x, y, color=None*)

Get or set the color of a given pixel.

**class** adafruit\_ht16k33.matrix.**Matrix8x8x2** (*i2c, address=112*)

A bi-color matrix.

**fill** (*color*)

Fill the whole display with the given color.

**pixel** (*x, y, color=None*)

Get or set the color of a given pixel.

## 5.4 Segment Displays

**class** adafruit\_ht16k33.segments.**Seg14x4** (*i2c, address=112*)

Alpha-numeric, 14-segment display.

**hex** (*number*)

Display the specified hexadecimal number.

**number** (*number*)

Display the specified decimal number.

**push** (*char*)

Scroll the display and add a character at the end.

**put** (*char*, *index=0*)

Put a character at the specified place.

**scroll** (*count=1*)

Scroll the display by specified number of places.

**text** (*text*)

Display the specified text.

**class** `adafruit_ht16k33.segments.Seg7x4` (*i2c*, *address=112*)

Numeric 7-segment display. It has the same methods as the alphanumeric display, but only supports displaying decimal and hex digits, period and a minus sign.

**push** (*char*)

Scroll the display and add a character at the end.

**put** (*char*, *index=0*)

Put a character at the specified place.

**scroll** (*count=1*)

Scroll the display by specified number of places.

## CHAPTER 6

---

### Indices and tables

---

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



### a

`adafruit_ht16k33.ht16k33`, [12](#)  
`adafruit_ht16k33.matrix`, [13](#)  
`adafruit_ht16k33.segments`, [13](#)



## A

adafruit\_ht16k33.ht16k33 (module), 12  
adafruit\_ht16k33.matrix (module), 13  
adafruit\_ht16k33.segments (module), 13

## B

blink\_rate() (adafruit\_ht16k33.ht16k33.HT16K33  
method), 13  
brightness() (adafruit\_ht16k33.ht16k33.HT16K33  
method), 13

## F

fill() (adafruit\_ht16k33.ht16k33.HT16K33 method), 13  
fill() (adafruit\_ht16k33.matrix.Matrix8x8x2 method), 13

## H

hex() (adafruit\_ht16k33.segments.Seg14x4 method), 13  
HT16K33 (class in adafruit\_ht16k33.ht16k33), 13

## M

Matrix16x8 (class in adafruit\_ht16k33.matrix), 13  
Matrix8x8 (class in adafruit\_ht16k33.matrix), 13  
Matrix8x8x2 (class in adafruit\_ht16k33.matrix), 13

## N

number() (adafruit\_ht16k33.segments.Seg14x4 method),  
13

## P

pixel() (adafruit\_ht16k33.matrix.Matrix16x8 method), 13  
pixel() (adafruit\_ht16k33.matrix.Matrix8x8 method), 13  
pixel() (adafruit\_ht16k33.matrix.Matrix8x8x2 method),  
13  
push() (adafruit\_ht16k33.segments.Seg14x4 method), 13  
push() (adafruit\_ht16k33.segments.Seg7x4 method), 14  
put() (adafruit\_ht16k33.segments.Seg14x4 method), 13  
put() (adafruit\_ht16k33.segments.Seg7x4 method), 14

## S

scroll() (adafruit\_ht16k33.segments.Seg14x4 method), 14  
scroll() (adafruit\_ht16k33.segments.Seg7x4 method), 14  
Seg14x4 (class in adafruit\_ht16k33.segments), 13  
Seg7x4 (class in adafruit\_ht16k33.segments), 14  
show() (adafruit\_ht16k33.ht16k33.HT16K33 method), 13

## T

text() (adafruit\_ht16k33.segments.Seg14x4 method), 14