sparkfun$_{q}wiic_{t}wist$

Release 0.0.9

Jul 14, 2021

Contents:

1 Contents **3**

2 Dependencies **5**

3 Documentation **7**

4 Installation **9**
- 4.1 PyPi Installation . . . 9
- 4.2 Local Installation . . . 9

5 Example Use **11**

6 Table of Contents **13**
- 6.1 API Reference . . . 13
  - 6.1.1 qwiic_twist . . . 13
- 6.2 Read a Position and Button State . . . 18
- 6.3 Random Colors . . . 19
- 6.4 Twist Connected Colors . . . 21

7 Indices and tables **25**

Python Module Index **27**

Index **29**

Python module for the qwiic twist, which is part of the SparkFun Qwiic Twist

This python package is a port of the existing SparkFun Qwiic Twist Arduino Library

This package can be used in conjunction with the overall SparkFun qwiic Python Package

New to qwiic? Take a look at the entire SparkFun qwiic ecosystem.

CHAPTER 1

Contents

- *Dependencies*
- *Installation*
- *Documentation*
- *Example Use*

CHAPTER 2

Dependencies

This driver package depends on the qwiic I2C driver: Qwiic_I2C_Py

CHAPTER 3

Documentation

The SparkFun qwiic Twist module documentation is hosted at ReadTheDocs

CHAPTER 4

Installation

4.1 PyPi Installation

This repository is hosted on PyPi as the sparkfun-qwiic-twist package. On systems that support PyPi installation via pip, this library is installed using the following commands

For all users (note: the user must have sudo privileges):

```
sudo pip install sparkfun-qwiic-twist
```

For the current user:

```
pip install sparkfun-qwiic-twist
```

4.2 Local Installation

To install, make sure the setuptools package is installed on the system.

Direct installation at the command line:

```
python setup.py install
```

To build a package for use with pip:

```
python setup.py sdist
```

A package file is built and placed in a subdirectory called dist. This package file can be installed using pip.

```
cd dist
pip install sparkfun_qwiic_twist-<version>.tar.gz
```

CHAPTER 5

Example Use

See the examples directory for more detailed use examples.

```
import qwiic_twist
import time
import sys

def runExample():

    print("\nSparkFun qwiic Twist   Example 1\n")
    myTwist = qwiic_twist.QwiicTwist()

    if myTwist.connected == False:
        print("The Qwiic twist device isn't connected to the system. Please check␣
↪your connection", \
            file=sys.stderr)
        return

    myTwist.begin()

    # Set the knob color to pink
    myTwist.set_color(100, 10, 50)

    while True:

        print("Count: %d, Pressed: %s" % (myTwist.count, \
            "YES" if myTwist.pressed else "NO", \
            ))

        time.sleep(.3)

if __name__ == '__main__':
    try:
        runExample()
    except (KeyboardInterrupt, SystemExit) as exErr:
```

(continues on next page)

(continued from previous page)

```
        print("\nEnding Example 1")
        sys.exit(0)
```

CHAPTER 6

Table of Contents

6.1 API Reference

6.1.1 qwiic_twist

Python module for the[SparkFun Qwiic Twist](https://www.sparkfun.com/products/15083)

This python package is a port of the existing [SparkFun Qwiic Twist Arduino Library](https://github.com/sparkfun/SparkFun_Qwiic_Twist_Arduino_Library)

This package can be used in conjunction with the overall [SparkFun qwiic Python Package](https://github.com/sparkfun/Qwiic_Py)

New to qwiic? Take a look at the entire [SparkFun qwiic ecosystem](https://www.sparkfun.com/qwiic).

class `qwiic_twist.`**`QwiicTwist`**(*address=None*, *i2c_driver=None*)

Parameters

- **address** – The I2C address to use for the device. If not provided, the default address is used.
- **i2c_driver** – An existing i2c driver object. If not provided a driver object is created.

Returns The QwiicTwist device object.

Return type Object

begin()

Initialize the operation of the Twist module

Returns Returns true of the initializtion was successful, otherwise False.

Return type bool

blue

Gets the blue color of the encoder LEDs

Returns blue component of the color

Return type integer

clear_interrupts()

Clears the moved, clicked, and pressed bits

Returns No return Value

clicked

Returns true if a click event has occurred

Returns Click event state

Return type Boolean

connect_blue

Gets the connect blue color of the encoder LEDs

Returns Blue component of the color

connect_color(*red*, *green*, *blue*)

Sets the relation between each color and the twisting of the knob Connect the LED so it changes [amount] with each encoder tick Negative numbers are allowed (so LED gets brighter the more you turn the encoder down)

Parameters

- **red** – Red component
- **green** – Green component
- **blue** – Blue component

Returns No return value

connect_green

Gets the connect green color of the encoder LEDs

Returns green component of the color

connect_red

Gets the connect red color of the encoder LEDs

Returns Red component of the color

connected

Determine if a Tesit device is conntected to the system..

Returns True if the device is connected, otherwise False.

Return type bool

count

Returns the number of indents the user has twisted the knob

Returns number of indents

Return type word as integer

get_blue()

Gets the blue color of the encoder LEDs

Returns blue component of the color

Return type integer

get_connect_blue()

Gets the connect blue color of the encoder LEDs

Returns Blue component of the color

get_connect_green()
Gets the connect green color of the encoder LEDs

Returns green component of the color

get_connect_red()
Gets the connect red color of the encoder LEDs

Returns Red component of the color

get_count()
Returns the number of indents the user has twisted the knob

Returns number of indents

Return type word as integer

get_diff(*clear_value=False*)
Returns the number of ticks since last check

Parameters **clearValue** – Set to True to clear the current value. Default is False

Returns the difference

Return type integer

get_green()
Gets the green color of the encoder LEDs

Returns green component of the color

Return type integer

get_int_timeout()
Get number of milliseconds that elapse between end of knob turning and interrupt firing

Returns the timeout value

Return type integer

get_limit()
Returns the limit of allowed counts before wrapping. 0 is disabled

Returns The limit

Return type integer

get_red()
Gets the red color of the encoder LEDs

Returns Red component of the color

get_version()
Returns a integer of the firmware version number

Returns The firmware version

Return type integer

green
Gets the green color of the encoder LEDs

Returns green component of the color

Return type integer

has_moved()
Returns true if knob has been twisted

Returns Moved state

Return type Boolean

int_timeout
Get number of milliseconds that elapse between end of knob turning and interrupt firing

Returns the timeout value

Return type integer

is_connected()
Determine if a Tesit device is conntected to the system..

Returns True if the device is connected, otherwise False.

Return type bool

is_pressed()
Returns true if button is currently being pressed

Returns Button pressed state

Return type Boolean

limit
Returns the limit of allowed counts before wrapping. 0 is disabled

Returns The limit

Return type integer

moved
Returns true if knob has been twisted

Returns Moved state

Return type Boolean

pressed
Returns true if button is currently being pressed

Returns Button pressed state

Return type Boolean

red
Gets the red color of the encoder LEDs

Returns Red component of the color

set_blue(*blue*)
Sets the blue color of the encoder LEDs

Parameters **blue** – blue component

Returns No return value

set_color(*red*, *green*, *blue*)
Sets the color of the encoder LEDs

Parameters

- **red** – Red component

- **green** – Green component
- **blue** – Blue component

Returns No return value

set_connect_blue(*blue*)
Sets the connect blue color of the encoder LEDs

Parameters **blue** – blue component

Returns No return value

set_connect_green(*green*)
Sets the connect green color of the encoder LEDs

Parameters **green** – Green component

Returns No return value

set_connect_red(*red*)
Sets the connect red color of the encoder LEDs

Parameters **red** – Red component

Returns No return value

set_count(*amount*)
Set the encoder count to a specific amount

Parameters **amount** – the value to set the counter to

Returns no return value

set_green(*green*)
Sets the green color of the encoder LEDs

Parameters **green** – Green component

Returns No return value

Return type integer

set_int_timeout(*timeout*)
Set number of milliseconds that elapse between end of knob turning and interrupt firing

Parameters **timeout** – the timeout value in milliseconds

Returns No return value

set_limit(*amount*)
Set the encoder count limit to a specific amount

Parameters **amount** – the value to set the limit to

Returns no return value

set_red(*red*)
Sets the red color of the encoder LEDs

Parameters **red** – Red component

Returns No return value

since_last_movement(*clear_value=True*)
Returns the number of milliseconds since the last encoder movement By default, clear the current value

Parameters **clearValue** – Clear out the value? True by default

Returns time since last encoder movement

Return type integer

since_last_press(*clear_value=True*)

Returns the number of milliseconds since the last button event (press and release) By default, clear the current value

Parameters **clearValue** – Clear out the value? False by default

Returns time since last button press

Return type integer

version

Returns a integer of the firmware version number

Returns The firmware version

Return type integer

was_clicked()

Returns true if a click event has occurred

Returns Click event state

Return type Boolean

6.2 Read a Position and Button State

Listing 1: examples/qwiic_twist_ex1.py

```
#!/usr/bin/env python
#------------------------------------------------------------------------------
# qwiic_twist_ex1.py
#
# Simple Example for the Qwiic Twist Device
#-------------------------------------------------------------------------
#
# Written by  SparkFun Electronics, May 2019
#
# This python library supports the SparkFun Electroncis qwiic
# qwiic sensor/board ecosystem on a Raspberry Pi (and compatable) single
# board computers.
#
# More information on qwiic is at https://www.sparkfun.com/qwiic
#
# Do you like this library? Help support SparkFun. Buy a board!
#
#==============================================================================
# Copyright (c) 2019 SparkFun Electronics
#
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
```

(continues on next page)

(continued from previous page)

```
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.
#==============================================================================
# Example 1
#

from __future__ import print_function
import qwiic_twist
import time
import sys

def runExample():

        print("\nSparkFun qwiic Twist   Example 1\n")
        myTwist = qwiic_twist.QwiicTwist()

        if myTwist.connected == False:
                print("The Qwiic twist device isn't connected to the system. Please␣
↪check your connection", \
                        file=sys.stderr)
                return

        myTwist.begin()

        # Set the knob color to pink
        myTwist.set_color(100, 10, 50)

        while True:

                print("Count: %d, Pressed: %s" % (myTwist.count, \
                        "YES" if myTwist.pressed else "NO", \
                        ))

                time.sleep(.3)

if __name__ == '__main__':
        try:
                runExample()
        except (KeyboardInterrupt, SystemExit) as exErr:
                print("\nEnding Example 1")
                sys.exit(0)


```

6.3 Random Colors

Listing 2: examples/qwiic_twist_ex2.py

```
#!/usr/bin/env python
#------------------------------------------------------------------------------
# qwiic_twist_ex2.py
#
# Simple Example for the Qwiic Twist Device
#------------------------------------------------------------------------------
#
# Written by  SparkFun Electronics, May 2019
#
# This python library supports the SparkFun Electroncis qwiic
# qwiic sensor/board ecosystem on a Raspberry Pi (and compatable) single
# board computers.
#
# More information on qwiic is at https://www.sparkfun.com/qwiic
#
# Do you like this library? Help support SparkFun. Buy a board!
#
#==============================================================================
# Copyright (c) 2019 SparkFun Electronics
#
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.
#==============================================================================
# Example 2
#

from __future__ import print_function
import qwiic_twist
import time
import random
import sys

def runExample():

        print("\nSparkFun qwiic Twist   Example 2 - crazy colors\n")
        myTwist = qwiic_twist.QwiicTwist()

        if myTwist.connected == False:
                print("The Qwiic twist device isn't connected to the system. Please␣
↪check your connection", \
```

(continues on next page)

(continued from previous page)

```
                        file=sys.stderr)
                return

        myTwist.begin()

        while True:

                print("Count: %d, Pressed: %s" % (myTwist.count, \
                        "YES" if myTwist.pressed else "NO", \
                        ))

                myTwist.set_color( random.randint(0,256), random.randint(0,256),
↪random.randint(0,256))

                time.sleep(.3)

if __name__ == '__main__':
        try:
                runExample()
        except (KeyboardInterrupt, SystemExit) as exErr:
                print("\nEnding Example 2 - Crazy Colors")
                sys.exit(0)


```

6.4 Twist Connected Colors

Listing 3: examples/qwiic_twist_ex3.py

```
#!/usr/bin/env python
#------------------------------------------------------------------------------
# qwiic_twist_ex3.py
#
# Simple Example for the Qwiic Twist Device
#------------------------------------------------------------------------
#
# Written by  SparkFun Electronics, May 2019
#
# This python library supports the SparkFun Electroncis qwiic
# qwiic sensor/board ecosystem on a Raspberry Pi (and compatable) single
# board computers.
#
# More information on qwiic is at https://www.sparkfun.com/qwiic
#
# Do you like this library? Help support SparkFun. Buy a board!
#
#==============================================================================
# Copyright (c) 2019 SparkFun Electronics
#
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
```

(continues on next page)

(continued from previous page)

```
# furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.
#==============================================================================
# Example 1
#

from __future__ import print_function
import qwiic_twist
import time
import sys

def runExample():

        print("\nSparkFun qwiic Twist   Example 3\n")
        myTwist = qwiic_twist.QwiicTwist()

        if myTwist.connected == False:
                print("The Qwiic twist device isn't connected to the system. Please␣
↪check your connection", \
                        file=sys.stderr)
                return

        myTwist.begin()

        myTwist.set_color(255/2, 0, 255/2) #Set Red and Blue LED brightnesses to half␣
↪of max.

        myTwist.connect_red = -10  # Red LED will go down 10 in brightness with each␣
↪encoder tick
        myTwist.connect_blue = 10 #Blue LED will go up 10 in brightness with each␣
↪encoder tick

        while True:

                print("Count: %d, Pressed: %s" % (myTwist.count, \
                        "YES" if myTwist.pressed else "NO", \
                        ))

                time.sleep(.3)

if __name__ == '__main__':
        try:
                runExample()
        except (KeyboardInterrupt, SystemExit) as exErr:
                print("\nEnding Example 3")
                sys.exit(0)

```

(continues on next page)

(continued from previous page)

```
79
```

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

q

qwiic_twist, 13

Index

B

`begin()` (*qwiic_twist.QwiicTwist method*), 13
`blue` (*qwiic_twist.QwiicTwist attribute*), 13

C

`clear_interrupts()` (*qwiic_twist.QwiicTwist method*), 14
`clicked` (*qwiic_twist.QwiicTwist attribute*), 14
`connect_blue` (*qwiic_twist.QwiicTwist attribute*), 14
`connect_color()` (*qwiic_twist.QwiicTwist method*), 14
`connect_green` (*qwiic_twist.QwiicTwist attribute*), 14
`connect_red` (*qwiic_twist.QwiicTwist attribute*), 14
`connected` (*qwiic_twist.QwiicTwist attribute*), 14
`count` (*qwiic_twist.QwiicTwist attribute*), 14

G

`get_blue()` (*qwiic_twist.QwiicTwist method*), 14
`get_connect_blue()` (*qwiic_twist.QwiicTwist method*), 14
`get_connect_green()` (*qwiic_twist.QwiicTwist method*), 15
`get_connect_red()` (*qwiic_twist.QwiicTwist method*), 15
`get_count()` (*qwiic_twist.QwiicTwist method*), 15
`get_diff()` (*qwiic_twist.QwiicTwist method*), 15
`get_green()` (*qwiic_twist.QwiicTwist method*), 15
`get_int_timeout()` (*qwiic_twist.QwiicTwist method*), 15
`get_limit()` (*qwiic_twist.QwiicTwist method*), 15
`get_red()` (*qwiic_twist.QwiicTwist method*), 15
`get_version()` (*qwiic_twist.QwiicTwist method*), 15
`green` (*qwiic_twist.QwiicTwist attribute*), 15

H

`has_moved()` (*qwiic_twist.QwiicTwist method*), 15

I

`int_timeout` (*qwiic_twist.QwiicTwist attribute*), 16
`is_connected()` (*qwiic_twist.QwiicTwist method*), 16
`is_pressed()` (*qwiic_twist.QwiicTwist method*), 16

L

`limit` (*qwiic_twist.QwiicTwist attribute*), 16

M

`moved` (*qwiic_twist.QwiicTwist attribute*), 16

P

`pressed` (*qwiic_twist.QwiicTwist attribute*), 16

Q

`qwiic_twist` (*module*), 13
`QwiicTwist` (*class in qwiic_twist*), 13

R

`red` (*qwiic_twist.QwiicTwist attribute*), 16

S

`set_blue()` (*qwiic_twist.QwiicTwist method*), 16
`set_color()` (*qwiic_twist.QwiicTwist method*), 16
`set_connect_blue()` (*qwiic_twist.QwiicTwist method*), 17
`set_connect_green()` (*qwiic_twist.QwiicTwist method*), 17
`set_connect_red()` (*qwiic_twist.QwiicTwist method*), 17
`set_count()` (*qwiic_twist.QwiicTwist method*), 17
`set_green()` (*qwiic_twist.QwiicTwist method*), 17
`set_int_timeout()` (*qwiic_twist.QwiicTwist method*), 17
`set_limit()` (*qwiic_twist.QwiicTwist method*), 17
`set_red()` (*qwiic_twist.QwiicTwist method*), 17
`since_last_movement()` (*qwiic_twist.QwiicTwist method*), 17
`since_last_press()` (*qwiic_twist.QwiicTwist method*), 18

V

`version` (*qwiic_twist.QwiicTwist attribute*), 18

W

`was_clicked()` (*qwiic_twist.QwiicTwist method*), 18