JupyterLab:
Building Blocks for Interactive Computing

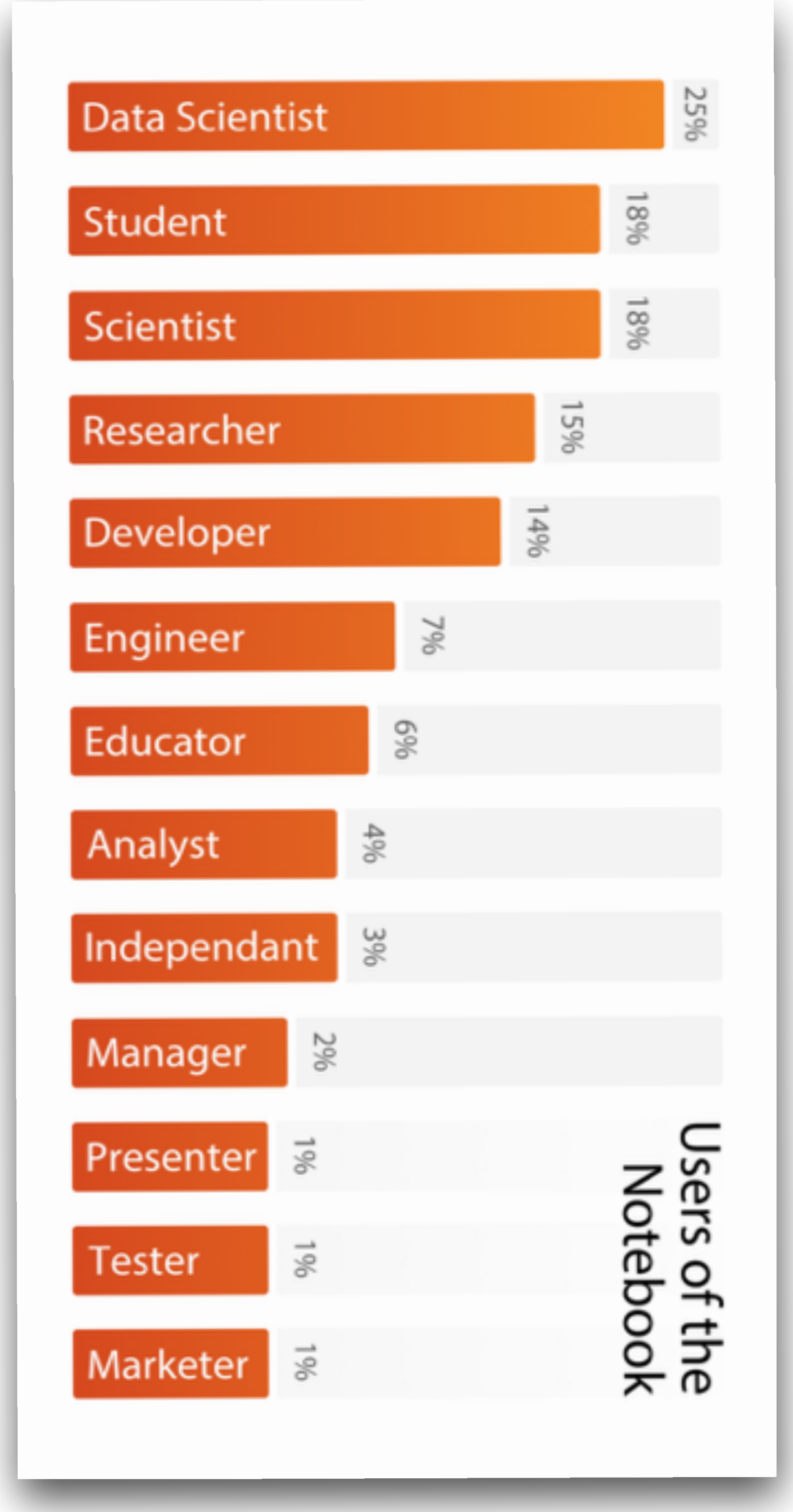
jupyter

Jupyter Days Atlanta, August 2016
Steven Silvester
Continuum Analytics

Brian E. Granger, Cal Poly
Jason Grout, Bloomberg LP
Chris Colbert, Continuum
Sylvain Corlay, Bloomberg
Afshin Darian, Continuum

Cameron Oelsen, Cal Poly
Fernando Perez, LBNL/Berkeley
David Willmer
The larger Jupyter Team

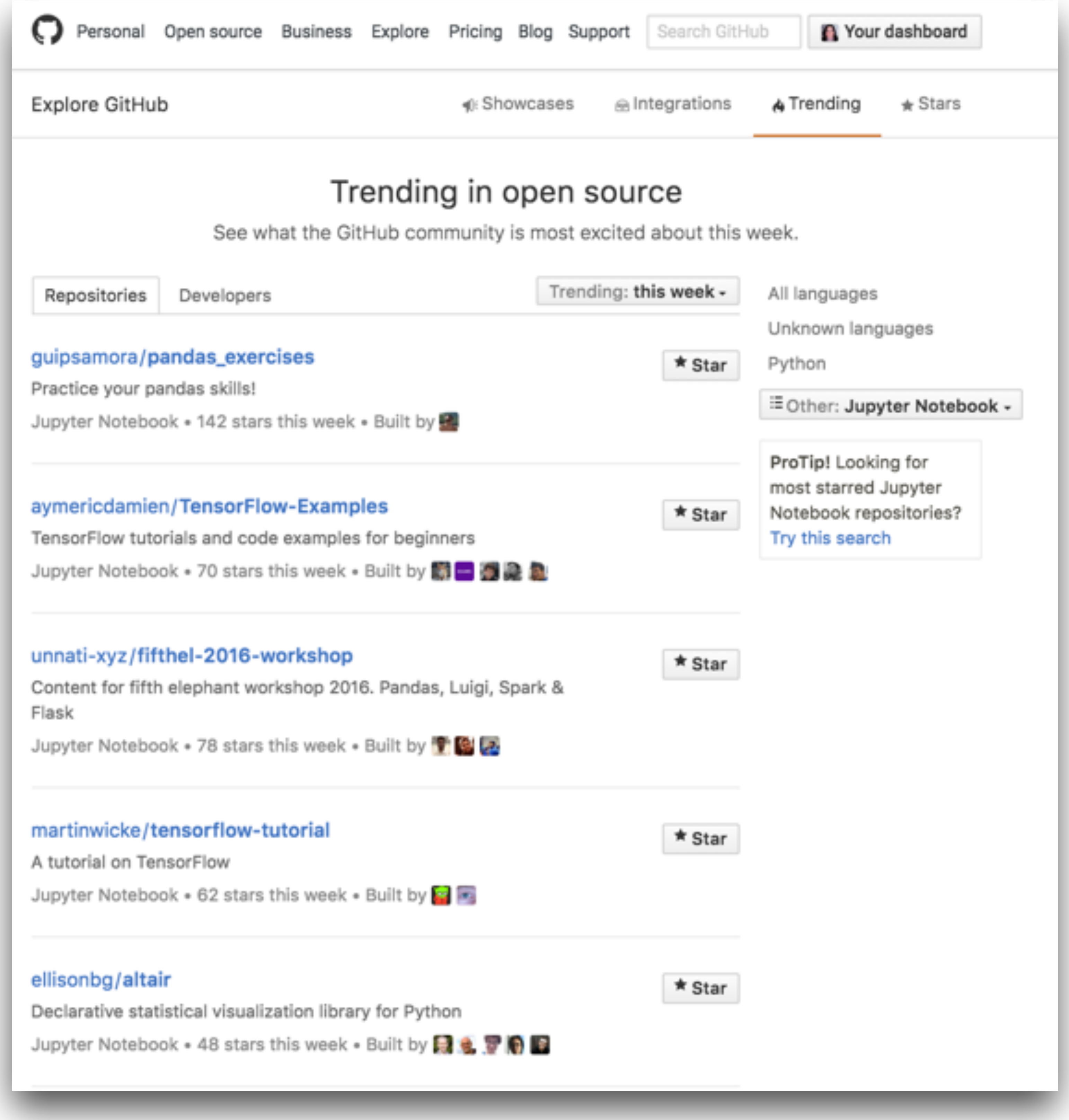
Where Are We Today?



~3M Jupyter Users

https://github.com/jupyter/design/blob/master/surveys/2015-notebook-ux/analysis/report_dashboard.ipynb

Over 500k Notebooks on GitHub



Trending in open source

See what the GitHub community is most excited about this week.

guipsamora/pandas_exercises

Practice your pandas skills!

Jupyter Notebook • 142 stars this week • Built by

aymericdamien/TensorFlow-Examples

TensorFlow tutorials and code examples for beginners

Jupyter Notebook • 70 stars this week • Built by

unnati-xyz/fifthel-2016-workshop

Content for fifth elephant workshop 2016. Pandas, Luigi, Spark & Flask

Jupyter Notebook • 78 stars this week • Built by

martinwicke/tensorflow-tutorial

A tutorial on TensorFlow

Jupyter Notebook • 62 stars this week • Built by

ellisonbg/altair

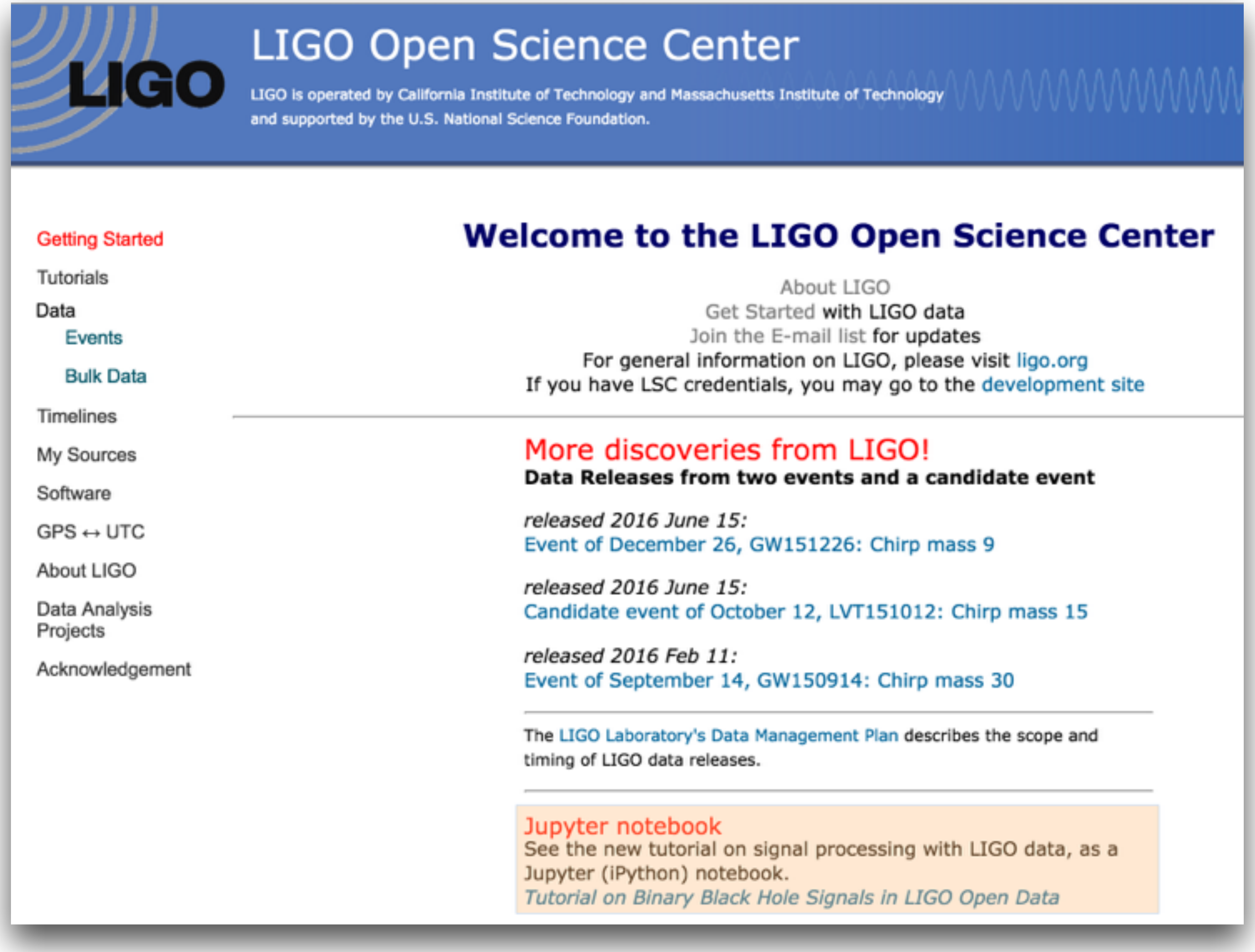
Declarative statistical visualization library for Python

Jupyter Notebook • 48 stars this week • Built by

ProTip! Looking for most starred Jupyter Notebook repositories? Try this search

https://github.com/trending/jupyter-notebook?since=weekly

Enabling Reproducible Science



LIGO Open Science Center

LIGO is operated by California Institute of Technology and Massachusetts Institute of Technology and supported by the U.S. National Science Foundation.

Getting Started
Tutorials
Data
Events
Bulk Data
Timelines
My Sources
Software
GPS ↔ UTC
About LIGO
Data Analysis Projects
Acknowledgement

Welcome to the LIGO Open Science Center

About LIGO
Get Started with LIGO data
Join the E-mail list for updates
For general information on LIGO, please visit ligo.org
If you have LSC credentials, you may go to the development site

More discoveries from LIGO!

Data Releases from two events and a candidate event

released 2016 June 15:
Event of December 26, GW151226: Chirp mass 9

released 2016 June 15:
Candidate event of October 12, LVT151012: Chirp mass 15

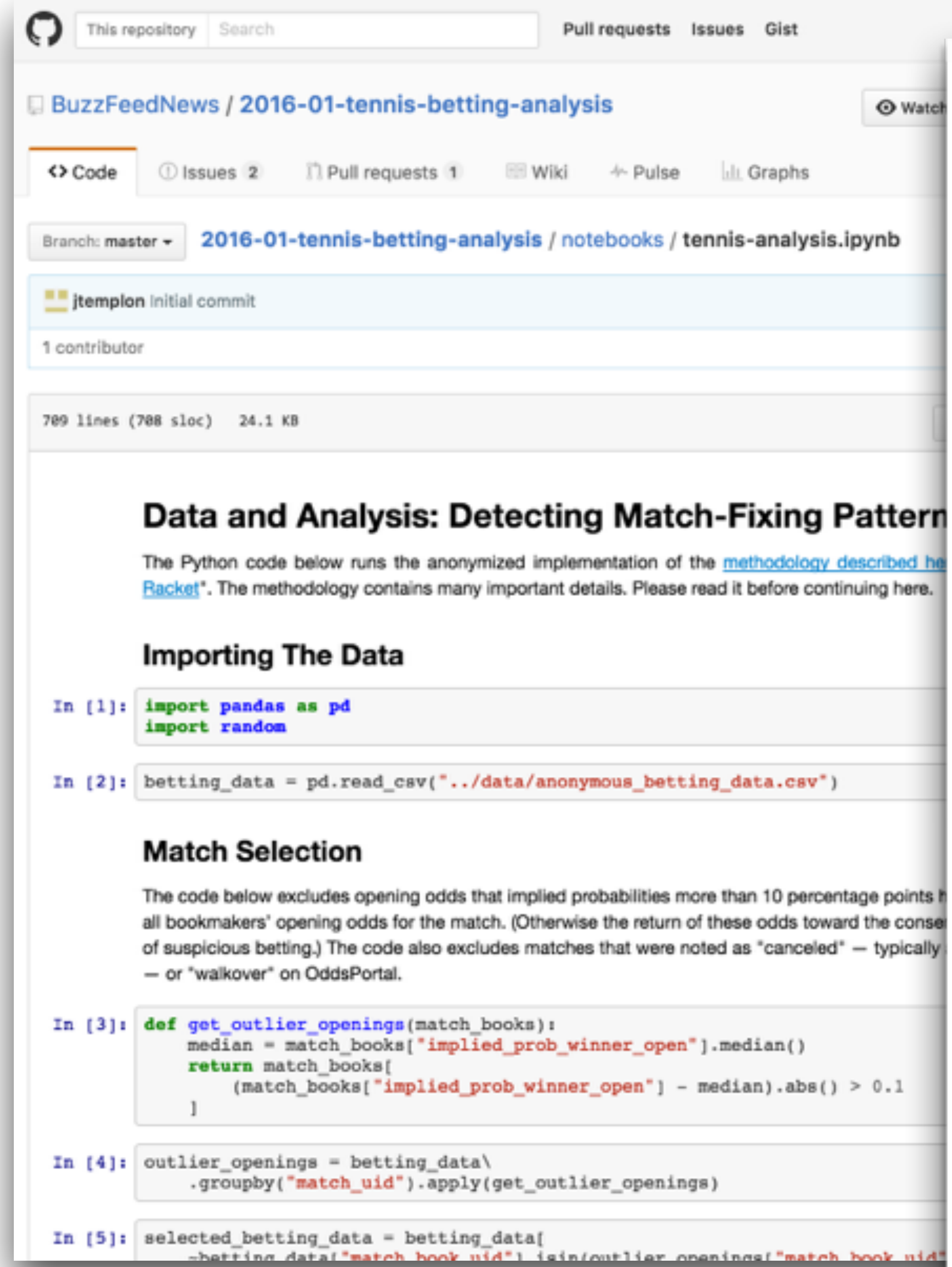
released 2016 Feb 11:
Event of September 14, GW150914: Chirp mass 30

The LIGO Laboratory's Data Management Plan describes the scope and timing of LIGO data releases.

Jupyter notebook
See the new tutorial on signal processing with LIGO data, as a Jupyter (iPython) notebook.
Tutorial on Binary Black Hole Signals in LIGO Open Data

https://losc.ligo.org/about/

Enabling Open Data Journalism



BuzzFeedNews / 2016-01-tennis-betting-analysis

2016-01-tennis-betting-analysis / notebooks / tennis-analysis.ipynb

jtemplon Initial commit

1 contributor

709 lines (708 sloc) 24.1 KB

Data and Analysis: Detecting Match-Fixing Pattern

The Python code below runs the anonymized implementation of the methodology described he Racket". The methodology contains many important details. Please read it before continuing here.

Importing The Data

```
In [1]: import pandas as pd
        import random
```

```
In [2]: betting_data = pd.read_csv("../data/anonymous_betting_data.csv")
```

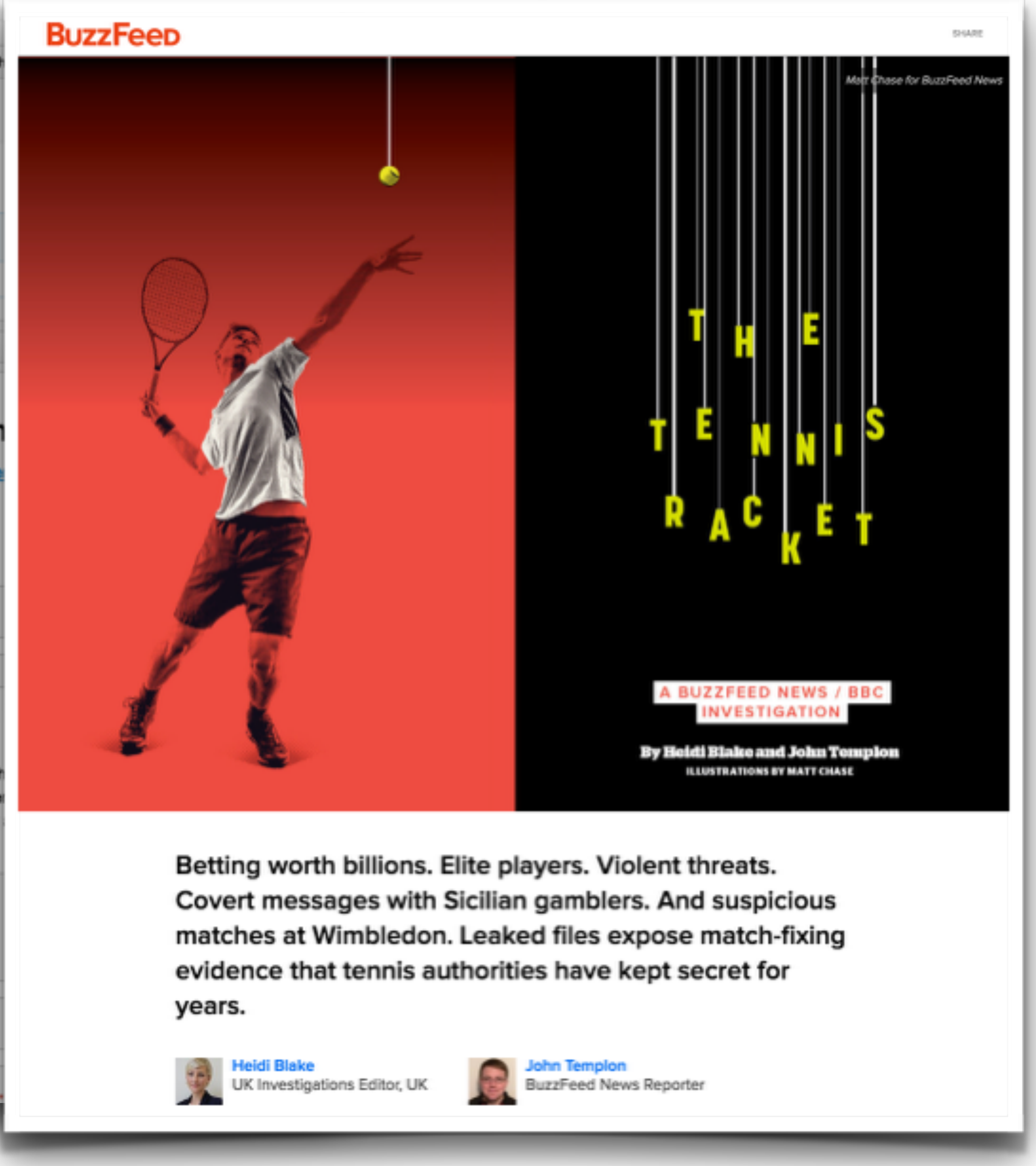
Match Selection

The code below excludes opening odds that implied probabilities more than 10 percentage points f all bookmakers' opening odds for the match. (Otherwise the return of these odds toward the conse of suspicious betting.) The code also excludes matches that were noted as "canceled" — typically — or "walkover" on OddsPortal.

```
In [3]: def get_outlier_openings(match_books):
            median = match_books["implied_prob_winner_open"].median()
            return match_books[
                (match_books["implied_prob_winner_open"] - median).abs() > 0.1
            ]
```

```
In [4]: outlier_openings = betting_data\
            .groupby("match_uid").apply(get_outlier_openings)
```

```
In [5]: selected_betting_data = betting_data[
            ~betting_data["match_book_uid"].isin(outlier_openings["match_book_uid"
```



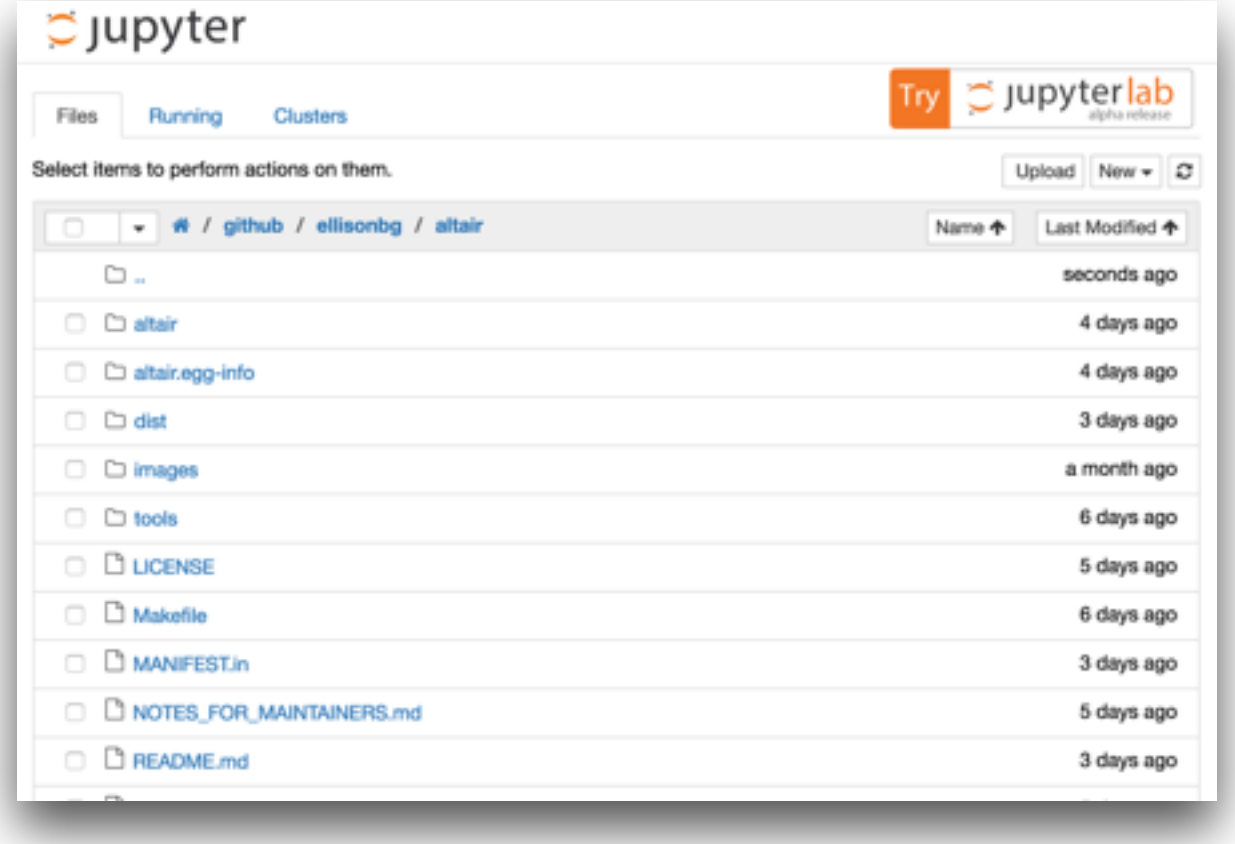
BuzzFeed
SHARE
Matt Chase for BuzzFeed News
THE TENNIS RACKET
A BUZZFEED NEWS / BBC INVESTIGATION
By Heidi Blake and John Templon
ILLUSTRATIONS BY MATT CHASE

Betting worth billions. Elite players. Violent threats. Covert messages with Sicilian gamblers. And suspicious matches at Wimbledon. Leaked files expose match-fixing evidence that tennis authorities have kept secret for years.

Heidi Blake
UK Investigations Editor, UK

John Templon
BuzzFeed News Reporter

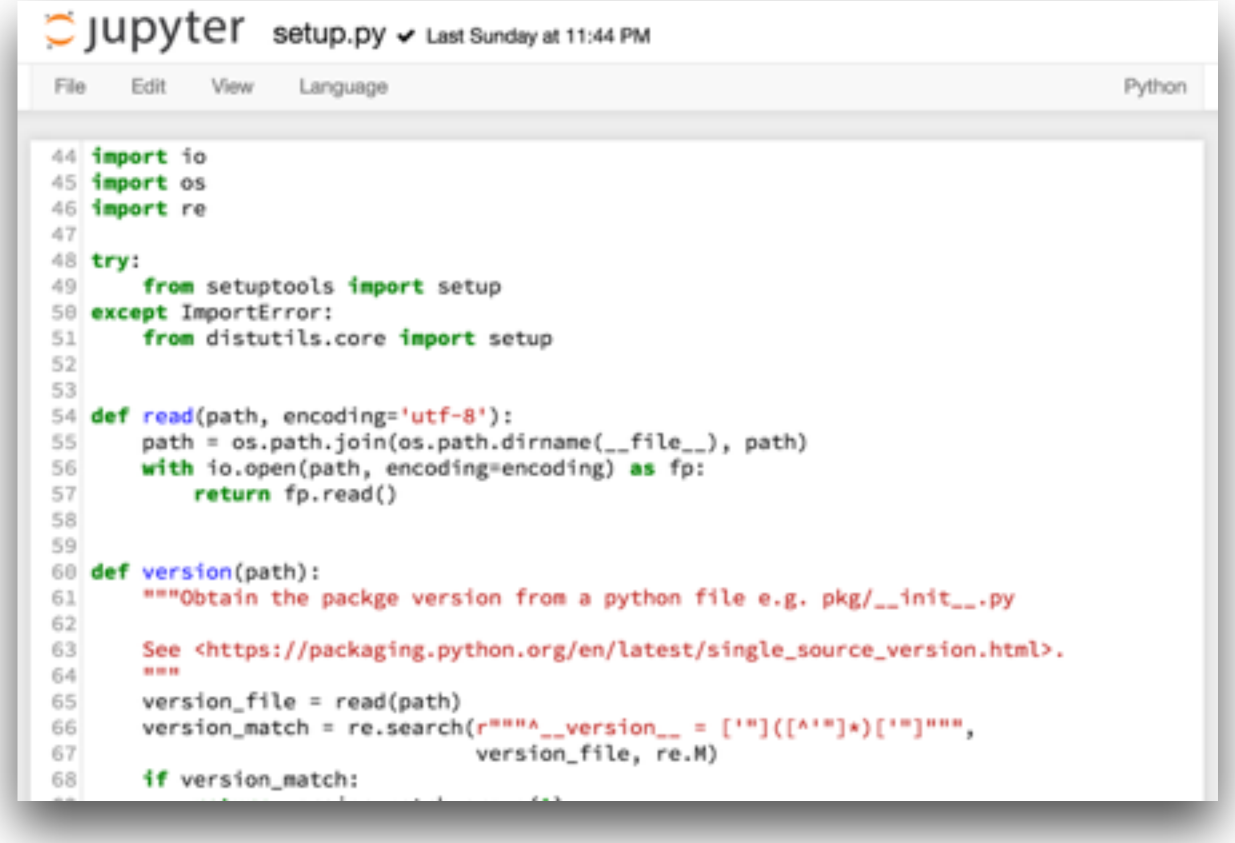
More Than Just Notebooks



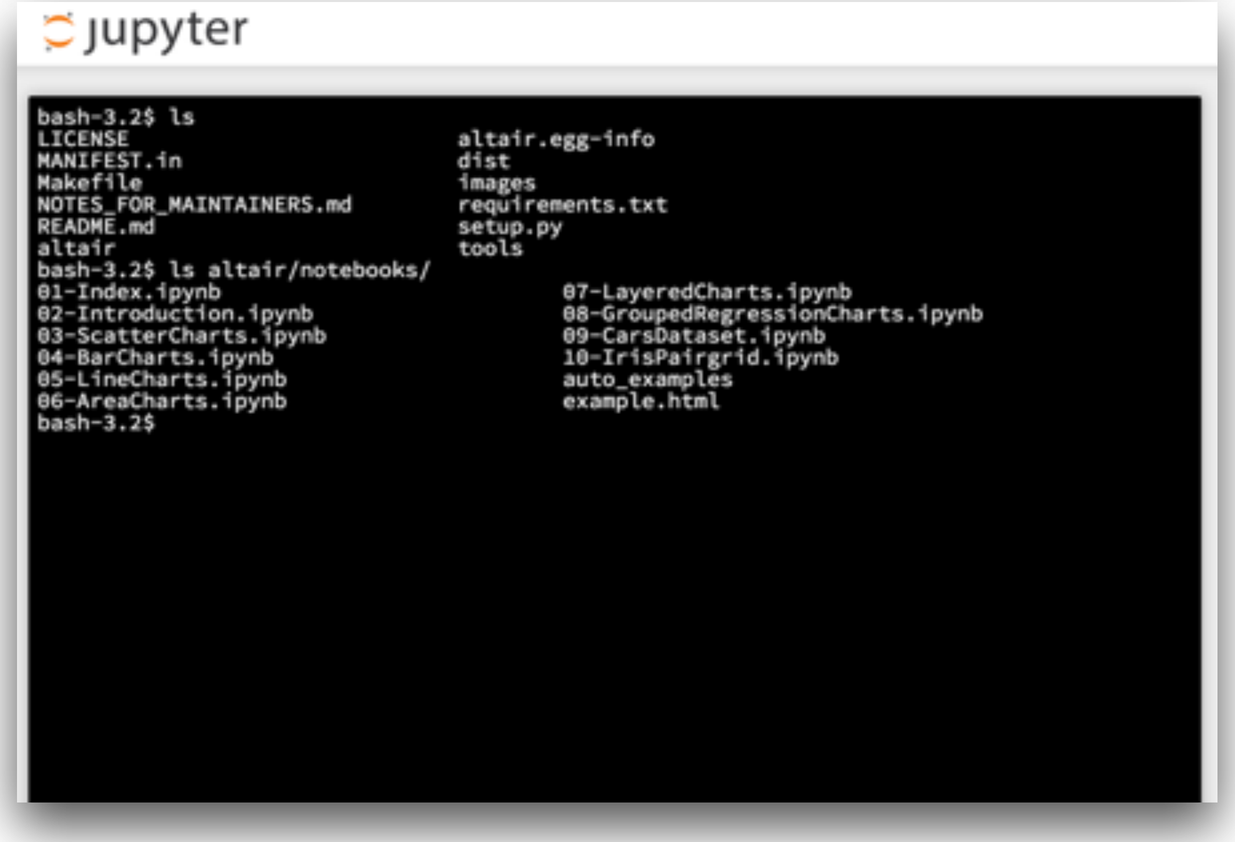
Select items to perform actions on them.

/ github / ellisonbg / altair

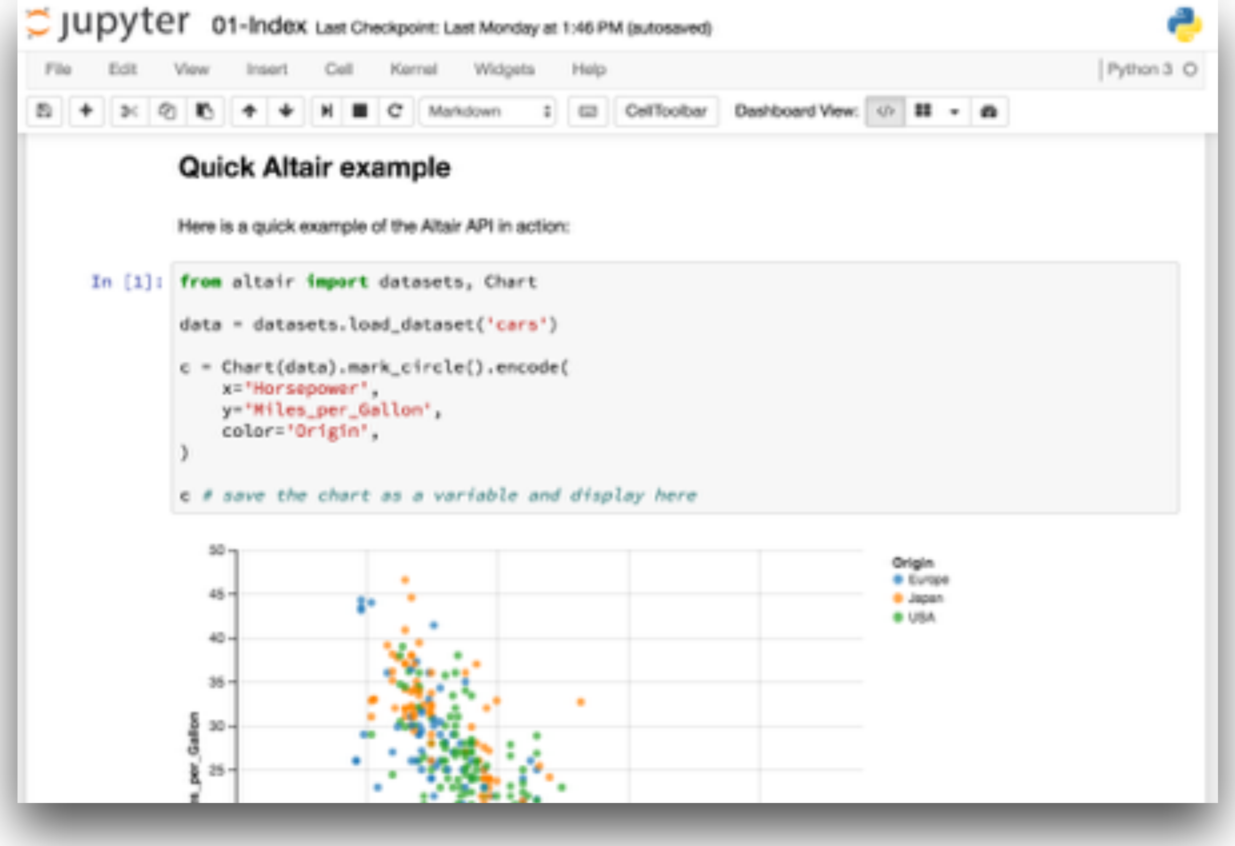
Name	Last Modified
..	seconds ago
altair	4 days ago
altair.egg-info	4 days ago
dist	3 days ago
images	a month ago
tools	6 days ago
LICENSE	5 days ago
Makefile	6 days ago
MANIFEST.in	3 days ago
NOTES_FOR_MAINTAINERS.md	5 days ago
README.md	3 days ago



```
import io
import os
import re

try:
    from setuptools import setup
except ImportError:
    from distutils.core import setup


def read(path, encoding='utf-8'):
    path = os.path.join(os.path.dirname(__file__), path)
    with io.open(path, encoding=encoding) as fp:
        return fp.read()


def version(path):
    """Obtain the packge version from a python file e.g. pkg/__init__.py

    See <https://packaging.python.org/en/latest/single_source_version.html>.
    """
    version_file = read(path)
    version_match = re.search(r"""^__version__ = ['"]([^'"]*)['"]""",
                              version_file, re.M)
    if version_match:
```



```
bash-3.2$ ls
LICENSE                                 altair.egg-info
MANIFEST.in                             dist
Makefile                                images
NOTES_FOR_MAINTAINERS.md                requirements.txt
README.md                               setup.py
altair                                  tools
bash-3.2$ ls altair/notebooks/
01-Index.ipynb                                  07-LayeredCharts.ipynb
02-Introduction.ipynb                           08-GroupedRegressionCharts.ipynb
03-ScatterCharts.ipynb                          09-CarsDataset.ipynb
04-BarCharts.ipynb                              10-IrisPairgrid.ipynb
05-LineCharts.ipynb                             auto_examples
06-AreaCharts.ipynb                             example.html
bash-3.2$
```

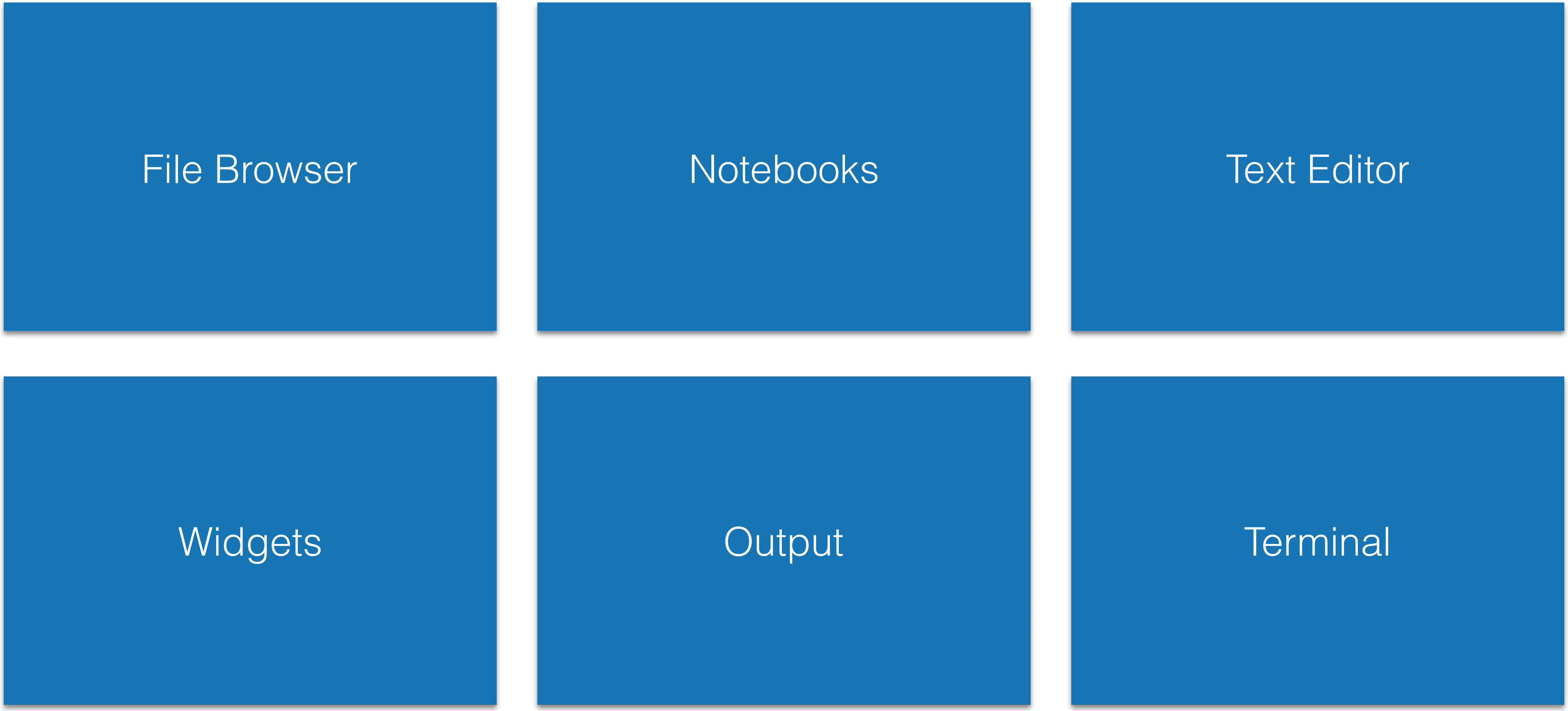


Quick Altair example

Here is a quick example of the Altair API in action:

```
In [1]: from altair import datasets, Chart

        data = datasets.load_dataset('cars')

        c = Chart(data).mark_circle().encode(
            x='Horsepower',
            y='Miles_per_Gallon',
            color='Origin',
        )

        c # save the chart as a variable and display here
```

Building Blocks



File Browser

Notebooks

Text Editor

Widgets

Output

Terminal

What Are We Hearing From Users?

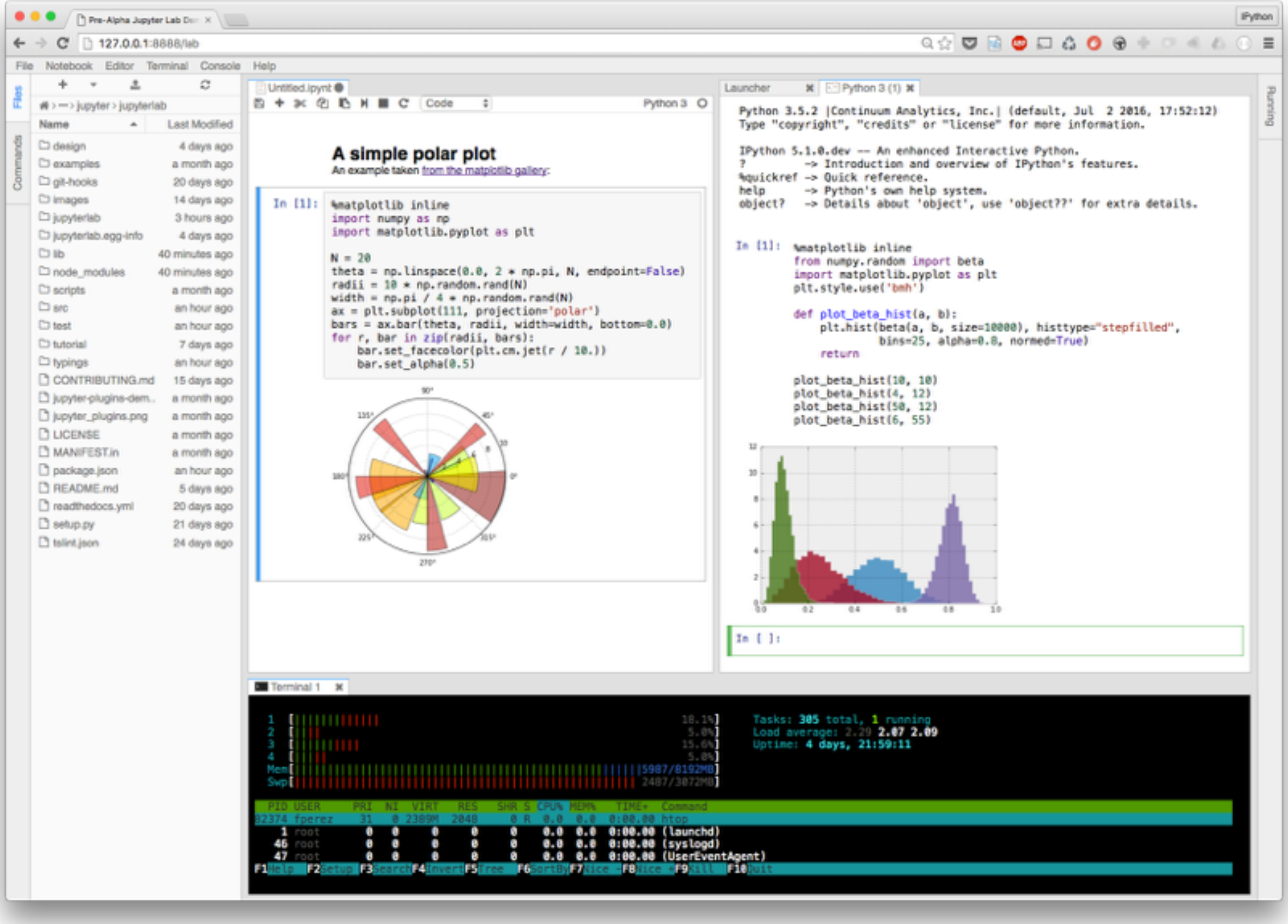
2015 User Experience Survey

- Mostly daily/weekly users
- Love the notebook workflow and user experience
- Top needs:
  - Integration with version control systems (git/GitHub)
  - Code/text editing
  - Layout/integration of building blocks
  - Debugger, profiler, variable inspector, etc.

https://github.com/jupyter/design/blob/master/surveys/2015-notebook-ux/analysis/report_dashboard.ipynb

Introducing JupyterLab (alpha)

JupyterLab: unifying these ideas



~ > jupyter > jupyterlab

Name	Last Modified
design	4 days ago
examples	a month ago
git-hooks	20 days ago
images	14 days ago
jupyterlab	3 hours ago
jupyterlab.egg-info	4 days ago
lib	40 minutes ago
node_modules	40 minutes ago
scripts	a month ago
src	an hour ago
test	an hour ago
tutorial	7 days ago
typings	an hour ago
CONTRIBUTING.md	15 days ago
jupyter-plugins-dem...	a month ago
jupyter_plugins.png	a month ago
LICENSE	a month ago
MANIFEST.in	a month ago
package.json	an hour ago
README.md	5 days ago
readthedocs.yml	20 days ago
setup.py	21 days ago
tslint.json	24 days ago

A simple polar plot

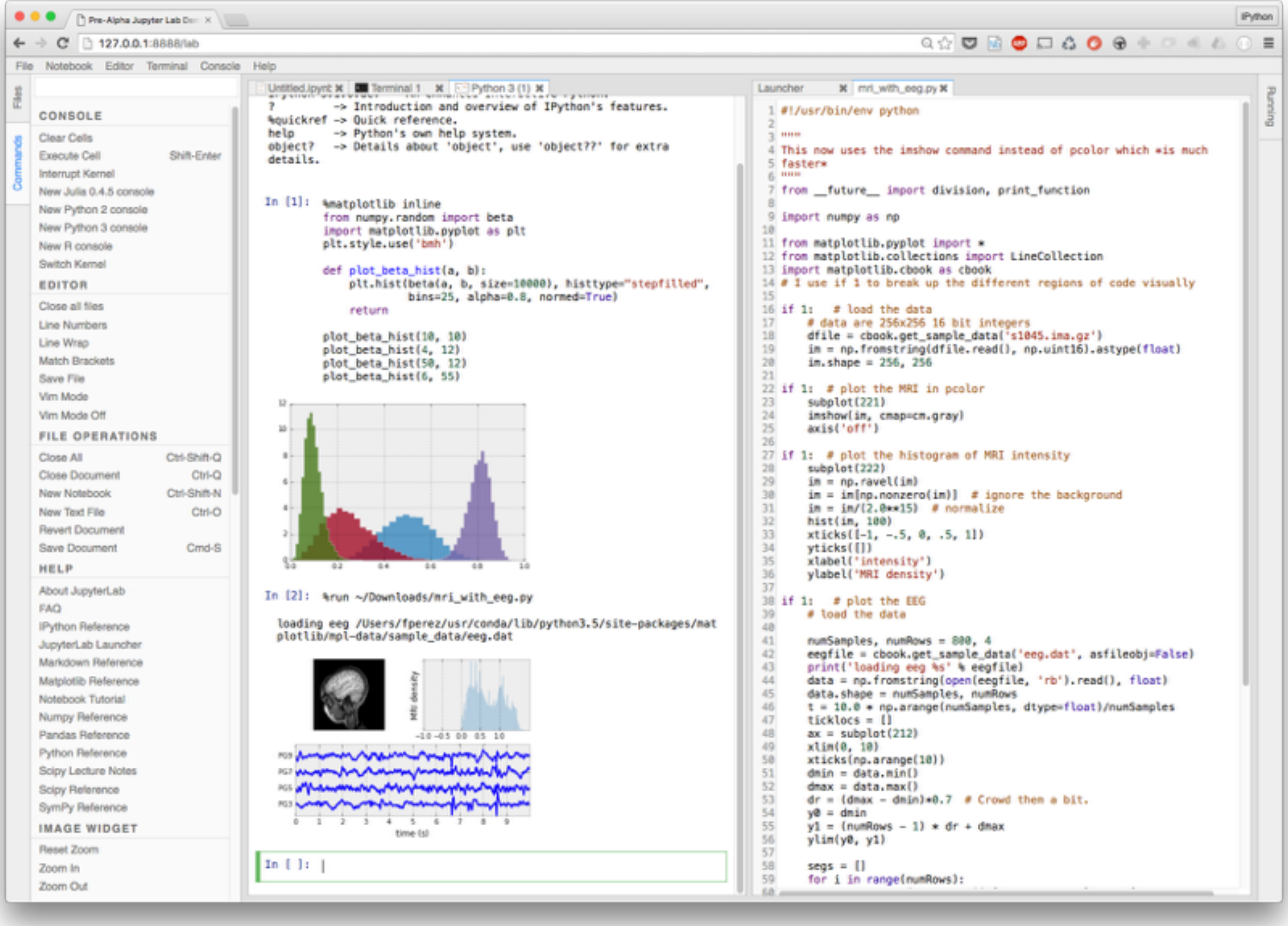
An example taken from the matplotlib gallery:

```
In [1]: %matplotlib inline
        import numpy as np
        import matplotlib.pyplot as plt

        N = 20
        theta = np.linspace(0.0, 2 * np.pi, N, endpoint=False)
        radii = 10 * np.random.rand(N)
        width = np.pi / 4 * np.random.rand(N)
        ax = plt.subplot(111, projection='polar')
        bars = ax.bar(theta, radii, width=width, bottom=0.0)
        for r, bar in zip(radii, bars):
            bar.set_facecolor(plt.cm.jet(r / 10.))
            bar.set_alpha(0.5)
```

```
Python 3.5.2 |Continuum Analytics, Inc.| (default, Jul  2 2016, 17:52:12)
Type "copyright", "credits" or "license" for more information.

IPython 5.1.0.dev -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.


In [1]: %matplotlib inline
        from numpy.random import beta
        import matplotlib.pyplot as plt
        plt.style.use('bmh')

        def plot_beta_hist(a, b):
            plt.hist(beta(a, b, size=10000), histtype="stepfilled",
                     bins=25, alpha=0.8, normed=True)
            return

        plot_beta_hist(10, 10)
        plot_beta_hist(4, 12)
        plot_beta_hist(50, 12)
        plot_beta_hist(6, 55)

In [ ]:
```

```
  1  [||||||||||||||                                   18.1%]   Tasks: 305 total, 1 running
  2  [||||                                              5.0%]   Load average: 2.29 2.07 2.09
  3  [||||||||||                                       15.6%]   Uptime: 4 days, 21:59:11
  4  [||||                                              5.0%]
  Mem[|||||||||||||||||||||||||||||||||||||||||||5987/8192MB]
  Swp[|||||||||||||||||||||||||||||||||||||||||  2487/3072MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
82374 fperez     31   0 2389M  2048     0 R  0.0  0.0  0:00.00 htop
    1 root        0   0     0     0     0    0.0  0.0  0:00.00 (launchd)
   46 root        0   0     0     0     0    0.0  0.0  0:00.00 (syslogd)
   47 root        0   0     0     0     0    0.0  0.0  0:00.00 (UserEventAgent)
F1Help  F2Setup F3SearchF4InvertF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

A completely modular architecture



```
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra
details.


In [1]:  %matplotlib inline
         from numpy.random import beta
         import matplotlib.pyplot as plt
         plt.style.use('bmh')

         def plot_beta_hist(a, b):
             plt.hist(beta(a, b, size=10000), histtype="stepfilled",
                      bins=25, alpha=0.8, normed=True)
             return

         plot_beta_hist(10, 10)
         plot_beta_hist(4, 12)
         plot_beta_hist(50, 12)
         plot_beta_hist(6, 55)

In [2]:  %run ~/Downloads/mri_with_eeg.py

  loading eeg /Users/fperez/usr/conda/lib/python3.5/site-packages/mat
  plotlib/mpl-data/sample_data/eeg.dat

In [ ]:  |
```

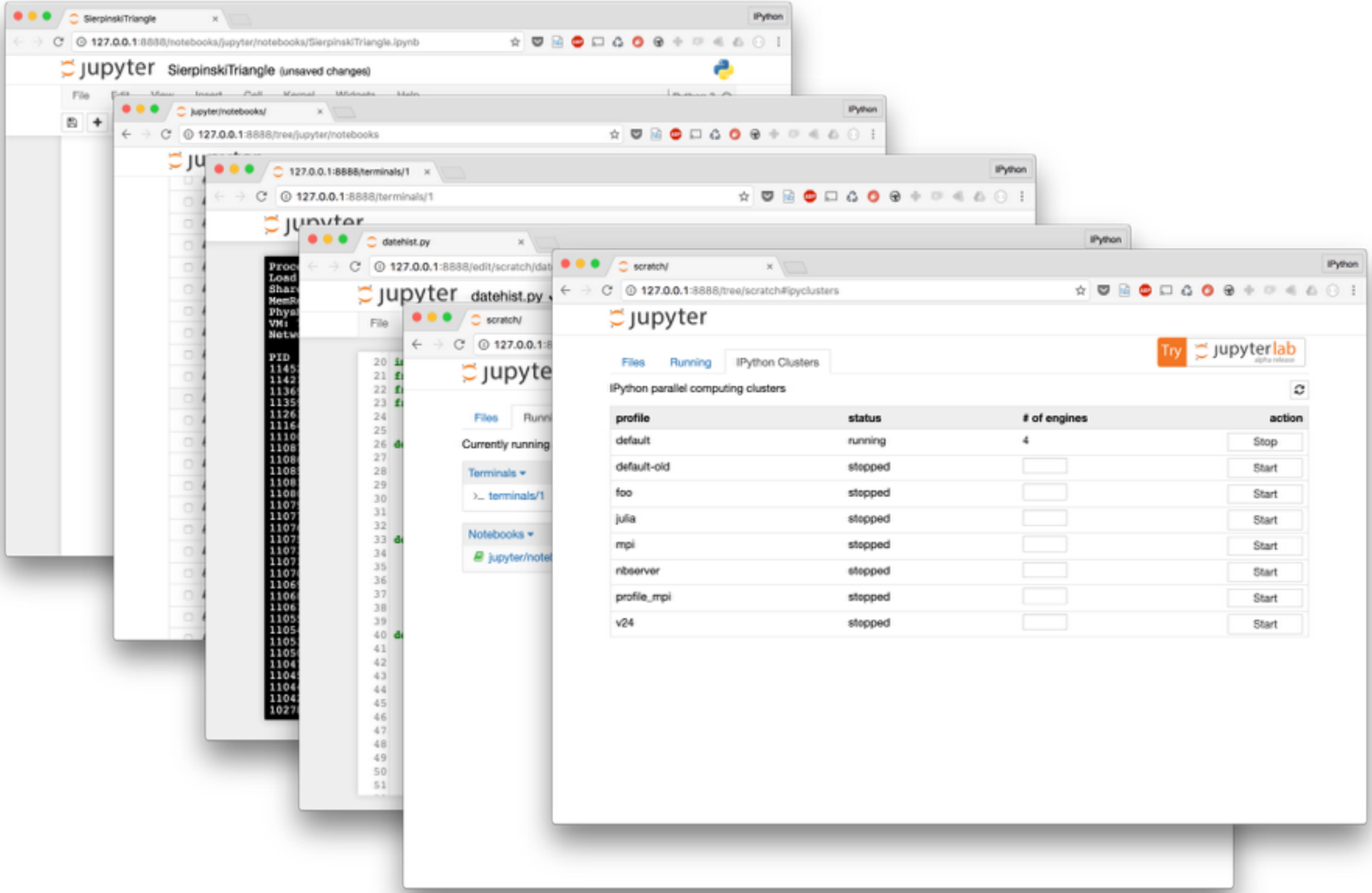
```python
#!/usr/bin/env python

"""
This now uses the imshow command instead of pcolor which *is much
faster*
"""
from __future__ import division, print_function

import numpy as np

from matplotlib.pyplot import *
from matplotlib.collections import LineCollection
import matplotlib.cbook as cbook
# I use if 1 to break up the different regions of code visually

if 1:   # load the data
    # data are 256x256 16 bit integers
    dfile = cbook.get_sample_data('s1045.ima.gz')
    im = np.fromstring(dfile.read(), np.uint16).astype(float)
    im.shape = 256, 256

if 1:  # plot the MRI in pcolor
    subplot(221)
    imshow(im, cmap=cm.gray)
    axis('off')

if 1:  # plot the histogram of MRI intensity
    subplot(222)
    im = np.ravel(im)
    im = im[np.nonzero(im)]  # ignore the background
    im = im/(2.0**15)  # normalize
    hist(im, 100)
    xticks([-1, -.5, 0, .5, 1])
    yticks([])
    xlabel('intensity')
    ylabel('MRI density')

if 1:   # plot the EEG
    # load the data

    numSamples, numRows = 800, 4
    eegfile = cbook.get_sample_data('eeg.dat', asfileobj=False)
    print('loading eeg %s' % eegfile)
    data = np.fromstring(open(eegfile, 'rb').read(), float)
    data.shape = numSamples, numRows
    t = 10.0 * np.arange(numSamples, dtype=float)/numSamples
    ticklocs = []
    ax = subplot(212)
    xlim(0, 10)
    xticks(np.arange(10))
    dmin = data.min()
    dmax = data.max()
    dr = (dmax - dmin)*0.7  # Crowd them a bit.
    y0 = dmin
    y1 = (numRows - 1) * dr + dmax
    ylim(y0, y1)

    segs = []
    for i in range(numRows):
```

JupyterLab

- JupyterLab is the natural evolution of the Jupyter Notebook user interface
- JupyterLab is an IDE: *Interactive* Development Environment
- Flexible user interface for assembling the fundamental building blocks of interactive computing
- Modernized JavaScript architecture based on npm/webpack, plugin system, model/view separation
- Built using PhosphorJS (http://phosphorjs.github.io/)
- Design-driven development process

https://github.com/jupyter/jupyterlab

The “Notebook”?



SierpinskiTriangle
127.0.0.1:8888/notebooks/jupyter/notebooks/SierpinskiTriangle.ipynb
jupyter SierpinskiTriangle (unsaved changes)
jupyter/notebooks/
127.0.0.1:8888/tree/jupyter/notebooks
127.0.0.1:8888/terminals/1
datehist.py
jupyter datehist.py
File
scratch/
Files
Currently running
Terminals
terminals/1
Notebooks
jupyter/note
127.0.0.1:8888/tree/scratch#ipyclusters
IPython
jupyter
Files
Running
IPython Clusters
Try jupyterlab alpha release

IPython parallel computing clusters

profile	status	# of engines	action
default	running	4	Stop
default-old	stopped		Start
foo	stopped		Start
julia	stopped		Start
mpi	stopped		Start
nbserver	stopped		Start
profile_mpi	stopped		Start
v24	stopped		Start

Roadmap

- Today (August 2016) JupyterLab is an early preview only
- Not suggested for general usage:
    - Visual design, UI, UX, interactions, code all still changing rapidly!
- Phases:
    - 1) Series of alpha/beta releases of JupyterLab available as an alternative UI alongside the classic notebook
    - 2) JupyterLab 1.0 = Lab notebook component has feature parity with classic notebook
    - 3) JupyterLab becomes the default UI, but classic notebook is still available
    - 4) Classic notebook only available as a separate download

Downloading JupyterLab

GitHub

jupyter/jupyterlab*

*JupyterLab is a very early developer preview, and is not suitable for general usage yet. Features and implementation are subject to change.

https://github.com/jupyter/jupyterlab

Getting started

Prerequisite

Jupyter notebook version 4.2 or later. To check your notebook version:

```
jupyter notebook —version
```

User installation

From the command line:

```
pip install jupyterlab
jupyter serverextension enable --py jupyterlab
```

(or conda install -c condaforge jupyterlab)- **no server enable step**

Start up JupyterLab

```
jupyter lab
```

JupyterLab will open automatically in your browser.

*JupyterLab is a very early developer preview, and is not suitable for general usage yet. Features and implementation are subject to change.

Live Demo

What next?

Alpha (rapid iteration, major changes, may break)

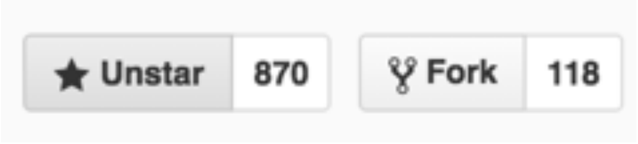
```
pip install jupyterlab
jupyter serverextension enable —sys-prefix jupyterlab
```

```
(or conda install -c condaforge jupyterlab)
```

```
jupyter lab
```

Update Since Scipy 2016

- Brian Granger and Jason Grout presented
- 67% Complete on our Milestone 0.9 goal of "A version of JupyterLab that is reasonably feature-complete and stable enough to be usable for day-to-day work"
- Introspection for Notebooks and Consoles
- Auto-save for all open documents
- Handling of relative urls for images and links in rendered outputs
- Numerous Notebook feature-parity fixes/additions
- 870 Stars on Github (please help increase that number!)



- Working on a scalable method for third party extensions using a cookiecutter template (coming soon!)

How to Contribute

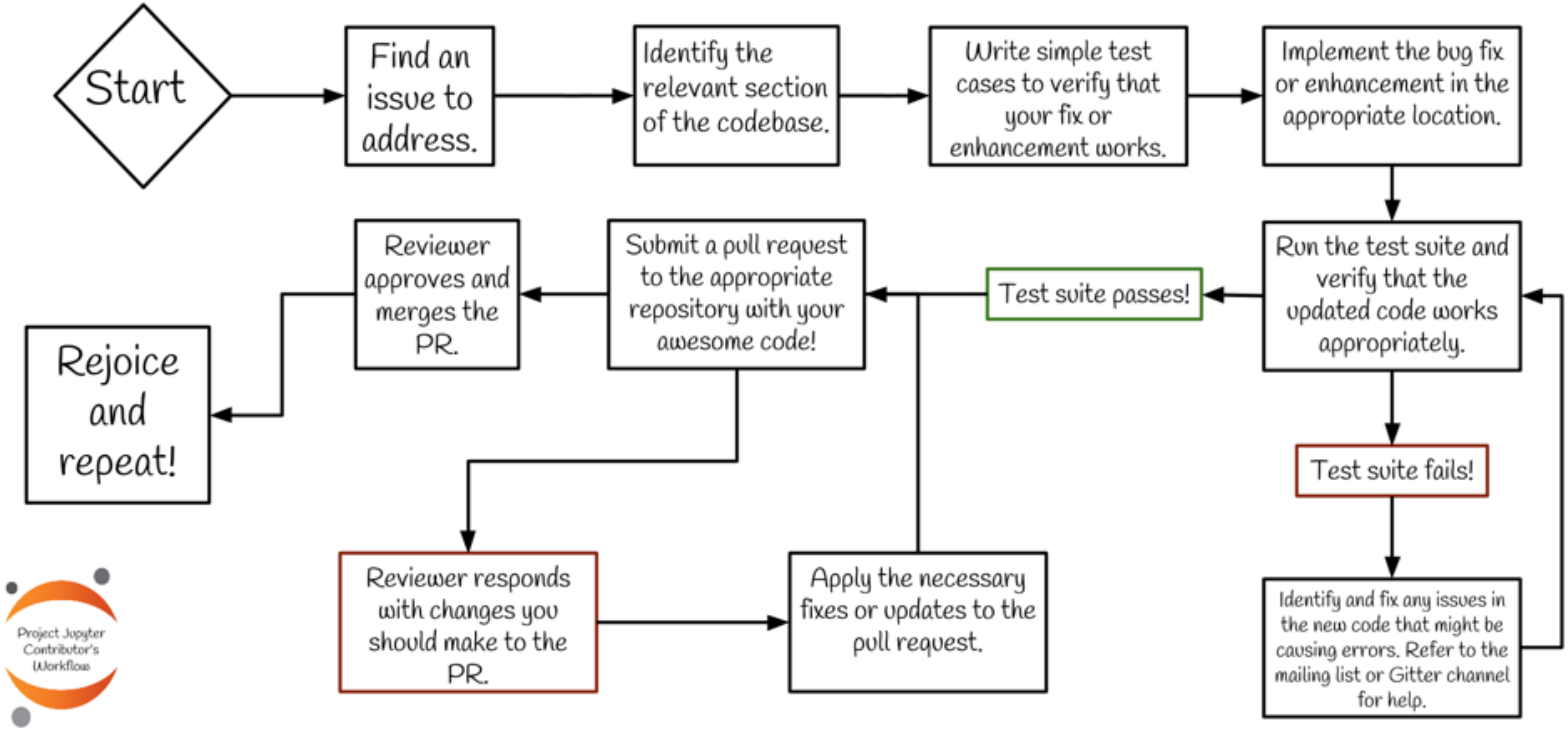
- Regular JupyterLab progress meetings on Fridays
- Follow repo on Github

https://github.com/jupyter/jupyterlab

- Step-by-step instructions on how to contribute in our documentation!

http://jupyter.readthedocs.io

Contribution Workflow



Start

Find an issue to address.

Identify the relevant section of the codebase.

Write simple test cases to verify that your fix or enhancement works.

Implement the bug fix or enhancement in the appropriate location.

Run the test suite and verify that the updated code works appropriately.

Test suite fails!

Identify and fix any issues in the new code that might be causing errors. Refer to the mailing list or Gitter channel for help.

Test suite passes!

Submit a pull request to the appropriate repository with your awesome code!

Reviewer responds with changes you should make to the PR.

Apply the necessary fixes or updates to the pull request.

Reviewer approves and merges the PR.

Rejoice and repeat!

Project Jupyter Contributor's Workflow

Thanks!

- Bloomberg, Continuum Analytics, Jupyter Team, especially Jason Grout, Brian Granger, Chris Colbert, Steve Silvester, Afshin Darian, and Dave Willmer

- Moore, Sloan, and Helmsley Foundations

Thank You!