sparkfun$_{q}wiic$

Release 0.9.1

Jul 14, 2021

Contents:

The SparkFun qwiic python package aggregates all python qwiic drivers/modules to provide a single entity for qwiic within a python environment. The qwiic package delivers the high-level functionality needed to dynamically discover connected qwiic devices and construct their associated driver object.

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

CHAPTER 1

Contents

CHAPTER 2

Supported Platforms

The qwiic Python package current supports the following platforms:

- Raspberry Pi
- NVidia Jetson Nano
- Google Coral Development Board

CHAPTER 3

Structure

Each qwiic board has an independent driver library that implements the required logic for the specific board. This driver implementation is structured as a python package that supports standard python package management operations and tools. Additionally, each driver is deployed in a distinct GitHub repository which provides a central area for package management and development.

To provide dynamic discovery and instantiation capabilities, the qwiic package imports all the underlying qwiic driver packages at runtime. As such the qwiic driver packages must be installed prior to using this package. These packages can be installed manually, or the overall package will install them automatically when using a PyPi based package manger (aka pip).

CHAPTER 4

Dependent Modules

To make development and evaluation easer, the modules this package is dependent on are included in this repository as git submodules. This allows rapid checkout and access to the entire qwiic python ecosystem if needed.

This structure has the following layout:

```
Qwiic_Py/
   |
   +--- qwiic_i2c/                                    --> Link to the qwiic_i2c␣
↪submodule repository
   |      |--- __index__.py
   |      `--- ... The cross platform I2C bus access driver
   |
   +--- qwiic/
   |      |--- __index__.py
   |      +--- ... Package Implementation
   |      `--- drivers/
   |            |--- qwiic_bme280               --> The qwiic_bme280 submodule
   |            |--- qwiic_micro_oled           --> The qwiic_micro_oled submodule
   |            `--- ... links to qwiic driver submodule repositories
   |
   +--- README.md
   +--- setup.py
   `--- ...etc
```

CHAPTER 5

Dependencies

The qwiic package depends on the qwiic I2C driver: Qwiic_I2C_Py

This package is also dependent on the driver packages contained in the drivers directory.

CHAPTER 6

Documentation

The SparkFun qwiic package documentation is hosted at ReadTheDocs

CHAPTER 7

Checkout Commands

To clone this repository, a standard git clone command will create a local copy of this repository:

```
git clone https://github.com/sparkfun/Qwiic_Py
```

This will create a local version of this repository, but the submodule directories (drivers/*, and qwiic_i2c/) will be empty. To clone the git repository and include the submodule contents, use the following command:

```
git clone --recurse-submodules https://github.com/sparkfun/Qwiic_Py.git
```

CHAPTER 8

Installation

8.1 PyPi Installation

This repository is hosted on PyPi as the sparkfun-qwiic package. On systems that support PyPi installation via pip, this package is installed using the following commands For all users (note: the user must have sudo privileges):

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

For the current user:

```
pip install sparkfun-qwiic
```

This process will also install all modules the qwiic package requires for operation, including the needed qwiic driver packages.

8.2 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_-<version>.tar.gz
```

CHAPTER 9

Table of Contents

9.1 API Reference

9.1.1 qwiic

The SparkFun qwiic python package aggregates all python qwiic drivers/modules to provide a single entity for qwiic within a python environment. The qwiic package delivers the high-level functionality needed to dynamically discover connected qwiic devices and construct their associated driver object.

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

`qwiic.`**`create_device`**(*device=None*)

Used to create a device object for a specific qwiic device

Parameters **`device`** – The I2C address (int), Name or Class name (str) of the device to created, or selection from listed connected devices (tuple). The device address should be an integer value, of the hex value. For an example, for an *0x60* (hex) address enter *qwiic.create_device(60)*.

Returns A qwiic device object for the specified qwiic device, using the I2C address that the device is connected at. If the specified device isn't found, None is returned.

Return type Object

Example

```
>>> import qwiic
>>> results = qwiic.list_devices()
>>> print(results)
[(16, 'Qwiic 4m Distance Sensor (ToF)', 'QwiicVL53L1X'), (16, 'Qwiic Titan GPS',
 ↪'QwiicTitanGps'), (41, 'Qwiic 4m Distance Sensor (ToF)', 'QwiicVL53L1X'), (60,
 ↪'Qwiic Micro OLED (64x48)', 'QwiicMicroOled'), (60, 'Qwiic 4m Distance Sensor␣
 ↪(ToF)', 'QwiicVL53L1X'), (60, 'SSD1306 Display Driver', 'QwiicSSD1306Driver'),␣
 ↪(60, 'Qwiic OLED Display (128x32)', 'QwiicOledDisplay'), (61, 'Qwiic Micro OLED␣
 ↪(64x48)', 'QwiicMicroOled'), (61, 'Qwiic 4m Distance Sensor (ToF)',
 ↪'QwiicVL53L1X'), (61, 'SSD1306 Display Driver', 'QwiicSSD1306Driver'), (61,
 ↪'Qwiic OLED Display (128x32)', 'QwiicOledDisplay')]
```

(continues on next page)

(continued from previous page)

```
>>> mydevice1 = qwiic.create_device(results[47])
>>> print(mydevice1)
<qwiic_micro_oled.qwiic_micro_oled.QwiicMicroOled object at 0x743558f0>
>>> mydevice2 = qwiic.create_device(41)
>>> print(mydevice2)
<qwiic_vl53l1x.QwiicVL53L1X object at 0x743e7bb0>
>>> mydevice3 = qwiic.create_device("QwiicTitanGps")
================================================================================
Message about Titan GPS package...
================================================================================
>>> print(mydevice3)
<qwiic_titan_gps.QwiicTitanGps object at 0x739c4430>
```

qwiic.**get_devices**()

Used to create device objects for all qwiic devices attached to the computer.

Returns A list of qwiic device objects. If no qwiic devices are an empty list is returned.

Return type list

Example

```
>>> import qwiic
```

```
>>> qwiic.get_devices()
[<qwiic_micro_oled.qwiic_micro_oled.QwiicMicroOled at 0x76081ef0>,
<qwiic_ccs811.QwiicCcs811 at 0x752b78b0>,
<qwiic_proximity.QwiicProximity at 0x752b0e10>,
<qwiic_bme280.QwiicBme280 at 0x752b0a30>]
```

qwiic.**list_available_drivers**(*device_address=None*)

Returns a list of known drivers/packages for qwiic devices.

Parameters **device_address** – A list with an I2C address or addresses. If no value was given, the I2C bus will be scanned and the address(es) of the connected qwiic devices will be used.

Returns A list of qwiic drivers/packages associated with the address(es) in the list. Each element of the list a tuple that contains the following values (Device I2C Address, Device Name, Device Driver Class Name)

Return type list

Example

```
>>> import qwiic
>>> qwiic.list_available_drivers([32])
[(32, 'Qwiic GPIO', 'QwiicGPIO'),
(32, 'Qwiic 4m Distance Sensor (ToF)', 'QwiicVL53L1X'),
(32, 'SparkFun Qwiic Joystick', 'QwiicJoystick')]
>>> qwiic.list_available_drivers([61,91])
[(61, 'Qwiic Micro OLED', 'QwiicMicroOled'),
(61, 'Qwiic 4m Distance Sensor (ToF)', 'QwiicVL53L1X'),
(91, 'Qwiic PCA9685', 'QwiicPCA9685'),
(91, 'Qwiic 4m Distance Sensor (ToF)', 'QwiicVL53L1X'),
(91, 'Qwiic CCS811', 'QwiicCcs811')]
```

qwiic.**list_devices**()

Returns a list of known qwiic driver/packages for I2c address(es) of device(s) connected to the I2C bus.

Returns A list of the qwiic devices associated with the I2C address(es) scanned from the I2C bus. Each element of the list a tuple that contains the following values (Device I2C Address, Device Name, Device Driver Class Name) If no devices are attached, an empty list is returned.

Return type list

Example

```
>>> import qwiic
>>> qwiic.list_devices()
[(61, 'Qwiic Micro OLED', 'QwiicMicroOled'),
(61, 'Qwiic 4m Distance Sensor (ToF)', 'QwiicVL53L1X'),
(91, 'Qwiic PCA9685', 'QwiicPCA9685'),
(91, 'Qwiic 4m Distance Sensor (ToF)', 'QwiicVL53L1X'),
(91, 'Qwiic CCS811', 'QwiicCcs811'),
(96, 'Qwiic PCA9685', 'QwiicPCA9685'),
(96, 'Qwiic 4m Distance Sensor (ToF)', 'QwiicVL53L1X'),
(96, 'Qwiic Proximity Sensor', 'QwiicProximity'),
(119, 'Qwiic PCA9685', 'QwiicPCA9685'),
(119, 'Qwiic 4m Distance Sensor (ToF)', 'QwiicVL53L1X'),
(119, 'Qwiic Mux', 'QwiicTCA9548A'),
(119, 'Qwiic BME280', 'QwiicBme280')]
```

qwiic.**scan**()

Used to scan the I2C bus, returning a list of I2C address attached to the computer.

Returns A list of I2C addresses. If no devices are attached, an empty list is returned.

Return type list

Example

```
>>> import qwiic
>>> [2]: qwiic.scan()
[61, 91, 96, 119]
```

CHAPTER 10

Indices and tables

- genindex
- modindex
- search

Python Module Index

q

Index