OpenSign Library Documentation

Release 1.0

Melissa LeBlanc-Williams

May 24, 2022

Contents

1 Dependencies **3**

2 Installing from PyPI **5**

3 Table of Contents **7**

3.1 Raspberry Pi Setup . . . 7
3.2 Package Updates . . . 8
3.3 Install Dependencies . . . 8
3.4 Additional Fonts . . . 8
3.5 RGB LED Matrix Python bindings . . . 8
3.6 Install OpenSign . . . 8
3.7 Automatically Start on Boot . . . 8
3.8 `opensign` . . . 9
3.8.1 Implementation Notes . . . 9
3.9 Demo . . . 14

4 Indices and tables **17**

Python Module Index **19**

Index **21**

A library to facilitate easy RGB Matrix Sign Animations.

CHAPTER 1

Dependencies

This library depends on:

- Henner Zeller RPi RGB LED Matrix
- Python Bindings for RGB Matrix Library
- Python Imaging Library (Pillow)

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 opensign
```

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

```
sudo pip3 install opensign
```

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 opensign
```

CHAPTER 3

Table of Contents

3.1 Raspberry Pi Setup

Start by download Raspberry Pi OS Lite from https://www.raspberrypi.org/downloads/raspberry-pi-os/. Once it is done downloading, flash the image onto a Micro SD card using a program such as balenaEtcher.

Mount the SD card on a computer and create an empty file named "ssh" inside the boot partition. If you only have a WiFi networking, you'll want to create a wpa_supplicant.conf file as well following these instructions. Otherwise, you can wait a bit and use raspi-config to do it the easy way.

Eject the card from your computer and insert the card into the Raspberry Pi.

If you didn't set up WiFi already, be sure the Raspberry Pi is plugged into your network.

Plug in power and let the Raspberry Pi boot.

Wait for a few minutes and then SSH into the Raspberry Pi using an SSH client such as PuTTY for windows or command line SSH on macOS or Linux.

Start raspi-config:

```
sudo raspi-config
```

You can change your password by going to `System Options` > `Password`, which is recommended for security if necessary.

If you haven't already and want to enable WiFi, you can do so now by choosing `System Options` > `Wireless LAN` and follow the prompts.

Be sure to set your Timezone by going to `Localisation Options` > `Change Timezone`.

If you would like to change your hostname so that it doesn't conflict with any other Raspberry Pi, you can do so by choosing `System Options` > `Hostname`.

After that you can exit the utility.

3.2 Package Updates

Next, you'll want to perform an update with the following commands:

```
sudo apt update
sudo apt upgrade
```

3.3 Install Dependencies

Install Python and other dependencies using this command:

```
sudo apt install -y git python3-dev python3-pillow python3-pip libatlas-base-dev␣
↪libtiff-dev libtiff5-dev libopenjp2-7-dev zlib1g-dev libfreetype6-dev liblcms2-dev␣
↪libwebp-dev tcl8.6-dev tk8.6-dev python3-tk libharfbuzz-dev libfribidi-dev libxcb1-
↪dev
```

3.4 Additional Fonts

Install additional fonts:

```
sudo apt install -y fonts-dejavu msttcorefonts fonts-noto
```

3.5 RGB LED Matrix Python bindings

Install the Henner Zeller RPi RGB LED Matrix Python Bindings:

```
git clone https://github.com/hzeller/rpi-rgb-led-matrix
cd rpi-rgb-led-matrix/bindings/python
make build-python PYTHON=$(which python3)
sudo make install-python PYTHON=$(which python3)
```

3.6 Install OpenSign

Install OpenSign via PyPI:

```
sudo pip3 install opensign
```

3.7 Automatically Start on Boot

To automatically start a python script on boot, the easiest way is to put it in **/etc/rc.local**

> **Warning:** If you are using any images in your script, be sure to use absolute paths because the script is not run from your home folder.

> **Warning:** Verify your script is working before doing this by running it from the command line first. If there are any bugs, it just won't start. You can check the specific errors by running `systemctl status rc.local.service`.

Edit the file using:

```
sudo nano /etc/rc.local
```

Insert a new line right before `exit 0`

Assuming your username is pi and your script is in your home directory and called startup.py, the new line should be:

```
sudo python3 /home/pi/startup.py &
```

Save, exit, and reboot. Your sign should come to life.

3.8 opensign

A library to facilitate easy RGB Matrix Sign Animations.

- Author(s): Melissa LeBlanc-Williams

3.8.1 Implementation Notes

Software and Dependencies:

- Henner Zeller RGB Matrix Library: https://github.com/hzeller/rpi-rgb-led-matrix
- Python Imaging Library (Pillow)

class opensign.**OpenSign**(*, *rows=16*, *columns=32*, *chain=1*, *brightness=100*, *gpio_mapping='adafruit-hat'*, *parallel=1*, *pwm_bits=11*, *panel_type=''*, *rgb_sequence='rgb'*, *show_refresh=False*, *slowdown_gpio=None*, *no_hardware_pulse=False*, *pwm_lsb_nanoseconds=130*, *row_addr_type=0*, *multiplexing=0*, *pixel_mapper=''*)

Main class that controls the sign and graphics effects.

blink(*canvas*, *count=3*, *duration=1*)

Blink the foreground on and off a centain number of times over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **count** (*float*) – (optional) The number of times to blink. (default=3)
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=1)

fade_in(*canvas*, *duration=1*, *steps=50*)

Fade the foreground in over a certain period of time by a certain number of steps. More steps is smoother, but too high of a number may slow down the animation too much.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.

- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=1)
- **steps** (*float*) – (optional) The number of steps to perform the animation. (default=50)

fade_out (*canvas*, *duration=1*, *steps=50*)

Fade the foreground out over a certain period of time by a certain number of steps. More steps is smoother, but too high of a number may slow down the animation too much.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=1)
- **steps** (*float*) – (optional) The number of steps to perform the animation. (default=50)

flash (*canvas*, *count=3*, *duration=1*)

Fade the foreground in and out a centain number of times over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **count** (*float*) – (optional) The number of times to flash. (default=3)
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=1)

height

Returns the height in pixels

hide (*canvas*)

Hide the canvas at its current position.

Parameters **canvas** (*OpenSignCanvas*) – The canvas to hide.

join_in_horizontally (*canvas*, *duration=0.5*)

Show the effect of a split canvas joining horizontally over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=0.5)

join_in_vertically (*canvas*, *duration=0.5*)

Show the effect of a split canvas joining vertically over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=0.5)

loop_down (*canvas*, *duration=0.5*, *count=1*)

Loop a canvas towards the bottom side of the display over a certain period of time by a certain number of times. The canvas will re-enter from the top and end up back a the starting position.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **count** (*float*) – (optional) The number of times to loop. (default=1)

- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=1)

loop_left (*canvas*, *duration=1*, *count=1*)

Loop a canvas towards the left side of the display over a certain period of time by a certain number of times. The canvas will re-enter from the right and end up back a the starting position.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **count** (*float*) – (optional) The number of times to loop. (default=1)
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=1)

loop_right (*canvas*, *duration=1*, *count=1*)

Loop a canvas towards the right side of the display over a certain period of time by a certain number of times. The canvas will re-enter from the left and end up back a the starting position.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **count** (*float*) – (optional) The number of times to loop. (default=1)
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=1)

loop_up (*canvas*, *duration=0.5*, *count=1*)

Loop a canvas towards the top side of the display over a certain period of time by a certain number of times. The canvas will re-enter from the bottom and end up back a the starting position.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **count** (*float*) – (optional) The number of times to loop. (default=1)
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=1)

scroll_from_to (*canvas*, *duration*, *start_x*, *start_y*, *end_x*, *end_y*)

Scroll the canvas from one position to another over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – The period of time to perform the animation over in seconds.
- **start_x** (*int*) – The Starting X Position
- **start_yx** (*int*) – The Starting Y Position
- **end_x** (*int*) – The Ending X Position
- **end_y** (*int*) – The Ending Y Position

scroll_in_from_bottom (*canvas*, *duration=1*, *y=0*)

Scroll a canvas in from the bottom side of the display over a certain period of time. The final position is centered.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.

- **duration** (*float*) – (optional) The period of time to perform the animation over in seconds. (default=1)
- **y** (*int*) – (optional) The amount of y-offset from the center position (default=0)

scroll_in_from_left(*canvas*, *duration=1*, *x=0*)

Scroll a canvas in from the left side of the display over a certain period of time. The final position is centered.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over in seconds. (default=1)
- **x** (*int*) – (optional) The amount of x-offset from the center position (default=0)

scroll_in_from_right(*canvas*, *duration=1*, *x=0*)

Scroll a canvas in from the right side of the display over a certain period of time. The final position is centered.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over in seconds. (default=1)
- **x** (*int*) – (optional) The amount of x-offset from the center position (default=0)

scroll_in_from_top(*canvas*, *duration=1*, *y=0*)

Scroll a canvas in from the top side of the display over a certain period of time. The final position is centered.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over in seconds. (default=1)
- **y** (*int*) – (optional) The amount of y-offset from the center position (default=0)

scroll_out_to_bottom(*canvas*, *duration=1*)

Scroll a canvas off the display from its current position towards the bottom over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over in seconds. (default=1)

scroll_out_to_left(*canvas*, *duration=1*)

Scroll a canvas off the display from its current position towards the left over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over in seconds. (default=1)

scroll_out_to_right(*canvas*, *duration=1*)

Scroll a canvas off the display from its current position towards the right over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over in seconds. (default=1)

scroll_out_to_top(*canvas*, *duration=1*)
Scroll a canvas off the display from its current position towards the top over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over in seconds. (default=1)

set_background_color(*color*)
Sets the background to a solid color. The color should be a 3 or 4 value tuple or list or an hexidecimal value in the format of 0xRRGGBB.

Parameters **color** (*tuple or list or int*) – The time to sleep in seconds.

set_background_image(*file*)
Sets the background to an image

Parameters **file** (*string*) – The file location of the image to display.

set_position(*canvas*, *x=0*, *y=0*)
Instantly move the canvas to a specific location. (0, 0) is the top-left corner.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to move.
- **x** (*int*) – (optional) The x-position to move the canvas to. (default=0)
- **y** (*int*) – (optional) The y-position to move the canvas to. (default=0)

show(*canvas*)
Show the canvas at its current position.

Parameters **canvas** (*OpenSignCanvas*) – The canvas to show.

split_out_horizontally(*canvas*, *duration=0.5*)
Show the effect of a canvas splitting horizontally over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=0.5)

split_out_vertically(*canvas*, *duration=0.5*)
Show the effect of a canvas splitting vertically over a certain period of time.

Parameters

- **canvas** (*OpenSignCanvas*) – The canvas to animate.
- **duration** (*float*) – (optional) The period of time to perform the animation over. (default=0.5)

width
Returns the width in pixels

3.9 Demo

Listing 1: examples/demo.py

```
#!/usr/bin/env python
import time
from opensign import OpenSign
from opensign.canvas import OpenSignCanvas


def main():
    message1 = OpenSignCanvas()
    message1.add_font("dejavu", "/usr/share/fonts/truetype/dejavu/DejaVuSans.ttf", 14)
    message1.set_stroke(1, (255, 255, 255))
    message1.add_text("Scroll Text In", color=(255, 0, 0))
    message1.set_shadow()

    message2 = OpenSignCanvas()
    message2.add_font(
        "comic", "/usr/share/fonts/truetype/msttcorefonts/Comic_Sans_MS.ttf", 14
    )
    message2.set_stroke(1, (0, 0, 0))
    message2.add_image("logo.png")
    message2.add_text("Maker Melissa's Lab", color=(255, 255, 0), y_offset=-2)
    message2.set_shadow()

    message3 = OpenSignCanvas()
    message3.add_font("dejavu", "/usr/share/fonts/truetype/dejavu/DejaVuSans.ttf", 14)
    message3.set_stroke(1, (0, 0, 0))
    message3.add_text("https://makermelissa.com/", color=(255, 0, 0))

    message4 = OpenSignCanvas()
    message4.add_font("dejavu", "/usr/share/fonts/truetype/dejavu/DejaVuSans.ttf", 14)
    message4.set_stroke(1, (0, 0, 0))
    message4.add_text("Subscribe to my Channel", color=(255, 255, 255))
    message4.set_shadow()

    sign = OpenSign(chain=6)
    sign.set_background_image("background.jpg")

    while True:
        sign.scroll_in_from_left(message1)
        time.sleep(1)
        message1.clear()
        message1.add_text("Change Messages")
        sign.show(message1)
        time.sleep(1)
        message1.clear()
        message1.add_text("And Scroll Out")
        sign.show(message1)
        sign.scroll_out_to_right(message1)
        time.sleep(1)

        sign.join_in_vertically(message1)
        sign.loop_left(message1)
        sign.flash(message1, count=3)
        sign.split_out_vertically(message1)
```

(continues on next page)

(continued from previous page)

```
        time.sleep(1)
        sign.set_background_color((0, 255, 0))
        sign.fade_in(message2)
        time.sleep(1)
        sign.fade_out(message2)
        sign.scroll_in_from_top(message3)
        time.sleep(1)
        sign.scroll_out_to_bottom(message3)
        sign.scroll_in_from_right(message4)
        time.sleep(1)


# Main function
if __name__ == "__main__":
    main()
```

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

O

opensign, 9

Index

B

`blink()` (*opensign.OpenSign method*), 9

F

`fade_in()` (*opensign.OpenSign method*), 9
`fade_out()` (*opensign.OpenSign method*), 10
`flash()` (*opensign.OpenSign method*), 10

H

`height` (*opensign.OpenSign attribute*), 10
`hide()` (*opensign.OpenSign method*), 10

J

`join_in_horizontally()` (*opensign.OpenSign method*), 10
`join_in_vertically()` (*opensign.OpenSign method*), 10

L

`loop_down()` (*opensign.OpenSign method*), 10
`loop_left()` (*opensign.OpenSign method*), 11
`loop_right()` (*opensign.OpenSign method*), 11
`loop_up()` (*opensign.OpenSign method*), 11

O

`OpenSign` (*class in opensign*), 9
`opensign` (*module*), 9

S

`scroll_from_to()` (*opensign.OpenSign method*), 11
`scroll_in_from_bottom()` (*opensign.OpenSign method*), 11
`scroll_in_from_left()` (*opensign.OpenSign method*), 12
`scroll_in_from_right()` (*opensign.OpenSign method*), 12
`scroll_in_from_top()` (*opensign.OpenSign method*), 12
`scroll_out_to_bottom()` (*opensign.OpenSign method*), 12
`scroll_out_to_left()` (*opensign.OpenSign method*), 12
`scroll_out_to_right()` (*opensign.OpenSign method*), 12
`scroll_out_to_top()` (*opensign.OpenSign method*), 13
`set_background_color()` (*opensign.OpenSign method*), 13
`set_background_image()` (*opensign.OpenSign method*), 13
`set_position()` (*opensign.OpenSign method*), 13
`show()` (*opensign.OpenSign method*), 13
`split_out_horizontally()` (*opensign.OpenSign method*), 13
`split_out_vertically()` (*opensign.OpenSign method*), 13

W

`width` (*opensign.OpenSign attribute*), 13