
domplotlib

Release 0.4.0

Dom's extensions to matplotlib

Dominic Davis-Foster

Apr 26, 2024

Contents

1	Installation	1
1.1	from PyPI	1
1.2	from GitHub	1
I	API Reference	3
2	domplotlib	5
2.1	create_figure	5
2.2	horizontal_legend	5
2.3	save_svg	5
2.4	transpose	6
3	domplotlib.plots	7
3.1	pie_from_tally	7
4	domplotlib.styles	9
II	Contributing	11
5	Overview	13
6	Coding style	15
7	Automated tests	17
8	Type Annotations	19
9	Build documentation locally	21
10	Downloading source code	23
10.1	Building from source	24
11	License	25
	Python Module Index	27
	Index	29

Installation

1.1 from PyPI

```
$ python3 -m pip install domplotlib --user
```

1.2 from GitHub

```
$ python3 -m pip install git+https://github.com/domdfcoding/domplotlib@master --user
```


Part I

API Reference

Dom's extensions to matplotlib.

Functions:

<code>create_figure(pagesize[, left, bottom, ...])</code>	Creates a figure with the given margins, and returns a tuple of the figure and its axes.
<code>horizontal_legend(fig[, handles, labels, ncol])</code>	Place a legend on the figure, with the items arranged to read right to left rather than top to bottom.
<code>save_svg(figure, fname, *[dpi, facecolor, ...])</code>	Save the given figure as an SVG.
<code>transpose(iterable, ncol)</code>	Transposes the contents of <code>iterable</code> so they are ordered right to left rather than top to bottom.

create_figure (*pagesize*, *left*=0.2, *bottom*=0.14, *right*=0.025, *top*=0.13)

Creates a figure with the given margins, and returns a tuple of the figure and its axes.

Parameters

- **pagesize** (`PageSize`)
- **left** (`float`) – Left margin. Default 0.2.
- **bottom** (`float`) – Bottom margin. Default 0.14.
- **right** (`float`) – Right margin. Default 0.025.
- **top** (`float`) – Top margin. Default 0.13.

Return type `Tuple[Figure, Axes]`

horizontal_legend (*fig*, *handles*=None, *labels*=None, *, *ncol*=1, ***kwargs*)

Place a legend on the figure, with the items arranged to read right to left rather than top to bottom.

Parameters

- **fig** (`Figure`) – The figure to plot the legend on.
- **handles** (`Optional[Iterable[Artist]]`) – Default None.
- **labels** (`Optional[Iterable[str]]`) – Default None.
- **ncol** (`int`) – The number of columns in the legend. Default 1.
- **kwargs** – Addition keyword arguments passed to `matplotlib.figure.Figure.legend()`.

Return type `Legend`

save_svg (*figure*, *fname*, *, *dpi*=None, *facecolor*='w', *edgecolor*='w', *orientation*='portrait',
transparent=False, *bbox_inches*=None, *pad_inches*=0.1, ***kwargs*)

Save the given figure as an SVG.

Parameters

- **figure** (`Figure`)
- **fname** (`Union[str, Path, PathLike, IO]`) – The file to save the SVG as. If `format` is set, it determines the output format, and the file is saved as `fname`. Note that `fname` is used verbatim, and there is no attempt to make the extension, if any, of `fname` match `format`, and no extension is appended.

If `format` is not set, then the format is inferred from the extension of `fname`, if there is one.
- **dpi** (`Union[float, Literal['figure'], None]`) – The resolution in dots per inch. If 'figure', use the figure's dpi value. Default `None`.
- **facecolor** (`Union[str, Literal['auto']]`) – The facecolor of the figure. If 'auto', use the current figure facecolor. Default 'w'.
- **edgecolor** (`Union[str, Literal['auto']]`) – The edgecolor of the figure. If 'auto', use the current figure edgecolor. Default 'w'.
- **orientation** (`Literal['portrait', 'landscape']`) – Currently only supported by the postscript backend. Default 'portrait'.
- **transparent** (`bool`) – If `True`, the axes patches will all be transparent; the figure patch will also be transparent unless `facecolor` and/or `edgecolor` are specified. This is useful, for example, for displaying a plot on top of a colored background on a web page. The transparency of these patches will be restored to their original values upon exit of this function. Default `False`.
- **bbox_inches** (`Optional[str]`) – Bounding box in inches: only the given portion of the figure is saved. If 'tight', try to figure out the tight bbox of the figure. Default `None`.
- **pad_inches** (`float`) – Amount of padding around the figure when `bbox_inches` is 'tight'. Default 0.1.
- ****kwargs** – Additional keyword arguments passed to `savefig()`.

transpose (`iterable, ncol`)

Transposes the contents of `iterable` so they are ordered right to left rather than top to bottom.

Parameters

- **iterable** (`Iterable[~T]`)
- **ncol** (`int`)

Return type `Iterable[~T]`

Returns An `Iterable` containing elements of the same type as `iterable`.

domplotlib.plots

Custom plotting functions.

New in version 0.2.0.

Functions:

<code>pie_from_tally(tally[, explode, percent, ...])</code>	Construct a pie chart from <code>cawdrey.tally.Tally</code> .
---	---

pie_from_tally (*tally*, *explode*=(), *, *percent*=False, *reverse*=False, *autopct*=None, ***kwargs*)
Construct a pie chart from `cawdrey.tally.Tally`.

Parameters

- **tally** (`Tally[str]`)
- **explode** (`Collection[str]`) – A list of key names to explode the segments for. Default `()`.
- **percent** (`bool`) – If `True`, shows the percentage of each element out of the sum of all elements. Default `False`.
- **reverse** (`bool`) – Order the wedges clockwise rather than anticlockwise.. Default `False`.
- ****kwargs** – Other keyword arguments taken by `matplotlib.axes.Axes.pie()`.

Return type `Tuple[List, ...]`

Returns

- **patches** (`list`) – A sequence of `matplotlib.patches.Wedge` instances
- **texts** (`list`) – A list of the label `.Text` instances.
- **autotexts** (`list`) – A list of `.Text` instances for the numeric labels. This will only be returned if the parameter *autopct* is not `None`.

Overloads

- `pie_from_tally(tally: Tally[str], explode = (), percent = ..., reverse = ..., autopct: None = ..., kwargs) -> Tuple[List[Wedge], List[Text]]`
- `pie_from_tally(tally: Tally[str], explode = (), percent = ..., reverse = ..., autopct: str, kwargs) -> Tuple[List[Wedge], List[Text], List[Text]]`

domplotlib.styles

Each of these styles expose `plt`, which is an alias of `matplotlib.pyplot`. Importing one of these styles configures matplotlib to use the desired style.

The styles currently available are:

- `default` – The default matplotlib style. Forces the backend to be `TkAgg` if `tkinter` is available.
- `domdf` – A theme adapted from `Solarize_Light2`.

Note: Importing a style for a second time will not change the current style. Use `importlib.reload()` to reload the module after importing to ensure the style is correctly set.

Part II

Contributing

Overview

domplotlib uses `tox` to automate testing and packaging, and `pre-commit` to maintain code quality.

Install `pre-commit` with `pip` and install the git hook:

```
$ python -m pip install pre-commit
$ pre-commit install
```


Coding style

`formate` is used for code formatting.

It can be run manually via `pre-commit`:

```
$ pre-commit run formate -a
```

Or, to run the complete autoformatting suite:

```
$ pre-commit run -a
```


Automated tests

Tests are run with `tox` and `pytest`. To run tests for a specific Python version, such as Python 3.6:

```
$ tox -e py36
```

To run tests for all Python versions, simply run:

```
$ tox
```


Type Annotations

Type annotations are checked using `mypy`. Run `mypy` using `tox`:

```
$ tox -e mypy
```


Build documentation locally

The documentation is powered by Sphinx. A local copy of the documentation can be built with `tox`:

```
$ tox -e docs
```


Downloading source code

The domplotlib source code is available on GitHub, and can be accessed from the following URL: <https://github.com/domdfcoding/domplotlib>

If you have git installed, you can clone the repository with the following command:

```
$ git clone https://github.com/domdfcoding/domplotlib
```

```
Cloning into 'domplotlib'...
remote: Enumerating objects: 47, done.
remote: Counting objects: 100% (47/47), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 173 (delta 16), reused 17 (delta 6), pack-reused 126
Receiving objects: 100% (173/173), 126.56 KiB | 678.00 KiB/s, done.
Resolving deltas: 100% (66/66), done.
```

Alternatively, the code can be downloaded in a 'zip' file by clicking:

Clone or download -> Download Zip

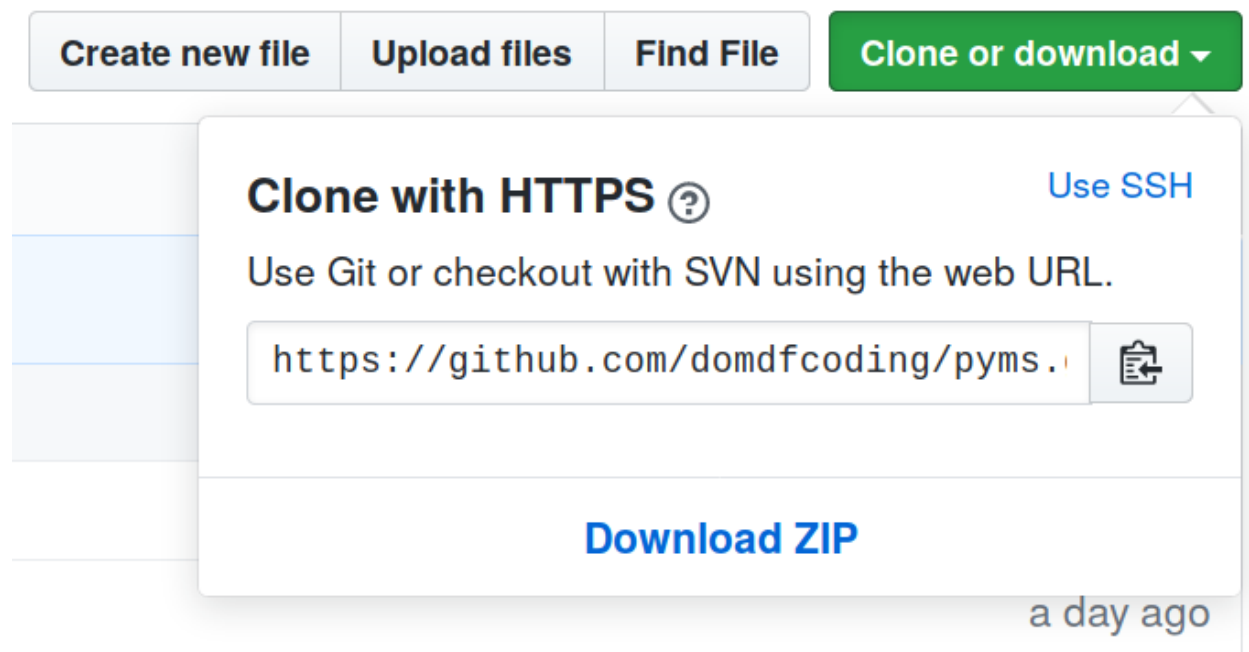


Fig. 1: Downloading a 'zip' file of the source code

10.1 Building from source

The recommended way to build `domplotlib` is to use `tox`:

```
$ tox -e build
```

The source and wheel distributions will be in the directory `dist`.

If you wish, you may also use `pep517.build` or another **PEP 517**-compatible build tool.

License

`domplotlib` is licensed under the [MIT License](#)

A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions

- Commercial use – The licensed material and derivatives may be used for commercial purposes.
- Modification – The licensed material may be modified.
- Distribution – The licensed material may be distributed.
- Private use – The licensed material may be used and modified in private.

Conditions

- License and copyright notice – A copy of the license and copyright notice must be included with the licensed material.

Limitations

- Liability – This license includes a limitation of liability.
- Warranty – This license explicitly states that it does NOT provide any warranty.

[See more information on choosealicense.com](#) ⇒

```
Copyright (c) 2020 Dominic Davis-Foster
```

```
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.
```


Python Module Index

d

`domplotlib`, [5](#)
`domplotlib.plots`, [7](#)

C

`create_figure()` (*in module domplotlib*), 5

D

`domplotlib`
 module, 5
`domplotlib.plots`
 module, 7

H

`horizontal_legend()` (*in module domplotlib*), 5

M

MIT License, 25
module
 `domplotlib`, 5
 `domplotlib.plots`, 7

P

`pie_from_tally()` (*in module domplotlib.plots*), 7
Python Enhancement Proposals
 PEP 517, 24

S

`save_svg()` (*in module domplotlib*), 5

T

`transpose()` (*in module domplotlib*), 6