AdafruitDisplay$_{S}hapesLibraryDocumentation$

Release 1.0

Limor Fried

Aug 05, 2020

Contents

Various common shapes for use with displayio

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.

1.1 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-display_shapes
```

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

```
sudo pip3 install adafruit-circuitpython-display_shapes
```

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-display_shapes
```

CHAPTER 2

Usage Example

```
import board
import displayio
from adafruit_display_shapes.rect import Rect
from adafruit_display_shapes.circle import Circle
from adafruit_display_shapes.roundrect import RoundRect

splash = displayio.Group(max_size=10)
board.DISPLAY.show(splash)

color_bitmap = displayio.Bitmap(320, 240, 1)
color_palette = displayio.Palette(1)
color_palette[0] = 0xFFFFFF
bg_sprite = displayio.TileGrid(color_bitmap, pixel_shader=color_palette, position=(0,␣
↪0))
print(bg_sprite.position)
splash.append(bg_sprite)

triangle = Triangle(170, 50, 120, 140, 210, 160, fill=0x00FF00, outline=0xFF00FF)
splash.append(triangle)

rect = Rect(80, 20, 41, 41, fill=0x0)
splash.append(rect)

circle = Circle(100, 100, 20, fill=0x00FF00, outline=0xFF00FF)
splash.append(circle)

rect2 = Rect(50, 100, 61, 81, outline=0x0, stroke=3)
splash.append(rect2)

roundrect = RoundRect(10, 10, 61, 81, 10, fill=0x0, outline=0xFF00FF, stroke=6)
splash.append(roundrect)

while True:
    pass
```

CHAPTER 3

Contributing

Contributions are welcome! Please read our Code of Conduct before contributing to help this project stay welcoming.

CHAPTER 4

Building locally

4.1 Zip release files

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-display_shapes --
↪library_location .
```

4.2 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.

CHAPTER 5

Table of Contents

5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/display_shapes_simpletest.py

```
import board
import displayio
from adafruit_display_shapes.rect import Rect
from adafruit_display_shapes.circle import Circle
from adafruit_display_shapes.roundrect import RoundRect
from adafruit_display_shapes.triangle import Triangle
from adafruit_display_shapes.line import Line
from adafruit_display_shapes.polygon import Polygon

# Make the display context
splash = displayio.Group(max_size=20)
board.DISPLAY.show(splash)

# Make a background color fill
color_bitmap = displayio.Bitmap(320, 240, 1)
color_palette = displayio.Palette(1)
color_palette[0] = 0xFFFFFF
bg_sprite = displayio.TileGrid(color_bitmap, x=0, y=0, pixel_shader=color_palette)
splash.append(bg_sprite)
################################################################################

splash.append(Line(220, 130, 270, 210, 0xFF0000))
splash.append(Line(270, 210, 220, 210, 0xFF0000))
splash.append(Line(220, 210, 270, 130, 0xFF0000))
splash.append(Line(270, 130, 220, 130, 0xFF0000))

# Draw a blue star
```

(continues on next page)

(continued from previous page)

```
polygon = Polygon(
    [
        (255, 40),
        (262, 62),
        (285, 62),
        (265, 76),
        (275, 100),
        (255, 84),
        (235, 100),
        (245, 76),
        (225, 62),
        (248, 62),
    ],
    outline=0x0000FF,
)
splash.append(polygon)

triangle = Triangle(170, 50, 120, 140, 210, 160, fill=0x00FF00, outline=0xFF00FF)
splash.append(triangle)

rect = Rect(80, 20, 41, 41, fill=0x0)
splash.append(rect)

circle = Circle(100, 100, 20, fill=0x00FF00, outline=0xFF00FF)
splash.append(circle)

rect2 = Rect(50, 100, 61, 81, outline=0x0, stroke=3)
splash.append(rect2)

roundrect = RoundRect(10, 10, 61, 81, 10, fill=0x0, outline=0xFF00FF, stroke=6)
splash.append(roundrect)


while True:
    pass
```

5.2 circle

Various common shapes for use with displayio - Circle shape!

- Author(s): Limor Fried

5.2.1 Implementation Notes

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases

class adafruit_display_shapes.circle.**Circle**(*x0*, *y0*, *r*, ***, *fill=None*, *outline=None*, *stroke=1*)

A circle.

Parameters

- **x0** – The x-position of the center.

- **y0** – The y-position of the center..
- **r** – The radius of the circle.
- **fill** – The color to fill the rounded-corner rectangle. Can be a hex value for a color or `None` for transparent.
- **outline** – The outline of the rounded-corner rectangle. Can be a hex value for a color or `None` for no outline.
- **stroke** – Used for the outline. Will not change the radius.

5.3 rect

Various common shapes for use with displayio - Rectangle shape!

- Author(s): Limor Fried

5.3.1 Implementation Notes

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases

class adafruit_display_shapes.rect.**Rect**(*x*, *y*, *width*, *height*, ***, *fill=None*, *outline=None*, *stroke=1*)

A rectangle.

Parameters

- **x** – The x-position of the top left corner.
- **y** – The y-position of the top left corner.
- **width** – The width of the rectangle.
- **height** – The height of the rectangle.
- **fill** – The color to fill the rectangle. Can be a hex value for a color or `None` for transparent.
- **outline** – The outline of the rectangle. Can be a hex value for a color or `None` for no outline.
- **stroke** – Used for the outline. Will not change the outer bound size set by `width` and `height`.

fill
The fill of the rectangle. Can be a hex value for a color or `None` for transparent.

outline
The outline of the rectangle. Can be a hex value for a color or `None` for no outline.

5.4 roundrect

A slightly modified version of Adafruit_CircuitPython_Display_Shapes that includes an explicit call to palette.make_opaque() in the fill color setter function.

class adafruit_display_shapes.roundrect.**RoundRect**(*x*, *y*, *width*, *height*, *r*, ***, *fill=None*, *outline=None*, *stroke=1*)

A round-corner rectangle.

Parameters

- **x** – The x-position of the top left corner.
- **y** – The y-position of the top left corner.
- **width** – The width of the rounded-corner rectangle.
- **height** – The height of the rounded-corner rectangle.
- **r** – The radius of the rounded corner.
- **fill** – The color to fill the rounded-corner rectangle. Can be a hex value for a color or `None` for transparent.
- **outline** – The outline of the rounded-corner rectangle. Can be a hex value for a color or `None` for no outline.
- **stroke** – Used for the outline. Will not change the outer bound size set by `width` and `height`.

fill
The fill of the rounded-corner rectangle. Can be a hex value for a color or `None` for transparent.

outline
The outline of the rounded-corner rectangle. Can be a hex value for a color or `None` for no outline.

5.5 triangle

Various common shapes for use with displayio - Triangle shape!

- Author(s): Melissa LeBlanc-Williams

5.5.1 Implementation Notes

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases

class adafruit_display_shapes.triangle.**Triangle**(*x0*, *y0*, *x1*, *y1*, *x2*, *y2*, ***, *fill=None*, *outline=None*)

A triangle.

Parameters

- **x0** – The x-position of the first vertex.
- **y0** – The y-position of the first vertex.
- **x1** – The x-position of the second vertex.
- **y1** – The y-position of the second vertex.
- **x2** – The x-position of the third vertex.
- **y2** – The y-position of the third vertex.
- **fill** – The color to fill the triangle. Can be a hex value for a color or `None` for transparent.

- **outline** – The outline of the triangle. Can be a hex value for a color or `None` for no outline.

fill
The fill of the triangle. Can be a hex value for a color or `None` for transparent.

5.6 line

Various common shapes for use with displayio - Line shape!

- Author(s): Melissa LeBlanc-Williams

5.6.1 Implementation Notes

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases

class `adafruit_display_shapes.line.`**Line**(*x0*, *y0*, *x1*, *y1*, *color*)
A line.

Parameters

- **x0** – The x-position of the first vertex.
- **y0** – The y-position of the first vertex.
- **x1** – The x-position of the second vertex.
- **y1** – The y-position of the second vertex.
- **color** – The color of the line.

5.7 polygon

Various common shapes for use with displayio - Polygon shape!

- Author(s): Melissa LeBlanc-Williams

5.7.1 Implementation Notes

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases

class `adafruit_display_shapes.polygon.`**Polygon**(*points*, *, *outline=None*)
A polygon.

Parameters

- **points** – A list of (x, y) tuples of the points
- **fill** – The color to fill the polygon. Can be a hex value for a color or `None` for transparent.
- **outline** – The outline of the polygon. Can be a hex value for a color or `None` for no outline.

outline
The outline of the polygon. Can be a hex value for a color or `None` for no outline.

5.8 sparkline

Various common shapes for use with displayio - Sparkline!

- Author(s): Kevin Matocha

5.8.1 Implementation Notes

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases

class adafruit_display_shapes.sparkline.**Sparkline**(*width*, *height*, *max_items*, *y_min=None*, *y_max=None*, *x=0*, *y=0*, *color=16777215*)

A sparkline graph.

: param width: Width of the sparkline graph in pixels : param height: Height of the sparkline graph in pixels : param max_items: Maximum number of values housed in the sparkline : param y_min: Lower range for the y-axis. Set to None for autorange. : param y_max: Upper range for the y-axis. Set to None for autorange. : param x: X-position on the screen, in pixels : param y: Y-position on the screen, in pixels : param color: Line color, the default value is 0xFFFFFF (WHITE)

add_value(*value*)
Add a value to the sparkline. : param value: The value to be added to the sparkline

update()
Update the drawing of the sparkline

values()
Returns the values displayed on the sparkline

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index