CircuitPython DisplayIO $_{A}nnotation Library Documentation

Release 1.0

Kevin Matocha

Jul 05, 2021

CONTENTS

A CircuitPython DisplayIO widget for annotating other widgets or freeform positions.

CHAPTER

ONE

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 or individual libraries can be installed using circup.

CHAPTER

TWO

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 circuitpython-displayio-annotation
```

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

```
sudo pip3 install circuitpython-displayio-annotation
```

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 circuitpython-displayio-annotation
```

CHAPTER

THREE

USAGE EXAMPLE

See scripts in the examples directory of this repository.

CHAPTER

FOUR

CONTRIBUTING

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

CHAPTER

FIVE

DOCUMENTATION

For information on building library documentation, please check out this guide.

CHAPTER

SIX

TABLE OF CONTENTS

6.1 Simple test

Displays annotations, examples relative to SwitchRound widget or as freeform.

Listing 1: examples/displayio_annotation_simpletest.py

```
# SPDX-FileCopyrightText: 2021 Kevin Matocha
#
# SPDX-License-Identifier: MIT
"""
Example of the Annotation widget to annotate a Switch widget or
for freeform annotation.
"""

import time
import board
import displayio
import adafruit_touchscreen
from adafruit_displayio_layout.widgets.switch_round import SwitchRound as Switch
from adafruit_displayio_layout.widgets.annotation import Annotation

display = board.DISPLAY

ts = adafruit_touchscreen.Touchscreen(
    board.TOUCH_XL,
    board.TOUCH_XR,
    board.TOUCH_YD,
    board.TOUCH_YU,
    calibration=((5200, 59000), (5800, 57000)),
    size=(display.width, display.height),
)

# Create the switch widget
my_switch = Switch(190, 50)

# Create several annotations

# This annotation is positioned relative to the switch widget, with default values.
switch_annotation = Annotation(
    widget=my_switch,  # positions are relative to the switch
```

(continues on next page)

(continued from previous page)

```
    text="Widget Annotation: Switch",
)

# This annotation is positioned relative to the switch widget, with the line
# going in the downard direction and anchored at the middle bottom of the switch.
# The position is "nudged" downward using ``position_offset`` to create a 1 pixel
# gap between the end of the line and the switch.
# The text is positioned under the line by setting ``text_under`` to True.
switch_annotation_under = Annotation(
    widget=my_switch,  # positions are relative to the switch
    text="Annotation with: text_under = True",
    delta_x=-10,
    delta_y=15,  # line will go in downward direction (positive y)
    anchor_point=(0.5, 1.0),  # middle, bottom of switch
    position_offset=(0, 1),  # nudge downward by one pixel
    text_under=True,
)

# This is a freeform annotation that is positioned using (x,y) values at the bottom,␣
↪right
# corner of the display (display.width, display.height).
# The line direction is
freeform_annotation = Annotation(
    x=display.width,  # uses freeform (x,y) position
    y=display.height,
    text="Freeform annotation (display.width, height)",
)

my_group = displayio.Group()
my_group.append(my_switch)
my_group.append(switch_annotation)
my_group.append(switch_annotation_under)
my_group.append(freeform_annotation)

# Add my_group to the display
display.show(my_group)

# Start the main loop
while True:

    p = ts.touch_point  # get any touches on the screen

    if p:  # Check each switch if the touch point is within the switch touch area
        # If touched, then flip the switch with .selected
        if my_switch.contains(p):
            my_switch.selected(p)

    time.sleep(0.05)  # touch response on PyPortal is more accurate with a small delay
```

6.2 displayio_annotation

A widget for annotating other widgets or freeform positions.

- Author(s): Kevin Matocha

6.2.1 Implementation Notes

Hardware:

Software and Dependencies:

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

class displayio_annotation.**Annotation**(*x=None*, *y=None*, *text=None*, *font=terminalio.FONT*, *delta_x=-15*, *delta_y=- 10*, *widget=None*, *anchor_point=(0.0, 0.0)*, *anchored_position=None*, *position_offset=(0, 0)*, *stroke=3*, *line_color=16777215*, *text_color=None*, *text_offset=(0, - 1)*, *text_under=False*)

A widget to be used to annotate other widgets with text and lines, but can also be used freeform by using `(x,y)` parameter.

Parameters

- **x** (*int*) – x-direction pixel position for the end of the annotation line for freeform positioning, `(x,y)` will be ignored if a `widget` and `anchor_point` and/or `anchored_position` are provided.
- **y** (*int*) – y-direction pixel position for the end of the annotation line for freeform positioning.
- **widget** (*Widget*) – the widget to be annotated, all dimensions are relative to this widget. The annotation line position will be defined by either the `anchor_point` (in relative dimensions of the size of the widget) or the `anchored_position` (in raw pixel dimensions relative to the origin of the widget).
- **text** (*str*) – text to be displayed in the annotation.
- **font** (*Font*) – font to be used for the text.
- **anchor_point** (*Tuple[float, float]*) – starting point for the annotation line, where `anchor_point` is an (A,B) tuple in relative units of the size of the widget, for example (0.0, 0.0) is the upper left corner, and (1.0, 1.0) is the lower right corner of the widget. If `anchor_point` is `None`, then `anchored_position` is used to set the annotation line starting point, in widget size relative units (default is (0.0, 0.0)).
- **anchored_position** (*Tuple[int, int]*) – pixel position starting point for the annotation line where `anchored_position` is an (x,y) tuple in pixel units relative to the upper left corner of the widget, in pixel units (default is None).
- **position_offset** (*Tuple[int, int]*) – Used to *nudge* the line position to where you want, this is an (x,y) pixel offset added to the annotation line starting point, either set by `anchor_point` or `anchored_position` (in pixel units).
- **delta_x** (*int*) – the pixel x-offset for the second end of the line where the text will reside, in pixel units (default: -15).
- **delta_y** (*int*) – the pixel y-offset for the second end of the line where the text will reside, in pixel units (default: -10).
- **stroke** (*int*) – the annotation line width (in pixels). [NOT currently implemented]

- **line_color** (*int*) – the color of the annotation line (default: 0xFFFFFF).
- **text_color** (*int*) – the color of the text, if set to `None` color will be set to `line_color` (default: same as `line_color`).
- **text_offset** (*Tuple[int, int]*) – a (x,y) pixel offset to adjust text position relative to annotation line, in pixel units (default: (0,-1)).
- **text_under** (*Boolean*) – set `True` for text to be placed below the annotation line (default: False).

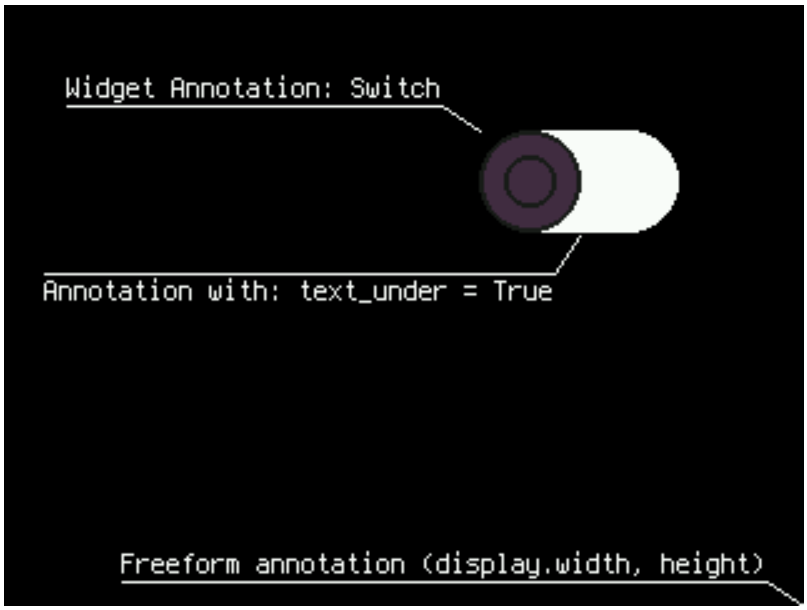

Fig. 1: Example of the annotation widget showing two widget annotations (using `widget` and `anchor_point` input parameters) and a freeform annotation (using `x` and `y` input parameters).
File location: *examples/displayio_annotation_simpletest.py*

Create a Group of a given size and scale. Scale is in one dimension. For example, scale=2 leads to a layer's pixel being 2x2 pixels when in the group.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

d

INDEX