Third-Party Packages

May 16, 2017

Overview

Overview

Package Management

Third Party Packages

Package Management

Python Package Management

Python Package Management

PyPI

`pip`

Python Package Index (PyPI)

Repository of software for the Python programming language

Python Package Index (PyPI)

Python Package Index (PyPI)

Python 3 Packages

Python Package Index (PyPI)

23,000+

Python 3 Packages

Python Package Index (PyPI)

Top 360 packages ported to Python 3

Python Package Index (PyPI)

343/360

Top 360 packages ported to Python 3

Python Package Index (PyPI)

Python Files

Python Package Index (PyPI)

500,000+

Python Files

TL;DR: If you want it, PyPI probably has it

Python Package Index (PyPI)

`pip`
the Python package manager

Installing Python Packages

Installing Python Packages

`pip` is the preferred Python package manager

Use `pip`!

Installing Python Packages

`pip` is the preferred Python package manager

Use `pip`!

When you can, use `pip` instead of:

Installing Python Packages

`pip` is the preferred Python package manager

Use `pip`!

When you can, use `pip` instead of:

`conda` - less flexible, less supported by the community

Installing Python Packages

`pip` is the preferred Python package manager

Use `pip`!

When you can, use `pip` instead of:

`conda` - less flexible, less supported by the community

`easy_install` - the old way to install packages

Installing Python Packages

`pip` is the preferred Python package manager

Use `pip`!

When you can, use `pip` instead of:

`conda` - less flexible, less supported by the community

`easy_install` - the old way to install packages

`python setup.py install` - build package from source

Using `pip` - common commands

Using `pip` - common commands

```
# Install/Uninstall a package
$ pip install package_name
$ pip uninstall package_name
```

Using `pip` - common commands

```
# Install/Uninstall a package
$ pip install package_name
$ pip uninstall package_name

# Upgrade an existing package to the newest version
$ pip install --upgrade package_name
```

Using `pip` - common commands

```
# Install/Uninstall a package
$ pip install package_name
$ pip uninstall package_name

# Upgrade an existing package to the newest version
$ pip install --upgrade package_name

# Install from requirements file
$ pip freeze > requirements.txt
$ pip install -r requirements.txt
```

Using `pip` - common commands

```
# Install/Uninstall a package
$ pip install package_name
$ pip uninstall package_name

# Upgrade an existing package to the newest version
$ pip install --upgrade package_name

# Install from requirements file
$ pip freeze > requirements.txt
$ pip install -r requirements.txt
```

```
# File: requirements.txt
...
ipython==4.2.0
matplotlib==1.5.1
numpy==1.11.0
pandas==0.18.0
pep8==1.7.0
Pillow==3.2.0
requests==2.9.1
scipy==0.17.0
...
```

Using `pip` - uncommon commands

Using `pip` - uncommon commands

```
# Install to Python user-specific install directory
$ pip install --user package_name
```

Using `pip` - uncommon commands

```
# Install to Python user-specific install directory
$ pip install --user package_name

# Require specific versions to be installed
$ pip install "package_name==4.1.0"
$ pip install "package_name >= 1.0, != 1.4.0, < 2.0"
```

Using `pip` - uncommon commands

```
# Install to Python user-specific install directory
$ pip install --user package_name

# Require specific versions to be installed
$ pip install "package_name==4.1.0"
$ pip install "package_name >= 1.0, != 1.4.0, < 2.0"

# Show information about a particular package
$ pip show package_name
```

Using `pip` - uncommon commands

```
# Install to Python user-specific install directory
$ pip install --user package_name

# Require specific versions to be installed
$ pip install "package_name==4.1.0"
$ pip install "package_name >= 1.0, != 1.4.0, < 2.0"

# Show information about a particular package
$ pip show package_name

# Search PyPI for matching packages
$ pip search query
```

Time-Out for Announcements

Assignment 2

Assignment 2

The Quest for the Holy Grail has finished!

Assignment 2

The Quest for the Holy Grail has finished!

Many knights trained dragons and concocted potions.

Assignment 2

The Quest for the Holy Grail has finished!

Many knights trained dragons and concocted potions.

In total, 11 valiant knights recovered the missing grail.

Assignment 3

Assignment 3

Stylize – Recipe Book

Assignment 3

Stylize – Recipe Book

Starter code is 100% functional, but terribly written

Assignment 3

Stylize – Recipe Book

Starter code is 100% functional, but terribly written

Your goal: fix the Python style and program design

Assignment 3

Stylize – Recipe Book

Starter code is 100% functional, but terribly written

Your goal: fix the Python style and program design

The starter code is ✓+/– for functionality/style

Assignment 3

Stylize – Recipe Book

Starter code is 100% functional, but terribly written

Your goal: fix the Python style and program design

The starter code is ✓+/– for functionality/style

Released tonight

Assignment 3

Assignment 3

Stylize – Alternate Challenge

Assignment 3

Stylize – Alternate Challenge

Find poorly written, functional Python code in the wild

Assignment 3

Stylize – Alternate Challenge

Find poorly written, functional Python code in the wild

- Other courses, open source projects, existing tools

Assignment 3

Stylize – Alternate Challenge

Find poorly written, functional Python code in the wild

- Other courses, open source projects, existing tools

With owner's permission, rewrite codebase for style

Assignment 3

Stylize – Alternate Challenge

Find poorly written, functional Python code in the wild

- Other courses, open source projects, existing tools

With owner's permission, rewrite codebase for style

Unusual – must check with course staff, IP law, etc.

Assignment 3

Assignment 3

Stylize – Bonus Challenge (Code Golf)

Assignment 3

Stylize – Bonus Challenge (Code Golf)

How short can you make the Recipe Book?

Assignment 3

Stylize – Bonus Challenge (Code Golf)

How short can you make the Recipe Book?

Toss style out the window – the fewer letters, the better

Assignment 3

Stylize – Bonus Challenge (Code Golf)

How short can you make the Recipe Book?

Toss style out the window – the fewer letters, the better

Must maintain all functionality as before

Assignment 3

Stylize – Bonus Challenge (Code Golf)

How short can you make the Recipe Book?

Toss style out the window – the fewer letters, the better

Must maintain all functionality as before

Beat us, and immediately get a ✓+/✓+ on the assignment

Assignment 4

Assignment 4

Final Project (Preview)

Assignment 4

Final Project (Preview)

Groups of up to three

Assignment 4

Final Project (Preview)

Groups of up to three

Incorporate Python in some meaningful way

Assignment 4

Final Project (Preview)

Groups of up to three

Incorporate Python in some meaningful way

Proposal (Google Form) by EOD Thursday + Final Project

Assignment 4

Final Project (Preview)

Groups of up to three

Incorporate Python in some meaningful way

Proposal (Google Form) by EOD Thursday + Final Project

Can be a final project for another class / thesis (with OK)

Assignment 4

Final Project (Preview)

Groups of up to three

Incorporate Python in some meaningful way

Proposal (Google Form) by EOD Thursday + Final Project

Can be a final project for another class / thesis (with OK)

- But we'll grade you on Python-specific practices

Back to Python!

The Very Best*
Third-Party Packages

Disclaimer

Disclaimer

Absurd number of Python packages

Disclaimer

Absurd number of Python packages

Some broadly applicable, some narrowly applicable

Disclaimer

Absurd number of Python packages

Some broadly applicable, some narrowly applicable

As before, assume all required imports are performed

Disclaimer

Absurd number of Python packages

Some broadly applicable, some narrowly applicable

As before, assume all required imports are performed

"Show-and-tell" slides - up to you to dive into the tools!

requests: HTTP for Humans

Quick Start

Requests

Requests

```
# A simple GET request
response = requests.get('https://api.example.com/users')
```

Requests

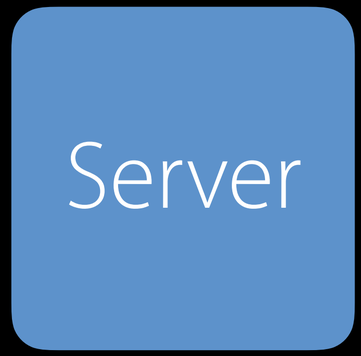
```
# A simple GET request
response = requests.get('https://api.example.com/users')
if response.ok:
    raw_data = response.content
    json_data = response.json()  # If server response is in JSON format
```

Requests

```
# A simple GET request
response = requests.get('https://api.example.com/users')
if response.ok:
    raw_data = response.content
    json_data = response.json()  # If server response is in JSON format
```

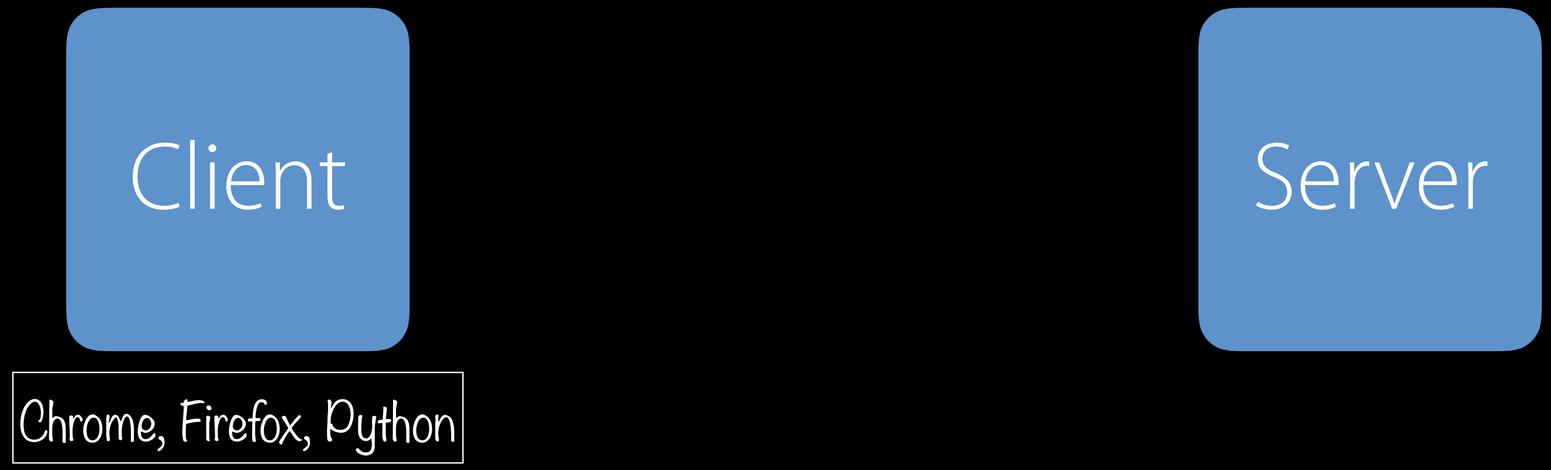
Requests

```
# A simple GET request
response = requests.get('https://api.example.com/users')
if response.ok:
    raw_data = response.content
    json_data = response.json()  # If server response is in JSON format
```



Requests

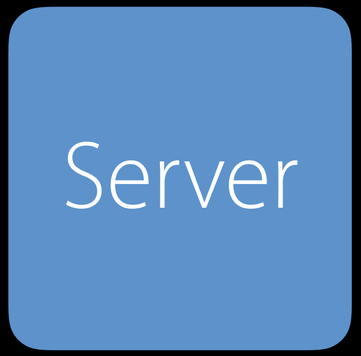
```
# A simple GET request
response = requests.get('https://api.example.com/users')
if response.ok:
    raw_data = response.content
    json_data = response.json()  # If server response is in JSON format
```



Requests

```
# A simple GET request
response = requests.get('https://api.example.com/users')
if response.ok:
    raw_data = response.content
    json_data = response.json()  # If server response is in JSON format
```

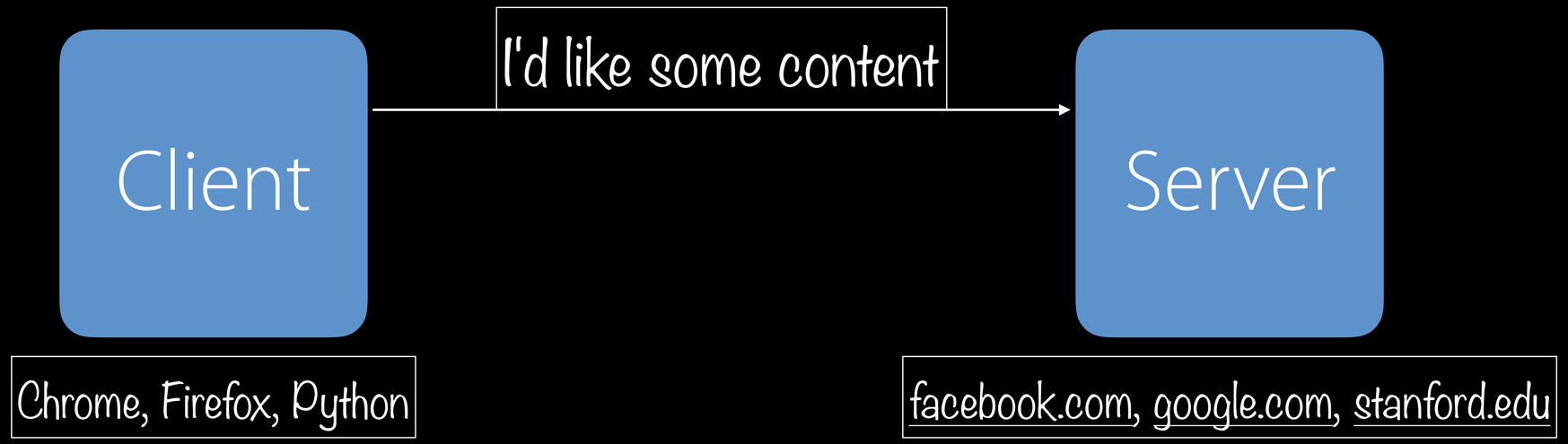
Chrome, Firefox, Python



facebook.com, google.com, stanford.edu

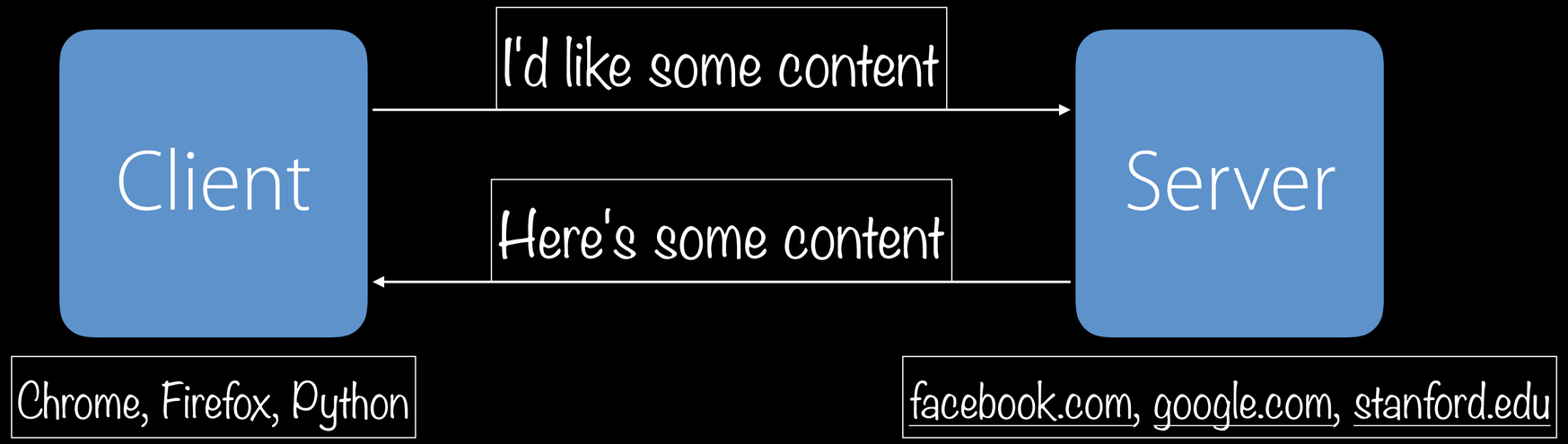
Requests

```
# A simple GET request
response = requests.get('https://api.example.com/users')
if response.ok:
    raw_data = response.content
    json_data = response.json()  # If server response is in JSON format
```



Requests

```
# A simple GET request
response = requests.get('https://api.example.com/users')
if response.ok:
    raw_data = response.content
    json_data = response.json()  # If server response is in JSON format
```



Requests

Requests

```
# GET with params
payload = {'name': 'sredmond', 'field': ['email', 'org']}
```

Requests

```
# GET with params
payload = {'name': 'sredmond', 'field': ['email', 'org']}
response = requests.get('https://api.example.com/users', params=payload)
```

Requests

```
# GET with params
payload = {'name': 'sredmond', 'field': ['email', 'org']}
response = requests.get('https://api.example.com/users', params=payload)
print(response.url)
# => https://api.example.com/users?name=sredmond&field=email&field=org
```

Requests

```
# GET with params
payload = {'name': 'sredmond', 'field': ['email', 'org']}
response = requests.get('https://api.example.com/users', params=payload)
print(response.url)
# => https://api.example.com/users?name=sredmond&field=email&field=org

# POST with params
payload = {'username': 'sredmond', 'password': 'pyth0n'}
```

Requests

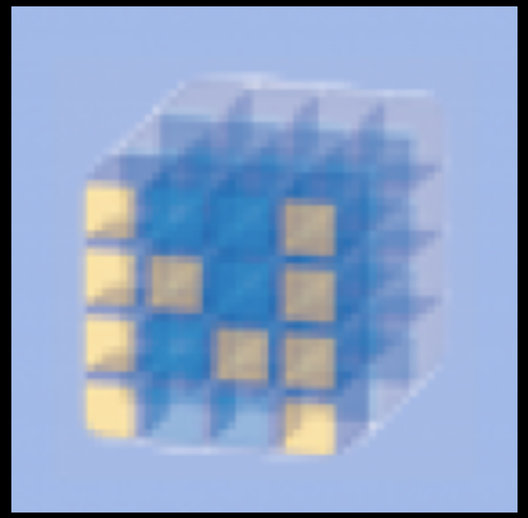
```
# GET with params
payload = {'name': 'sredmond', 'field': ['email', 'org']}
response = requests.get('https://api.example.com/users', params=payload)
print(response.url)
# => https://api.example.com/users?name=sredmond&field=email&field=org

# POST with params
payload = {'username': 'sredmond', 'password': 'pyth0n'}
response = requests.post('https://api.example.com/login', data=payload)
```

Example: /r/sneks

Intro to Scientific Python

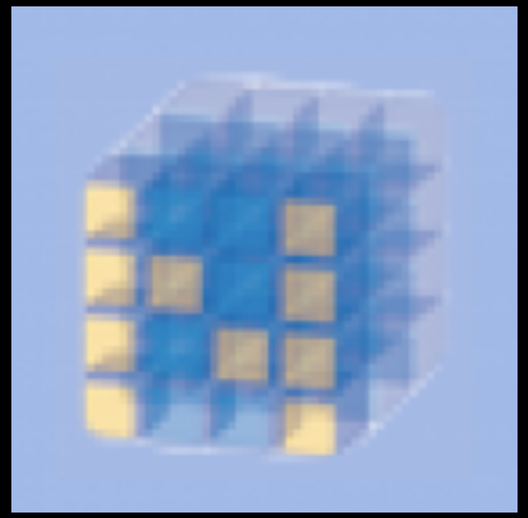
CME 193 for more

NumPy (Numerical Python)



NumPy (Numerical Python)

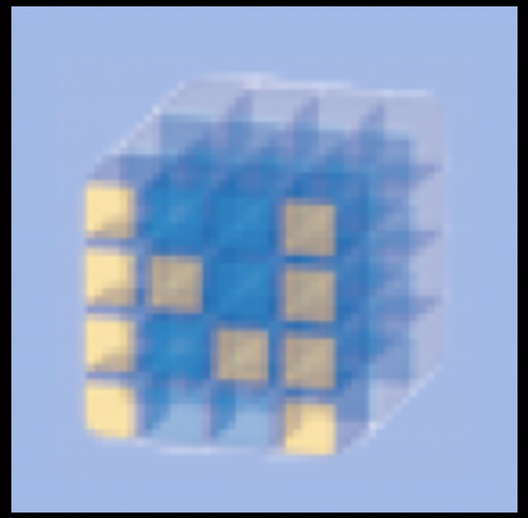
N-dimensional array object



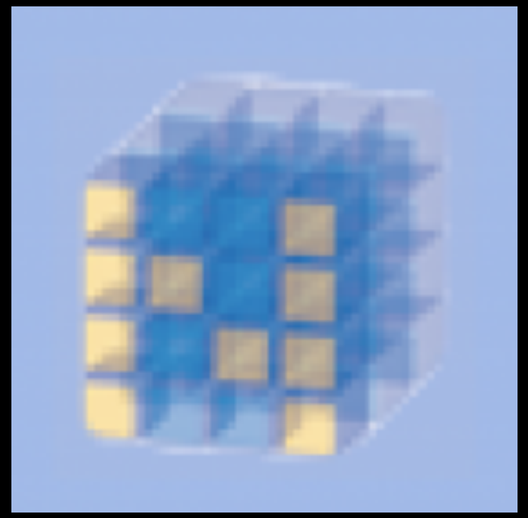
NumPy (Numerical Python)

N-dimensional array object

Sophisticated functions



NumPy (Numerical Python)



- N-dimensional array object
- Sophisticated functions
- Capabilities
  - Linear algebra
  - Fourier transform
  - Random sampling
- Docs

Using **numpy**

Using **numpy**

```
a = np.arange(15).reshape(3, 5)
```

Using **numpy**

```
a = np.arange(15).reshape(3, 5)
print(a)
# array([[ 0,  1,  2,  3,  4],
#        [ 5,  6,  7,  8,  9],
#        [10, 11, 12, 13, 14]])
```

Using **numpy**

```
a = np.arange(15).reshape(3, 5)
print(a)
# array([[ 0,  1,  2,  3,  4],
#        [ 5,  6,  7,  8,  9],
#        [10, 11, 12, 13, 14]])

a.shape  # => (3, 5)
a.ndim  # => 2
a.dtype.name  # => 'int64'
type(a)  # => numpy.ndarray
```

Using **numpy**

```
a = np.arange(15).reshape(3, 5)
print(a)
# array([[ 0,  1,  2,  3,  4],
#        [ 5,  6,  7,  8,  9],
#        [10, 11, 12, 13, 14]])

a.shape  # => (3, 5)
a.ndim  # => 2
a.dtype.name  # => 'int64'
type(a)  # => numpy.ndarray

print(a[:, 1])  # => array([ 1,  6, 11])
```

Using **numpy**

Using **numpy**

```
a = np.array([3, 4, 5])
a + 4  # => array([7, 8, 9])
a * 1.5  # => array([ 4.5,  6. ,  7.5])
```

Using **numpy**

```
a = np.array([3, 4, 5])
a + 4  # => array([7, 8, 9])
a * 1.5  # => array([ 4.5,  6. ,  7.5])

b = np.array([4, -1, 0])
np.dot(a, b)  # => 8 (= 3 * 4 + 4 * -1 + 5 * 0)
a.sum()  # => 12
```

Using **numpy**

```
a = np.array([3, 4, 5])
a + 4  # => array([7, 8, 9])
a * 1.5  # => array([ 4.5,  6. ,  7.5])

b = np.array([4, -1, 0])
np.dot(a, b)  # => 8 (= 3 * 4 + 4 * -1 + 5 * 0)
a.sum()  # => 12

space = np.linspace(0, 2 * np.pi, 100)
sinusoid = np.sin(b)
```

SciPy (Scientific Python)

SciPy (Scientific Python)

Everything you need for mathematics, science, and engineering.

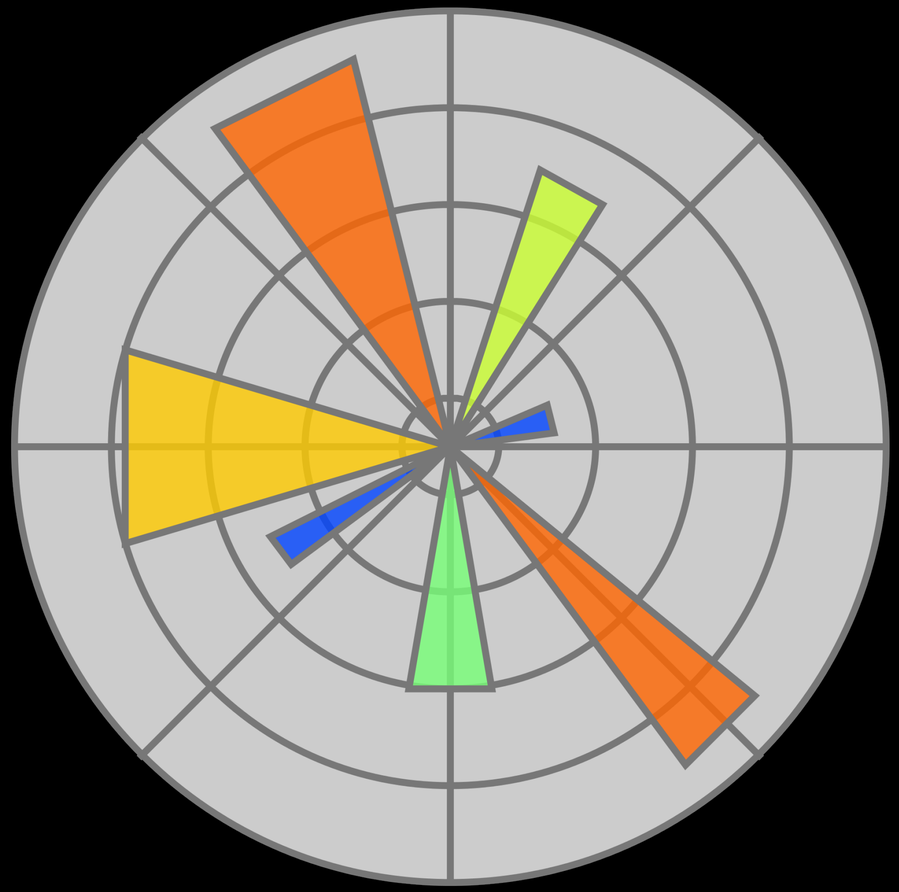
Docs

```
from scipy import spatial, linalg
```

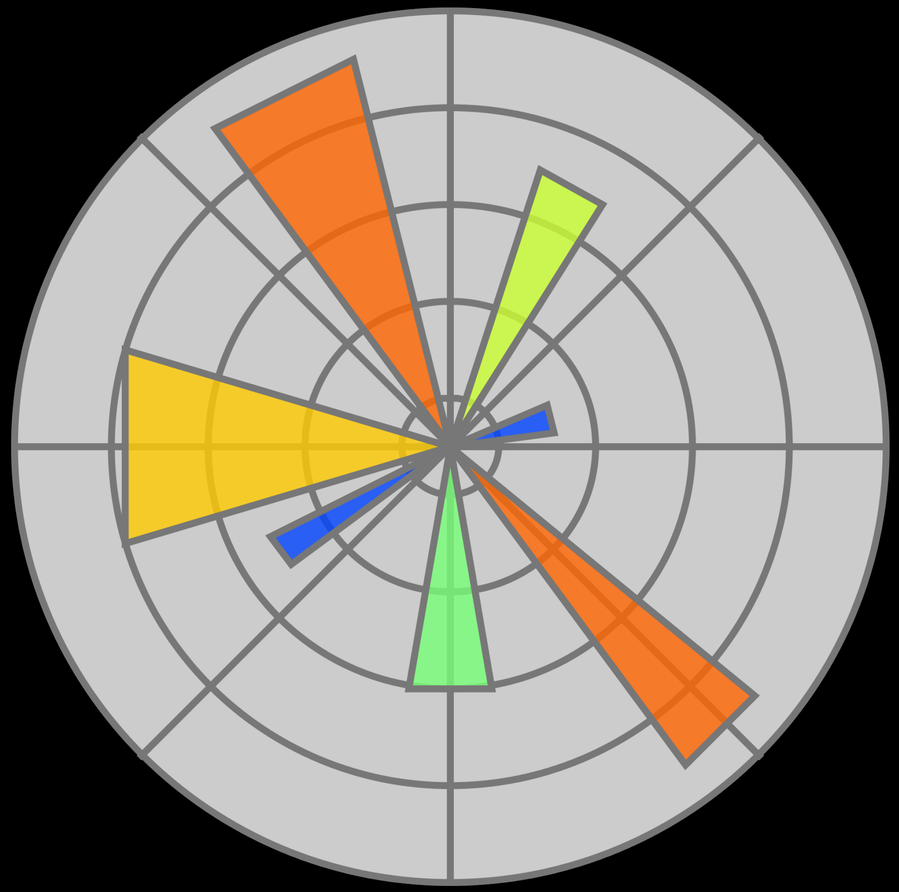
```
from scipy import spatial, linalg

# Build a KD Tree
points = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])
tree = spatial.KDTree(points)
tree.query([0.1, 0.1])  # => (0.14142135623730953, 0)
```

```
from scipy import spatial, linalg

# Build a KD Tree
points = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])
tree = spatial.KDTree(points)
tree.query([0.1, 0.1])  # => (0.14142135623730953, 0)

# Invert a matrix
A = np.array([[1, 2], [3, 4]])
print(linalg.inv(A))
# array([[-2. ,  1. ],
#        [ 1.5, -0.5]])
```

MatPlotLib + PyPlot



MatPlotLib + PyPlot



Python 2D Plotting Library

Think MATLAB + Python

Examples

Gallery

Using `pyplot`

Using `pyplot`

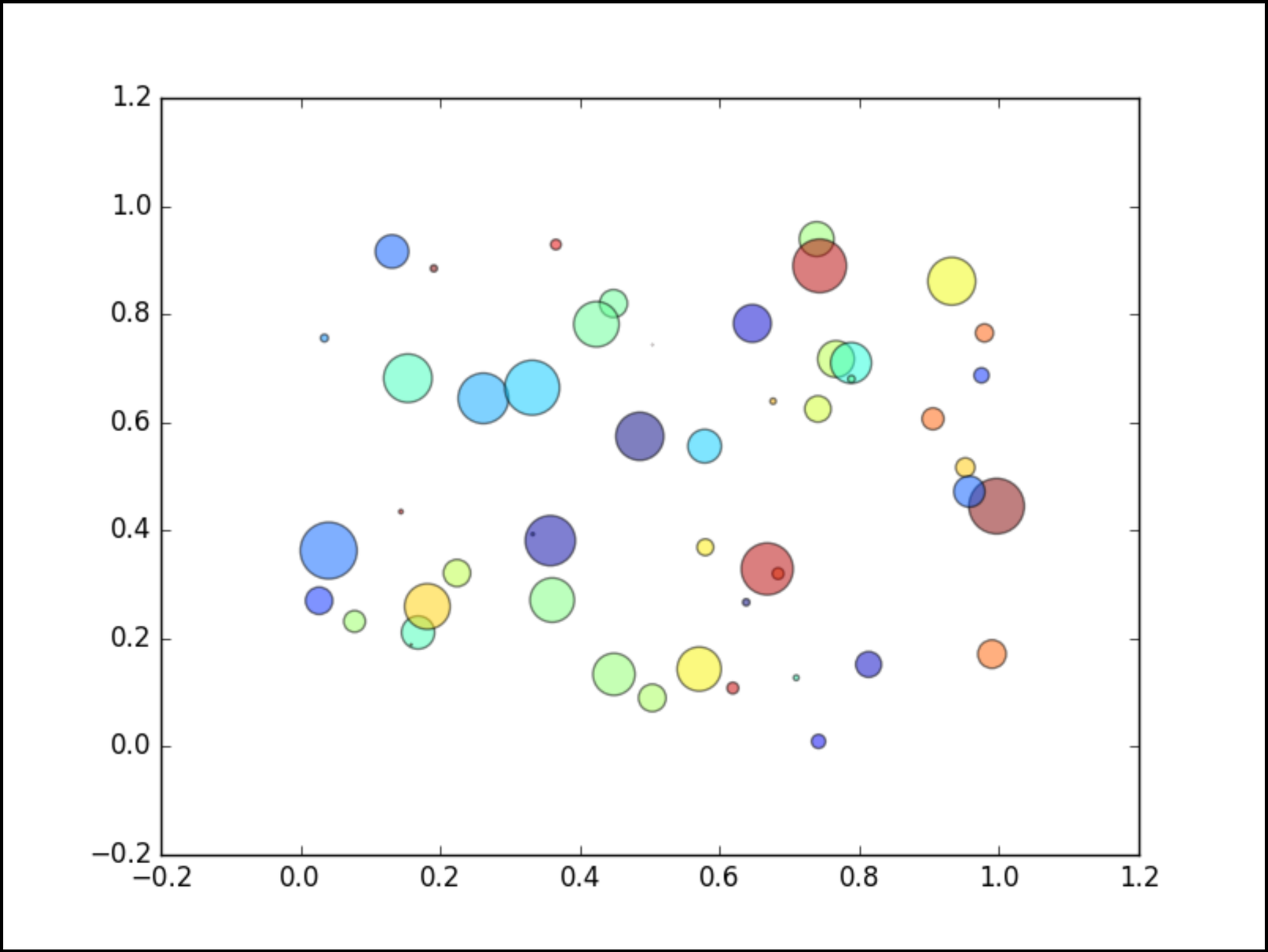
```
import numpy as np
import matplotlib.pyplot as plt
```

Using pyplot

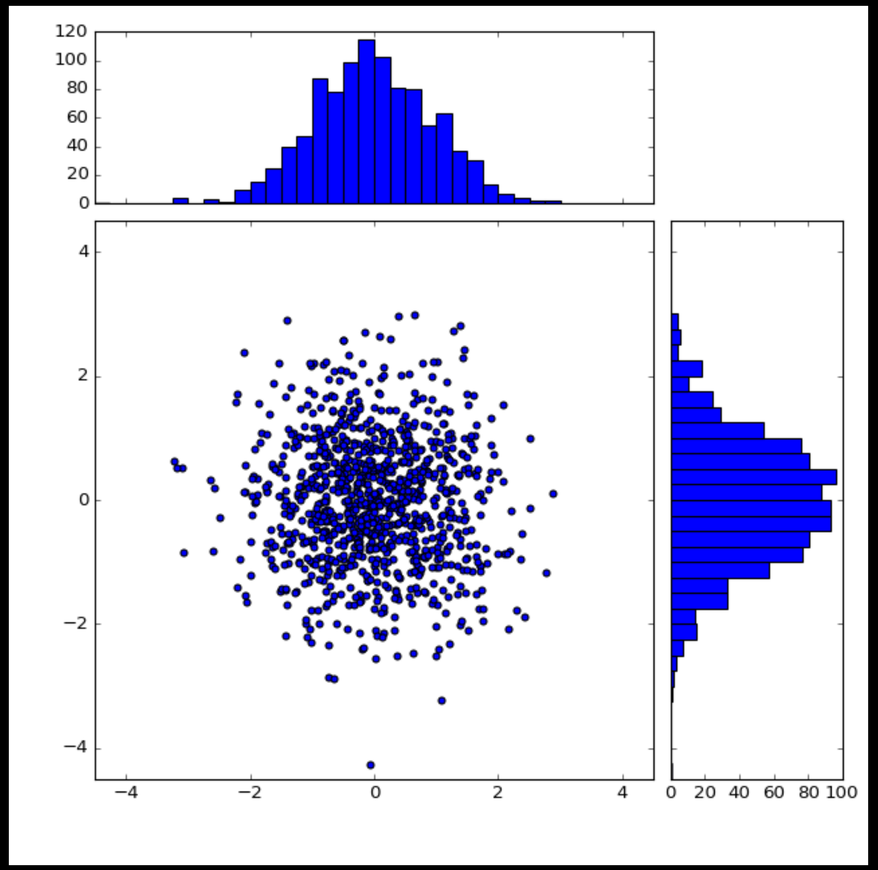
```
import numpy as np
import matplotlib.pyplot as plt

N = 50
x = np.random.rand(N)
y = np.random.rand(N)
colors = np.random.rand(N)
area = np.pi * (15 * np.random.rand(N))**2
```

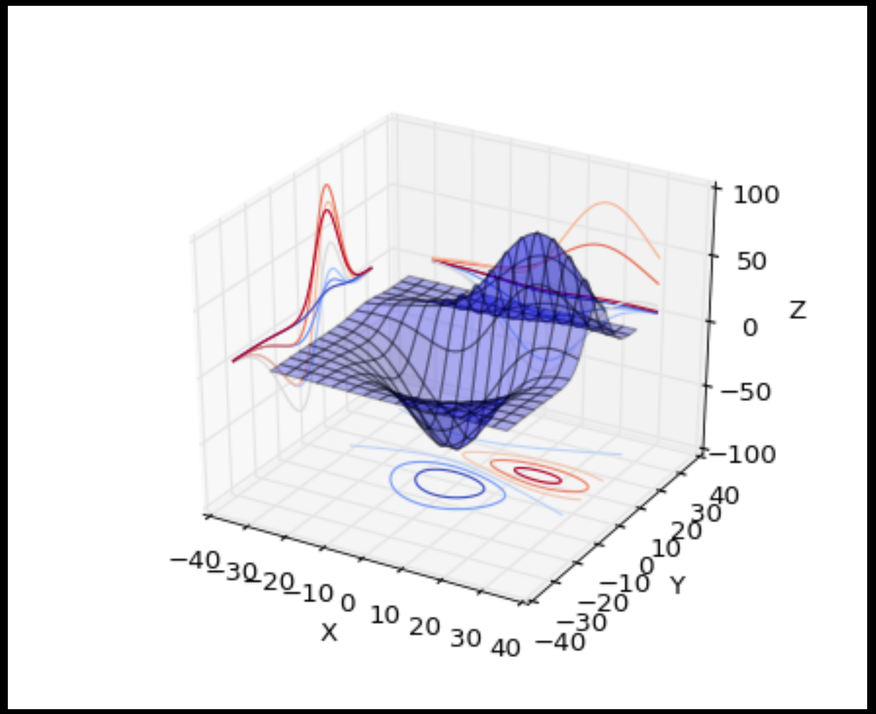
Using pyplot

```
import numpy as np
import matplotlib.pyplot as plt

N = 50
x = np.random.rand(N)
y = np.random.rand(N)
colors = np.random.rand(N)
area = np.pi * (15 * np.random.rand(N))**2

plt.scatter(x, y, s=area, c=colors, alpha=0.5)
plt.show()
```


1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2

And Many More!





Shoutouts

Shoutouts

tensorflow (MI)

keras (Deep Learning)

scikit-learn (ML)

nltk (Natural Language)

Development Tools

PEP8 and AutoPEP8

Style Enforcers for PEP8 – Python's Style Guide

- PEP8 identifies errors
- AutoPEP8 fixes errors

PEP8 Intro

AutoPEP8 Source

Using PEP8 and AutoPEP8

Using PEP8 and AutoPEP8

```
# Print PEP8 errors
$ pep8 mystery.py
```

Using PEP8 and AutoPEP8

```
# Print PEP8 errors
$ pep8 mystery.py
mystery.py:18:1: E302 expected 2 blank lines, found 1
mystery.py:22:19: E231 missing whitespace after ':'
...
```

Using PEP8 and AutoPEP8

```
# Print PEP8 errors
$ pep8 mystery.py
mystery.py:18:1: E302 expected 2 blank lines, found 1
mystery.py:22:19: E231 missing whitespace after ':'
...

# Overwrites file.py for PEP8 compliance
$ autopep8 --in-place --aggressive --aggressive file.py
$
```

Web Frameworks

Flask

Lightweight

Beginner-friendly

WSGI by Werkzeug

Templating by Jinja2

Documentation

Flask Mega-Tutorial

Flask Hello World

File: hello.py

Flask Hello World

```
from flask import Flask
app = Flask(__name__)
```

File: hello.py

Flask Hello World

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"
```

File: hello.py

Flask Hello World

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    app.run(debug=True)
```

File: hello.py

Flask Hello World

Flask Hello World

```
# Run a local server
```

Flask Hello World

```
# Run a local server
$ python hello.py
```

Flask Hello World

```
# Run a local server
$ python hello.py
* Running on http://localhost:5000/
```

Django

"Batteries-included"

Heavily-configurable

Industry-standard

Overview

Tutorial

Django Hello World

Django Hello World

```
# polls/views.py
```

Django Hello World

```
# polls/views.py
from django.http import HttpResponse
```

Django Hello World

```
# polls/views.py
from django.http import HttpResponse
def index(request):
    return HttpResponse("Hello, world. You're at the polls index.")
```

Django Hello World

```
# polls/views.py
from django.http import HttpResponse
def index(request):
    return HttpResponse("Hello, world. You're at the polls index.")

# polls/urls.py
```

Django Hello World

```
# polls/views.py
from django.http import HttpResponse
def index(request):
    return HttpResponse("Hello, world. You're at the polls index.")

# polls/urls.py
from django.conf.urls import url
from . import views
urlpatterns = [
    url(r'^$', views.index, name='index'),
]
```

Django Hello World

```
# polls/views.py
from django.http import HttpResponse
def index(request):
    return HttpResponse("Hello, world. You're at the polls index.")

# polls/urls.py
from django.conf.urls import url
from . import views
urlpatterns = [
    url(r'^$', views.index, name='index'),
]

# mysite/urls.py
from django.conf.urls import include, url
from django.contrib import admin

urlpatterns = [
    url(r'^polls/', include('polls.urls')),
    url(r'^admin/', include(admin.site.urls)),
]
```

Django Hello World

Django Hello World

```
$ python manage.py runserver
```

Twisted

Asynchronous, event-driven

Networking engine

Fast, but complicated

Echo Server

Echo Server

```
from twisted.internet import protocol, reactor, endpoints
```

Echo Server

```
from twisted.internet import protocol, reactor, endpoints

class Echo(protocol.Protocol):
    def dataReceived(self, data):
        self.transport.write(data)
```

Echo Server

```
from twisted.internet import protocol, reactor, endpoints

class Echo(protocol.Protocol):
    def dataReceived(self, data):
        self.transport.write(data)

class EchoFactory(protocol.Factory):
    def buildProtocol(self, addr):
        return Echo()
```

Echo Server

```
from twisted.internet import protocol, reactor, endpoints

class Echo(protocol.Protocol):
    def dataReceived(self, data):
        self.transport.write(data)

class EchoFactory(protocol.Factory):
    def buildProtocol(self, addr):
        return Echo()

endpoints.serverFromString(reactor, "tcp:1234").listen(EchoFactory())
```

Echo Server

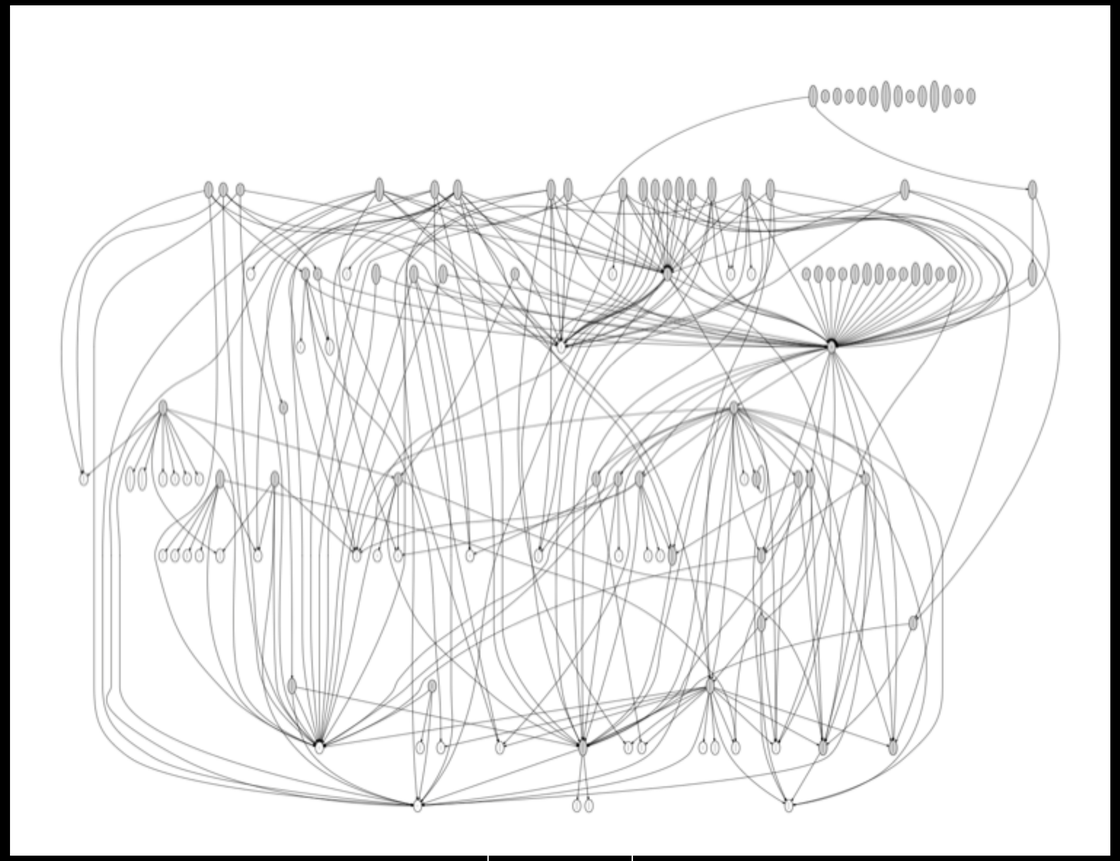
```
from twisted.internet import protocol, reactor, endpoints

class Echo(protocol.Protocol):
    def dataReceived(self, data):
        self.transport.write(data)

class EchoFactory(protocol.Factory):
    def buildProtocol(self, addr):
        return Echo()

endpoints.serverFromString(reactor, "tcp:1234").listen(EchoFactory())
reactor.run()
```

Codebase Complexity

Codebase Complexity - `snakefood`

Module Dependencies

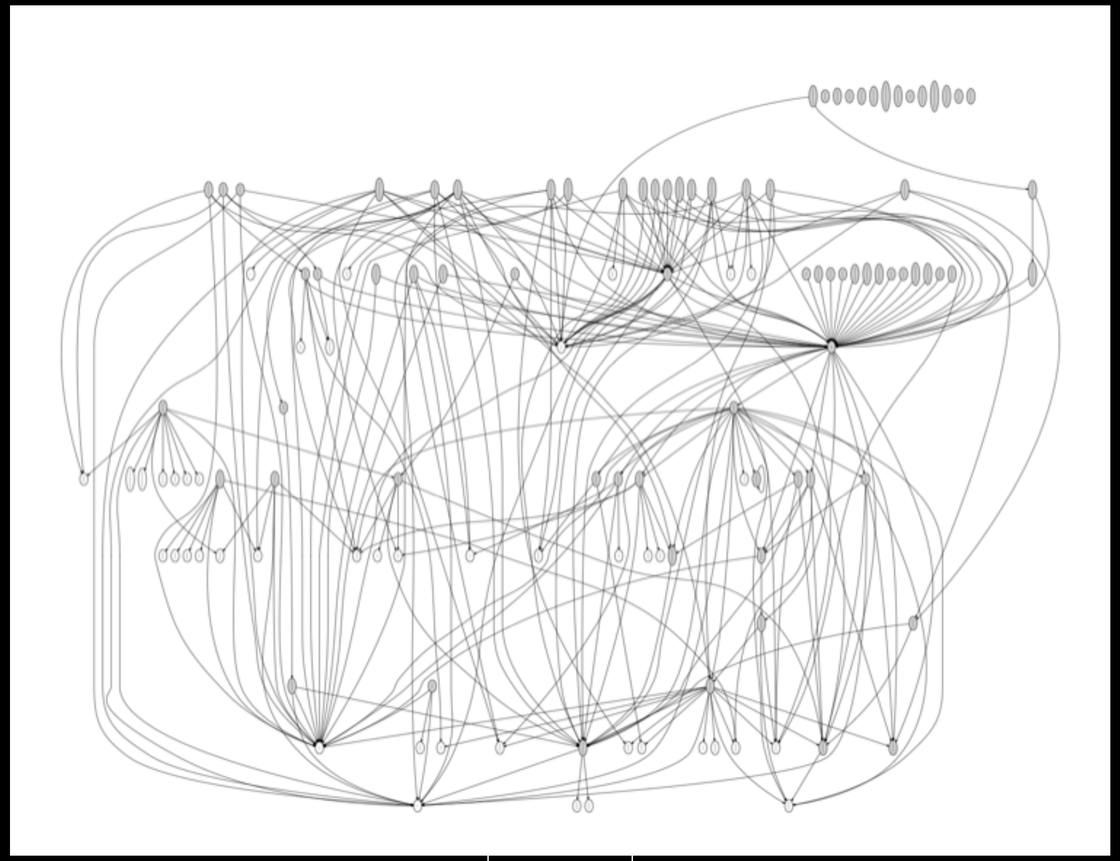
Codebase Complexity - `snakefood`



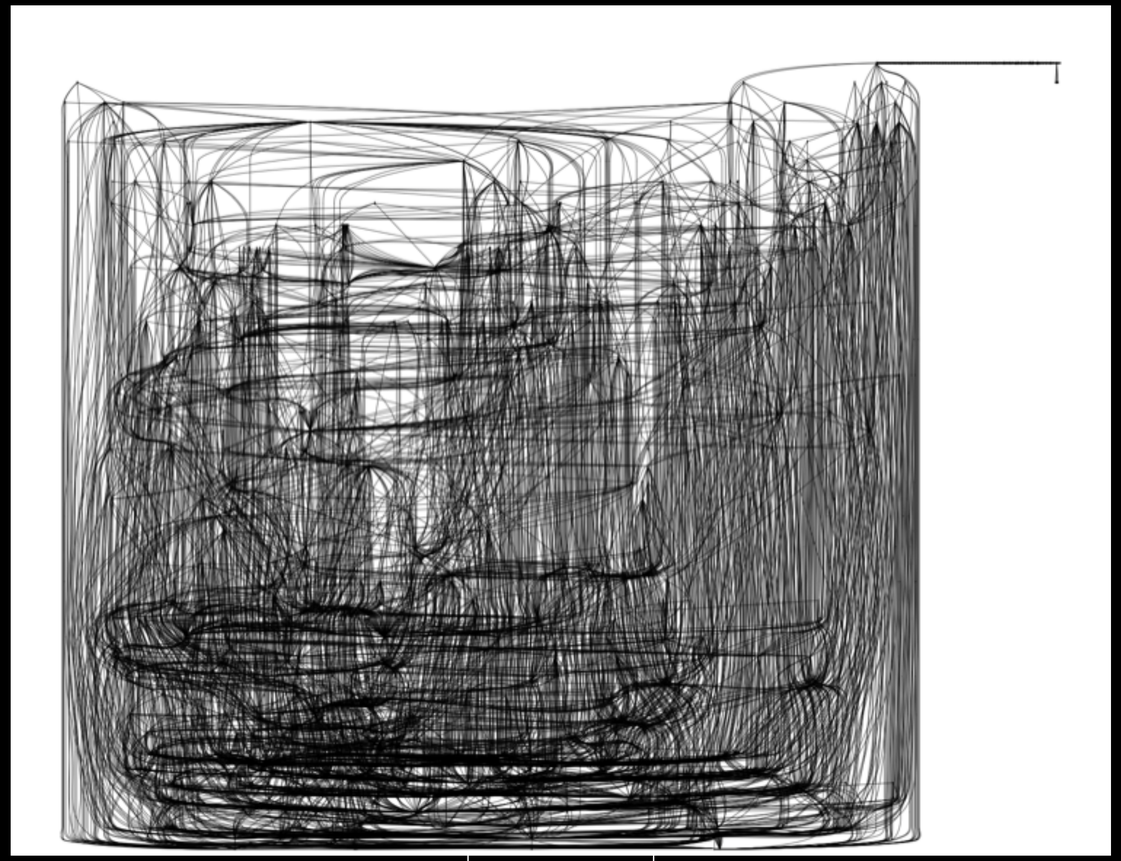
Flask

Module Dependencies

Codebase Complexity - `snakefood`



Flask



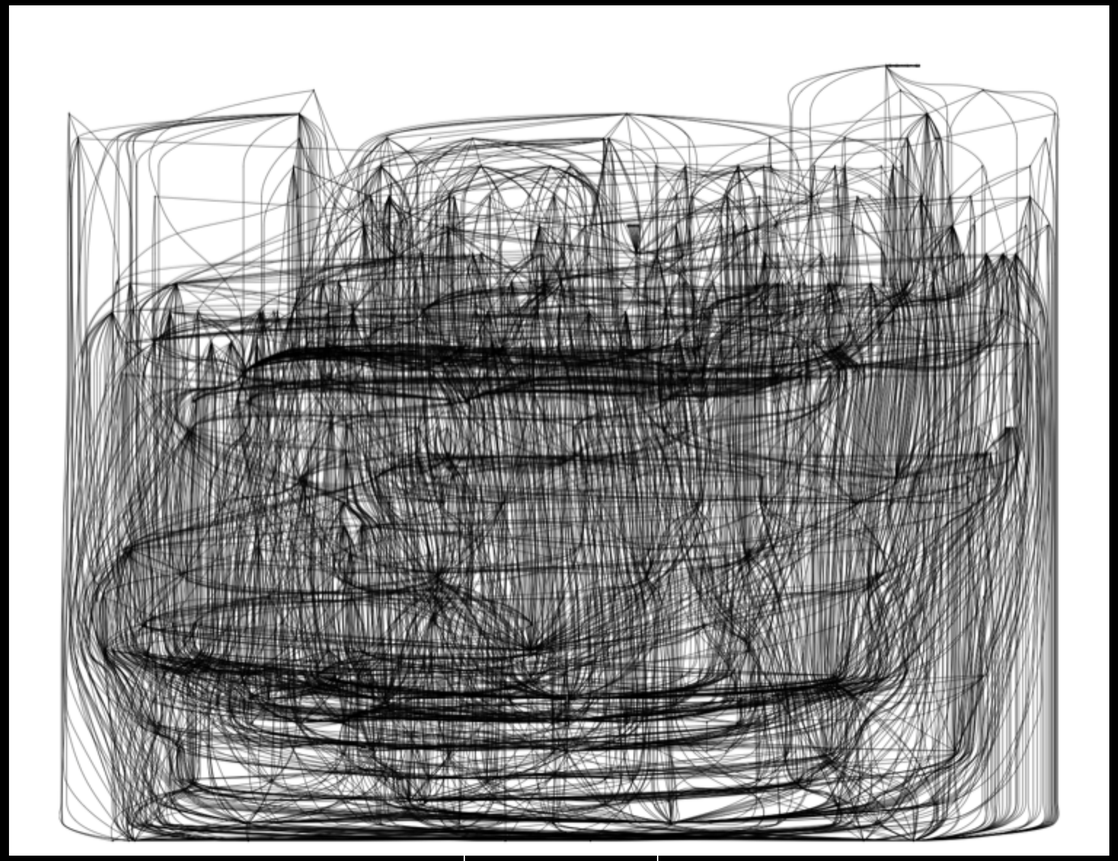
Django

Module Dependencies

Codebase Complexity - `snakefood`

Complex

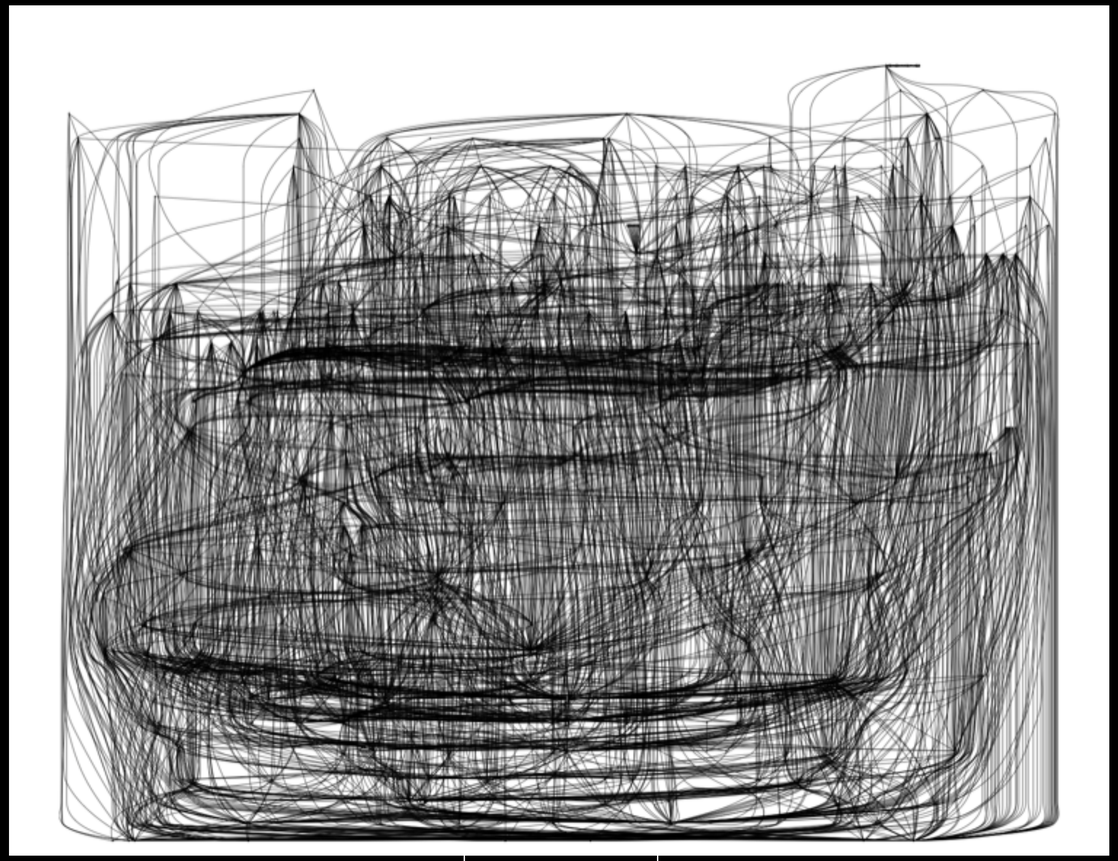
Codebase Complexity - `snakefood`



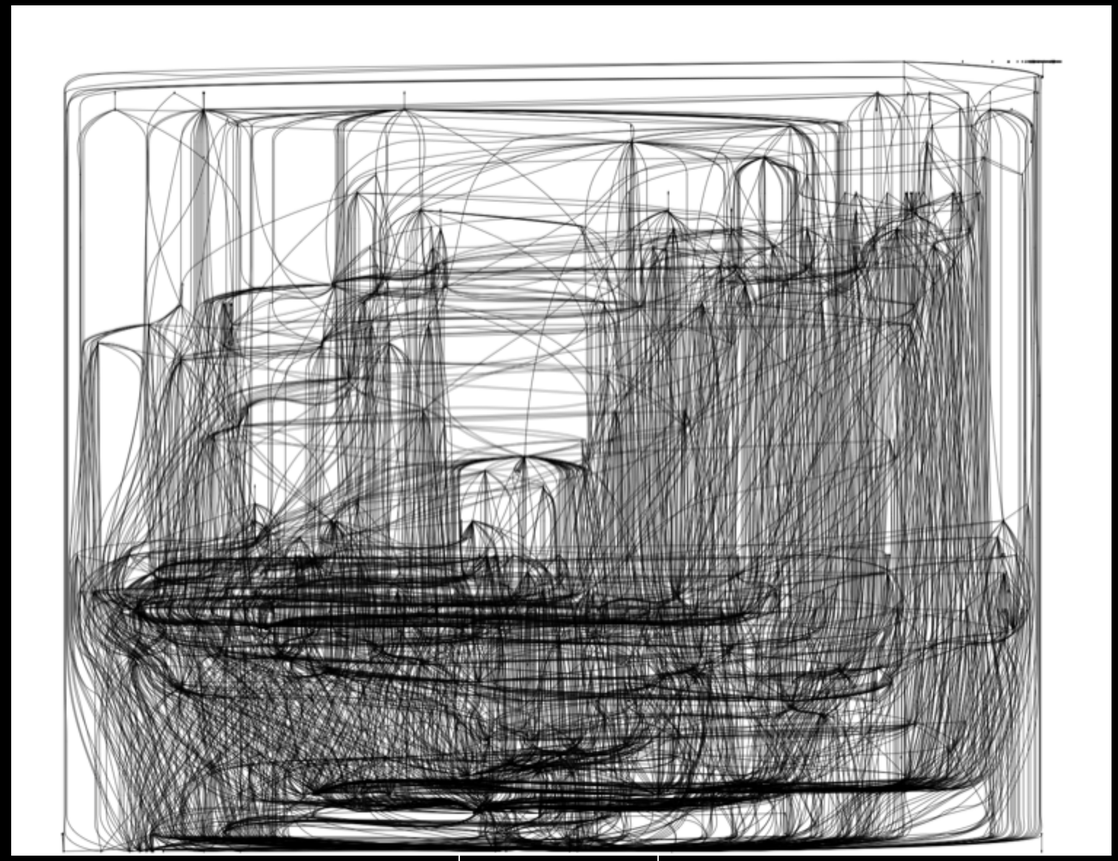
Twisted

Complex

Codebase Complexity - `snakefood`



Twisted

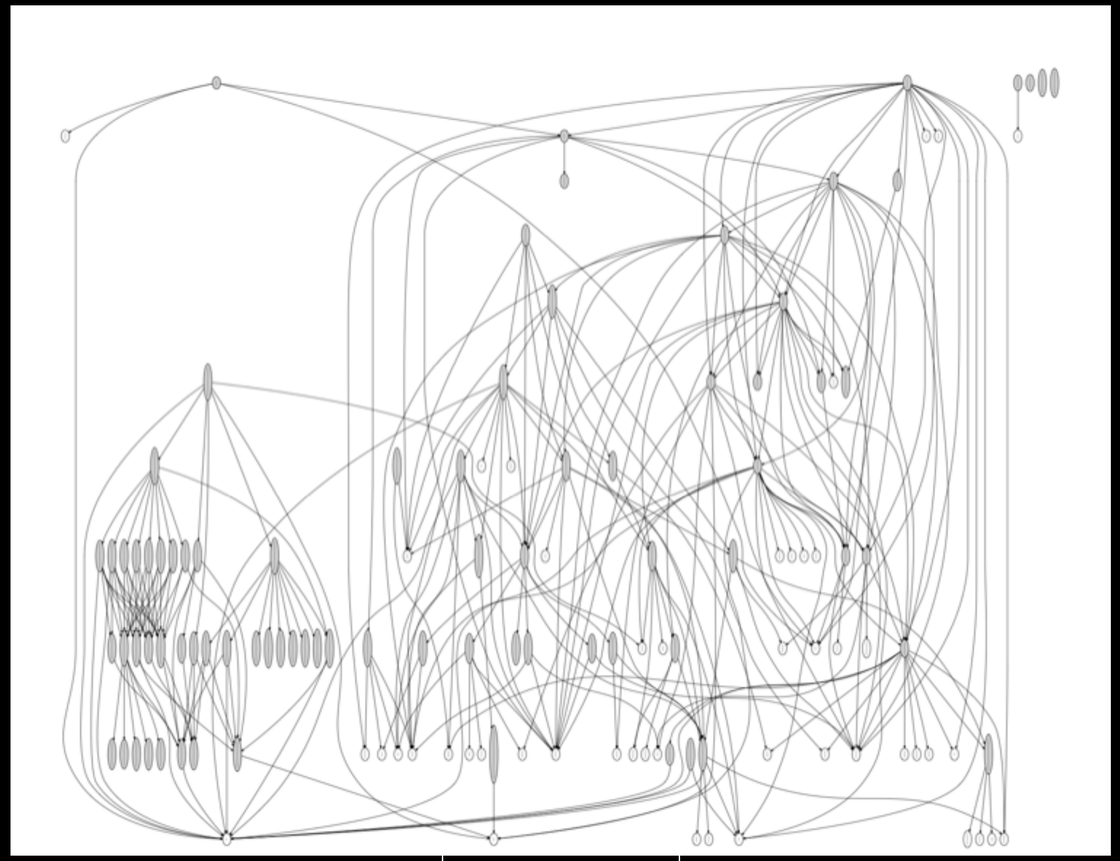


iPython

Complex

Codebase Complexity - `snakefood`

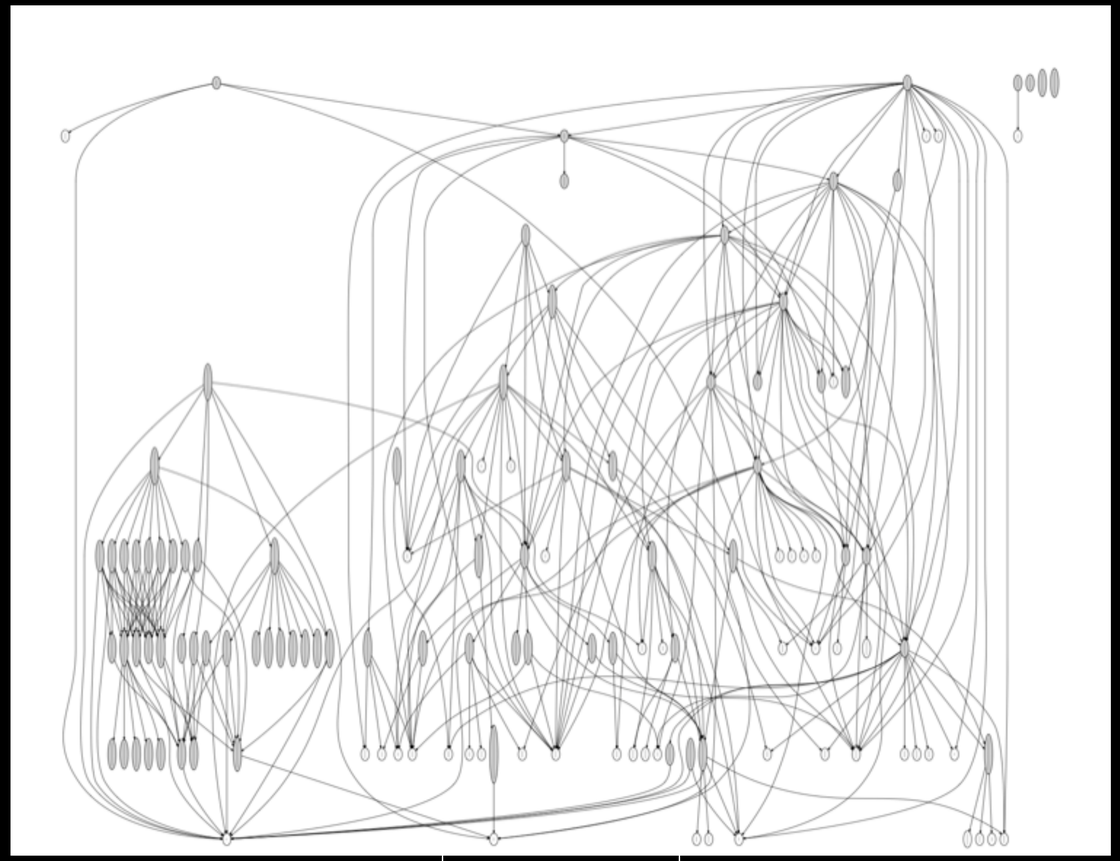
Codebase Complexity - `snakefood`



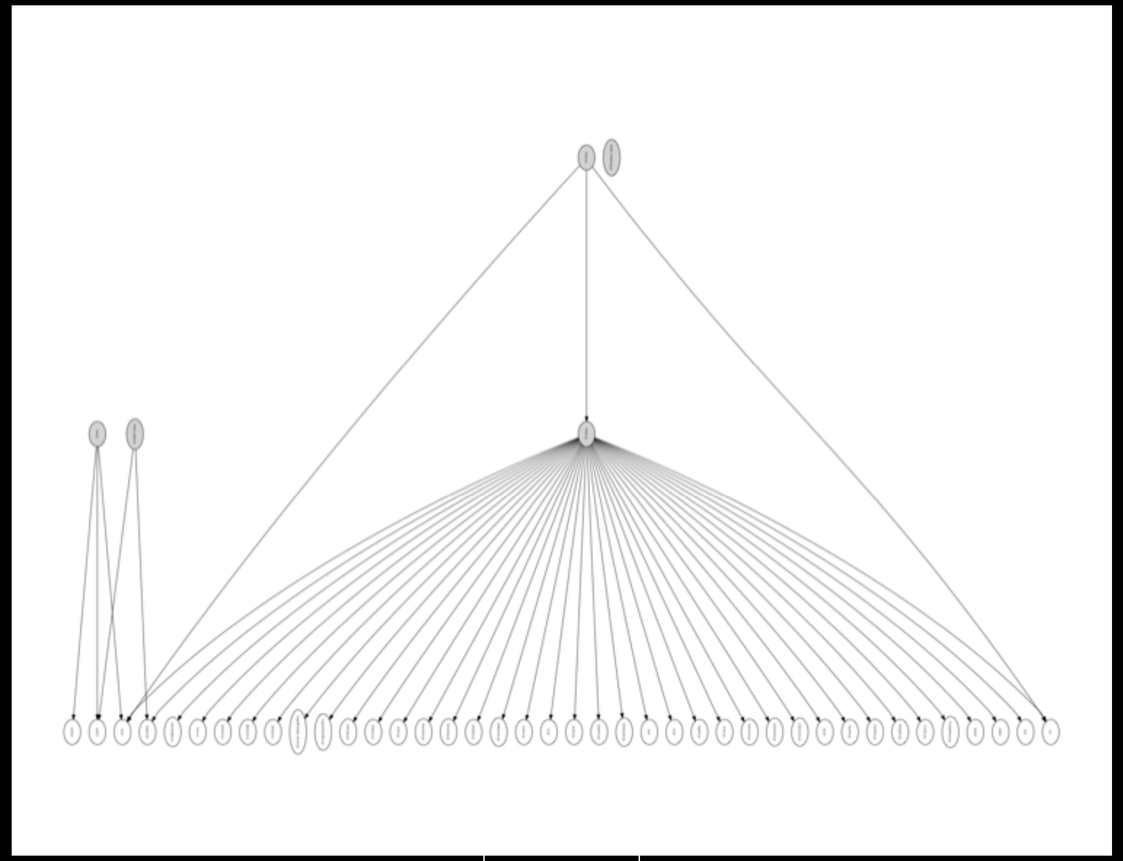
Requests

Simple

Codebase Complexity - `snakefood`



Requests



Bottle

Simple

Other Neat Packages

Extras

Extras

Selenium – Automated web client

Extras

Selenium – Automated web client

Dateutil – Better date/time handling

Extras

Selenium – Automated web client

Dateutil – Better date/time handling

SQLAlchemy – Build objects on top of database schemas

Extras

Selenium – Automated web client

Dateutil – Better date/time handling

SQLAlchemy – Build objects on top of database schemas

bcrypt / cryptography – Cryptographic tools

Extras

Selenium – Automated web client

Dateutil – Better date/time handling

SQLAlchemy – Build objects on top of database schemas

bcrypt / cryptography – Cryptographic tools

scrapy – Web scraping

Extras

Selenium – Automated web client

Dateutil – Better date/time handling

SQLAlchemy – Build objects on top of database schemas

bcrypt / cryptography – Cryptographic tools

scrapy – Web scraping

pandas – Vectorized data manipulation

Extras

Selenium – Automated web client

Dateutil – Better date/time handling

SQLAlchemy – Build objects on top of database schemas

bcrypt / cryptography – Cryptographic tools

scrapy – Web scraping

pandas – Vectorized data manipulation

boto – Python interface to Amazon Web Services

Extras

Extras

PyEphem – Track planets and satellites.

Extras

PyEphem – Track planets and satellites.

Basemap – Plot 2D data on maps.

Extras

PyEphem – Track planets and satellites.

Basemap – Plot 2D data on maps.

PRAW – Play with reddit's API, write a reddit bot.

Extras

PyEphem – Track planets and satellites.

Basemap – Plot 2D data on maps.

PRAW – Play with reddit's API, write a reddit bot.

robobrowser / mechanize – Automate web tasks.

Extras

PyEphem – Track planets and satellites.

Basemap – Plot 2D data on maps.

PRAW – Play with reddit's API, write a reddit bot.

robobrowser / mechanize – Automate web tasks.

networkx – Visualize small graphs.

Extras

PyEphem – Track planets and satellites.

Basemap – Plot 2D data on maps.

PRAW – Play with reddit's API, write a reddit bot.

robobrowser / mechanize – Automate web tasks.

networkx – Visualize small graphs.

BeautifulSoup – Rapidly prototype HTML parsers.

Extras

PyEphem – Track planets and satellites.

Basemap – Plot 2D data on maps.

PRAW – Play with reddit's API, write a reddit bot.

robobrowser / mechanize – Automate web tasks.

networkx – Visualize small graphs.

BeautifulSoup – Rapidly prototype HTML parsers.

pygame – Basic game-playing tools.

Coming Soon?

Coming Soon?

GOvals and GRects?

Coming Soon?

GOvals and GRects?

(Extremely) alpha release

Coming Soon?

GOvals and GRects?

(Extremely) alpha release

Speak with us after class

What Now?

Build Build Build

Python has a strong open-source community - contribute!

What's available? Read the Standard Library

Exact Python specification? Read the Language Reference

Still have questions? Read the FAQ

Build something you love!

Next Time

Lab

Lab

Build a wallpaper scraper!

or… explore cool packages!

or… start a final project!

Next Week

Next Week

The Python Ecosystem

Python 2 vs. Python 3

Python Style

Joining a Python Codebase

Credit

Documentation for all these awesome libraries!

GrokCode for codebase complexity graphs```

FOSS community for making tools available.

PSF, of course.