JupyterLab: The Evolution of the Jupyter Notebook

The JupyterLab Team

Chris Colbert, Jupyter
Steven Silvester, Quansight
Afshin Darian, Jupyter
Ian Rose, Berkeley
Jason Grout, Bloomberg
Brian Granger, Cal Poly
Jessica Forde, Jupyter
Grant Nestor, Cal Poly
Cameron Oelsen, Cal Poly
Fernando Perez, LBNL/Berkeley
Cal Poly Interns
The Larger Jupyter Team

@jupyterlab on GitHub
@ProjectJupyter on Twitter

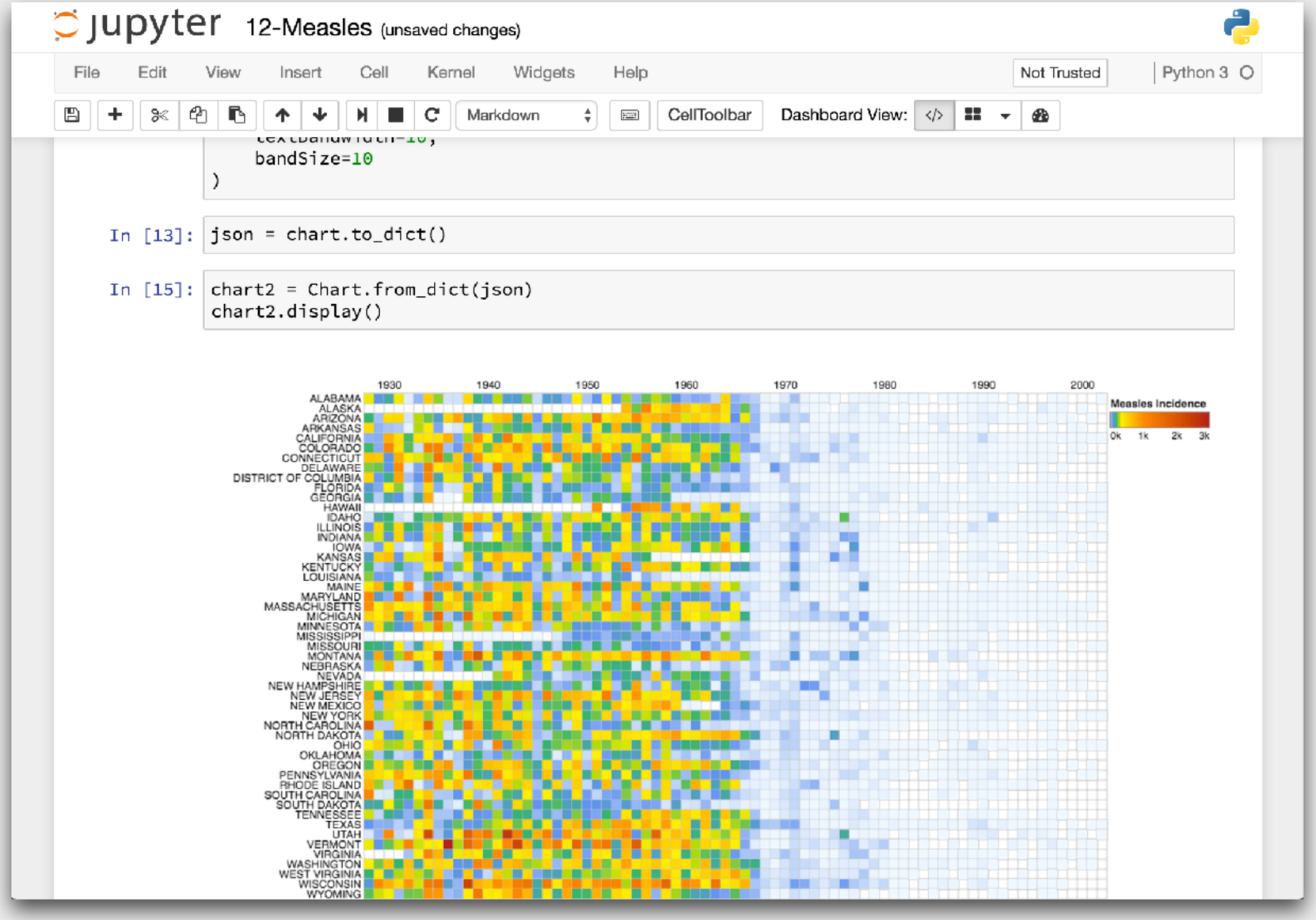
The Jupyter Notebook

Jupyter Notebook

Interactive, Exploratory, Reproducible

- Interactive, browser-based computing environment
- Exploratory data science, ML, visualization, analysis, stats
- Reproducible document format:
  - Code
  - Narrative text (markdown)
  - Equations (LaTeX)
  - Images, visualizations
- Over 50 programming languages
- Everything open-source (BSD license)

Jupyter Notebook



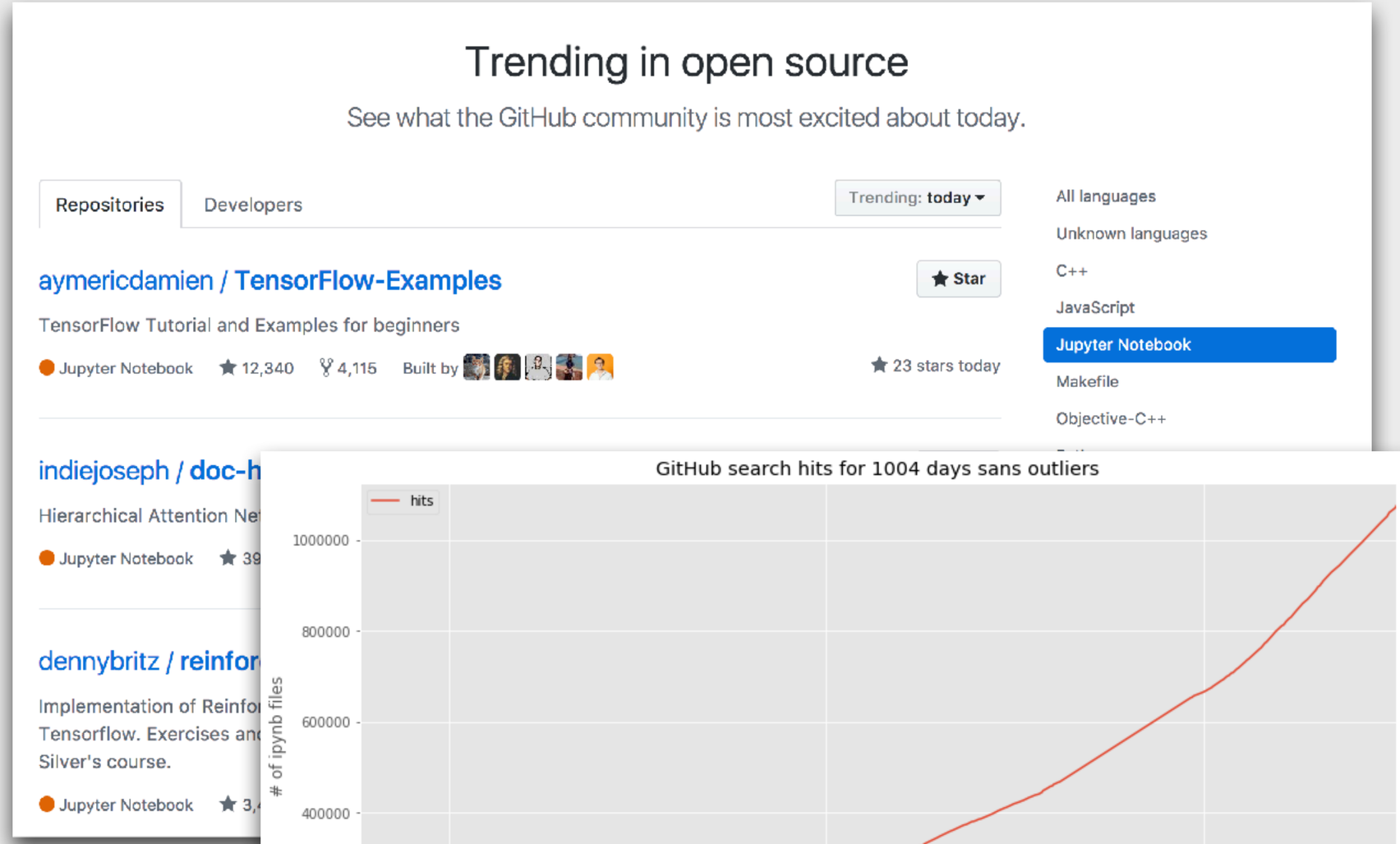
A Jupyter Notebook document with a visualization of measles data.

Project Jupyter: Where are we today?

Millions of Users

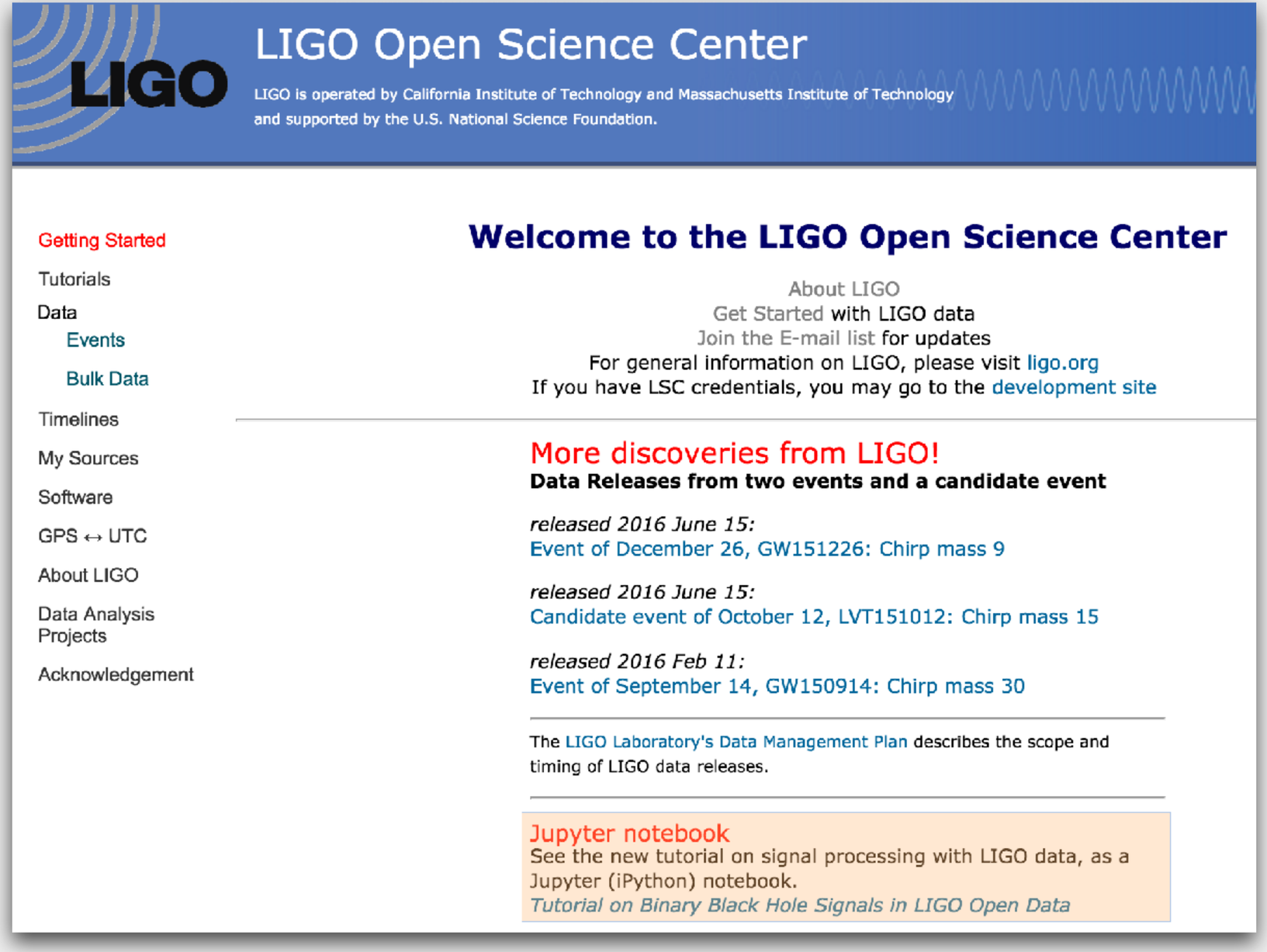
https://github.com/jupyter/design/blob/master/surveys/2015-notebook-ux/analysis/report_dashboard.ipynb

Millions of Notebooks



https://github.com/trending/jupyter-notebook

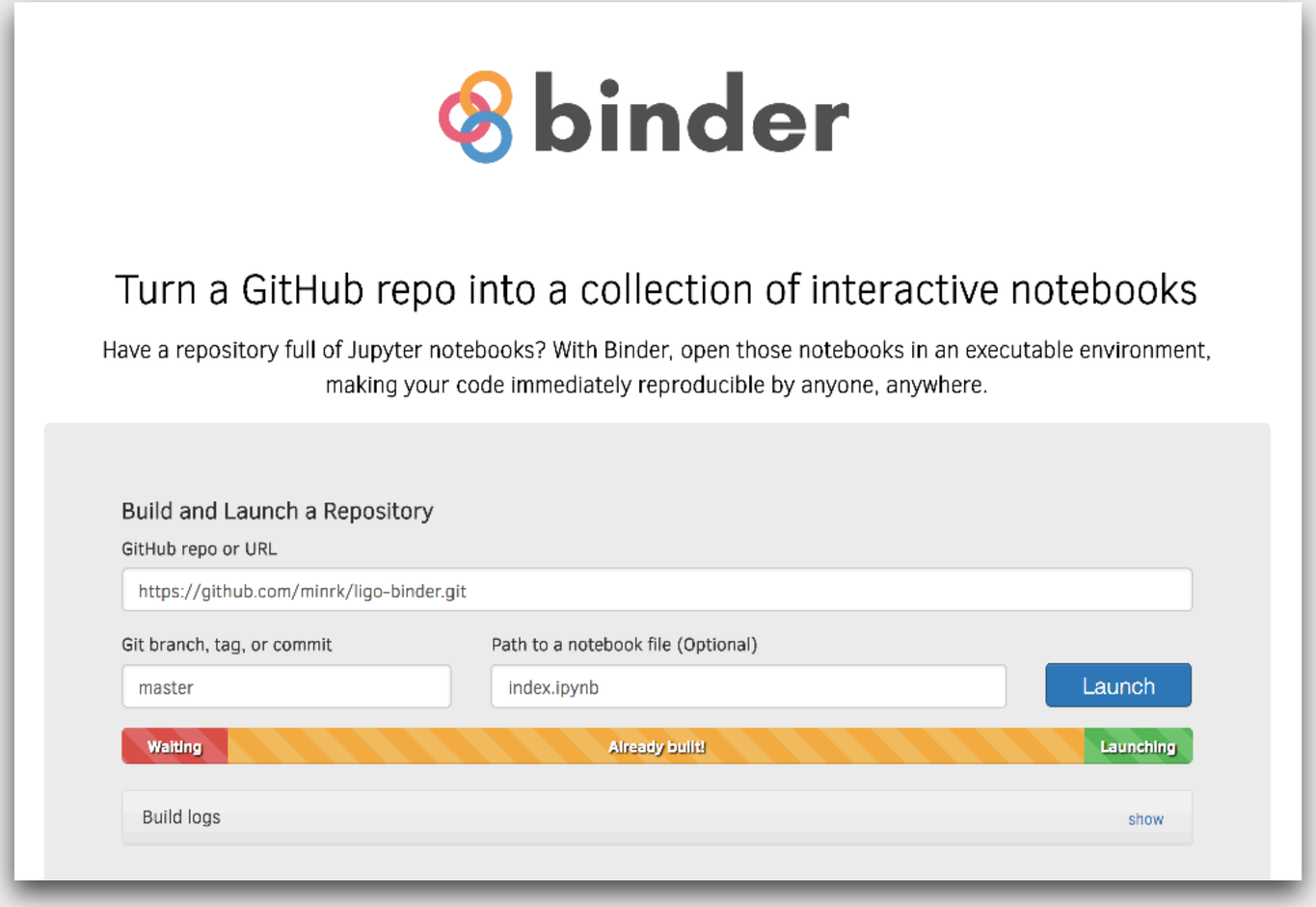
Enabling Reproducible Science



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

Live Code on Binder

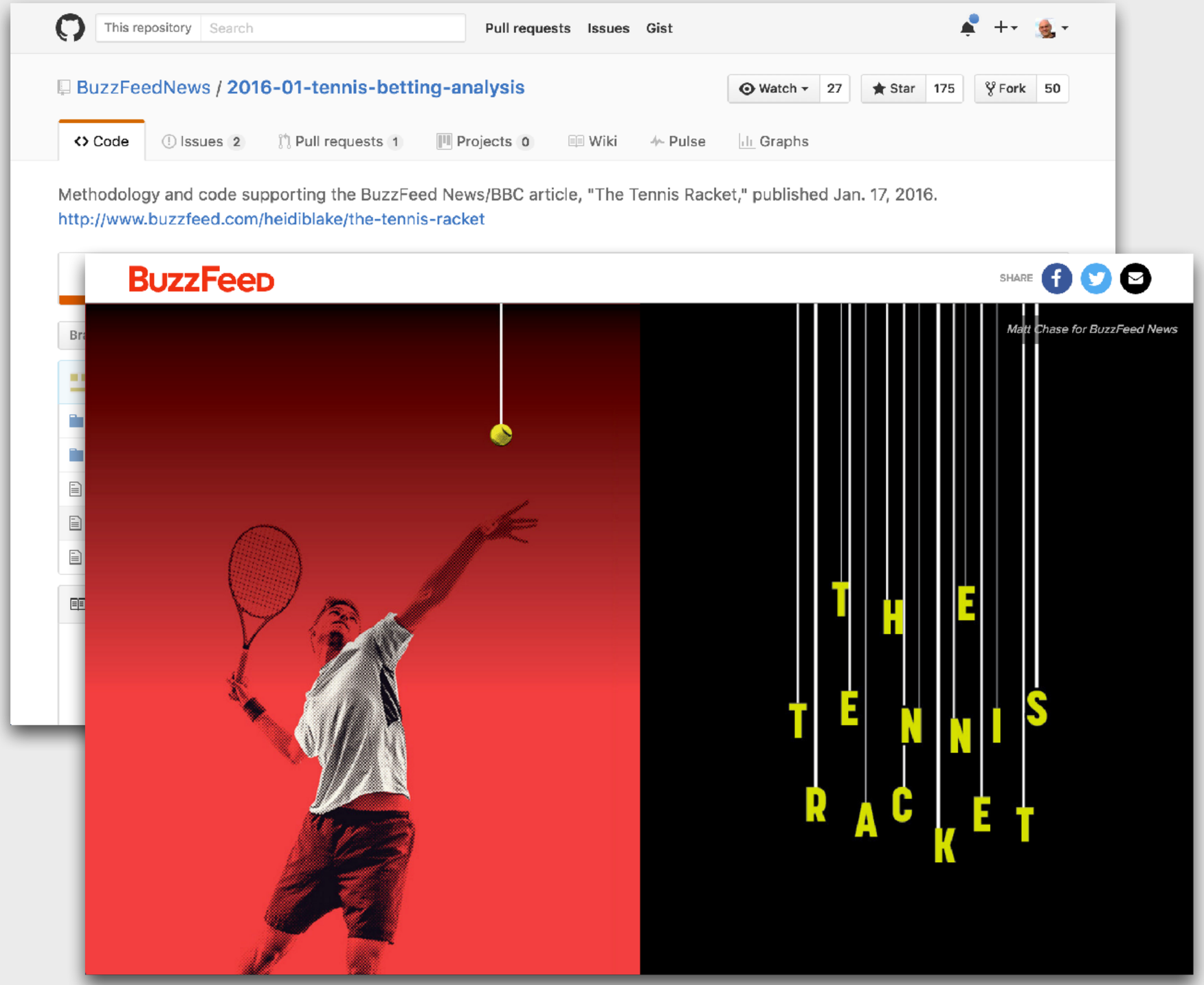
LIGO Binder



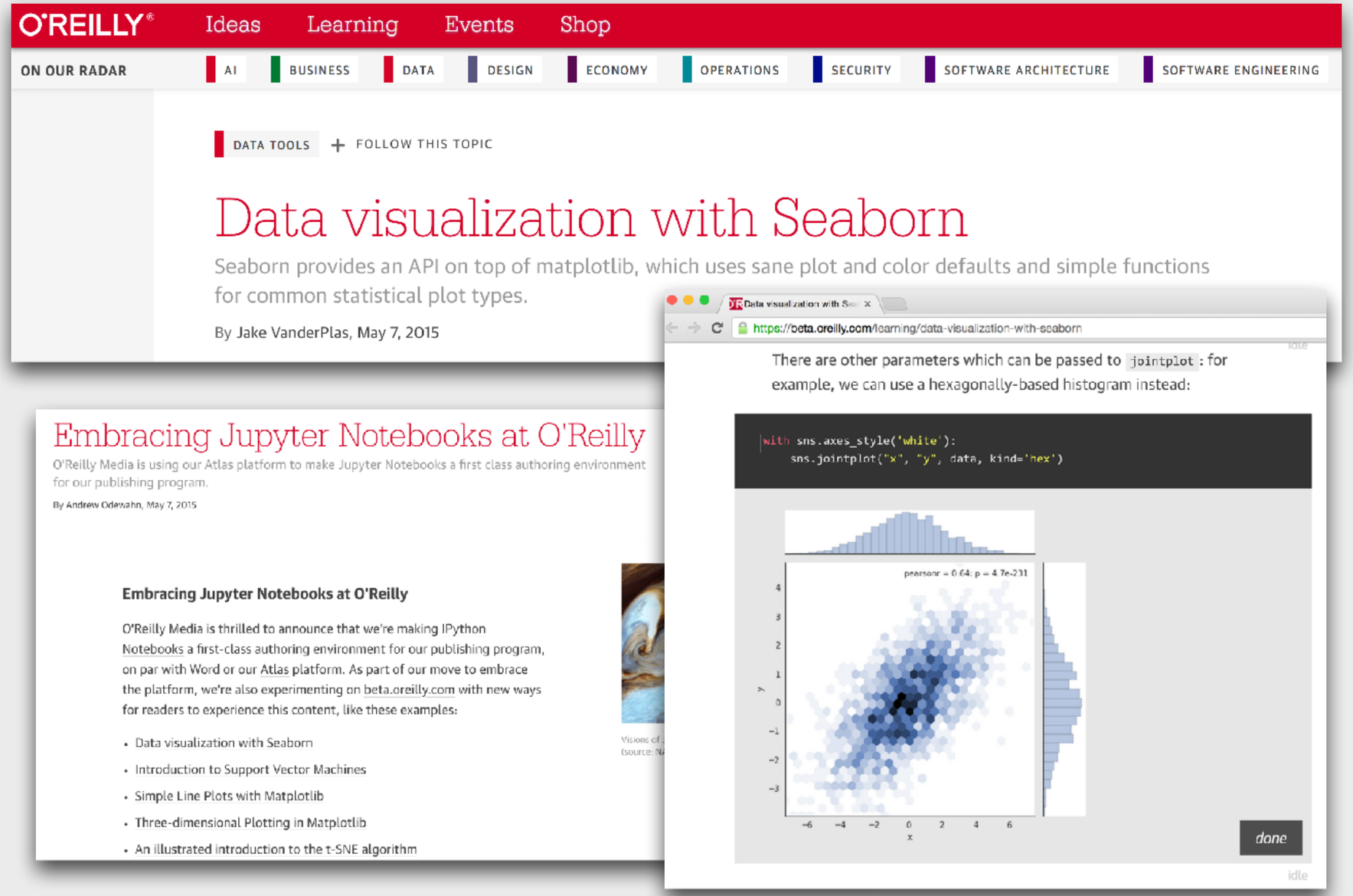
https://losc.ligo.org/tutorials/

https://beta.mybinder.org/v2/gh/minrk/ligo-binder/master?filepath=index.ipynb

Enabling Open Data Journalism



Authoring Interactive Books

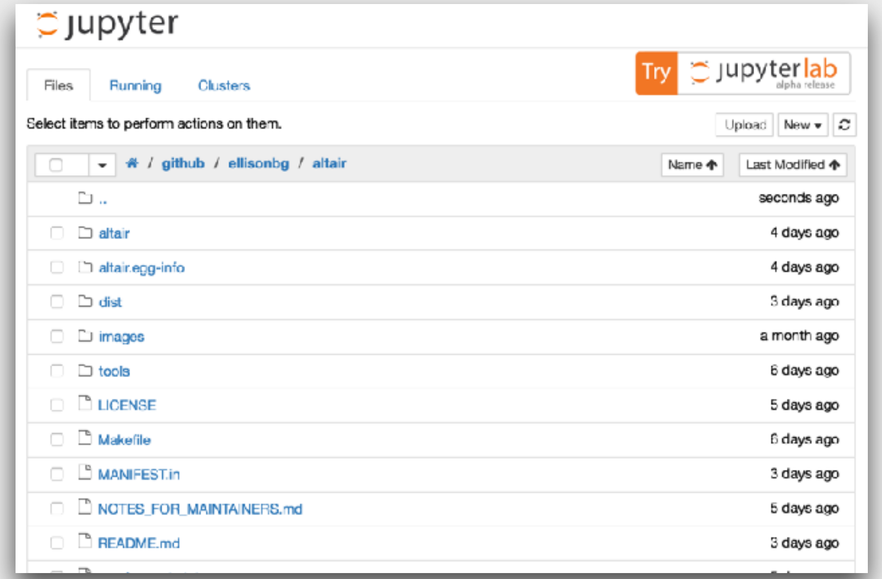


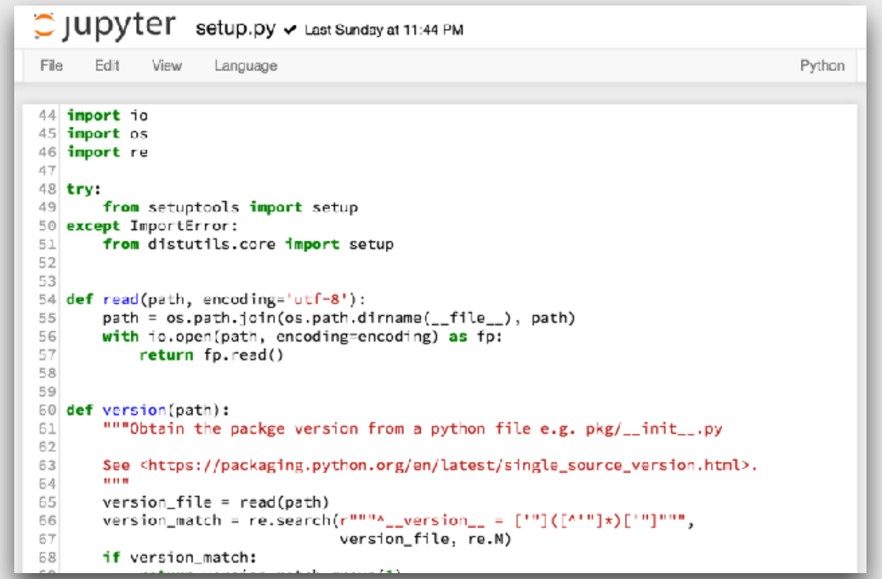
O'Reilly Atlas authoring platform incorporating live code

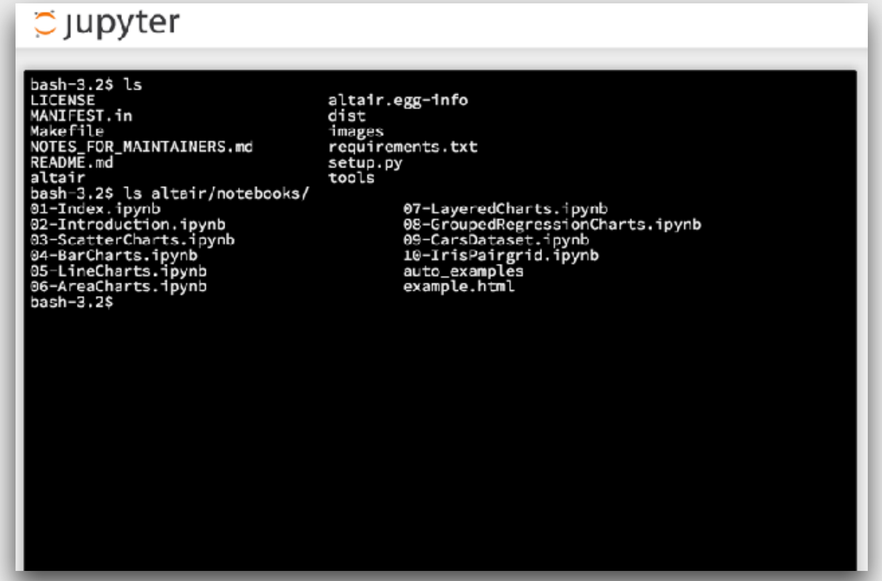
https://www.oreilly.com/ideas/jupyter-at-oreilly

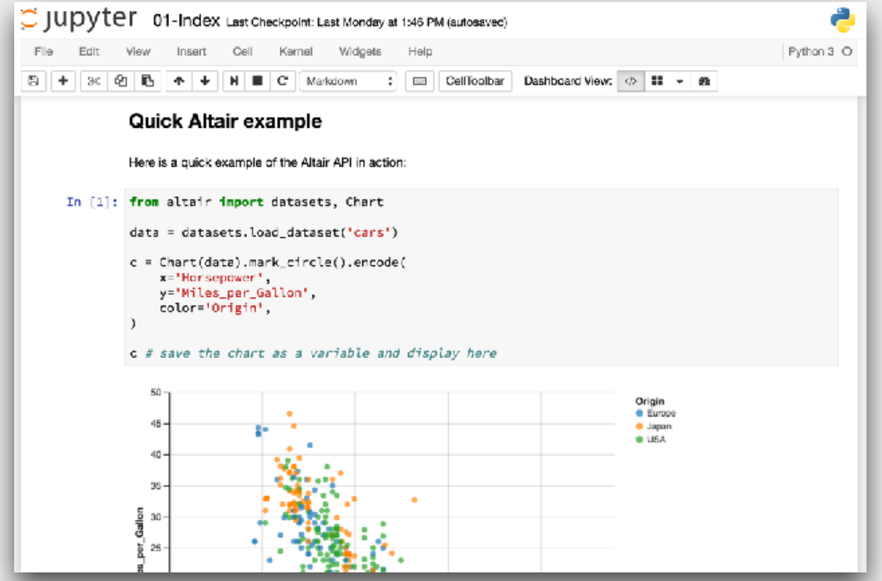
Building Blocks for Interactive Computing

Classic Jupyter: More Than Just Notebooks









Building Blocks

File Browser

Notebooks

Terminal

Text Editor

Kernels

Output

Introducing JupyterLab

JupyterLab: Integrated Experience

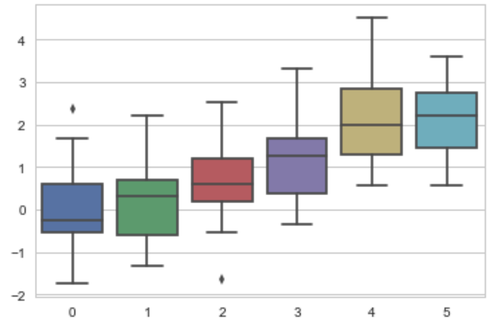
File Notebook Editor Terminal Console Help

Files Running Commands Cell Tools Tabs

Launcher | aesthetics.ipyn | categorical.py

Code jlabdemo

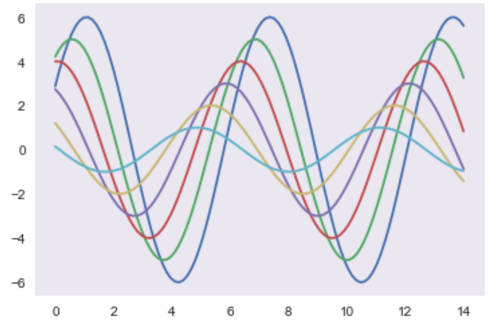
In [6]:
```
sns.set_style("whitegrid")
data = np.random.normal(size=(20, 6)) + np.arange(6) / 2
sns.boxplot(data=data);
```



For many plots, (especially for settings like talks, where you primarily want to use figures to provide impressions of patterns in the data), the grid is less necessary.

In [7]:
```
sns.set_style("dark")
```

In [8]:
```
sinplot()
```



In [9]:
```
sns.set_style("white")
```

categorical.py

```
class _BoxPlotter(_CategoricalPlotter):

    def __init__(self, x, y, hue, data, order, hue_order,
                 orient, color, palette, saturation,
                 width, dodge, fliersize, linewidth):

        self.establish_variables(x, y, hue, data, orient, order, hue_order)
        self.establish_colors(color, palette, saturation)

        self.dodge = dodge
        self.width = width
        self.fliersize = fliersize

        if linewidth is None:
            linewidth = mpl.rcParams["lines.linewidth"]
        self.linewidth = linewidth

    def draw_boxplot(self, ax, kws):
        """Use matplotlib to draw a boxplot on an Axes."""
        vert = self.orient == "v"

        props = {}
        for obj in ["box", "whisker", "cap", "median", "flier"]:
            props[obj] = kws.pop(obj + "props", {})

        for i, group_data in enumerate(self.plot_data):

            if self.plot_hues is None:

                # Handle case where there is data at this level
                if group_data.size == 0:
                    continue

                # Draw a single box or a set of boxes
                # with a single level of grouping
                box_data = remove_na(group_data)

                # Handle case where there is no non-null data
                if box_data.size == 0:
                    continue

                artist_dict = ax.boxplot(box_data,
                                         vert=vert,
                                         patch_artist=True,
                                         positions=[i],
                                         widths=self.width,
                                         **kws)
```

JupyterLab

Building Blocks

- Work with the building blocks in a flexible and integrated manner
- Modern JavaScript development: npm-based packaging,

  Typescript, phosphor.js
- Clean model/view separation
- Well separated public/private APIs
- Fully extensible by third parties
- High performance
- Design!

January
2018

JupyterLab Today

- https://github.com/jupyterlab
- ~2.5 years worth of development
- ~100 contributors, ~60 components
- ~1,800 releases (npm+python)
- Over 11,000 commits, ~classic notebook
- Currently Beta

Roadmap

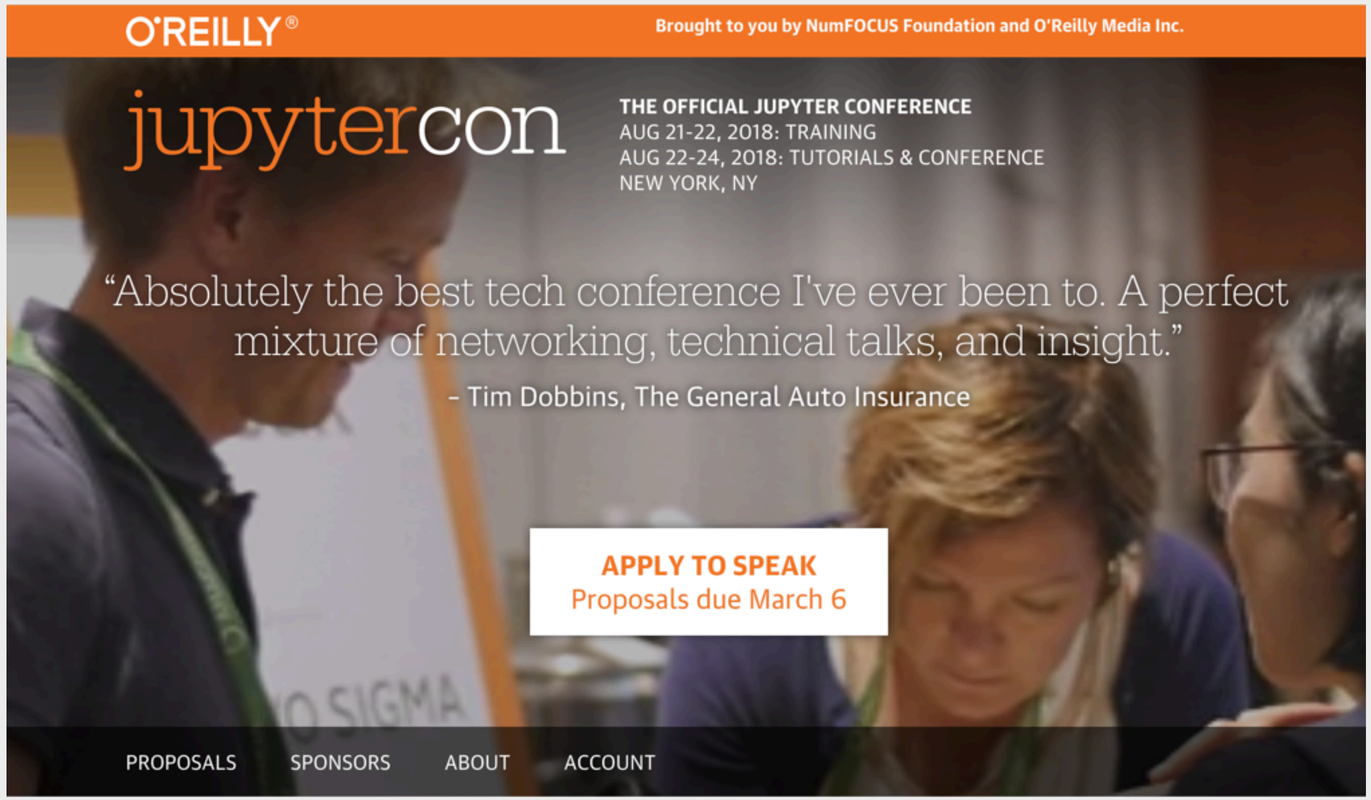
JupyterLab Beta: Use It Today

conda install -c conda-forge jupyterlab
or pip install jupyterlab

- Beta releases, January 2018
  - For all users
  - For adventurous extension developers
- 1.0 this year
  - For all users, extension developers
- Eventually:
  - Classic notebook will be retired

JupyterCon

JupyterCon, Aug 21-25, New York

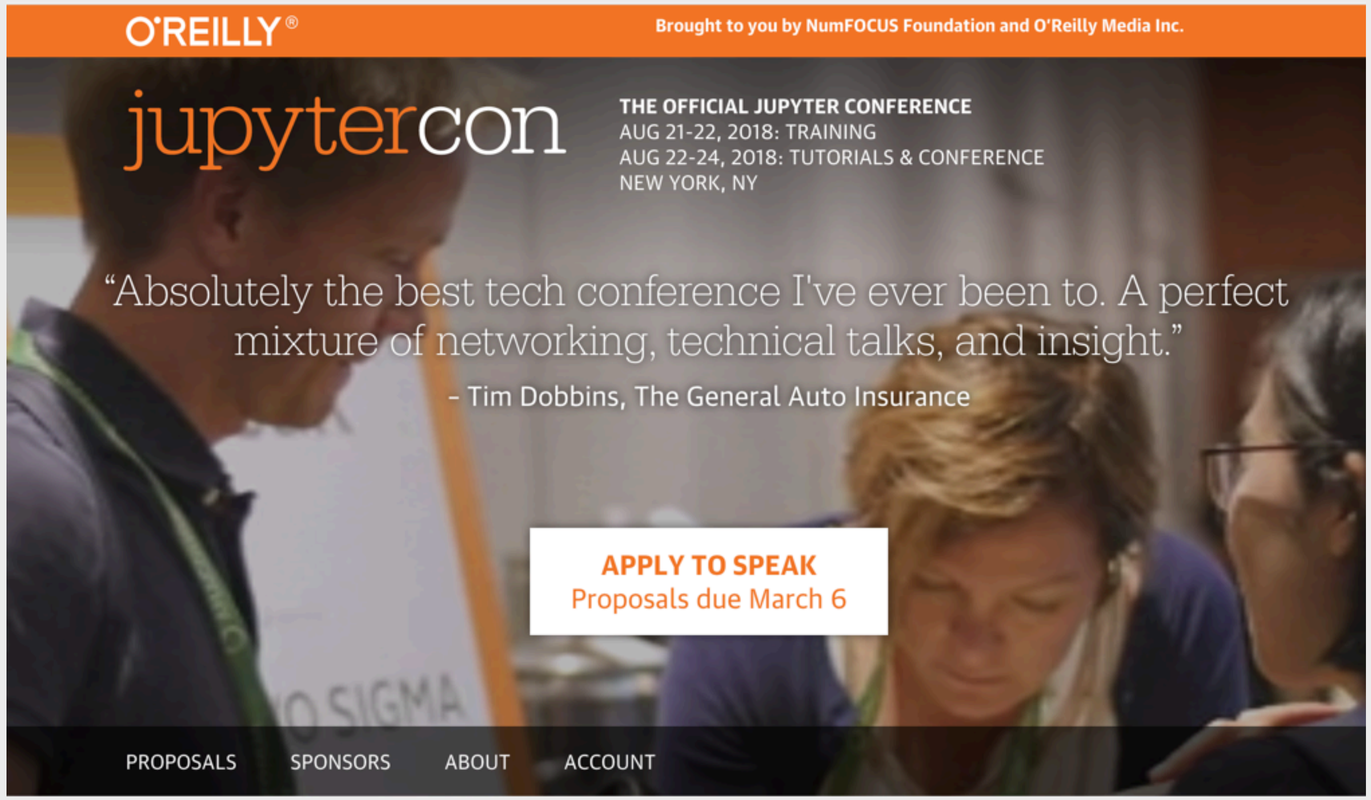


Discover how the most data-driven organizations are using Jupyter to analyze data, share insights, and create dynamic, reproducible data science.

Live Demos!

Thank You!

JupyterCon, Aug 21-25, New York



Discover how the most data-driven organizations are using Jupyter to analyze data, share insights, and create dynamic, reproducible data science.

conda install -c conda-forge jupyterlab
or pip install jupyterlab