charticle Documentation

Release 0.0.1

Jeremy G. Kahn

July 03, 2016

Contents

`charticle` is an MIT-licensed Python package designed to make it very easy to build new diagrams like the Drew Conway Data Science diagram.

This may not be wise.

Contents:

Example usages of charticle.

CHAPTER 1

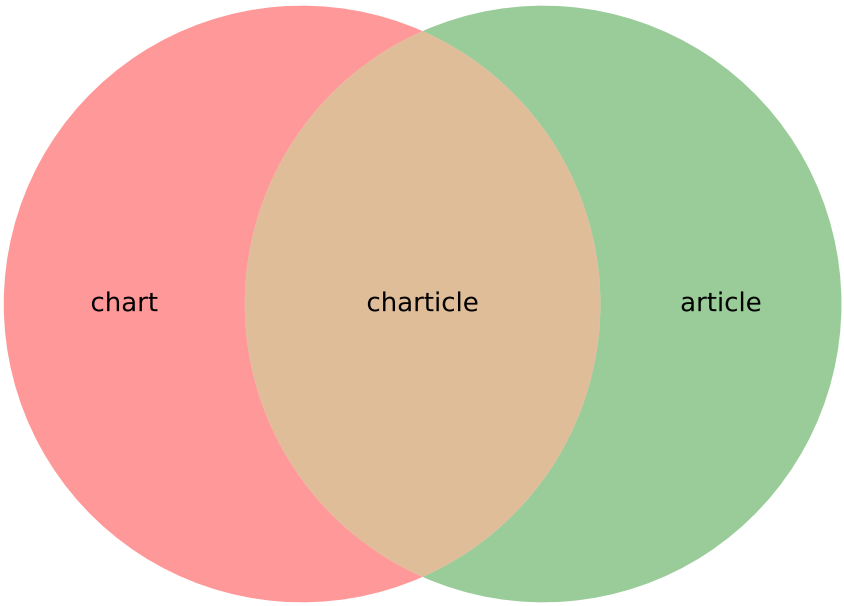
charticle.venn examples

The *charticle.venn* package contains classes for Venn diagrams.

1.1 Venn2

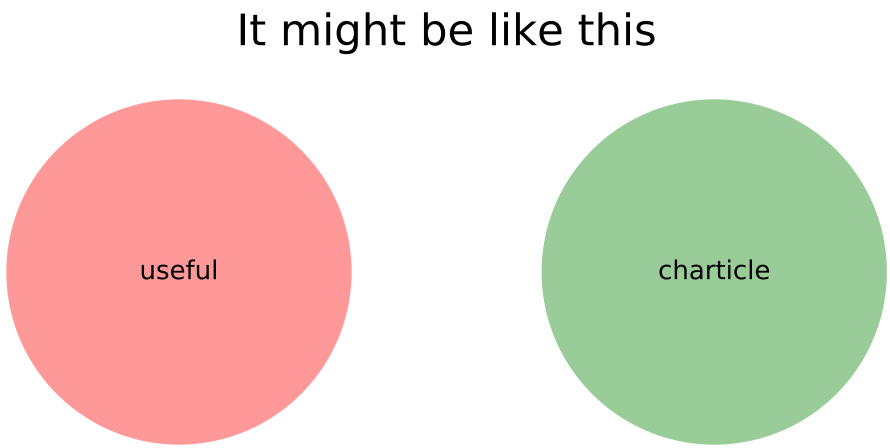
Two-circle Venn diagrams are supported (at v0.0.2+).

```
>>> from matplotlib import pyplot as plt
>>> from charticle.venn import Venn2
>>> v2 = Venn2(a="chart", b="article")
>>> v2.ab = "charticle"
>>> _ = v2.plot()
```



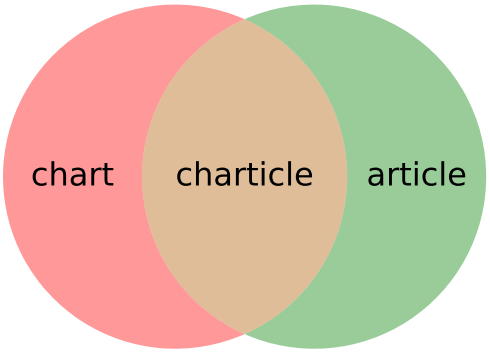
but beware:

```
>>> from charticle.venn import Venn2
>>> v = Venn2(title="It might be like this", a="useful", b="charticle",
...   sizes=Venn2.Sizes(ab=0.0))
>>> _ = v.plot()
```

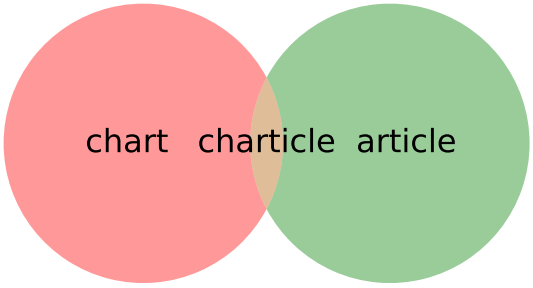


And you can apply these to subplots, too:

```
>>> import matplotlib.pyplot as plt
>>> from charticle.venn import Venn2
>>> fig = plt.figure()
>>> fig.set_size_inches(4, 6)
>>> v = Venn2(a="chart", b="article", ab="charticle")
>>> ax1, ax2 = (fig.add_subplot(211), fig.add_subplot(212))
>>> _ = v.plot(ax1)
>>> v.title = "but really it's"
>>> v.sizes.a *= 10
>>> v.sizes.b *= 10
>>> v.sizes.ab *= 0.5
>>> _ = v.plot(ax2)
```



but really it's

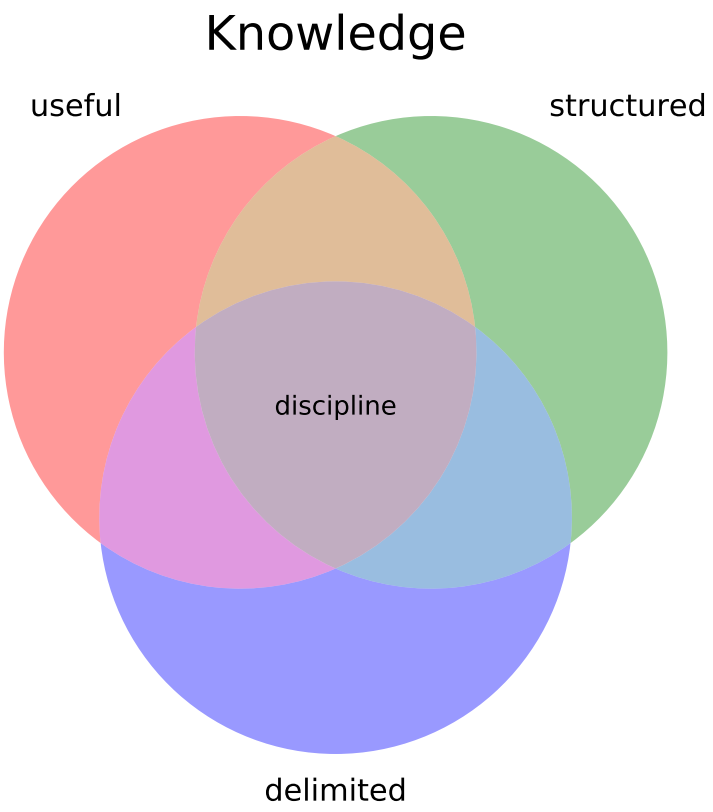


because the intersection of charts and articles is really quite tiny.

1.2 Venn3

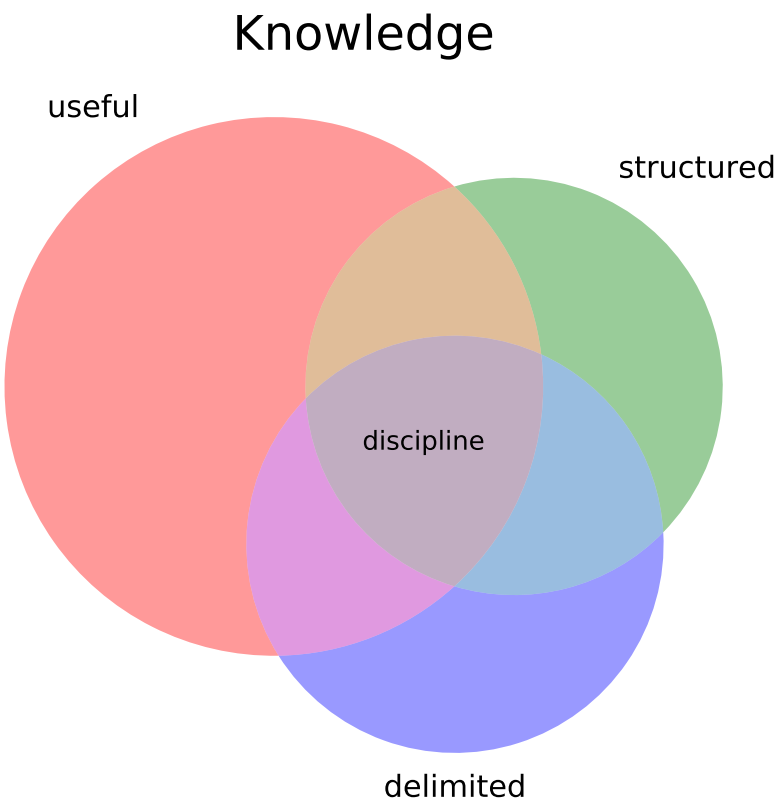
Three-circle Venn diagrams are supported, *pace* Drew Conway's data science (last in this section).

```
>>> from charticle.venn import Venn3
>>> v3 = Venn3(a_name="useful", b_name = "structured", c_name="delimited")
>>> v3.abc = "discipline"
>>> v3.title = "Knowledge"
>>> v3.fontsizes.title = 22
>>> v3
Venn3(a_name='useful', b_name='structured', c_name='delimited', ...))
>>> _ = v3.plot()
```



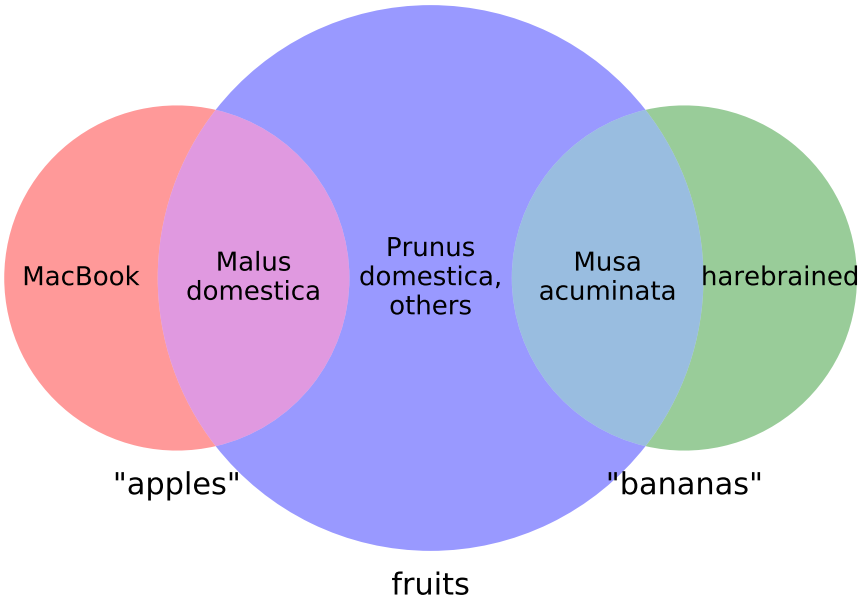
Further can set region sizes:

```
>>> v3.sizes
Venn3.Sizes(a=1.0, b=1.0, c=1.0, ab=1.0, ac=1.0, bc=1.0, abc=1.0, normalize=1.0)
>>> v3.sizes.set_single_weight(1.0) # moot
Venn3.Sizes(a=1.0, b=1.0, c=1.0, ab=1.0, ac=1.0, bc=1.0, abc=1.0, normalize=1.0)
>>> v3.sizes.a *= 5
>>> v3.sizes.set_double_weight(2.0)
Venn3.Sizes(a=5.0, b=1.0, c=1.0, ab=2.0, ac=2.0, bc=2.0, abc=1.0, normalize=1.0)
>>> _ = v3.plot()
```



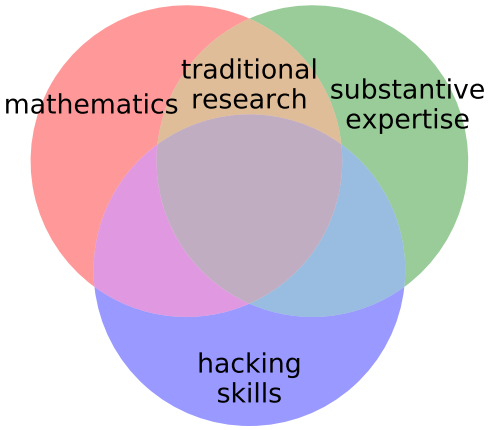
If regions are given a zero size, the diagram will reorganize:

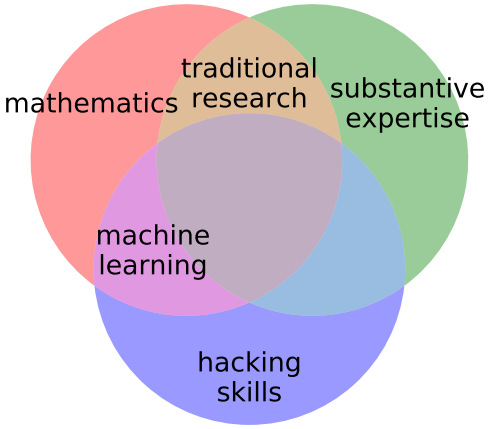
```
>>> from charticle.venn import Venn3
>>> v = Venn3(a_name='"apples"', b_name='"bananas"',  c_name="fruits")
>>> v.a = "MacBook"
>>> v.b = 'harebrained'
>>> v.c = "Prunus\ndomestica,\nothers"; v.sizes.c = 3
>>> v.ac = "Malus\ndomestica"
>>> v.bc = "Musa\nacuminata"
>>> v.sizes.ab = 0; v.sizes.abc = 0
>>> _ = v.plot()
```

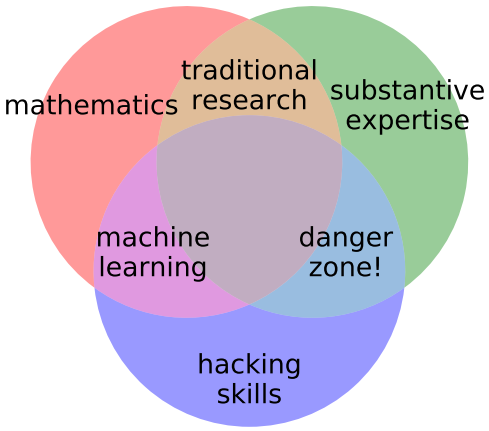


And you can still do multiple plots by passing an axis object to plot.

```
>>> import matplotlib.pyplot as plt
>>> from charticle.venn import Venn3, FontSizes
>>> fig = plt.figure()
>>> fig.set_size_inches(6,13)
>>> v = Venn3(a="mathematics", b="substantive\nexpertise",
...  c="hacking\nskills",
...  fontsizes=FontSizes(intersections=10),
...  sizes=Venn3.Sizes(normalize=30))
>>> ax1, ax2, ax3, ax4 = (fig.add_subplot(411), fig.add_subplot(412),
...   fig.add_subplot(413), fig.add_subplot(414))
>>> v.ab = "traditional\nresearch"; _ = v.plot(ax1)
>>> v.ac = "machine\nlearning"; _ = v.plot(ax2)
>>> v.bc = "danger\nzone!"; _ = v.plot(ax3)
>>> v.abc = "data\nscience"; _ = v.plot(ax4)
```


traditional
research
mathematics
substantive
expertise
hacking
skills


traditional
research
mathematics
substantive
expertise
machine
learning
hacking
skills


traditional
research
mathematics
substantive
expertise
machine
learning
danger
zone!
hacking
skills


traditional
substantive

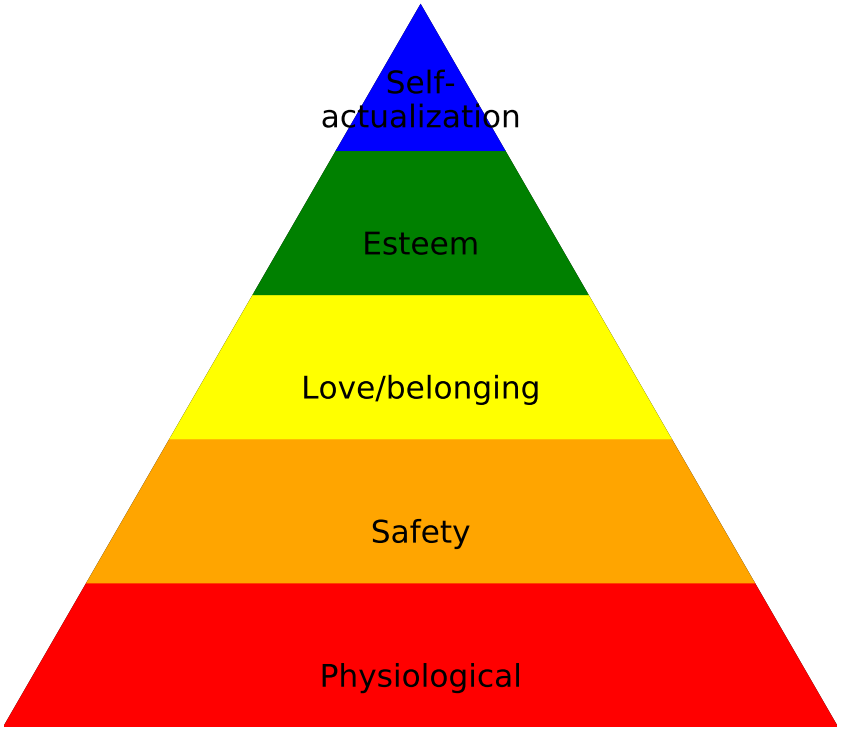
CHAPTER 2

charticle.hierarchy examples

The *charticle.hierarchy* package contains classes for pyramid diagrams, like Maslow's hierarchy.

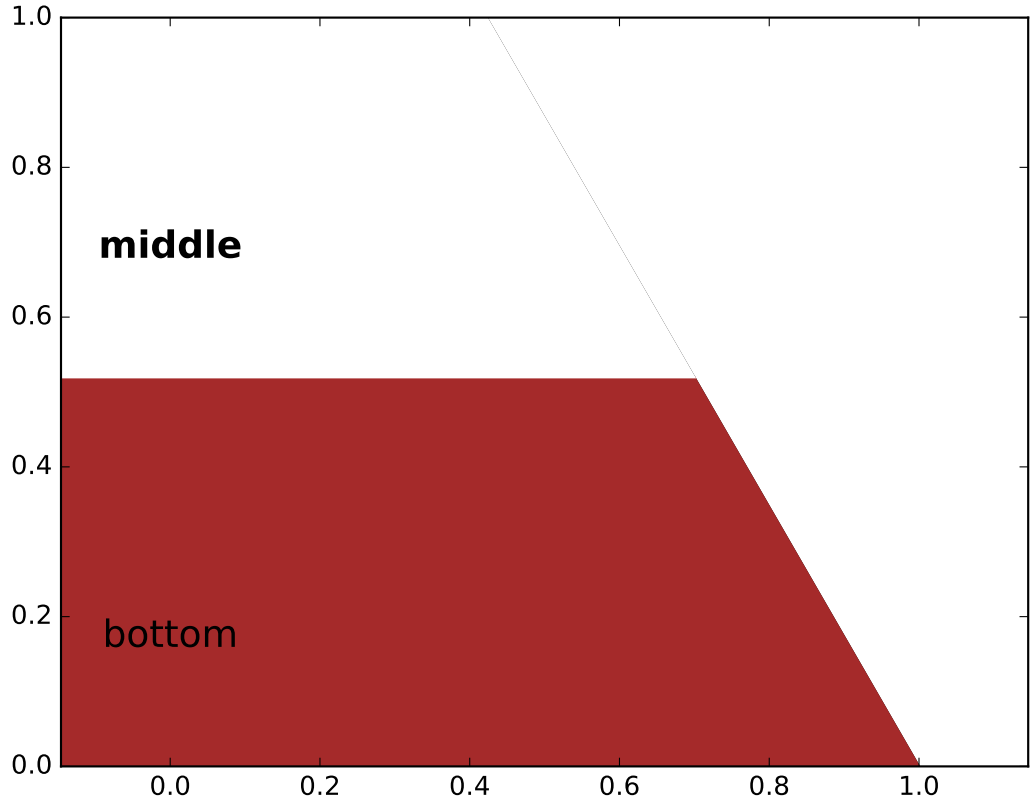
2.1 Hierarchy

```
>>> from matplotlib import pyplot as plt
>>> from charticle.hierarchy import Hierarchy
>>> h = Hierarchy(layer_text_defaults=dict(size='large'))
>>> _ = h.set_layers(['Physiological', 'Safety', 'Love/belonging',
...                   'Esteem','Self-\nactualization'])
>>> h.plot()
>>> _ = plt.axis('scaled'); _ = plt.axis('off')
```



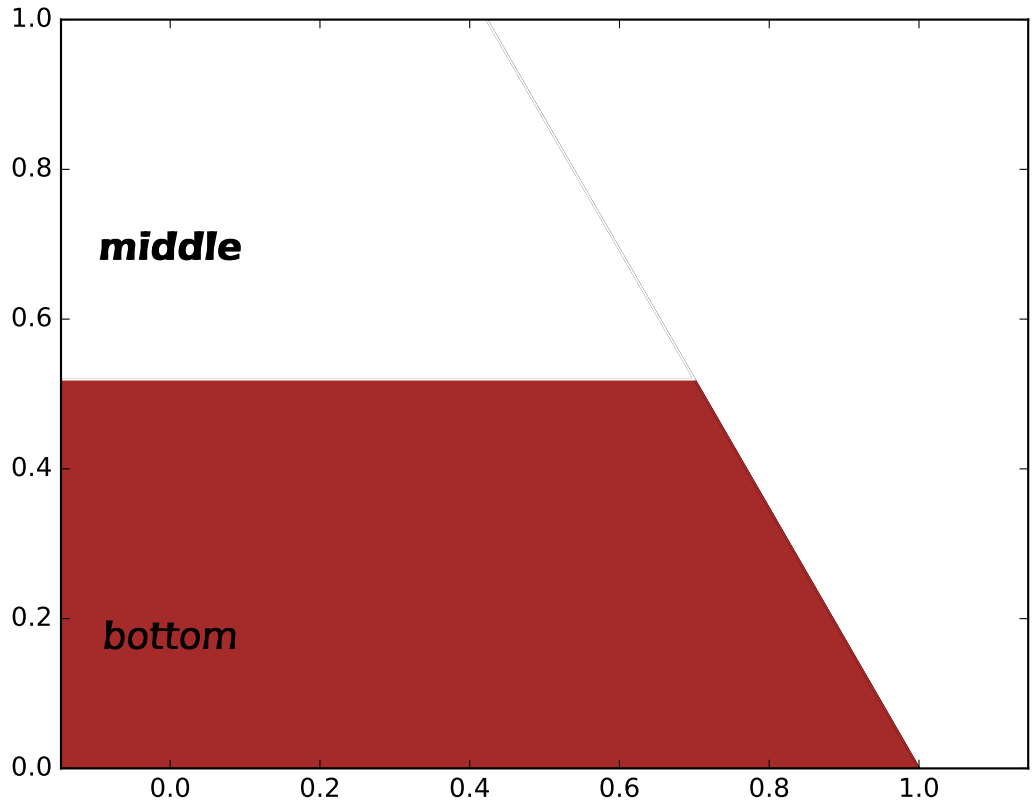
You may of course set the colors as a whole or override them one at a time.

```
>>> from matplotlib import pyplot as plt
>>> from charticle.hierarchy import Hierarchy
>>> h = Hierarchy(layer_text_defaults=dict(size='x-large'))
>>> h.set_color_cycle('brown', 'white', 'pink')
>>> l1 = h.add_layer(lower=0.0, upper=0.3, label='bottom')
>>> l2 = h.add_layer(lower=0.3, upper=0.6, label='middle',
...                  text={'weight': 'bold'})
>>> l3 = h.add_layer(lower=0.6, upper=1.0, label='top',
...                  polygon={'fill': True, 'color': 'green'})
>>> h.plot()
```



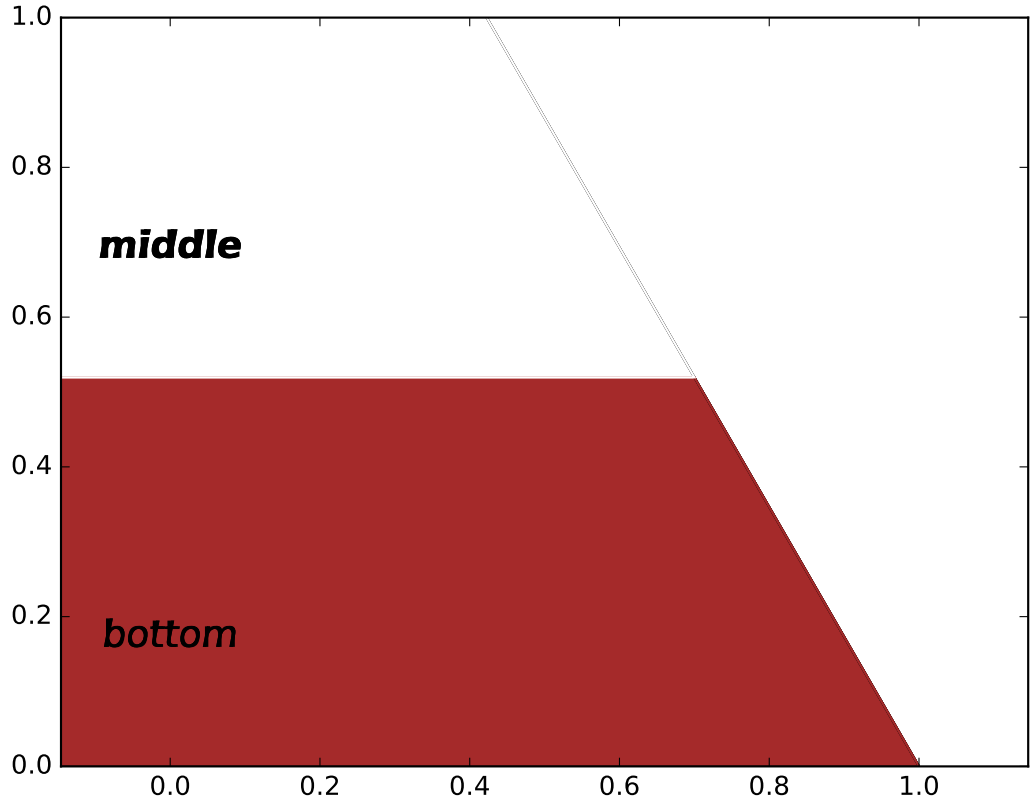
There are ways to set defaults for all the layer polygon forms:

```
>>> h.layer_polygon_defaults["fill"] = False
>>> h.layer_text_defaults["style"] = 'italic'
>>> h.plot()
```



And it works just fine with an axis passed in:

```
>>> ax = plt.gca()
>>> h.plot(ax=ax)
```


1.0
0.8
0.6
0.4
0.2
0.0
middle
bottom
0.0
0.2
0.4
0.6
0.8
1.0

CHAPTER 3

Future modules

Plans for *charticle.xy* as well?

CHAPTER 4

API

Documentation of the charticle APIs.

4.1 Venn diagrams in `charticle.venn`

Venn diagrams with labeled regions.

class charticle.venn.**FontSizes**(*title=20*, *sets=14*, *intersections=12*)
Utility class for font size tracking.

class charticle.venn.**Venn2**(*a_name=None*, *b_name=None*, *a=None*, *b=None*, *ab=None*, *title=None*, *sizes=NOTHING*, *fontsizes=NOTHING*, *palette=NOTHING*)
Object for a 2-circle Venn. Set attributes at init or by assignment.

Parameters

- **a_name** (*str*) –
- **b_name** (*str*) – Label text for outside the A & B circles.
- **a** (*str*) –
- **b** (*str*) – Label text for the 1-member crescents.
- **ab** (*str*) – Label text for the lenticular intersection of A & B.
- **title** (*str*) – Text for the title of the plot.
- **palette** (Venn2.Palette) – a color palette for the A & B sets.
- **fontsizes** (FontSizes) – the font sizes for various labels.

class Palette(*a='red'*, *b='green'*, *alpha=0.4*)
Container of color palette for both sets.

Parameters

- **a,b** (*legal html colornames or hex codes*) – color names for the two sets.
- **alpha** (*float in [0,1]*) – color combination alpha for intersection.

TODO: add some default "constant" palettes.

class Venn2.**Sizes**(*a=1.0*, *b=1.0*, *c=1.0*, *ab=1.0*, *normalize=1.0*)
Utility class for shaping the Venn2.

Venn2.**plot**(*ax=None*)
Produce a plot on the specified axes.

Puts label strings in the right places and produces the figure.

ax: the axis on which to plot this diagram. Defaults to current axes.

class charticle.venn.**Venn3**(*a_name=None*, *b_name=None*, *c_name=None*, *a=None*, *b=None*, *c=None*, *ab=None*, *bc=None*, *ac=None*, *abc=None*, *title=None*, *sizes=NOTHING*, *fontsizes=NOTHING*, *palette=NOTHING*)
Object for a 3-label venn. Set attributes at init or by assignment.

Parameters

- **a_name** (*str*) –
- **b_name** (*str*) –
- **c_name** (*str*) – Label text for the outer circles.
- **a** (*str*) –
- **b** (*str*) –
- **c** (*str*) – Label text for the 1-member patches.
- **ab** (*str*) –
- **ac** (*str*) –
- **bc** (*str*) – Label text for the 2-set-intersection patches.
- **abc** (*str*) – Label text for the full 3-set intersection.
- **title** (*str*) – Text for the title of the plot.
- **palette** (Venn3.Palette) – a color palette for the sets.
- **sizes** (Venn3.Sizes) – the region sizes (relative to 1.0).
- **fontsizes** (FontSizes) – the font sizes for various labels.

class **Palette**(*a='red'*, *b='green'*, *c='blue'*, *alpha=0.4*)
Container of color palette for all 3 items.

Parameters

- **a,b,c** (*legal html colornames or hex codes*) – color names for the three sets.
- **alpha** (*float in [0,1]*) – color combination alpha for intersections.

TODO: add some default "constant" palettes.

class Venn3.**Sizes**(*a=1.0*, *b=1.0*, *c=1.0*, *ab=1.0*, *ac=1.0*, *bc=1.0*, *abc=1.0*, *normalize=1.0*)
Utility class for shaping the Venn3.

Venn3.**plot**(*ax=None*)
Produce a plot on the specified axes.

Puts label strings in the right places and produces the figure.

ax: the axis on which to plot this diagram. Defaults to current axes.

4.2 Hierarchy pyramids in `charticle.hierarchy`

class charticle.hierarchy.**Hierarchy**(*scale=1.0, polygon=NOTHING, layers=NOTHING, layer_polygon_defaults=NOTHING, layer_text_defaults=NOTHING, color_cycle=cycler('color', ['red', 'orange', 'yellow', 'green', 'blue', 'purple'])*)

Draws a 'Maslow-style' hierarchy.

class **Layer**(*label*, *lower*, *upper*, *polygon=NOTHING*, *text=NOTHING*)

Container for layer information.

plot(*hierarchy*, *ax*, *default_color=None*)

Adds this layer to the hierarchy.

Hierarchy.**plot**(*ax=None*)

Write the hierarchy (onto specified axes, if given).

Hierarchy.**set_color_cycle**(**colors*)

Sets color cycle from iterable of color names.

Parameters **colors** (*valid matplotlib color names.*) – color names or None.

Hierarchy.**set_layers**(*layers_list*)

Write text labels into parameters.

CHAPTER 5

Changelog

Versions are trying to maintain major.minor.patch format.

Major version 0: 2016.

5.1 0.0.3 (3 July 2016)

5.1.1 Changes:

Add *charticle.hierarchy* diagrams and examples.

Supports general text and polygon arguments for outer triangle & layers.

Cleanups:

- wishlist, release procedures added to repo.
- Jupyter notebooks removed from repo.
- py27 testing included in *tox.ini*.
- refactoring validators.
- cleaned up release procedure.

5.2 0.0.2 (30 June 2016)

5.2.1 Changes:

Support Venn2 diagrams (and Venn3).

Improve documentation examples and testing.

5.3 0.0.1 (29 June 2016)

5.3.1 Changes:

First released version to pypi.

venn.Venn3 is only working charticle type right now.

Previously unreleased versions didn't have working docs or tests.

CHAPTER 6

Wishlist

6.1 Convenience functions

- Palettes and/or styles that can be applied across articles
- pass offset parameters to hierarchy objects so they can be plotted as legends to other graphs.
- Add color palette shorthands to Venn diagrams?

6.2 New decorations

- Add peripheral lines to circles?
- Add option to transform label into annotation, per @vennsplainer.

6.3 Refactors

- Allow venn objects to set polygon and text properties the way hierarchy does.

6.4 Bigger ideas

- Apply an 'indexed' style like the website.

CHAPTER 7

Release checklist

- Run the **tox** tests.

7.1 Land the current version

- update `CHANGELOG.rst`
- update `__version__` in `src/charticle/__init__.py` to have new version number.
- run **tox** again.
- commit changes to local git. repo
- **git** tag with current `v{v}`
- `git push origin v{v}` to make sure the tag has shipped to github.

7.2 Create the next launch version

- checkout a new branch named `v{v+1}.dev`.
- update `__version__` in `charticle/__init__.py` to `v{v+1}.dev` and `CHANGELOG.rst` to `v{v+1} (unreleased)`.
- commit these changes to the new branch.
- push the new branch to github (`git push --set-upstream origin v{v+1}-dev`).

7.3 Ship the tag to PyPI and advance `master`

- **git** `checkout` the v{v} tag.
- Ship to PyPI with `tox -e pypi`
- `git checkout master && git merge --ff-only v{v}`
- `git push origin master`
- (**git** checkout the v{v+1}-dev branch again.)

CHAPTER 8

License and Credits

8.1 License

`charticle` is licensed under the MIT license. The full license text can be also found in the source code repository.

8.2 Credits

`charticle` is written and maintained by Jeremy G. Kahn.

Collaboration and kibitzing due to Bill McNeill.

A full list of contributors can be found in GitHub's overview.

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

C

C

F

H

P

S

V