Jupyter notebook Tutorial

Author: Yuheng Qiu

How to set up jupyter notebook as slides? Check https://medium.com/@mjspeck/presenting-code-using-jupyter-notebook-slides-a8a3c3b59d67

In [40]:

```
__author__ = "Yuheng Qiu"
```

In this titorial, I will go through following part:

- Benifits and usage of jupyter
- visualization of jupyter
- basic mechanism of jupyter
- tools and utils

I will introduce some concepts and ideas rather than basic usage. Jupyter_logo.svg.png

Basic Mechanism

reference: https://jupyter.readthedocs.io/en/latest/architecture/how_jupyter_ipython_work.html#the-ipython-kernel

IPython kernel

The IPython Kernel is a separate process which is responsible for running user code, and things like computing possible completions.

- Frontends, like the notebook or the Qt console, communicate with the IPython Kernel using JSON messages sent over ZeroMQ sockets;
- the protocol used between the frontends and the IPython Kernel is described in Messaging in Jupyter.

In [43]:

```
__author__
```

Out[43]:

```
'Yuheng Qiu'
```

This design was intended to allow easy development of different frontends based on the same kernel, but it also made it possible to support new languages in the same frontends, by developing kernels in those languages, and we are refining IPython to make that more practical.

- Using HTML through `IPython`

In [38]:

```
from IPython.core.display import HTML
display(HTML("<a href='http://www.google.com' target='_blank'>www.google.com</a>"))
```

www.google.com

Notebook

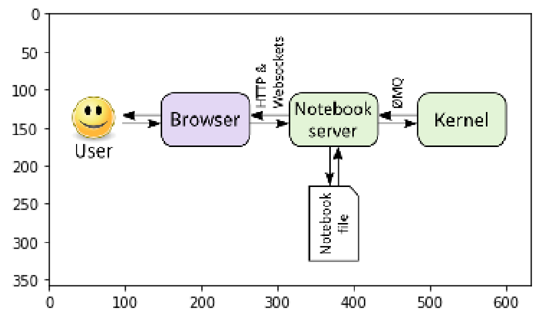
In addition to running your code, it stores code and output, together with markdown notes, in an editable document called a notebook. When you save it, this is sent from your browser to the notebook server, which saves it on disk as a JSON file with a `.ipynb` extension.

In [47]:

```
import matplotlib.pyplot as plt
img = plt.imread("./doc/notebook_components.png")
plt.imshow(img)
```

Out[47]:

```
<matplotlib.image.AxesImage at 0x11e1dfdd8>
```



List of hotkeys is shown in Help > Keyboard Shortcuts (list is extended from time to time, so don't hesitate to look at it again).

Sharing notebooks

Simplest way is to share notebook file (.ipynb), but not everyone is using notebooks, so the option is

- convert notebooks to html file

Benifits and features

1. checkpoint
2. easier to host readble data
3. easily host on any server

Easier to host readble data

- Text can be added to jupyter using `Markdown` cell. What you added to this `Markdown` cell includes code, HTML, formula and image.

```python
def f(x):
    """a docstring"""
    return x**2
```

With markdown cell, code you write is always well documented.

- Also inserting formular with Latex:

$$D(x) = \begin{cases} 1, & \text{if } x \in \mathbb{Q}; \\ 0, & \text{if } x \in \mathbb{R} \setminus \mathbb{Q}. \end{cases}$$

How to use `Markdown`? You pay a visit to this guild: https://www.markdownguide.org/basic-syntax/

easily host on any server

you can access to any jupyter notebook server.

For example, your can access LAN(local area network) server access:

10.26.2.4:8888

How to run and setup a notebook server, pay a visit to: https://jupyter-notebook.readthedocs.io/en/stable/public_server.html

In [19]:

```
import pandas as pd
import numpy as np
import scipy as sp
import plotly.plotly as py
import plotly
```

Install package with jupyter notebook

If you're in the jupyter notebook and you want to install a package with conda, you might be tempted to use the `!` notation to run conda directly as a shell command from the notebook:

```
import sys
!conda install --yes --prefix {sys.prefix} numpy
```

If you're using the Jupyter notebook and want to install a package with pip, you similarly might be inclined to run pip directly in the shell:

```
!pip install numpy
```

In [2]:

```
!pip install numpy
```

```
Collecting numpy
  Retrying (Retry(total=4, connect=None, read=None, redirect=None, status=None)) after conn
ection broken by
'ConnectTimeoutError(<pip._vendor.urllib3.connection.VerifiedHTTPSConnection object at 0x10
e001ef0>, 'Connection to files.pythonhosted.org timed out. (connect timeout=15)')': /packag
es/a6/6f/cb20ccd8f0f8581e0e090775c0e3c3e335b037818416e6fa945d924397d2/numpy-1.16.2-cp37-cp3
7m-
macosx_10_6_intel.macosx_10_9_intel.macosx_10_9_x86_64.macosx_10_10_intel.macosx_10_10_x86_6
.whl
  Downloading
https://files.pythonhosted.org/packages/a6/6f/cb20ccd8f0f8581e0e090775c0e3c3e335b037818416e6
a945d924397d2/numpy-1.16.2-cp37-cp37m-
macosx_10_6_intel.macosx_10_9_intel.macosx_10_9_x86_64.macosx_10_10_intel.macosx_10_10_x86_6
```

```
macosx_10_6_intel.macosx_10_9_intel.macosx_10_9_x86_64.macosx_10_10_intel.macosx_10_10_x86_(
.whl (13.9MB)
    100% |████████████████████████████████| 13.9MB 364kB/s ta 0:00:01    59% |█████████████
██████              | 8.3MB 423kB/s eta 0:00:14
Installing collected packages: numpy
Successfully installed numpy-1.16.2
```

Visualization

In this part I will introduce `matplolib` and `ploty`.

Matplotlib

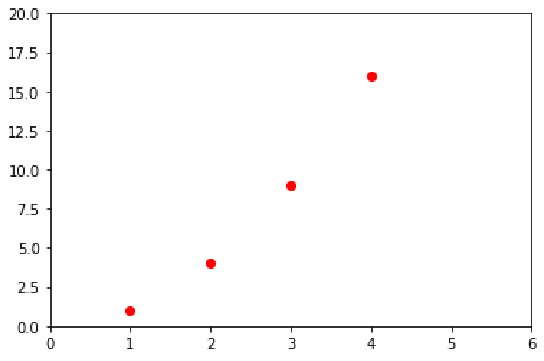
matplotlib.pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc

you can pay a visit to https://matplotlib.org/tutorials/index.html

In [50]:

In [50]:

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,4,9,16], 'ro')
plt.axis([0, 6, 0, 20])
plt.show()
```

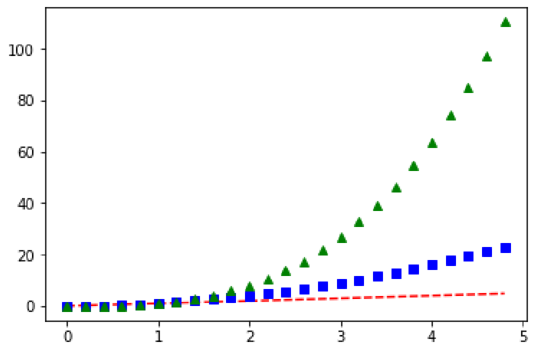


In [51]:

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

# evenly sampled time at 200ms intervals
t = np.arange(0., 5., 0.2)

# red dashes, blue squares and green triangles
plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
plt.show()
```



Ploty

`ploty`'s example is taken from ploty https://plot.ly/python/ipython-notebook-tutorial/ If you want to use ploty, you can go to https://plot.ly/settings/api sign up you own api

- Remenber to keep network connected.

In [23]:

```
plotly.tools.set_credentials_file(username='haleqiu', api_key='hThbTuv0srbERZ2g3Ogu')
```

In [24]:

```
import plotly.plotly as py
import plotly.figure_factory as ff
import pandas as pd

df = pd.read_csv("./doc/school_earnings.csv")

table = ff.create_table(df)
py.iplot(table, filename='jupyter-table1')
```

```
High five! You successfully sent some data to your account on plotly. View your plot in you
r browser at https://plot.ly/~haleqiu/0 or inside your plot.ly account where it is named 'j
upyter-table1'
```

```
/Users/qiuyuheng/Desktop/py3/lib/python3.7/site-packages/IPython/core/display.py:689: UserW
arning:

Consider using IPython.display.IFrame instead
```

Out[24]:

In [29]:

```
data = [plotly.graph_objs.Bar(x=df.School,y=df.Gap)]
py.iplot(data, filename='jupyter-basic_bar')
```

Out[29]:

In [48]:

```
import plotly.plotly as py
import plotly.graph_objs as go

import numpy as np

s = np.linspace(0, 2 * np.pi, 240)
t = np.linspace(0, np.pi, 240)
tGrid, sGrid = np.meshgrid(s, t)

r = 2 + np.sin(7 * sGrid + 5 * tGrid)  # r = 2 + sin(7s+5t)
x = r * np.cos(sGrid) * np.sin(tGrid)  # x = r*cos(s)*sin(t)
y = r * np.sin(sGrid) * np.sin(tGrid)  # y = r*sin(s)*sin(t)
z = r * np.cos(tGrid)                  # z = r*cos(t)

surface = go.Surface(x=x, y=y, z=z)
data = [surface]

layout = go.Layout(
    title='Parametric Plot',
    scene=dict(
        xaxis=dict(
            gridcolor='rgb(255, 255, 255)',
            zerolinecolor='rgb(255, 255, 255)',
            showbackground=True,
            backgroundcolor='rgb(230, 230,230)'
        ),
        yaxis=dict(
            gridcolor='rgb(255, 255, 255)',
            zerolinecolor='rgb(255, 255, 255)',
            showbackground=True,
            backgroundcolor='rgb(230, 230,230)'
        ),
        zaxis=dict(
            gridcolor='rgb(255, 255, 255)',
            zerolinecolor='rgb(255, 255, 255)',
            showbackground=True,
            backgroundcolor='rgb(230, 230,230)'
        )
    )
)

fig = go.Figure(data=data, layout=layout)
```

In [49]:

```
py.iplot(fig, filename='jupyter-parametric_plot')
```

```
/Users/qiuyuheng/Desktop/py3/lib/python3.7/site-packages/IPython/core/display.py:689: UserW
arning:

Consider using IPython.display.IFrame instead
```

Out[49]:

In [33]:

```
from IPython.display import YouTubeVideo
YouTubeVideo("wupToqz1e2g")
```

Out[33]:

Some extra tips

- Jupyter don't support `argv`
- import will not be auto load

In [12]:

```
%load_ext autoreload
%autoreload 2
```

In [ ]:

```
%run
```