PanTiltHAT Documentation

Release 0.0.4

Phil Howard

Aug 26, 2017

Contents

This documentation will guide you through the methods available in the Pan Tilt HAT python library.

Pan-Tilt HAT lets you mount and control one of our pan-tilt modules right on top of your Raspberry Pi. The HAT and its on-board microcontroller let you independently drive the two servos (pan and tilt), as well as driving up to 24 regular LED (with PWM control) or NeoPixel RGB (or RGBW) LEDs

- More information - https://shop.pimoroni.com/products/pan-tilt-hat
- Get the code - https://github.com/pimoroni/pantilt-hat
- Get help - http://forums.pimoroni.com/c/support

CHAPTER 1

At A Glance

class `pantilthat.`**`PanTilt`**(*enable_lights=True*, *idle_timeout=2*, *light_mode=1*, *light_type=0*, *servo1_min=575*, *servo1_max=2325*, *servo2_min=575*, *servo2_max=2325*, *address=21*, *i2c_bus=None*)

`brightness`(*brightness*)

`clear`()

`get_pan`()

`get_servo_one`()

`get_servo_two`()

`get_tilt`()

`idle_timeout`(*value*)

`light_mode`(*mode*)

`light_type`(*set_type*)

`num_pixels`()

`pan`(*angle*)

`servo_enable`(*index*, *state*)

`servo_one`(*angle*)

`servo_pulse_max`(*index*, *value*)

`servo_pulse_min`(*index*, *value*)

`servo_two`(*angle*)

`set_all`(*red*, *green*, *blue*, *white=None*)

`set_pixel`(*index*, *red*, *green*, *blue*, *white=None*)

`set_pixel_rgbw`(*index*, *red*, *green*, *blue*, *white*)

`show`()

`tilt`(*angle*)

CHAPTER 2

Set Brightness

`pantilthat.brightness(brightness)`

Set the brightness of the connected LED ring.

This only applies if light_mode has been set to PWM.

It will be ignored otherwise.

Parameters **`brightness`** – Brightness from 0 to 255

CHAPTER 3

Clear

```
pantilthat.clear()
```

Clear the buffer.

CHAPTER 4

Set Light Mode & Type

`pantilthat.`**`light_mode`**`(`*mode*`)`

Set the light mode for attached lights.

PanTiltHAT can drive either WS2812 or SK6812 pixels, or provide a PWM dimming signal for regular LEDs.

- PWM - PWM-dimmable LEDs
- WS2812 - 24 WS2812 or 18 SK6812 pixels

`pantilthat.`**`light_type`**`(`*set_type*`)`

Set the light type for attached lights.

Set the type of lighting strip connected:

- RGB - WS2812 pixels with RGB pixel order
- RGB - WS2812 pixels with GRB pixel order
- RGBW - SK6812 pixels with RGBW pixel order
- GRBW - SK6812 pixels with GRBW pixel order

CHAPTER 5

Pan

pantilthat.**pan**(*angle*)

Set position of servo 1 in degrees.

Parameters **angle** – Angle in degrees from -90 to 90

pantilthat.**servo_one**(*angle*)

Set position of servo 1 in degrees.

Parameters **angle** – Angle in degrees from -90 to 90

pantilthat.**get_pan**()

Get position of servo 1 in degrees.

CHAPTER 6

Tilt

`pantilthat.`**`tilt`**`(`*angle*`)`

Set position of servo 2 in degrees.

Parameters **`angle`** – Angle in degrees from -90 to 90

`pantilthat.`**`servo_two`**`(`*angle*`)`

Set position of servo 2 in degrees.

Parameters **`angle`** – Angle in degrees from -90 to 90

`pantilthat.`**`get_tilt`**`()`

Get position of servo 2 in degrees.

CHAPTER 7

Servo Enable

pantilthat.**servo_enable**(*index*, *state*)

Enable or disable a servo.

Disabling a servo turns off the drive signal.

It's good practise to do this if you don't want the Pan/Tilt to point in a certain direction and instead want to save power.

Parameters

- **index** – Servo index: either 1 or 2
- **state** – Servo state: True = on, False = off

CHAPTER 8

Servo Idle Timeout

pantilthat.**idle_timeout**(*value*)

Set the idle timeout for the servos

Configure the time, in seconds, after which the servos will be automatically disabled.

Parameters **value** – Timeout in seconds

CHAPTER 9

Servo Pulse Min

`pantilthat.`**`servo_pulse_min`**`(`*index*, *value*`)`

Set the minimum high pulse for a servo in microseconds.

Parameters **`value`** – Value in microseconds

CHAPTER 10

Servo Pulse Max

`pantilthat.servo_pulse_max(index, value)`

Set the maximum high pulse for a servo in microseconds.

Parameters **`value`** – Value in microseconds

CHAPTER 11

Set All LEDs

`pantilthat.set_all(red, green, blue, white=None)`

Set all pixels in the buffer.

Parameters

- **`red`** – Amount of red, from 0 to 255
- **`green`** – Amount of green, from 0 to 255
- **`blue`** – Amount of blue, from 0 to 255
- **`white`** – Optional amount of white for RGBW and GRBW strips

CHAPTER 12

Set A LED

`pantilthat.`**`set_pixel`**`(`*index*, *red*, *green*, *blue*, *white=None*`)`

Set a single pixel in the buffer.

Parameters

- **`index`** – Index of pixel from 0 to 23
- **`red`** – Amount of red, from 0 to 255
- **`green`** – Amount of green, from 0 to 255
- **`blue`** – Amount of blue, from 0 to 255
- **`white`** – Optional amount of white for RGBW and GRBW strips

CHAPTER 13

Set A LED (RGBW)

`pantilthat.`**`set_pixel_rgbw`**(*index*, *red*, *green*, *blue*, *white*)

Set a single pixel in the buffer for GRBW lighting stick

Parameters

- **`index`** – Index of pixel from 0 to 17
- **`red`** – Amount of red, from 0 to 255
- **`green`** – Amount of green, from 0 to 255
- **`blue`** – Amount of blue, from 0 to 255
- **`white`** – Amount of white, from 0 to 255

CHAPTER 14

Show

`pantilthat.show()`

Display the buffer on the connected WS2812 strip.

CHAPTER 15

Constants

- `WS2812 = 1` - used with `pantilthat.light_mode` to set WS2812/SK6812 LEDs
- `PWM = 0` - used with `pantilthat.light_mode` to set PWM dimmed LEDs
- `RGB = 0` - used with `pantilthat.light_type` to set RGB WS2812 LEDs
- `GRB = 1` - used with `pantilthat.light_type` to set GRB WS2812 LEDs
- `RGBW = 2` - used with `pantilthat.light_type` to set RGBW SK6812 LEDs
- `GRBW = 3` - used with `pantilthat.light_type` to set GRBW SK6812 LEDs

Index