



Tools for maintaining an open source Python project

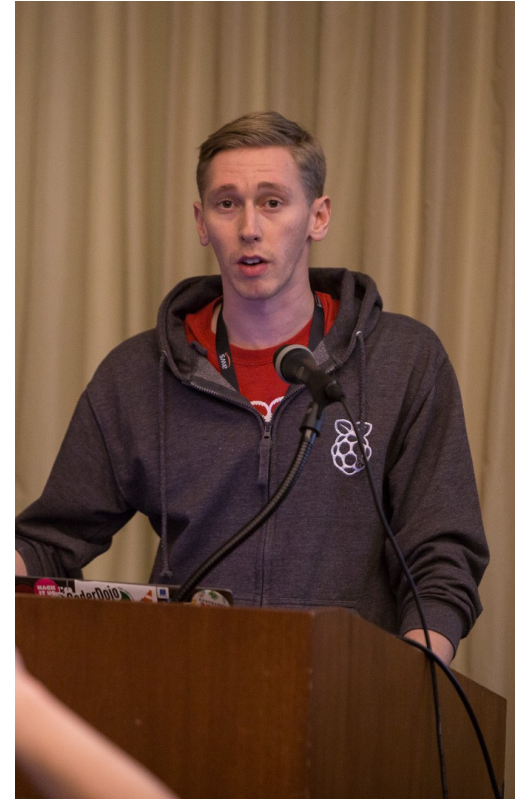
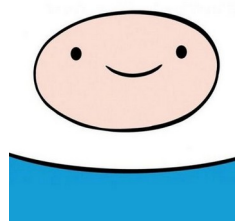
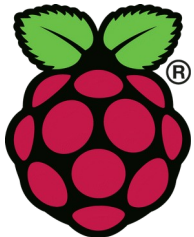
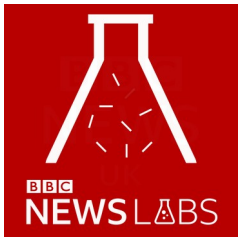
Ben Nuttall

@ben_nuttall



Ben Nuttall

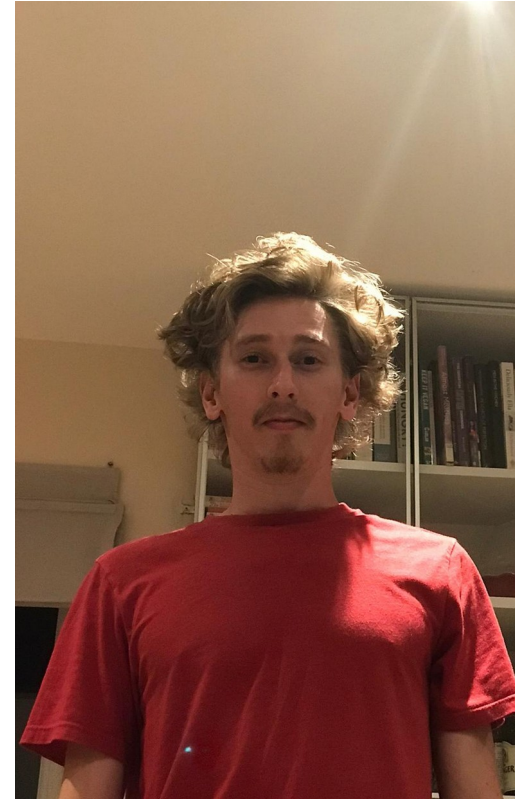
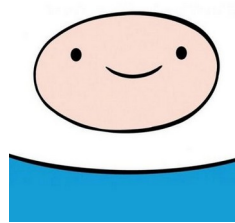
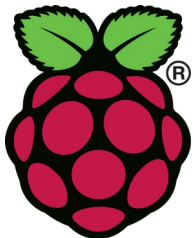
- Software engineer at BBC News Labs
- Formerly at Raspberry Pi Foundation
- Creator of gpiozero, piwheels and pyjokes
- Opensource.com correspondent
- twitter.com/ben_nuttall
- github.com/bennuttall



@ben_nuttall

Ben Nuttall

- Software engineer at BBC News Labs
- Formerly at Raspberry Pi Foundation
- Creator of gpiozero, piwheels and pyjokes
- Opensource.com correspondent
- twitter.com/ben_nuttall
- github.com/bennuttall



@ben_nuttall

What this talk covers

- Organising a Python module
- Distributing software
- Using git/GitHub
- Virtual environments
- Testing & automated testing
- Documentation
- Licensing software

What this talk is not

- A thorough follow-along tutorial on how to use each of the ~50 tools mentioned
- Me telling you which tools to use
- Me telling you that you need to know all of these tools inside-out in order to be considered a proper Python programmer



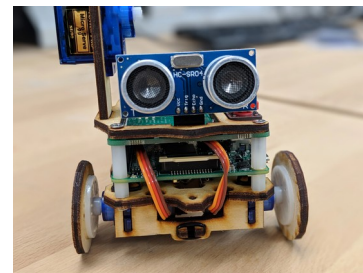
gpiozero

- Python library providing simple API for physical computing with Raspberry Pi
- Eases the learning curve for young people, beginners and educators
- Nice Pythonic API with advanced tooling for experienced programmers
- gpiozero.readthedocs.io
- github.com/gpiozero/gpiozero

```
from gpiozero import Button

button = Button(2)

while True:
    if button.is_pressed:
        print("Button is pressed")
    else:
        print("Button is not pressed")
```



Guido van Rossum ✓
@gvanrossum



GPIOzero I love you! gpiozero.readthedocs.io

♥ 360 5:26 AM - Jan 17, 2019



@ben_nuttall

piwheels

- Tooling for automating building wheels of everything on PyPI
- piwheels.org – pip-compatible repository hosting Arm wheels
- Natively compiled Arm wheels built on Pi 3 hardware
- Repository hosted on 1 × Pi serves 1 million downloads per month
- piwheels.org
- github.com/piwheels/piwheels

A screenshot of the piwheels.org website. The top navigation bar includes links for Packages, Package Issues, FAQ, Stats, Blog, Github, Docs, and Twitter. The main content area is titled "Python Wheels for the Raspberry Pi" and describes it as a PyPI package repository for ARM platforms. It includes a statistics table, a list of latest blog posts, and a configuration section with terminal commands for setting up a virtual environment and installing packages.

Package	Count
Wheels	214,958
Downloads (all time)	1,810,213
Downloads (last 30 days)	22,261,478
Downloads (last 30 days)	1,051,997

```
[github]
extra-index-url=https://www.piwheels.org/simple

$ sudo apt install virtualenv python3-virtualenv -y
$ virtualenv -p /usr/bin/python3 testpip
$ source testpip/bin/activate
(testpip) $ pip install scipy
...
(testpip) $ deactivate
$ rm -rf testpip/
```


Dave Jones

- Professional programmer, amateur dentist
- Responsible for implementing my crazy ideas
- I write the first 90%, he writes the next 90%
- Co-author of gpiozero and piwheels (also author of picamera, colorzero, picraft, sense-emu, lars, structa, compoundpi, pisense, pibootctl, ...)
- Introduced me to most of the tools in this talk



Writing a Python module

```
.  
└─ project.py
```

GitHub - personal

bennuttall / meme-overflow

Watch 1 Star 22 Fork 2

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 5 tags

Go to file Add file Code

bennuttall Do some replacements to ensure hashtags are valid 1760184 on 12 May 67 commits

memeoverflow	Do some replacements to ensure hashtags are valid	2 months ago
tests	Do some replacements to ensure hashtags are valid	2 months ago
.gitignore	Add tox config	5 months ago
.travis.yml	It's 'extras_require' not 'extra_requires'	5 months ago
LICENSE.md	Create LICENSE.md	17 months ago
MANIFEST.in	Add description file	17 months ago
Makefile	Release v0.7.1	4 months ago
README.md	Release v0.7.0	4 months ago
coverage.cfg	Add coverage analysis, improve test coverage	5 months ago
db_schema.md	Use hard-coded database of meme IDs instead of retrieving from imgfl...	6 months ago
description.rst	Add academia link to description file	5 months ago

About

Simple framework for Twitter bots creating memes from Stack Exchange questions

[twitter.com/pi_stack](#)

Readme

BSD-3-Clause License

Releases

5 tags

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

@ben_nuttall

GitHub - organisation

The screenshot shows the GitHub repository page for `gpiozero/gpiozero`. At the top, there are navigation links for Code, Issues (119), Pull requests (16), Actions, Projects (2), Wiki, Security, and Insights. The repository is currently on the `master` branch, with 4 branches and 19 tags. A commit by `bennuttall` is highlighted, titled "Correct ButtonBoard examples", with a commit hash of `965c21e` and 1,275 commits. Below the commit list, a file tree is shown with folders like `.github`, `debian`, `docs`, `gpiozero`, `gpiozerocli`, and `tests`, and files like `.gitignore`, `.travis.yml`, `LICENSE.rst`, `MANIFEST.in`, `Makefile`, `README.rst`, and `RELEASE.rst`. On the right side, there is an "About" section describing the project as a simple interface to GPIO devices with Raspberry Pi, a link to `gpiozero.readthedocs.io/`, tags for `gpio`, `raspberry-pi`, `python`, `education`, `zero-boilerplate`, `raspberrypi`, and `physical-computing`, a "Readme" link, and a "BSD-3-Clause License" link. There is also a "Releases" section with 19 tags and a "Create a new release" link, and a "Packages" section with no packages published and a "Publish your first package" link.

gpiozero / gpiozero

Unwatch 73 Unstar 1.1k Fork 217

<> Code Issues 119 Pull requests 16 Actions Projects 2 Wiki Security Insights ...

master 4 branches 19 tags Go to file Add file Code

bennuttall Correct ButtonBoard examples ✓ 965c21e 24 days ago 1,275 commits

📁 .github	Rename some more references to python-gpiozero -> gpio...	10 months ago
📁 debian	Update github links RPi-Distro/python-gpiozero -> gpiozero/...	10 months ago
📁 docs	Add missing Button import to led_button_remote_*.py exam...	last month
📁 gpiozero	Correct ButtonBoard examples	24 days ago
📁 gpiozerocli	Run copyright generator before release	2 years ago
📁 tests	Update copyrights	4 months ago
📄 .gitignore	Don't need these ignore lines	2 years ago
📄 .travis.yml	Target trusty on travis to fix test runner for older Python ver...	10 months ago
📄 LICENSE.rst	Create copyright generator	2 years ago
📄 MANIFEST.in	Get the MANIFEST filenames right	2 years ago
📄 Makefile	Don't use rename in Makefile	14 months ago
📄 README.rst	Add Trafficphat and update Ryantek boards	4 months ago
📄 RELEASE.rst	Rename some more references to python-gpiozero -> gpio...	10 months ago

About ⚙️

A simple interface to GPIO devices with Raspberry Pi

gpiozero.readthedocs.io/

gpio raspberry-pi python education zero-boilerplate raspberrypi physical-computing

📖 Readme

📄 BSD-3-Clause License

Releases

🏷️ 19 tags

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

@ben_nuttall

GitHub - organisation

The screenshot shows the GitHub repository page for `piwheels/piwheels`. At the top, there are navigation links for `Code`, `Issues` (32), `Pull requests` (5), `Actions`, `Projects` (2), `Wiki`, `Security`, `Insights`, and `Settings`. Below the navigation, there are buttons for `Go to file`, `Add file`, and `Code`. The repository is currently on the `master` branch, with 6 branches and 13 tags. The main content area displays a list of files and folders, each with a commit message and a timestamp. The files include `.github`, `docs`, `piwheels`, `tests`, `.gitignore`, `.travis.yml`, `CONTRIBUTING.md`, `LICENSE.txt`, `MANIFEST.in`, `Makefile`, `README.rst`, and `coverage.cfg`. On the right side, there is an `About` section with a description: "Python package repository providing wheels (pre-built binaries) for Raspberry Pi". Below the description, there is a link to `www.piwheels.org/` and several tags: `python`, `raspberrypi`, `wheels`, `pypi`, `arm`, `raspbian`, and `raspberry-pi`. There are also links for `Readme` and `View license`. The `Releases` section shows 13 tags and a link to `Create a new release`. The `Packages` section indicates that no packages have been published and provides a link to `Publish your first package`.

piwheels / piwheels

Unwatch 10 Unstar 156 Fork 16

<> Code Issues 32 Pull requests 5 Actions Projects 2 Wiki Security Insights Settings

master 6 branches 13 tags

Go to file Add file Code

waveform80 Use a stylesheet appropriate to piwheels c114cfd 14 days ago 729 commits

.github	Update config.yml	2 months ago
docs	Add tests for package description	17 days ago
piwheels	Use a stylesheet appropriate to piwheels	14 days ago
tests	Add tests for package description	17 days ago
.gitignore	Update gitignore	25 days ago
.travis.yml	Into the matrix	9 months ago
CONTRIBUTING.md	Update to new github URL and new package issue tracker URL	13 months ago
LICENSE.txt	Incorporate cbor2 library directly	2 years ago
MANIFEST.in	Exclude default Raspbian Stretch libraries from project page	17 months ago
Makefile	Make the test-bed a bit more flexible	25 days ago
README.rst	Merge pull request #91 from piwheels/status-badges	last month
coverage.cfg	All tests passing again	2 years ago

About

Python package repository providing wheels (pre-built binaries) for Raspberry Pi

www.piwheels.org/

python raspberrypi wheels pypi arm raspbian raspberry-pi

Readme

View license

Releases

13 tags

Create a new release

Packages

No packages published

Publish your first package



@ben_nuttall

GitHub - collaborators

Options
Manage access
Security & analysis
Branches
Webhooks
Notifications
Integrations
Deploy keys
Secrets
Actions


Moderation
Interaction limits
Reported content

Who has access

PUBLIC REPOSITORY 	BASE ROLE Read	DIRECT ACCESS 
This repository is public and visible to anyone. Manage	All 3 members can access this repository. Manage	0 teams or members have access this repository. Only Owners can contribute to this repository.

Manage access

[Create team](#)



You haven't added any teams or people yet

Organization owners can manage individual and team access to the organization's repositories. Team maintainers can also manage a team's repository access. [Learn more about organization access](#)

[Invite teams or people](#)

GitHub - branches

Default branch						
<code>master</code>	Updated a month ago by waveform80	✓		Default		Change default branch

Your branches						
<code>rgbledboard</code>	Updated 2 months ago by bennuttall	✗	196 10		#631	Open
<code>lurch-patch-1</code>	Updated a month ago by lurch	✗	0 1		#752	Open

Active branches						
<code>coverage_bump</code>	Updated a month ago by lurch	✓	0 1		#754	Open
<code>lurch-patch-1</code>	Updated a month ago by lurch	✗	0 1		#752	Open
<code>rgbledboard</code>	Updated 2 months ago by bennuttall	✗	196 10		#631	Open

GitHub - releases

Releases Tags

on 12 Feb  **v1.5.0** 

 9cba4aa  zip  tar.gz

on 21 Feb 2018  **v1.4.1** 

 5085b9c  zip  tar.gz





on 26 Jul 2017  **v1.4.0** 

 f791d22  zip  tar.gz

on 3 Mar 2017  **v1.3.2** 

 2e7543d  zip  tar.gz

GitHub - issues

<input type="checkbox"/>	🔔 89 Open ✓ 321 Closed	Author ▾	Projects ▾	Labels ▾	Milestones ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	🔔 ButtonBoard release events? bug						🗨 1
	#761 opened 2 days ago by bennuttall						
<input type="checkbox"/>	🔔 Add support for 5 channels line follower detector suggestion						🗨 2
	#760 opened 4 days ago by imanousar						
<input type="checkbox"/>	🔔 Question: multiprocessing and gpiozero question						🗨 3
	#759 opened 6 days ago by judalvarezca						
<input type="checkbox"/>	🔔 Servo docs don't make it clear that it can be driven to any position documentation						
	#758 opened 6 days ago by lurch						
<input type="checkbox"/>	🔔 Internal device values are still boolean suggestion						🗨 1
	#755 opened on 24 Feb by bennuttall						
<input type="checkbox"/>	🔔 Diagram inconsistencies documentation						
	#753 opened on 17 Feb by lurch						
<input type="checkbox"/>	🔔 Provide a method of setting per-device init params to composite devices suggestion						🗨 3
	#751 opened on 14 Feb by bennuttall						
<input type="checkbox"/>	🔔 Why doesn't pinout -x open in chromium? question						🗨 10
	#748 opened on 13 Feb by bennuttall						
<input type="checkbox"/>	🔔 New Feature Request - GPIO and Desktop Via Ethernet						🗨 2
	#747 opened on 13 Feb by JonnyAlpha						
<input type="checkbox"/>	🔔 Class init documentation inconsistencies documentation						🗨 1
	#746 opened on 13 Feb by lurch						
<input type="checkbox"/>	🔔 Request for better MockPin docs documentation						🗨 16
	#738 opened on 11 Feb by lurch						

@ben_nuttall

GitHub - pull requests

Add StepperMotor #757

Edit

[Open](#) m-alzam wants to merge 4 commits into `RPi-Distro:master` from `m-alzam:master`

Conversation 5

Commits 4

Checks 0

Files changed 3

+219 -1

Commits on Mar 16, 2019

Add StepperMotor

m-alzam committed 10 days ago



ef95f18



Remove *~ files

m-alzam committed 10 days ago



cd4cd83



Commits on Mar 17, 2019

Add StepperMotor documentation

m-alzam committed 9 days ago ✖



09d289a



Commits on Mar 20, 2019

Improve StepperMotor

m-alzam committed 6 days ago ✖



5fdf92a



GitHub - project boards

The screenshot displays five GitHub Project Board columns:

- 0 Triage**: Empty column.
- 7 Questions**:
 - Button bounce_time (#378, opened by lurch, question)
 - Addon boards compatibility question (#349, opened by lurch, question)
 - when_held callback repeatedly called even after button has been released (#697, opened by rommar, question)
 - Software debounce missing input events (#550, opened by thagrol, question)
 - MCP3201 readings are unstable (#564, opened by flohwie, question)
 - TimeOfDay questions (#263, opened by lurch)
- 18 Pending suggestions**:
 - Asymmetric SmoothedInputDevice? (#238, opened by lurch, suggestion)
 - RFC: Polymorphic Parameters for CompositeDevice subclasses (#219, opened by lurch, suggestion)
 - Add support for MCP23x08, MCP23x17 (#199, opened by elegantandrogynne, suggestion)
 - RGBLED.cycle() (#228, opened by bennuttall, suggestion)
 - Temperature Sensor (#93, opened by bennuttall, help wanted, new device)
 - Add circuit diagrams to docs (#51, opened by bennuttall, documentation, suggestion)
- 2 In progress**:
 - [WIP] Add RGBLEDBoard Class (#631, opened by bennuttall, help wanted, new device)
 - Servo enhancements (#419, opened by lurch, help wanted, suggestion)
- 12 Important / ready**:
 - pinout: rotate option (#723, opened by bennuttall, good first issue, suggestion)
 - Add Stepper Motor (#144, opened by bennuttall, suggestion)
 - Add support for more motors (#330, opened by bennuttall, suggestion)
 - pigpio pwm does not support float frequencies (#712, opened by bennuttall, bug)
 - Allow users to differentiate between pressed and held events (#708, opened by bennuttall, suggestion)
 - Document iterable behaviour (#407, opened by bennuttall, documentation, help wanted, suggestion)

Automated as **To do** Manage

@ben_nuttall

Distributing software – how?

Package managers:

- Linux – apt, rpm, yum
- Language package managers – pip, npm, gem
- Linux portable – snap, flatpak, AppImage
- Mac – homebrew

Download from sites:

- GitHub, GitLab, Sourceforge
- Personal websites
- curl

Distributing software – why?

- **Ease of access**
- **Expectations**
- **Trust and confidence**
- **Stability**

Licensing

- It's important to choose a licence for a project
- Think about what would annoy you
- It's important to specify which licence
- It's important to include the licence with the source code and distributions
- Refer to choosealicense.com

Choose an open source license

An open source license protects contributors and users. Businesses and savvy developers won't touch a project without this protection.

{ Which of the following best describes your situation? }



I need to work in a community.

Use the [license preferred by the community](#) you're contributing to or depending on. Your project will fit right in.

If you have a dependency that doesn't have a license, ask its maintainers to [add a license](#).



I want it simple and permissive.

The [MIT License](#) is short and to the point. It lets people do almost anything they want with your project, including to make and distribute closed source versions.

[Babel](#), [.NET Core](#), and [Rails](#) use the MIT License.



I care about sharing improvements.

The [GNU GPLv3](#) also lets people do almost anything they want with your project, *except* to distribute closed source versions.

[Ansible](#), [Bash](#), and [GIMP](#) use the GNU GPLv3.

{ What if none of these work for me? }

My project isn't software.

[There are licenses for that.](#)

I want more choices.

[More licenses are available.](#)

I don't want to choose a license.

[Here's what happens if you don't.](#)

Packaging a Python module

```
.  
├── project  
│   ├── __init__.py  
│   └── project.py  
├── README.rst  
└── setup.py
```

Packaging a Python module – setup.py

```
import os
from setuptools import setup, find_packages

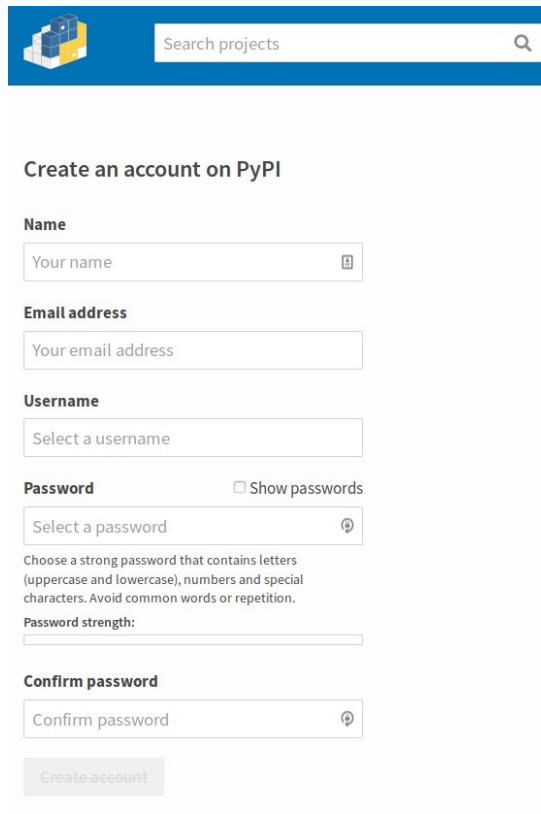
def read(fname):
    return open(os.path.join(os.path.dirname(__file__), fname)).read()

setup(
    name="project",
    version="0.1.0",
    author="Ben Nuttall",
    description="Really cool project",
    license="MIT",
    keywords=["sample", "project"],
    url="https://github.com/bennuttall/project",
    packages=find_packages(),
    long_description=read('README.rst'),
)
```

Publishing a Python module on PyPI

- Register an account on pypi.org
- Put your account details in ~/.pypirc:

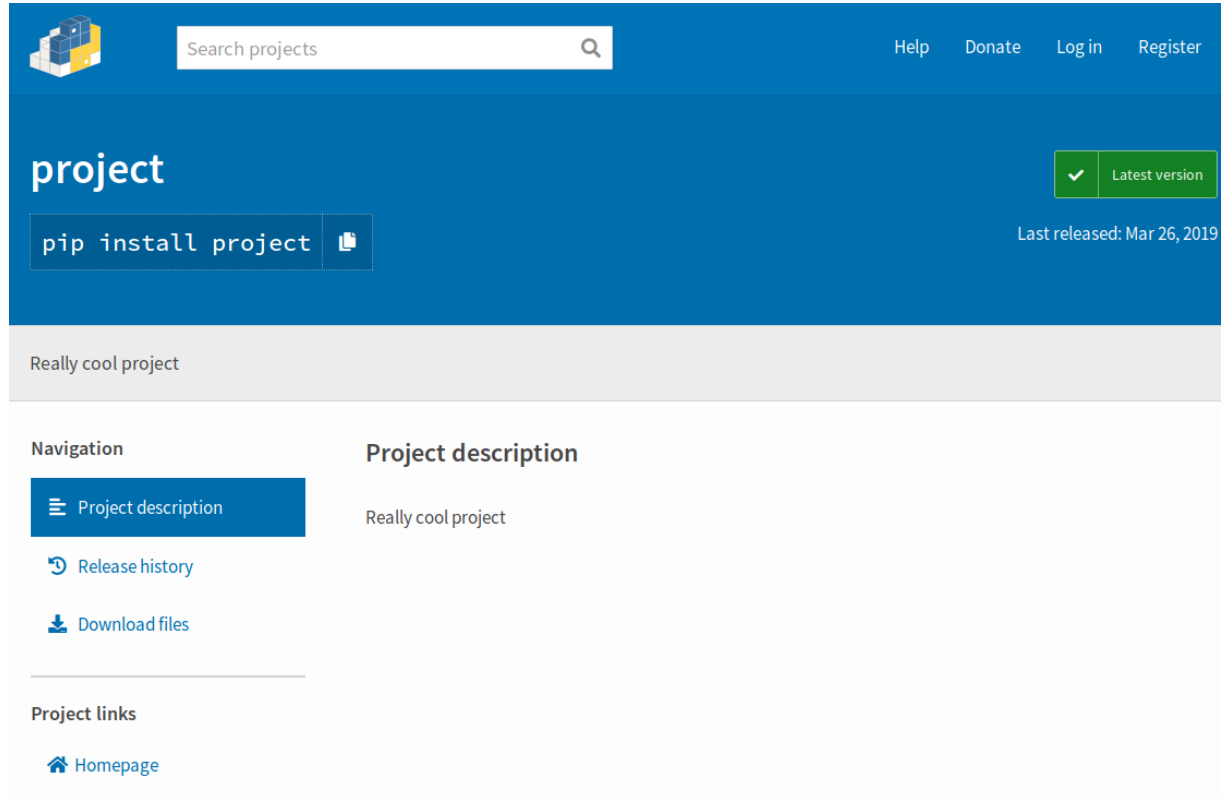
```
ben@magicman:~ $ cat .pypirc  
[pypi]  
username: bennuttall  
password: correcthorsebatterystaple
```
- Install Twine:
 - pip install twine



The screenshot shows the PyPI website's account creation page. At the top, there is a search bar with the text "Search projects" and a magnifying glass icon. Below the search bar, the heading "Create an account on PyPI" is displayed. The form consists of several input fields: "Name" (with placeholder "Your name"), "Email address" (with placeholder "Your email address"), "Username" (with placeholder "Select a username"), and "Password" (with placeholder "Select a password" and a "Show passwords" checkbox). Below the password field, there is a note: "Choose a strong password that contains letters (uppercase and lowercase), numbers and special characters. Avoid common words or repetition." and a "Password strength" indicator. At the bottom of the form, there is a "Confirm password" field (with placeholder "Confirm password") and a "Create account" button.

@ben_nuttall

Publishing a Python module on PyPI



The screenshot shows a PyPI project page for a package named 'project'. The page has a blue header with the PyPI logo, a search bar, and navigation links for Help, Donate, Log in, and Register. Below the header, the project name 'project' is displayed in large white text on a blue background. To the right of the name is a green button with a checkmark and the text 'Latest version'. Below the name is a dark blue button with the text 'pip install project' and a small icon of a document. To the right of this button is the text 'Last released: Mar 26, 2019'. Below the blue header is a light gray section with the text 'Really cool project'. The main content area is divided into two columns. The left column is titled 'Navigation' and contains three links: 'Project description' (highlighted with a blue background), 'Release history', and 'Download files'. The right column is titled 'Project description' and contains the text 'Really cool project'. Below the navigation links is a section titled 'Project links' with a single link 'Homepage'.

@ben_nuttall

__init__.py choices: gpiozero

User:

```
from gpiozero import LED
```

__init__.py:

```
from .input_devices import LED, ...
```

setup.py

```
__version__ = '1.5.1'
```

```
setup(  
    version=__version__  
    ...  
)
```


__init__.py choices: piwheels

User:

```
from piwheels.master.pypi import  
PiWheelsTransport
```

__init__.py:

```
__version__ = '0.17'
```

setup.py

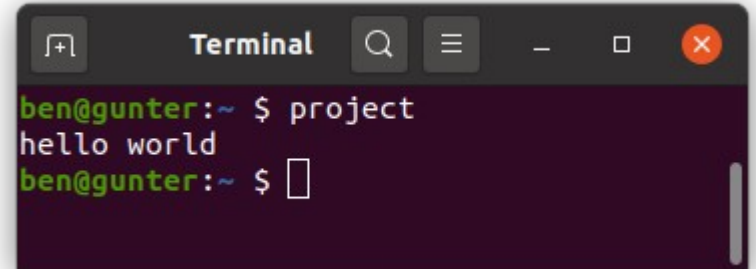
```
import piwheels as app  
  
setup(  
    version=app.__version__,  
    ...  
)
```

Entry points

setup.py:

```
entry_points = {  
    'console_scripts': [  
        'project = project.cli:main',  
    ],  
}
```

```
setup(  
    ...  
    entry_points=entry_points,  
    ...  
)
```



A terminal window titled "Terminal" with standard window controls (maximize, search, menu, close). The prompt is "ben@gunter:~". The user enters "project" and the output is "hello world". The prompt returns to "ben@gunter:~ \$".

```
ben@gunter:~ $ project  
hello world  
ben@gunter:~ $
```

Virtual environments

- Virtual environment for a Python project
- You create the environment, pip install into it
- Isolated from your system Python and system packages
- Build your project inside it, with changes "installed" in real time

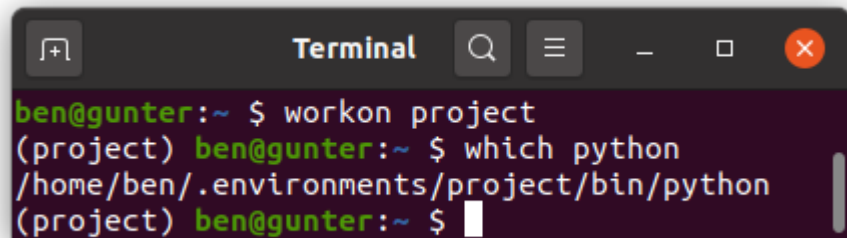
```
mkvirtualenv -p /usr/bin/python3 project
```

```
pip install -e .
```

```
pip install ipython
```

```
deactivate
```

```
workon project
```

A terminal window titled "Terminal" with standard window controls (maximize, search, menu, close). The terminal shows the following commands and output:

```
ben@gunter:~ $ workon project
(project) ben@gunter:~ $ which python
/home/ben/.environments/project/bin/python
(project) ben@gunter:~ $
```

Makefiles

```
all:  
    @echo "make install - Install on local system"  
    @echo "make develop - Install symlinks for development"  
  
install:  
    pip install .  
  
develop:  
    pip install -e .
```

Testing

- Write tests to validate what your code is supposed to do
- Keep your old tests to make sure nothing breaks in future
- For maximum effect, write tests before you write code!
- Testing can be performed quickly locally
- Testing can be automated – e.g. Travis after push
- Be pragmatic! Test edge cases, don't be exhaustive

Testing - assert

```
from project import add
```

```
assert add(2, 2) == 4
```


Testing - pytest

```
from project import add

def test_add():
    assert add(2, 2) == 4
```

Testing - layout

```
.  
├─ Makefile  
├─ project  
│  └─ __init__.py  
│    └─ project.py  
├─ setup.py  
└─ tests  
    └─ test_add.py
```

pytest

```
Terminal
(project) ben@gunter:~/Projects/bennuttall/project $ tree
.
├── Makefile
├── project
│   ├── __init__.py
│   └── project.py
├── setup.py
└── tests
    ├── __pycache__
    │   └── test_add.cpython-38-pytest-5.4.3.pyc
    └── test_add.py

3 directories, 6 files
(project) ben@gunter:~/Projects/bennuttall/project $ pytest -v
===== test session starts =====
platform linux -- Python 3.8.2, pytest-5.4.3, py-1.9.0, pluggy-0.13.1 -- /home/
ben/.environments/project/bin/python
cachedir: .pytest_cache
rootdir: /home/ben/Projects/bennuttall/project
collected 1 item

tests/test_add.py::test_add PASSED [100%]

===== 1 passed in 0.01s =====
(project) ben@gunter:~/Projects/bennuttall/project $
```

Testing - pytest

```
from project import add
import pytest

assert add(2, 2) == 4
with pytest.raises(TypeError):
    add("foo", "bar")
```

Testing – mock

```
>>> from unittest.mock import Mock
>>> m = Mock(msg=Mock(return_value="hello"))
>>> m
<Mock id='140656083514272'>
>>> m.msg()
'hello'
```

Testing – mock patch

```
def test_timeofday_value(mock_factory):
    with TimeOfDay(time(7), time(8), utc=False) as tod:
        assert repr(tod).startswith('<gpiozero.TimeOfDay object')
        assert tod.start_time == time(7)
        assert tod.end_time == time(8)
        assert not tod.utc
    with patch('gpiozero.internal_devices.datetime') as dt:
        dt.now.return_value = datetime(2018, 1, 1, 6, 59, 0)
        assert not tod.is_active
        dt.now.return_value = datetime(2018, 1, 1, 7, 0, 0)
        assert tod.is_active
        dt.now.return_value = datetime(2018, 1, 2, 8, 0, 0)
        assert tod.is_active
        dt.now.return_value = datetime(2018, 1, 2, 8, 1, 0)
        assert not tod.is_active
```

Tox

- Run tests in multiple Python versions simultaneously
- Ubuntu users – look for "Deadsnakes PPA"
- tox.ini:

```
[tox]
```

```
envlist = {py27, py35, py36, py37, py38}
```


Coverage.py

- **Measuring code coverage of Python programs**
- **Monitors your program, noting which parts of the code have been executed**
- **Analyses the source to identify code that could have been executed but was not**
- **Typically used to gauge the effectiveness of tests**
- **Shows which parts of your code are being touched by your tests, and which are not**

Coverage

```
coverage report --rcfile coverage.cfg
Name                               Stmts  Miss Branch BrPart  Cover   Missing
-----
gpiozero/__init__.py                11      0      0      0  100%
gpiozero/boards.py                  463      1     181      2   99%  686, 293->exit, 685->686
gpiozero/compat.py                   96      4      30      0   97%  52-53, 98, 120
gpiozero/devices.py                 232      2      78      1   99%  442-443, 462->461
gpiozero/exc.py                      60      0      0      0  100%
gpiozero/input_devices.py           252     19      36      5   92%  48-49, 138-139, 290, 301-302, 305, 514-516, 597-599, 676-678, 846, 946, 289->290, 304->305, 845->846, 919->922, 937->946
gpiozero/internal_devices.py         141     10      20      0   94%  100-101, 192-193, 294-295, 380-381, 475-476
gpiozero/mixins.py                   236      8      74      1   97%  48-49, 87, 179, 506-508, 558-560, 113->exit
gpiozero/output_devices.py          518     22     169      6   96%  49-50, 171-172, 428-429, 650-651, 665-668, 698, 714-717, 721, 738-739, 759, 1360, 1727, 1786, 661->665, 691->693, 694->698, 1359->1360, 1723->1727, 1779->1786
gpiozero/pins/__init__.py           117     30      51      1   78%  135, 149, 160, 185, 232, 240, 275, 278-279, 298, 301, 330, 333, 350, 353-354, 370, 373-374, 412, 415, 442, 445, 632, 659, 662, 711, 714, 756, 759, 96->94
gpiozero/pins/data.py                305      5     108      5   97%  1048-1049, 1188, 1255, 1331, 713->exit, 1187->1188, 1234->1252, 1295->exit, 1330->1331
gpiozero/pins/local.py               165      8      18      1   95%  48-49, 98, 179-180, 230, 253-254, 97->98
gpiozero/pins/mock.py               302      9     104     16   94%  46-47, 243-248, 472, 474-476, 478, 204->209, 234->exit, 235->exit, 318->exit, 336->exit, 341->exit, 372->374, 374->exit, 388->390, 390->399, 442->444, 466->468, 468->470, 471->472, 473->474, 477->478
gpiozero/pins/native.py              307     192      50      3   34%  52-53, 92-100, 106-107, 110-120, 123, 126, 144-146, 149, 152, 155, 158, 161, 164-202, 205-218, 221-222, 225-229, 244-246, 249-251, 254-255, 261-267, 279-280, 289-299, 330-339, 369-394, 397-401, 404, 407-411, 418, 421-426, 429, 432-446, 449, 452, 455-462, 465-476, 479-480, 483, 259->exit, 260->261, 284->exit
gpiozero/pins/pi.py                  121     10      40      1   93%  44-46, 107-113, 291, 290->291
gpiozero/pins/pigpio.py              297     293      68      0      1%  44-580
gpiozero/pins/rpigpio.py             126     122      30      0      3%  43-253
gpiozero/pins/rpio.py                125     121      32      0      3%  43-247
gpiozero/pins/spi.py                  67      2      28      4   94%  102-103, 57->59, 59->exit, 101->102, 111->114
gpiozero/spi_devices.py              160      7      34      1   96%  45-46, 85-88, 104, 113-114, 103->104
gpiozero/threads.py                  32      2      10      2   90%  77-80, 59->61, 76->77
gpiozero/tones.py                    86      8      28      3   90%  16-17, 91, 108, 116-117, 122-123, 90->91, 94->102, 105->108
gpiozero/tools.py                    231      4     130      4   98%  54-55, 58-59, 466->exit, 497->exit, 648->exit, 675->exit
-----
TOTAL                                4450     879     1319      56   81%
(gpiozero-env) ben@magicman:~/Projects/rpi-distro/python-gpiozero (master) $
```

Testing – Travis CI

gpiozero / gpiozero 


Current Branches Build History Pull Requests > **Build #1412**

✓ **master** Correct ButtonBoard examples ↔ #1412 passed

↗ Commit 965c21e [↗](#) 🕒 Ran for 3 min 10 sec

🔍 Compare 76261e0...965c21e [↗](#) 🕒 Total time 12 min 7 sec

📁 Branch master [↗](#) 📅 24 days ago

 Ben Nuttall

[Build jobs](#) [View config](#)

✓ # 1412.1	AMD64	Trusty	</> Python: 3.6	no environment variables set
✓ # 1412.2	AMD64	Trusty	</> Python: 3.5	no environment variables set
✓ # 1412.3	AMD64	Trusty	</> Python: 3.4	no environment variables set
✓ # 1412.4	AMD64	Trusty	</> Python: 3.3	no environment variables set
✓ # 1412.5	AMD64	Trusty	</> Python: 3.2	no environment variables set
✓ # 1412.6	AMD64	Trusty	</> Python: 2.7	no environment variables set
✓ # 1412.7	AMD64	Trusty	</> Python: pypy	no environment variables set
✓ # 1412.8	AMD64	Trusty	</> Python: pypy3	no environment variables set
✓ # 1412.9	AMD64	Xenial	</> Python: 3.7	no environment variables set

@ben_nuttall

GitHub – Travis CI & codecov integration



codecov-io commented 9 days ago • edited ▾

Codecov Report

Merging #757 into master will decrease coverage by 1.19%.
The diff coverage is 19.78%.



	Coverage	Diff	
@@	master	#757	+/- @@
##			##
- Coverage	79.46%	78.26%	-1.2%
Files	23	23	
Lines	4470	4560	+90
Branches	650	667	+17
+ Hits	3552	3569	+17
- Misses	863	936	+73
Partials	55	55	

Impacted Files	Coverage Δ	
gpiozero/__init__.py	100% <0> (0)	↑
gpiozero/output_devices.py	83.71% <19.78%> (-11.27%)	↓

Default branch

master Updated a month ago by waveform80 ✓

Your branches

rgbledboard Updated 2 months ago by bennuttall ✗

lurch-patch-1 Updated a month ago by lurch ✗

Active branches

coverage_bump Updated a month ago by lurch ✓

lurch-patch-1 Updated a month ago by lurch ✗

rgbledboard Updated 2 months ago by bennuttall ✗

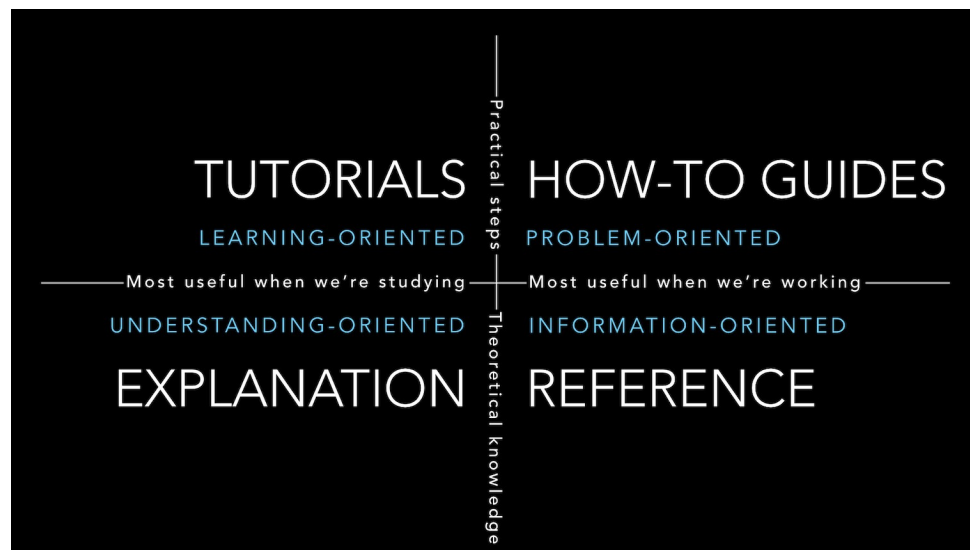
@ben_nuttall

Makefiles

```
all:  
@echo "make install - Install on local system"  
@echo "make develop - Install symlinks for development"  
@echo "make test - Run tests"  
  
install:  
    pip install .  
  
develop:  
    pip install -e .  
  
test:  
    coverage run --rcfile coverage.cfg -m pytest -v tests  
    coverage report --rcfile coverage.cfg
```

Documentation

- Tutorials
- How-to guides
- Explanation
- Reference
- <https://documentation.divio.com/>



Documentation - GitHub

Run your own instance

1. Sign up for a [Twitter](#) account, [create an app](#) and get your four API keys.
2. Sign up for an [Imgflip](#) account and note your username and password.
3. Register for a [Stack Exchange App Key](#)
4. Install this project:

```
sudo pip3 install memoverflow
```

5. Copy the example script `example.py` (e.g. to `raspberrypi.py`) and edit your copy to specify:
 - the Stack Exchange site you wish to follow (get the exact string [here](#)) and your Stack Exchange API key
 - your Twitter account's API keys
 - your imgflip's username and password

6. Run it:

```
python3 raspberrypi.py
```

README.md

Raspberry Pi revision codes

Each distinct Raspberry Pi model revision has a unique revision code. You can look up a Raspberry Pi's revision code by running:

```
cat /proc/cpuinfo
```

The last three lines show the hardware type, the revision code, and the Pi's unique serial number. For example:

```
Hardware : BCM2835
Revision : a02082
Serial   : 00000000765fc593
```

Note: As of the 4.9 kernel, all Pis report `BCM2835`, even those with `BCM2836` and `BCM2837` processors. You should not use this string to detect the processor.

Old-style revision codes

The first set of Raspberry Pi revisions were given sequential hex revision codes from `0002` to `0015`:

Documentation - Markdown

Title

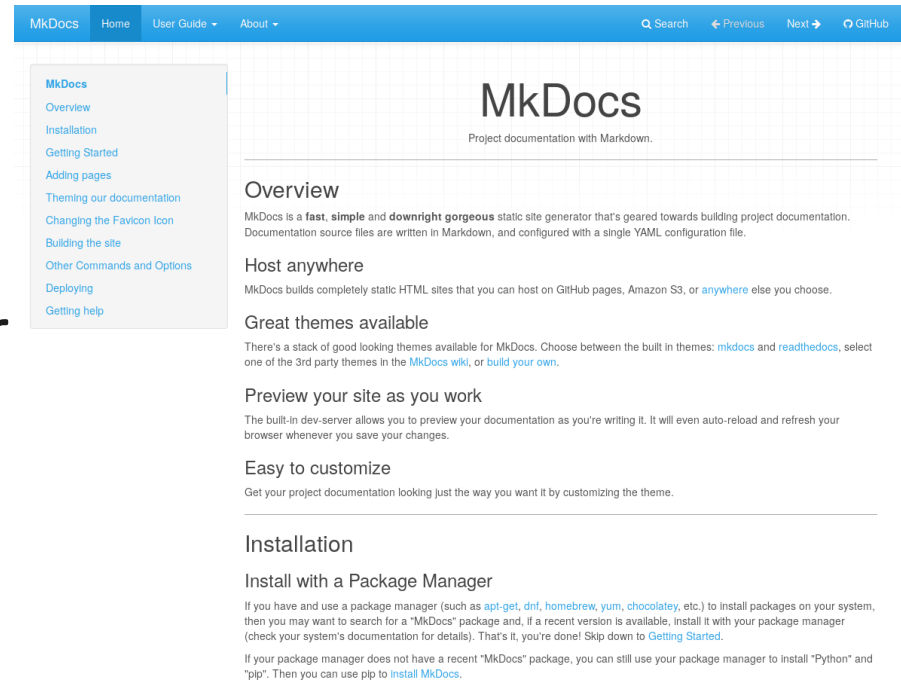
Some text

Header 2

- List item**
- [link](http://foo.com/)**

Documentation - mkdocs

- Markdown-based documentation builder
- Exports to static HTML
- Easy to write, easy to deploy
- Can host anywhere – e.g. GitHub pages or self-hosted



The screenshot shows the MkDocs website interface. At the top, there is a blue navigation bar with links for 'MkDocs', 'Home', 'User Guide', and 'About'. A search bar and navigation arrows are also present. Below the navigation bar is a sidebar menu with the following items: 'MkDocs', 'Overview', 'Installation', 'Getting Started', 'Adding pages', 'Theming our documentation', 'Changing the Favicon Icon', 'Building the site', 'Other Commands and Options', 'Deploying', and 'Getting help'. The main content area features the 'MkDocs' logo and the tagline 'Project documentation with Markdown.' Below this, there are several sections: 'Overview' (describing MkDocs as a fast, simple, and downright gorgeous static site generator), 'Host anywhere' (listing GitHub pages, Amazon S3, and other options), 'Great themes available' (mentioning built-in themes like 'mkdocs' and 'readthedocs'), 'Preview your site as you work' (describing the built-in dev-server), 'Easy to customize' (explaining how to customize the theme), and 'Installation' (with a sub-section 'Install with a Package Manager' that lists various package managers like apt-get, dnf, homebrew, yum, and chocolatey).

Documentation – ReST (ReStructured Text)

```
=====
```

```
Title
```

```
=====
```

```
Some text
```

```
Header 2
```

```
=====
```

```
* List item
```

```
* :doc:`api_input`
```

Documentation - sphinx

- ReST
- Extracts docs from docstrings
- Can embed additional bespoke docs
- Multiple outputs:
 - HTML
 - PDF
 - Epub
- Language docs linking (e.g. gpiozero can link to Python docs using ReST)
- Cross-project linking (e.g. gpiozero can link to picamera using ReST)

The screenshot shows the Sphinx Python Documentation Generator website. The header features the Sphinx logo and the text 'SPHINX Python Documentation Generator'. Navigation links include 'Home', 'Get it', 'Docs', and 'Extend/Develop'. The main content area is titled 'Welcome' and describes Sphinx as a tool for creating intelligent and beautiful documentation. It lists features such as output formats (HTML, LaTeX, ePub, etc.), cross-references, hierarchical structure, automatic indices, code handling, and extensions. A 'What users say' quote is also present. The right sidebar contains a 'Download' section with the current version (v1.8.5) and a 'pip install -U Sphinx' command. Below that is a 'Questions?' section with a link to a mailing list and a 'Subscribe' button. At the bottom, there is a 'Quick search' field with a 'Go' button.

Documentation - sphinx

Regular Classes

=====

The following classes are intended for general use with the devices they represent. All classes in this section are concrete (not abstract).

LED

```
.. autoclass:: LED(pin, *, active_high=True, initial_value=False, pin_factory=None)
   :members: on, off, toggle, blink, pin, is_lit, value
```

PWMLED

```
.. autoclass:: PWMLED(pin, *, active_high=True, initial_value=0, frequency=100, pin_factory=None)
   :members: on, off, toggle, blink, pulse, pin, is_lit, value
```

14. API - Output Devices

These output device component interfaces have been provided for simple use of everyday components. Components must be wired up correctly before use in code.

Note

All GPIO pin numbers use Broadcom (BCM) numbering by default. See the [Pin Numbering](#) section for more information.

14.1. Regular Classes

The following classes are intended for general use with the devices they represent. All classes in this section are concrete (not abstract).

14.1.1. LED

```
class gpiozero.LED(pin, *, active_high=True, initial_value=False, pin_factory=None) \[source\]
```

Extends [DigitalOutputDevice](#) and represents a light emitting diode (LED).

Connect the cathode (short leg, flat side) of the LED to a ground pin; connect the anode (longer leg) to a limiting resistor; connect the other side of the limiting resistor to a GPIO pin (the limiting resistor can be placed either side of the LED).

The following example will light the LED:

```
from gpiozero import LED

led = LED(17)
led.on()
```

Parameters:

- `pin` (*int or str*) - The GPIO pin which the LED is connected to. See [Pin Numbering](#) for valid pin numbers. If this is `None` a `GPIODeviceError` will be raised.
- `active_high` (*bool*) - If `True` (the default), the LED will operate normally with the

Documentation - sphinx

Python » English » 3.8.5 » Documentation » The Python Standard Library » Data Types »

Table of Contents

- datetime** — Basic date and time types
 - Aware and Naive Objects
 - Constants
 - Available Types
 - Common Properties
 - Determining if an Object is Aware or Naive
 - **timedelta** Objects
 - Examples of usage: **timedelta**
 - **date** Objects
 - Examples of Usage: **date**
 - **datetime** Objects
 - Examples of Usage: **datetime**
 - **time** Objects
 - Examples of Usage: **time**
 - **tzinfo** Objects
 - **timezone** Objects
 - **strptime()** and **strftime()** Behavior
 - **strptime()** and **strftime()** Format Codes
 - Technical Detail

Previous topic

Data Types

Next topic

calendar — General calendar-related functions

This Page

Report a Bug
Show Source

datetime — Basic date and time types

Source code: [Lib/datetime.py](#)

The `datetime` module supplies classes for manipulating dates and times.

While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation.

See also:

Module `calendar`

General calendar related functions.

Module `time`

Time access and conversions.

Package `dateutil`

Third-party library with expanded time zone and parsing support.

Aware and Naive Objects

Date and time objects may be categorized as “aware” or “naive” depending on whether or not they include timezone information.

With sufficient knowledge of applicable algorithmic and political time adjustments, such as time zone and daylight saving time information, an **aware** object can locate itself relative to other aware objects. An aware object represents a specific moment in time that is not open to interpretation. [1]

A **naive** object does not contain enough information to unambiguously locate itself relative to other date/time objects. Whether a naive object represents Coordinated Universal Time (UTC), local time, or time in some other timezone is purely up to the program, just like it is up to the program whether a particular number represents metres, miles, or mass. Naive objects are easy to understand and to work with, at the cost of ignoring some aspects of reality.

For applications requiring aware objects, `datetime` and `time` objects have an optional time zone information attribute, `tzinfo`, that can be set to an instance of a subclass of the abstract `tzinfo` class. These `tzinfo` objects capture information about the offset from UTC time, the time zone name, and whether daylight saving

@ben_nuttall

Documentation - ReadTheDocs

- Multiple versions (v1.0, v1.1, v1.2...)
- Branches
- Multi-user management
- Easy to integrate with GitHub to automate building

Read the Docs v: stable

Docs » gpiozero

[Edit on GitHub](#)

gpiozero

pypi package: 1.5.0 build: passing coverage: 79%

A simple interface to GPIO devices with Raspberry Pi.

Created by [Ben Nuttall](#) of the [Raspberry Pi Foundation](#), [Dave Jones](#), and other contributors.

About

Component interfaces are provided to allow a frictionless way to get started with physical computing:

```
from gpiozero import LED
from time import sleep

led = LED(17)

while True:
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```

With very little code, you can quickly get going connecting your components together:

```
from gpiozero import LED, Button
from signal import pause

led = LED(17)
button = Button(3)

button.when_pressed = led.on
button.when_released = led.off

pause()
```

The library includes interfaces to many simple everyday components, as well as some more

@ben_nuttall

Documentation - Graphviz

7. Source/Values

GPIO Zero provides a method of using the declarative programming paradigm to connect devices together: feeding the values of one device into another, for example the values of a button into an LED:

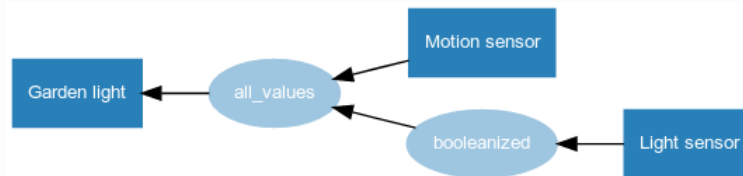


```
from gpiozero import LED, Button
from signal import pause

led = LED(17)
button = Button(2)

led.source = button

pause()
```



```
from gpiozero import LED, MotionSensor, LightSensor
from gpiozero.tools import booleanized, all_values
from signal import pause

garden = LED(2)
motion = MotionSensor(4)
light = LightSensor(5)

garden.source = all_values(booleanized(light, 0, 0.1), motion)

pause()
```

Documentation - Graphviz

```
digraph {  
    graph [rankdir=RL];  
    node [shape=rect, style=filled, color="#2980b9", fontname=Sans, fontcolor="#ffffff", fontsize=10];  
    edge [arrowhead=normal, style=solid];  
  
    Button -> LED;  
}
```



Documentation - Graphviz

```
digraph {
  graph [rankdir=RL];
  edge [arrowhead=normal, style=solid];

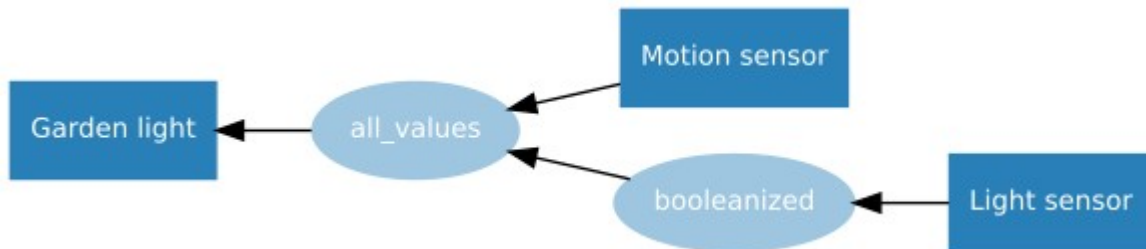
  /* Devices */
  node [shape=rect, style=filled, color="#2980b9", fontname=Sans, fontcolor="#ffffff", fontsize=10];

  led [label="Garden light"]
  light [label="Light sensor"]
  motion [label="Motion sensor"]

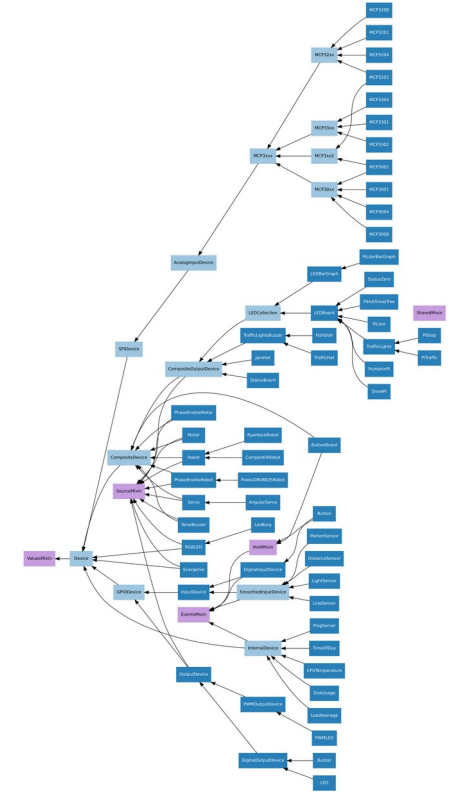
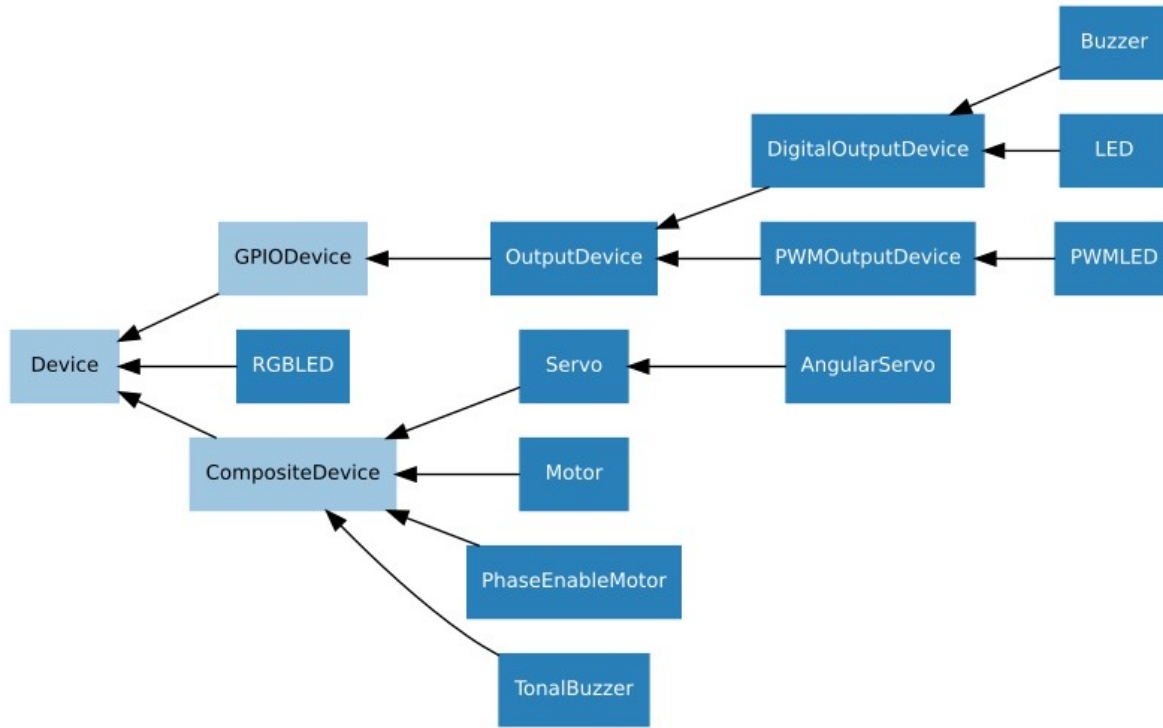
  /* functions */
  node [shape=oval, style=filled, color="#9ec6e0", fontcolor="#ffffff"];

  booleanized
  all_values

  all_values -> led;
  booleanized -> all_values;
  motion -> all_values;
  light -> booleanized;
}
```



Documentation - Graphviz



Project structure

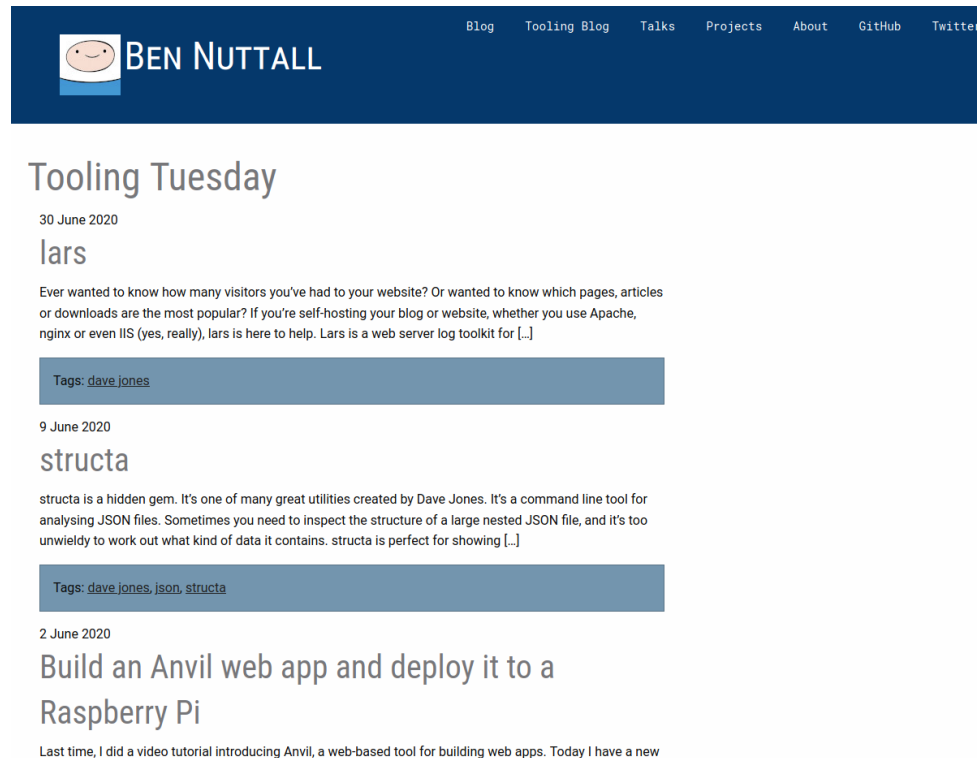
```
.  
├── coverage.cfg  
├── docs/  
├── .github/  
├── .gitignore  
├── LICENSE  
├── Makefile  
├── project/  
├── setup.py  
├── tests/  
├── tox.ini  
└── .travis.yml
```

What this talk covered

- Organising a Python module – module structure, setup.py, Makefiles
- Distributing software – PyPI, pip
- Using git/GitHub – repositories, users & orgs, collaborators, issues, PRs, project boards, integrations
- Virtual environments – virtualenvwrapper
- Testing & automated testing – assert, pytest, mock, coverage, tox, Travis CI
- Documentation – markdown, ReST, mkdocs, sphinx, graphviz
- Licensing software – choosealicense.org

Tooling Tuesday

- My tooling blog:
<https://tooling.bennuttall.com/>
- Inspired by Les Pounder:
<https://bigl.es/>
- ~~New posts every Tuesday~~
- ~~New posts every other Tuesday~~
- New posts every now and then, sometimes on a Tuesday



The screenshot shows the top of a blog post on a dark blue background. On the left is a circular profile picture of a smiling face with a blue background, followed by the name 'BEN NUTTALL' in white. To the right is a navigation menu with links: 'Blog', 'Tooling Blog', 'Talks', 'Projects', 'About', 'GitHub', and 'Twitter'. Below the header, the title 'Tooling Tuesday' is displayed in a light grey font. The date '30 June 2020' is shown in a smaller font. The main title of the post is 'lars' in a large, light grey font. The first paragraph of the post reads: 'Ever wanted to know how many visitors you've had to your website? Or wanted to know which pages, articles or downloads are the most popular? If you're self-hosting your blog or website, whether you use Apache, nginx or even IIS (yes, really), lars is here to help. Lars is a web server log toolkit for [...]'. Below the text is a light blue tag bar containing the text 'Tags: [dave jones](#)'. The date '9 June 2020' is shown in a smaller font. The main title of the post is 'structa' in a large, light grey font. The first paragraph of the post reads: 'structa is a hidden gem. It's one of many great utilities created by Dave Jones. It's a command line tool for analysing JSON files. Sometimes you need to inspect the structure of a large nested JSON file, and it's too unwieldy to work out what kind of data it contains. structa is perfect for showing [...]'. Below the text is a light blue tag bar containing the text 'Tags: [dave jones](#), [json](#), [structa](#)'. The date '2 June 2020' is shown in a smaller font. The main title of the post is 'Build an Anvil web app and deploy it to a Raspberry Pi' in a large, light grey font. The first paragraph of the post reads: 'Last time, I did a video tutorial introducing Anvil, a web-based tool for building web apps. Today I have a new

@ben_nuttall



Tools for maintaining an open source Python project

Ben Nuttall

@ben_nuttall

