PYCON CHINA 2018

使用Jupyter赋能☁服务

演讲人: 丁来强

使用Jupyter赋能☁服务

PYCON CHINA 2018

- 关于Jupyter
- 扩展Jupyter
- 云计算与Jupyter
- 演示

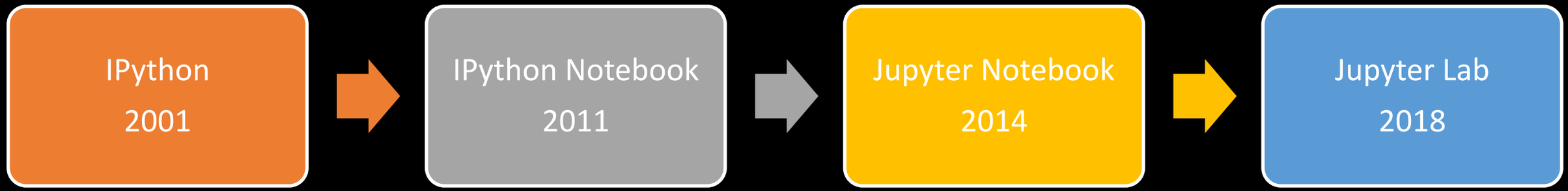
1

关于Jupyter

介绍Jupyter历史、现状、未来趋势等

Jupyter的前世今生

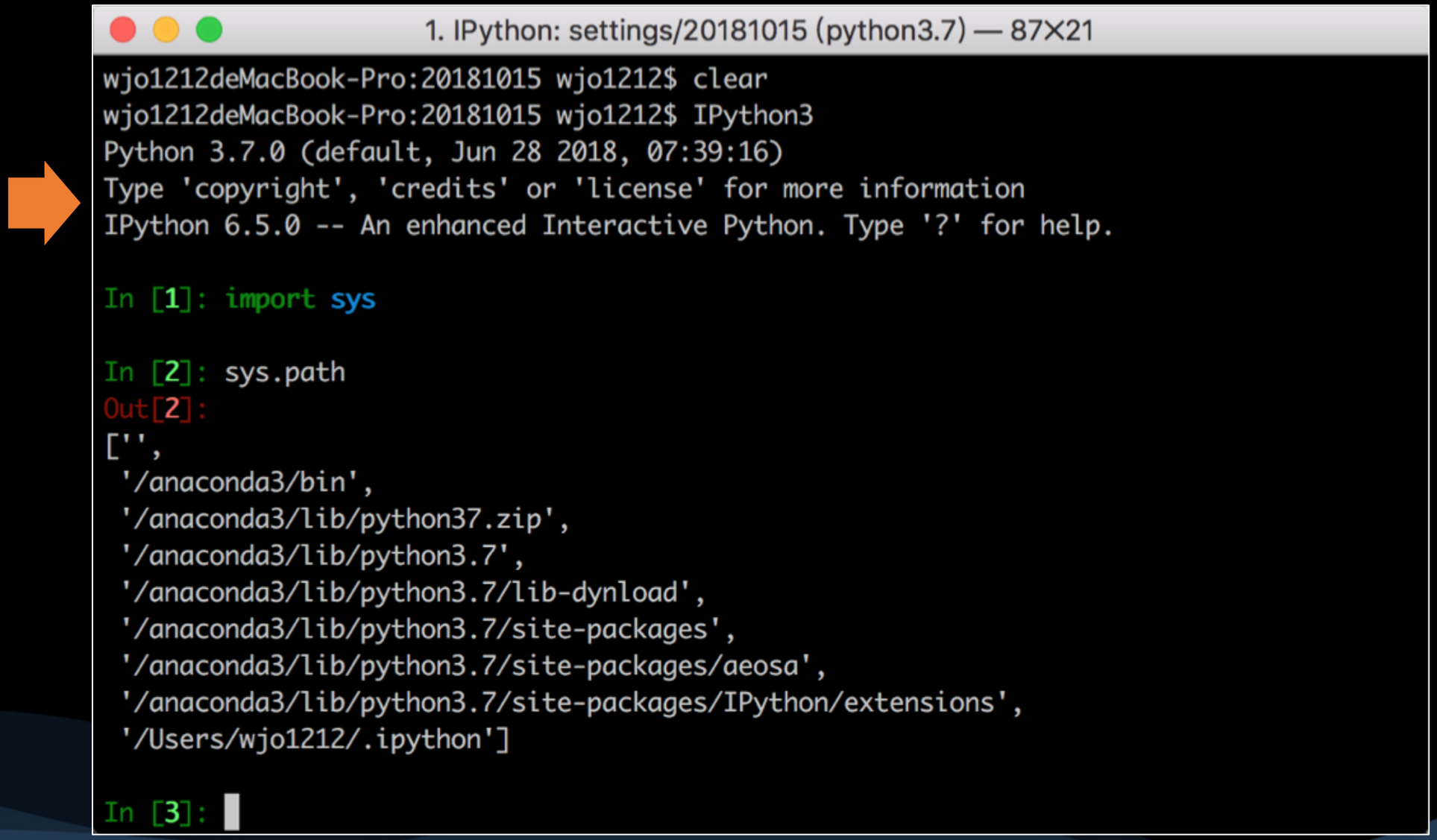
PYCON CHINA 2018



Jupyter的前世今生

PYCON CHINA 2018

IPython
2001

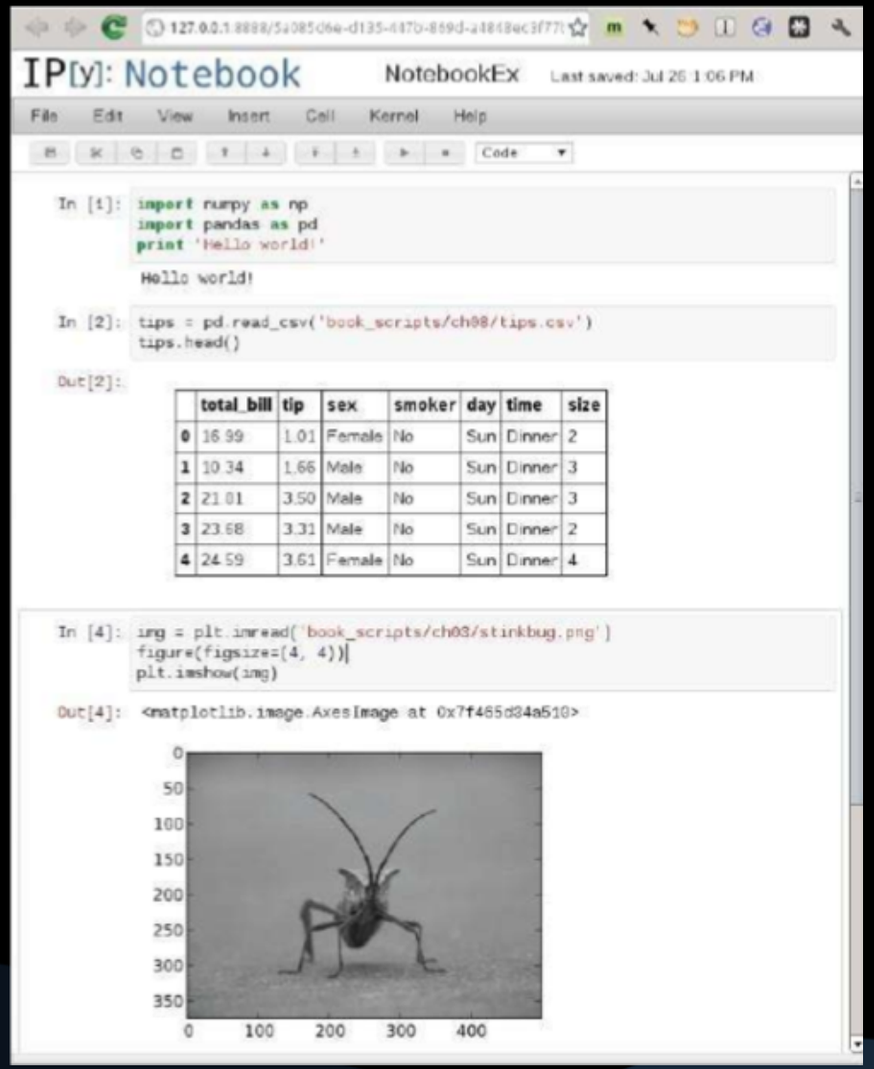


Jupyter的前世今生

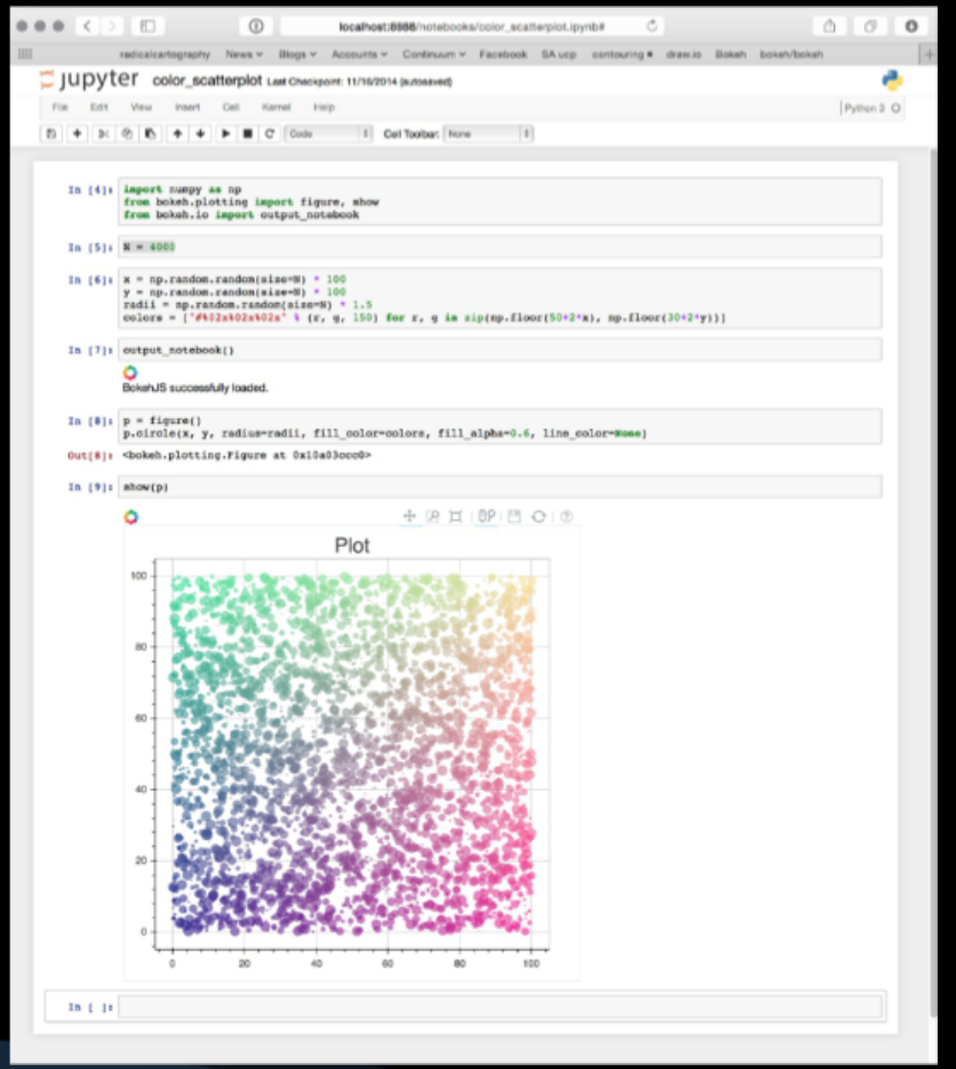
PYCON CHINA 2018

IPython Notebook
2011

Jupyter Notebook
2014



	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

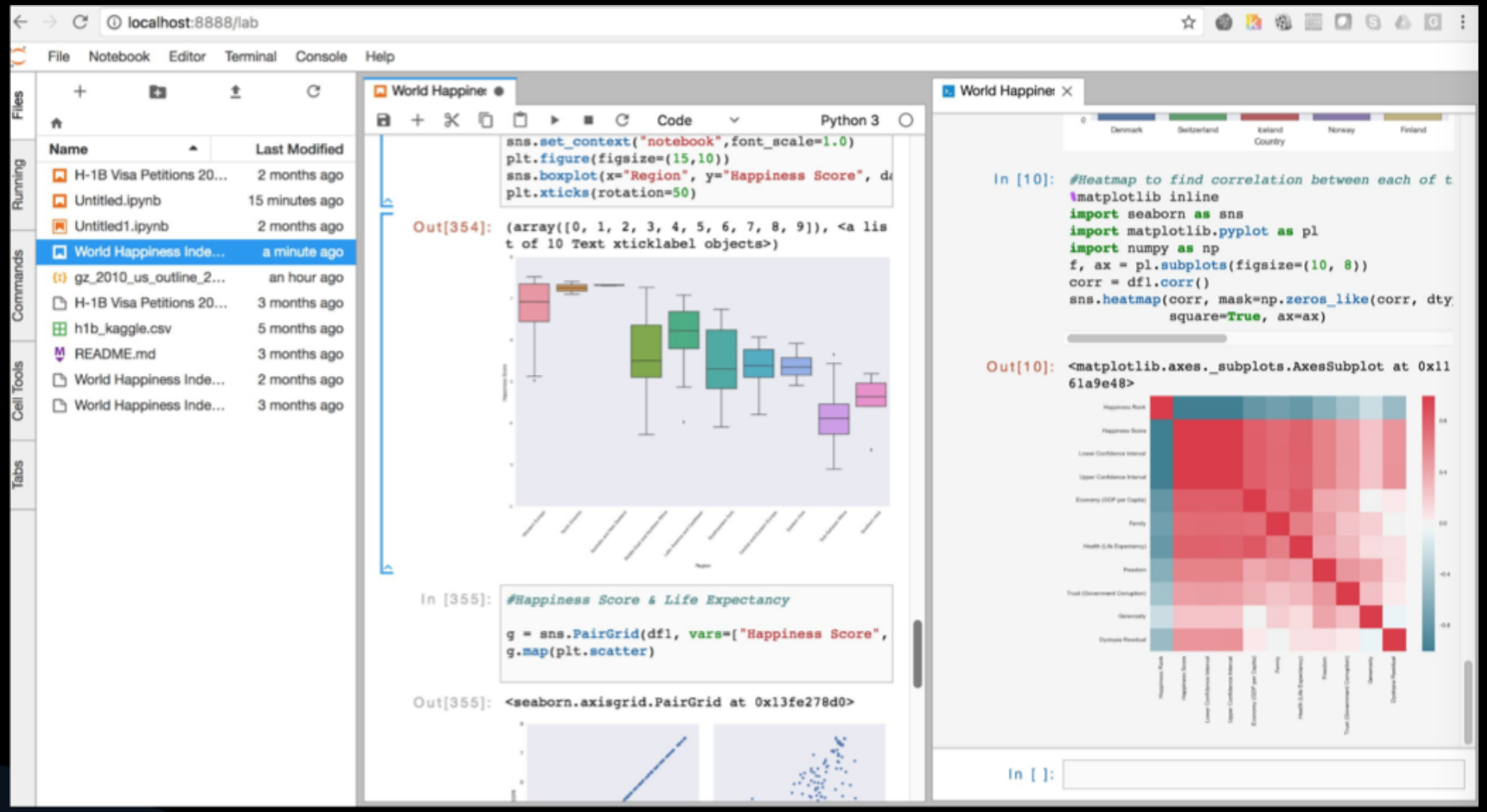


Jupyter的前世今生

PYCON CHINA 2018

Jupyter Lab

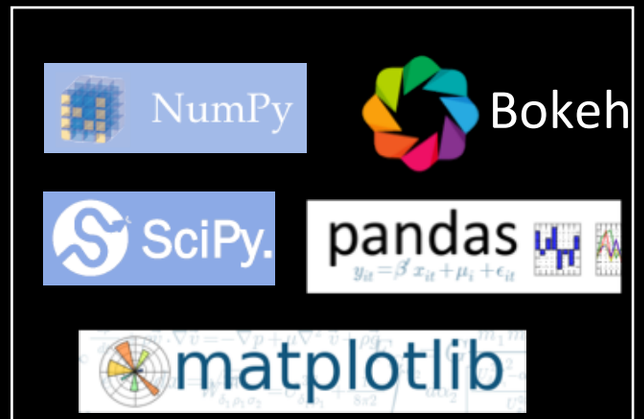
2018



Jupyter现状：很流行

PYCON CHINA 2018

01

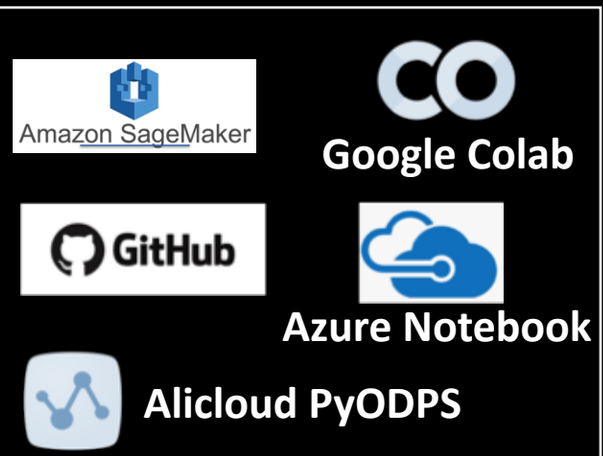


Python科学生态入口

02

开放生态、50+语言

03



云计算界面

交互式计算趋势：Jupyter为啥流行？

PYCON CHINA 2018

1 强大、正经

交互式计算变得“正经”：工具越来越强大、丰富。也逐渐走向开放标准化、生态丰富。

2 计算性叙述

计算性叙述变得“流行”：代码、文本、可视化融合。

3 分析洞察编程

分析洞察”式编程正在“兴起”：区别于泛化任务式编程。

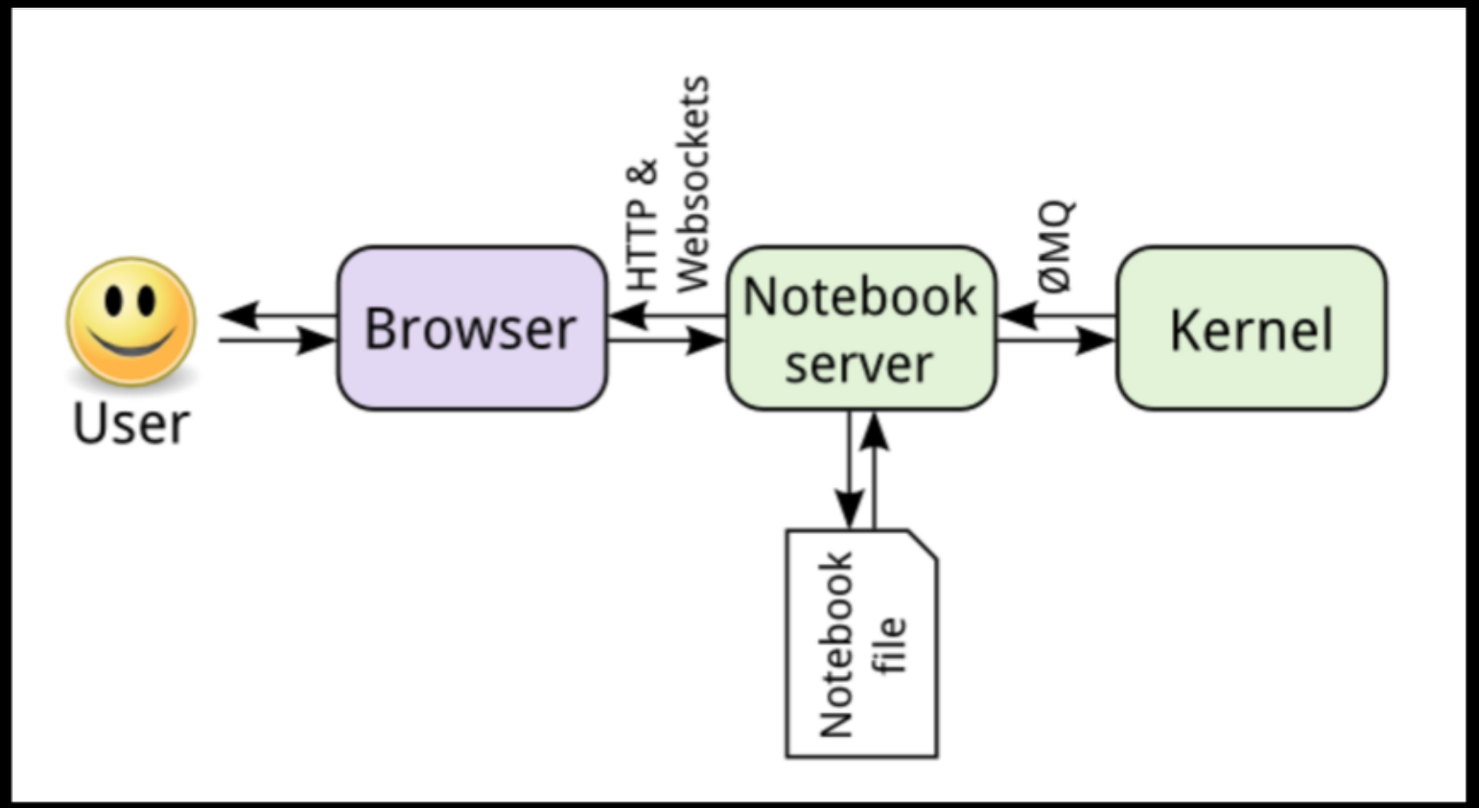
4 拥抱多语言

个人与组织拥抱多语言：跨语言的统一运行、交互与协作。

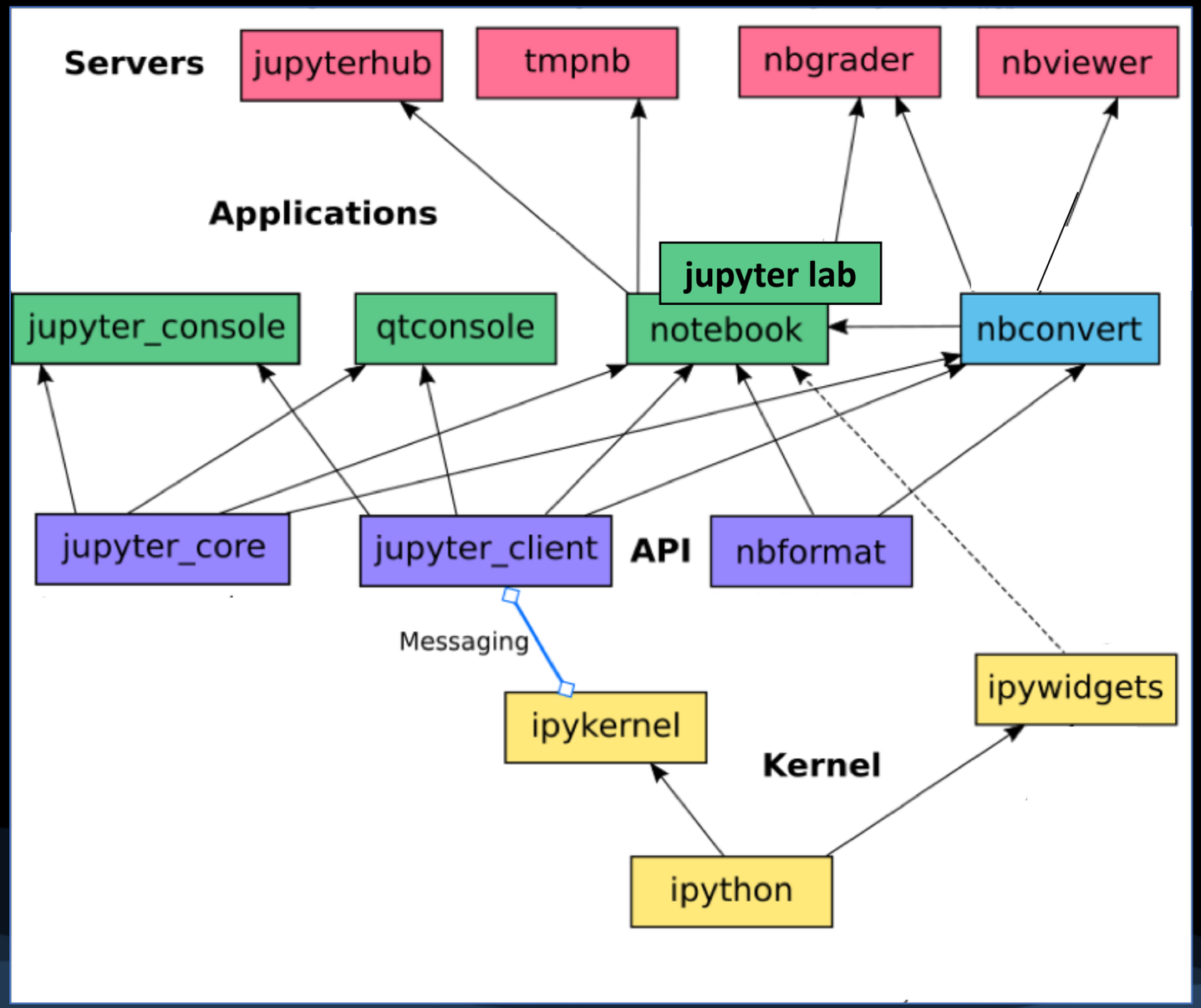
5 超越数据共享

共享不再是仅仅数据：还包括计算、结果与叙述等。

Jupyter架构


User
Browser
HTTP &
Websockets
Notebook
server
ØMQ
Kernel
Notebook
file

Jupyter架构



2

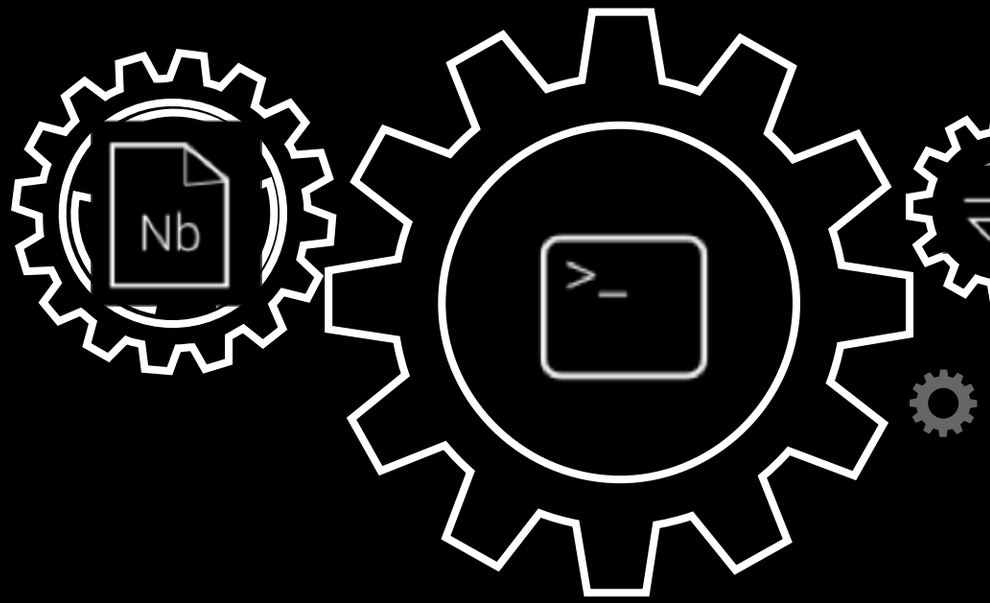
扩展Jupyter

Jupyter如何被扩展？扩展的场景、模式是什么？

Jupyter的开放标准

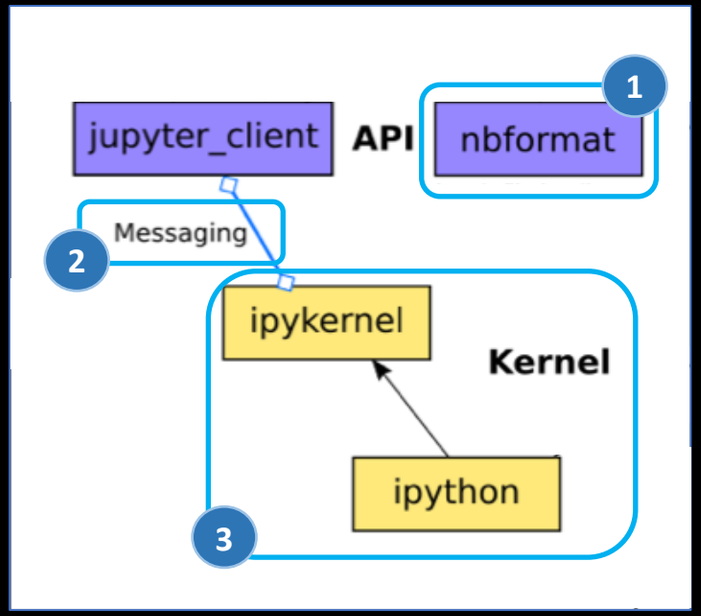
PYCON CHINA 2018

1. Notebook 文档格式



2. 交互计算协议

3. Kernel



Jupyter NB 格式

PYCON CHINA 2018

In [28]:
```
%time
for x in range(10):
    print(x)
```

```
CPU times: user 3 µs, sys: 1 µs, total: 4 µs

Wall time: 10 µs

0

1

2

3

4

5
```


```
{
 "cell_type": "code",
 "execution_count": 28,
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "CPU times: user 3 µs, sys: 1 µs, total: 4 µs\n",
    "Wall time: 10 µs\n",
    "0\n",
    "1\n",
    "2\n",
    "3\n",
    "4\n",
    "5\n",
    "6\n",
    "7\n",
    "8\n",
    "9\n"
   ]
  }
 ],
 "source": [
  "%time\n",
  "for x in range(10):\n",
  "    print(x)"
 ]
},
```

Jupyter扩展策略

PYCON CHINA 2018

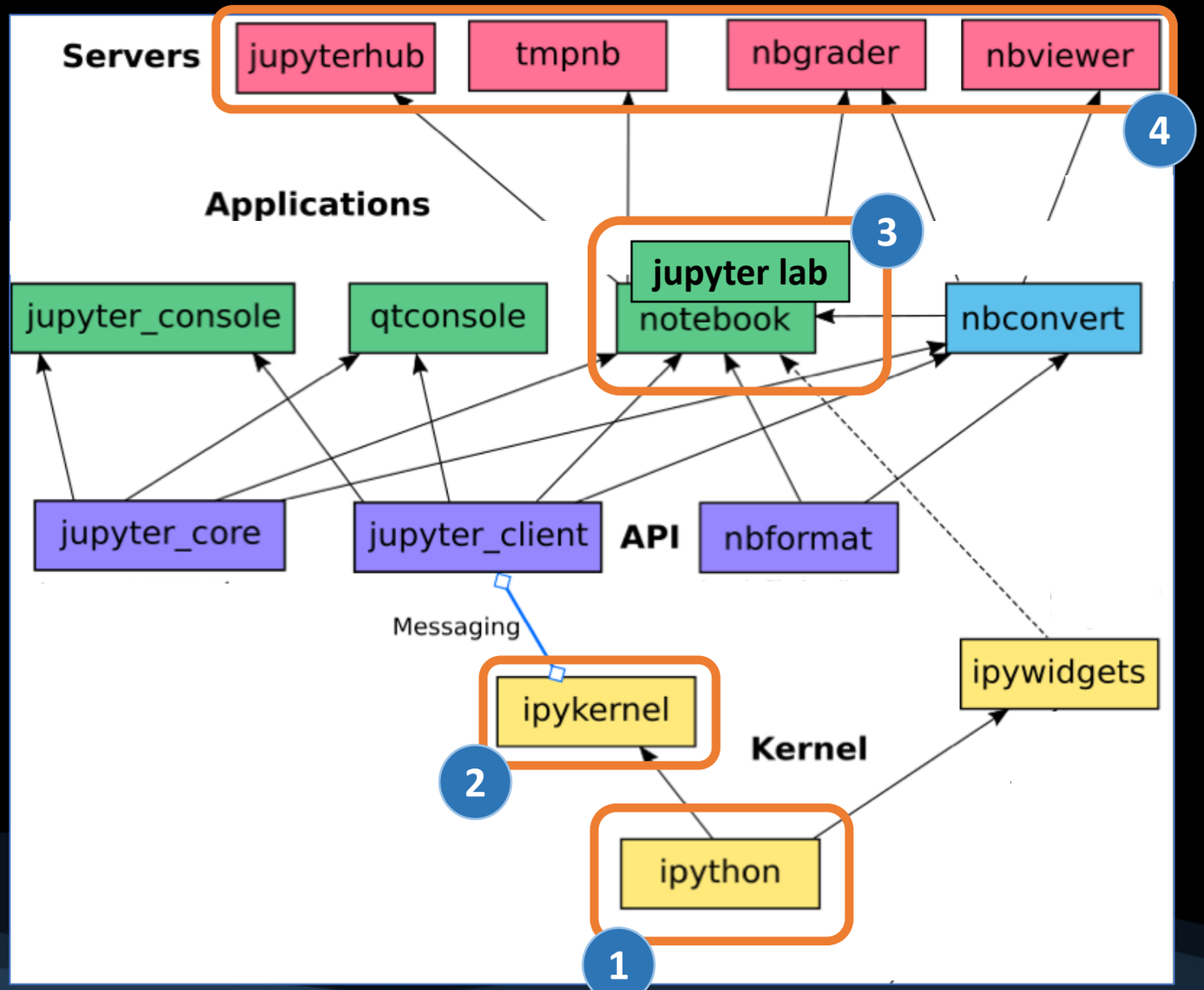
1. 扩展 IPython Kernel

2. 扩展新Kernel

3. 扩展Notebook

4. 扩展架构

Jupyter扩展策略



#1. 扩展 IPython Kernel

PYCON CHINA 2018

- 加载插件
- 扩展Magic Command
- 扩展输入格式

加载插件

- %load_ext ， %unload_ext ， %reload_ext

```
In [ ]: %load_ext my_extension
```

my_extension.py

```
def load_ipython_extension(ipython):
    ipython.register_magic(SLSMagics)
    sls_client = LogClient(....)
    ipython.push({'sls', sls_client})

def unload_ipython_extension(ipython):
    # do some cleanup work
```

扩展 Magic Command

PYCON CHINA 2018

- 如何添加一个魔法方法，执行一些任务？

```
In [11]: %manage_log

Connection to Alicloud Log Service: Verified.
Created Log Service Client as 'log'
Created OSS Service Client as 'oss'
```

扩展 Magic Command

PYCON CHINA 2018

```python
from IPython.core.magic import (Magics,
    magics_class, line_cell_magic)

@magics_class
class SLSMagics(Magics):

    @line_cell_magic
    def manage_log(self, line, cell=None):
        ....

    @line_cell_magic
    def log(self, line, cell=None):
        ....

def load_ipython_extension(ipython):
    ipython.register_magic(SLSMagics)
    log_client = LogClient(....)
    ipython.push({'log', log_client})
```

```
In [11]: %manage_log

Connection to Alicloud Log Service: Verified.
Created Log Service Client as 'log'
Created OSS Service Client as 'oss'
```

Magic Command 支持库

PYCON CHINA 2018

In [8]:
```
%log
population > 1000
| select province as "省份", count(1) as "订单" group by province desc
```

Out[8]:

	省份	订单
0	上海	10000
1	江苏	2000
2	山东	4900

```
# 函数标记
from IPython.core.magic import line_cell_magic, needs_local_scope, line_magic

# 类标记
from IPython.core.magic import magics_class

# 命令参数支持
from IPython.core.magic_arguments import argument, magic_arguments
```

扩展输入格式

- 如何自动将Python2代码转换为Python3？

扩展输入格式

- 如何自动将Python2代码转换为Python3？

```
In [ ]: %load_ext my_extension
```

my_extension.py

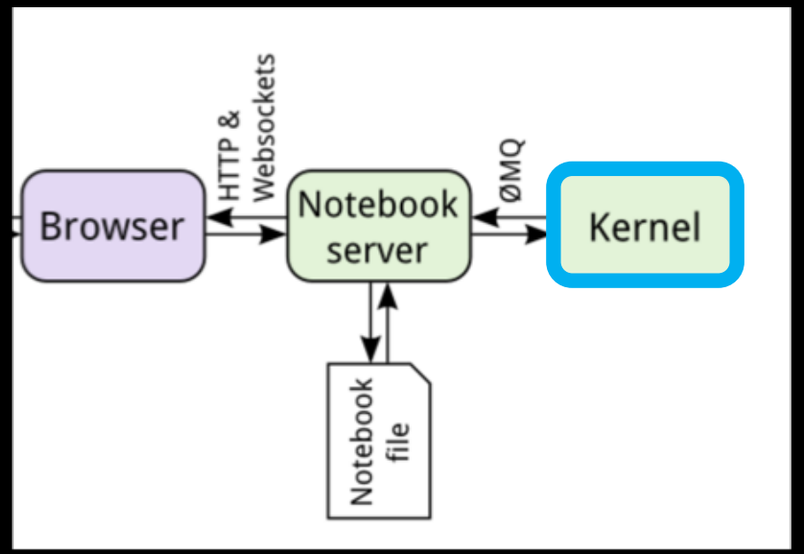
```
def auto2to3(lines):
    code = ''.join(lines)
    return autoformat_2to3(code)

def load_ipython_extension(ipython):
    ipython.input_transformers_cleanup.append(auto2to3)
```

#2. 扩展新 Kernel

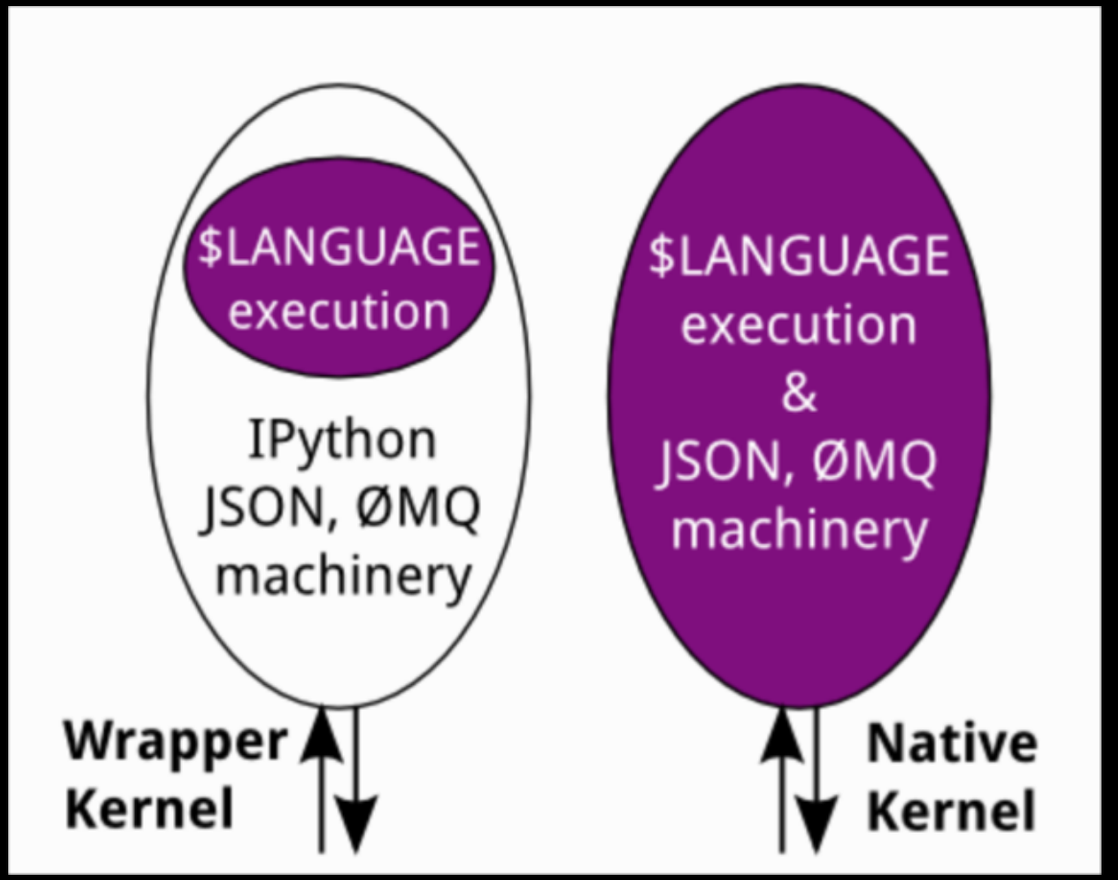
PYCON CHINA 2018

- 场景：
  - 支持一种新的语言
    - 让个这个语言支持交互（例如C++）
  - 支持一种新的DSL
    - 也可能是多个语言的混合或者子集
  - 提供一种新的执行环境
    - 环境可能在Docker、远程、云端等

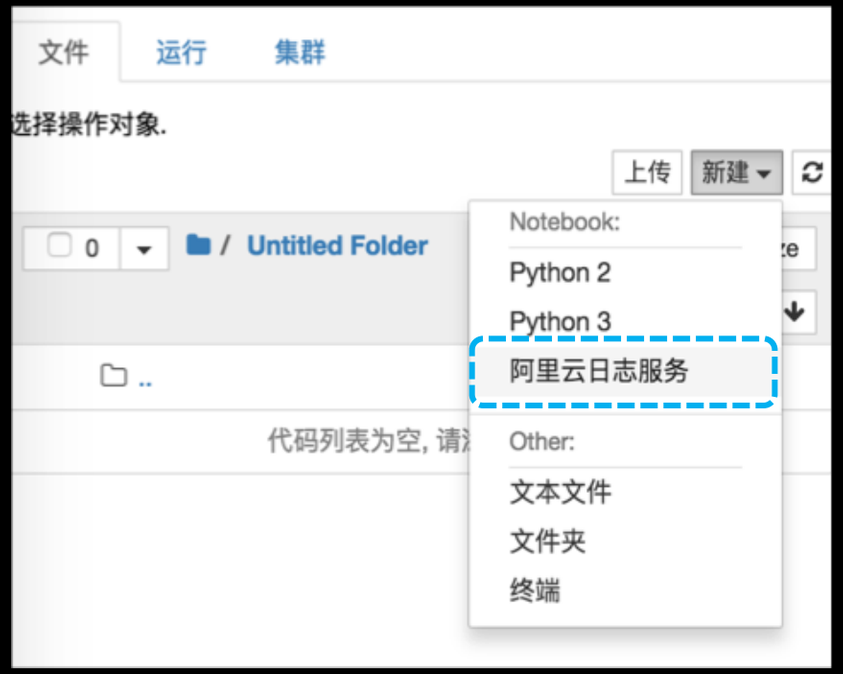


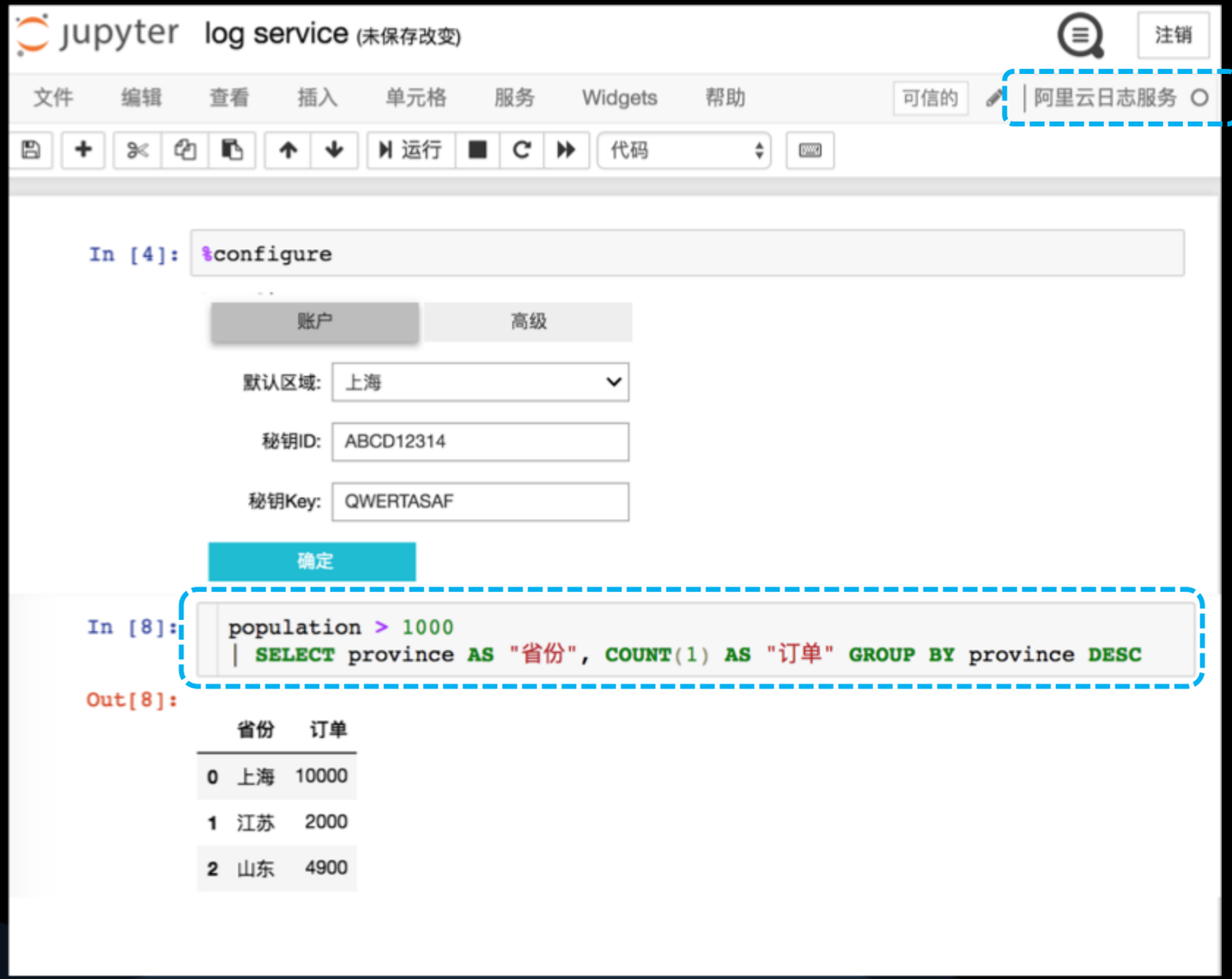
实现策略

- 1. 复用Wrapper Kernel
  - 只需实现语言执行部分
  - 简单高效
- 2. 重头实现
  - 需要重现通讯与执行部分
  - 复杂强大



创建一个Kernel

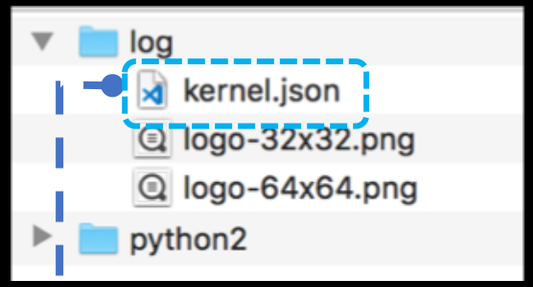

文件
运行
集群
选择操作对象.
上传
新建
0
/ Untitled Folder
..
代码列表为空, 请
Notebook:
Python 2
Python 3
阿里云日志服务
Other:
文本文件
文件夹
终端


jupyter log service (未保存改变)
注销
文件 编辑 查看 插入 单元格 服务 Widgets 帮助
可信的
阿里云日志服务
运行
代码
In [4]: %configure
账户
高级
默认区域: 上海
秘钥ID: ABCD12314
秘钥Key: QWERTASAF
确定
In [8]:
population > 1000
| SELECT province AS "省份", COUNT(1) AS "订单" GROUP BY province DESC
Out[8]:
省份 订单
0 上海 10000
1 江苏 2000
2 山东 4900

创建一个Kernel

PYCON CHINA 2018

/usr/local/share/jupyter/kernels/



kernel.json

```
{
  "argv": [
    "python",
    "-m",
    "alicloud_log",
    "-f",
    "{connection_file}"
  ],
  "display_name": "阿里云日志服务"
}
```

alicloud_log.py

```
from ipykernel.kernelbase import Kernel

class LogKernel(Kernel):
    language_info = {
        'name': 'sql', 'mimetype': 'text/x-sql',
        'codemirror_mode': 'text/x-sql',
        'pygments_lexer': 'sql'
    }

    def do_execute(self, code, silent, store_history=True,
                   user_expressions=None, allow_stdin=False):
        result = log.get_data(code)

        self.send_response(self.iopub_socket, 'stream', result)

        return {'status': 'ok',
                'execution_count': self.execution_count,
                'payload': [], 'user_expressions': {}, }

if __name__ == '__main__':
    from ipykernel.kernelapp import IPKernelApp
    IPKernelApp.launch_instance(kernel_class=LogKernel)
```

#3. 扩展 Notebook

PYCON CHINA 2018

- 对象自定义可视化
  - 如何将一个对象展示位一个图片、图表、交互图？
- 自定义命令
  - 如何添加一个命令?
- 插件管理
  - 构建NB插件
  - NB插件库

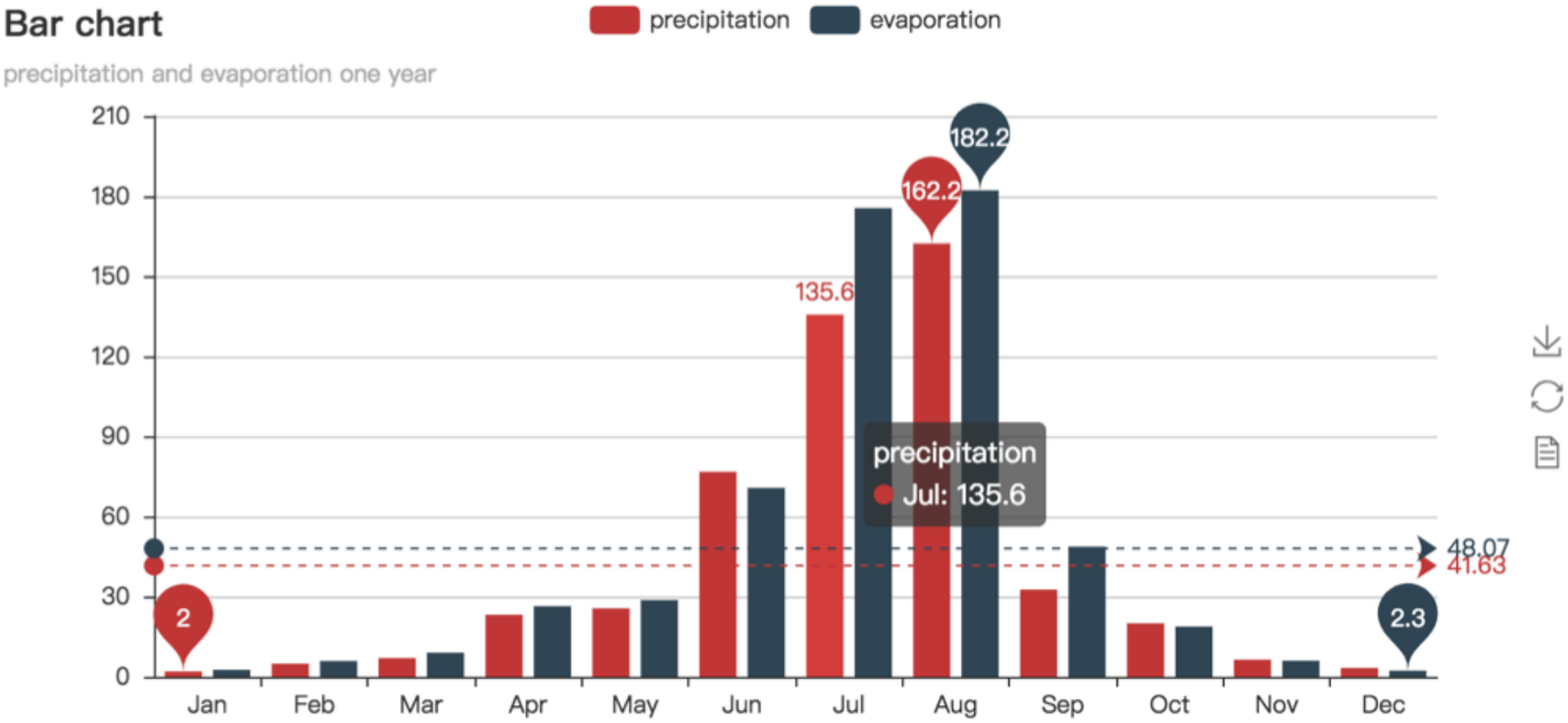
对象自定义可视化

PYCON CHINA 2018

- 如何将一个对象展示为一个图片、图表、交互图？

In [12]:
```
data
```

Out[12]:



对象自定义可视化

PYCON CHINA 2018

- data._repr_html_()

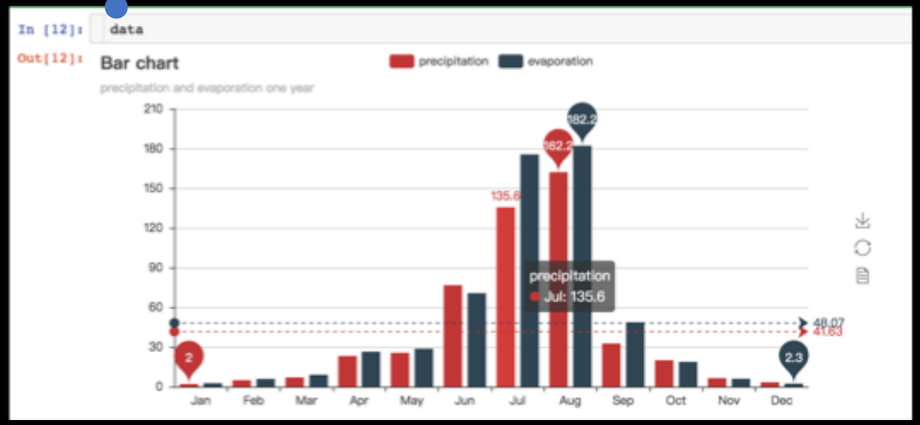
```
<script>
    require.config({...});
</script>

<div id="d1"></div>

<script>
require(['echarts'], function(echarts) {
    var chart1 = echarts.init(
        document.getElementById('d1'),
        'light',
        {renderer: 'canvas'});

    var option_1 = {... };

    chart1.setOption(option_1);
});
</script>
```



对象自定义可视化

- 基于ipwidgets进一步扩展

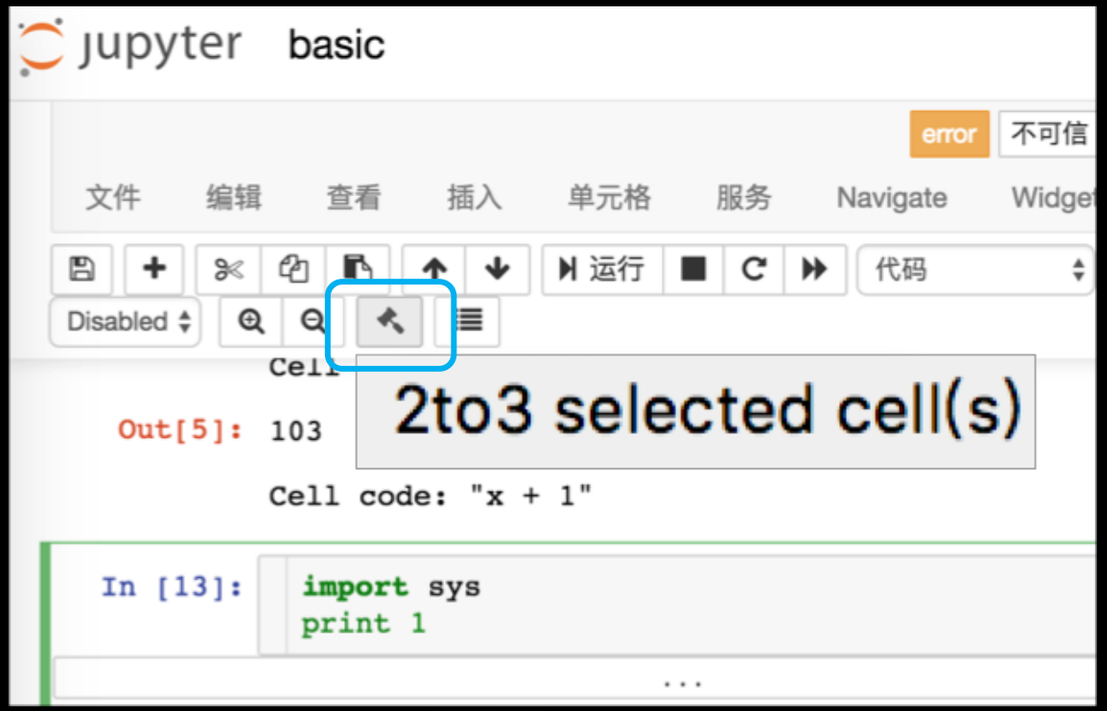
```
from ipywidgets import VBox, Output,
    Button, HTML, HBox, Dropdown, Checkbox
    , ToggleButtons, Text, Textarea, Tab

class MyChart(VBox):
    ....
```

自定义命令

PYCON CHINA 2018

- 如何给工具栏和命令行添加一个功能？



自定义命令

- 如何给工具栏和命令行添加一个功能？

```
%%javascript
var autoformat_cells = function({...});

Jupyter.keyboard_manager.actions.register(
        {
            help: 'convert selected cell(s)',
            icon: 'fa-legal',
            handler: function(evt) { autoformat_cells(); },
        }, 'process_selected_cells', '2to3');
```

```
Jupyter.keyboard_manager.command_shortcuts.add_shortcut('Ctrl+L', function (event)
    autoformat_cells();
    return false;
});
```

构建NB插件

PYCON CHINA 2018

~/.ipython/profile_default/nbconfig/notebook.json

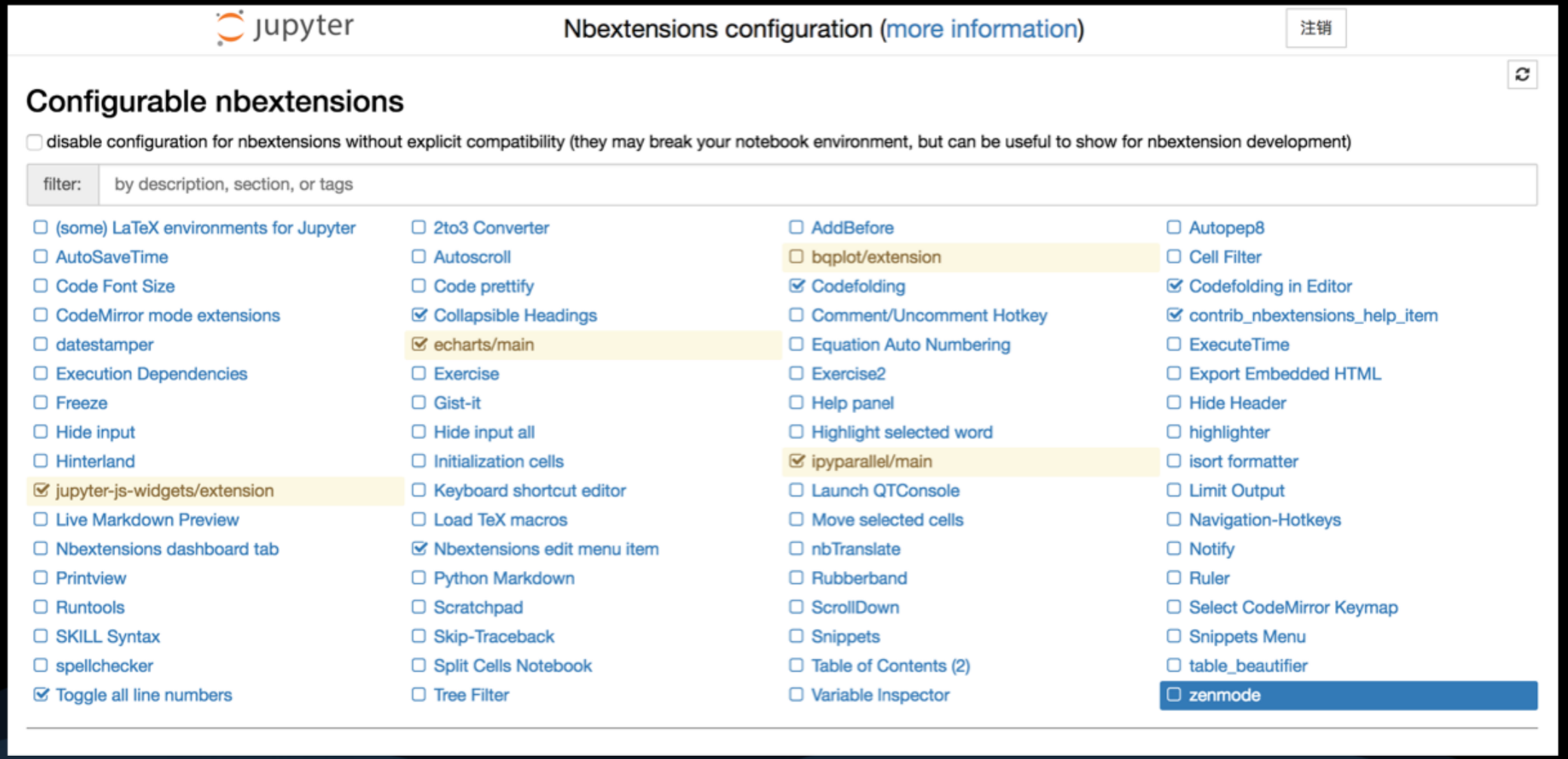
```
{
  "load_extensions": {
    "my_ext/main": true
  }
}
```

~/Library/Jupyter/nbextensions/myext/main.js

```
define(function(){
  return {
    load_ipython_extension: function(){
        // logic here
        IPython.keyboard_manager.XXXXX
    }
  };
})
```

NB插件库

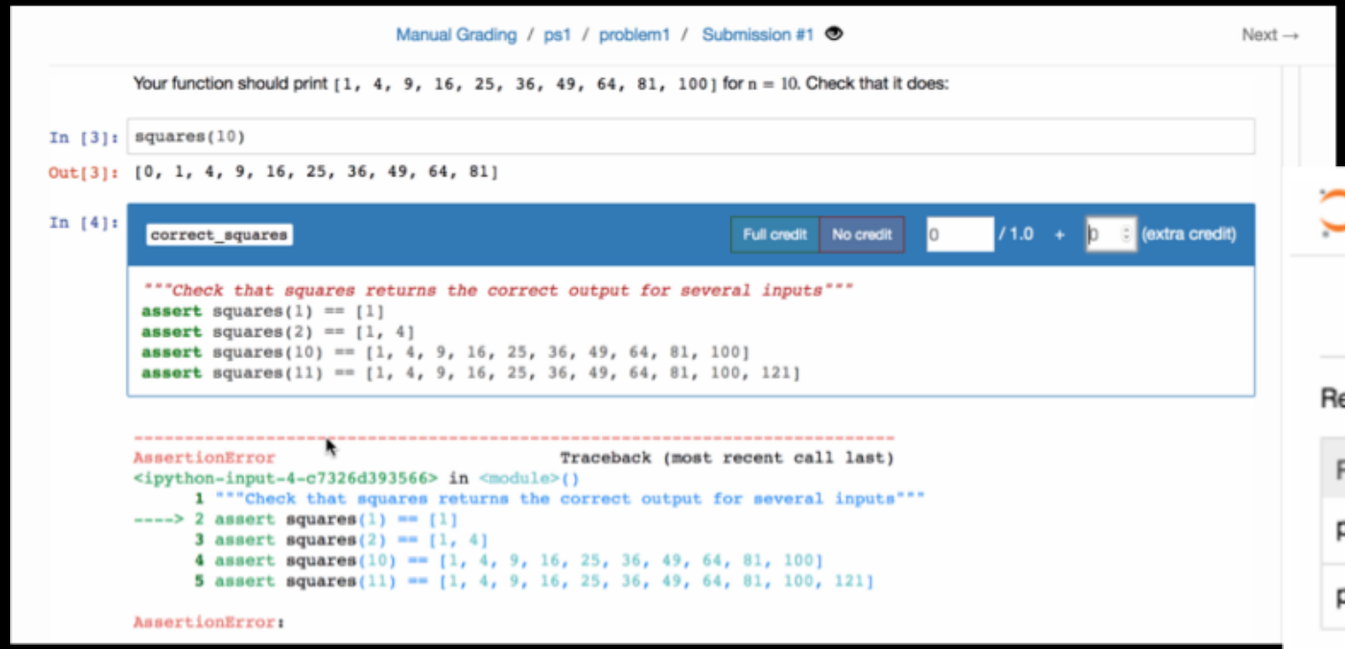
ipython-contrib/jupyter_contrib_nbextensions：一个jupyter NB插件库（接近60个）

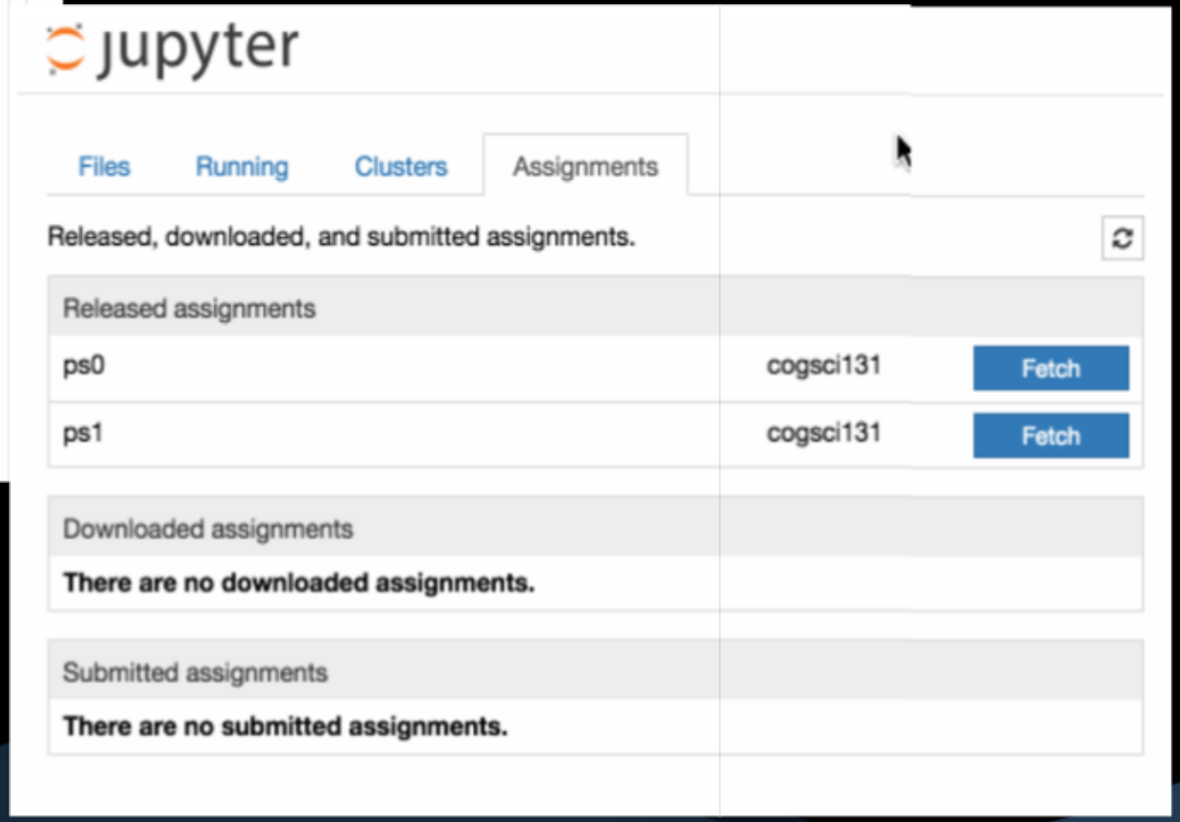


#4. 扩展架构 - NBGrader

PYCON CHINA 2018

- 将Jupyter变成一个考试系统





Jupyter生态中的其他开源项目

PYCON CHINA 2018

- 网络与调度：
  - JupyterHub：Jupyter Notebook的网关
  - Jupyter Kernel Gateway：Kernel的网关，转换MQ协议为REST
  - Ipyparallel: 对Ipython Kenel进行集群管理与调度
- 工作流与结果转换：
  - nteract/papermill: 基于NB的工作流参数化运行、分析与结果汇总管理。
  - Nbconvert：NB文件的运行与格式转换工具（如PDF）
- 可视化改进：
  - RISE：将Jupyter变为PPT播放模式
  - Jupyter Dashboards：因此代码，调整布局，将NB变成报表样式

3

云服务与Jupyter

使用Jupyter扩展云服务场景、技术方法与策略

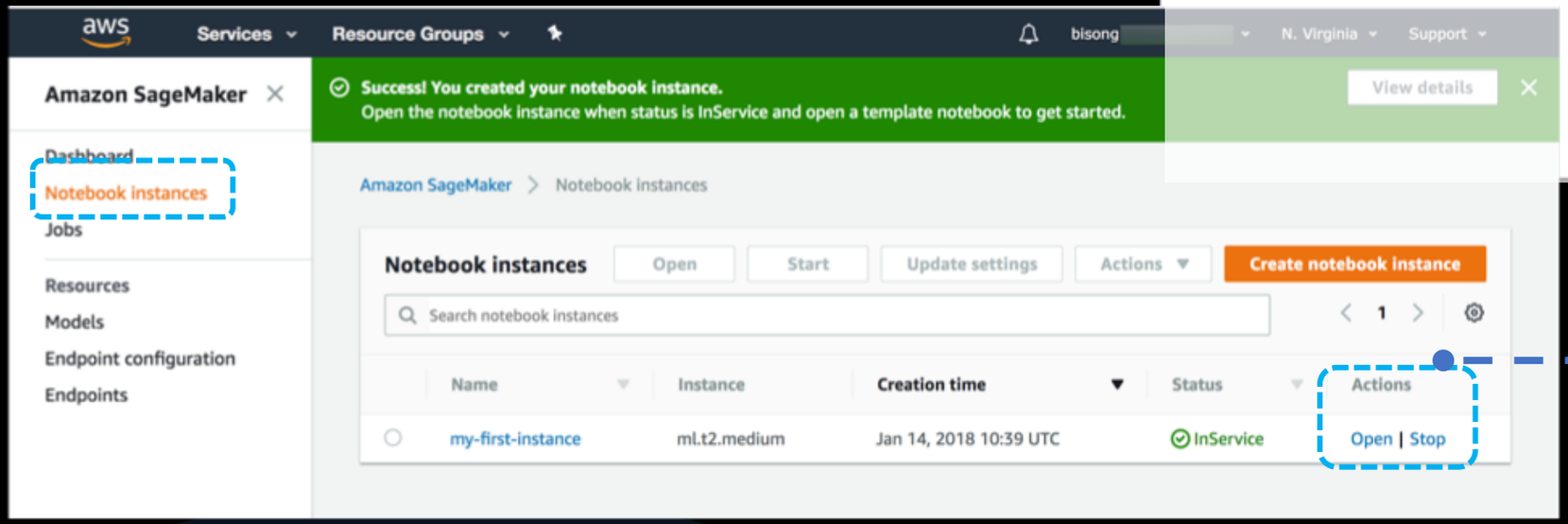
云服务与Jupyter模式

- 托管服务：
  - 提供Notebook的PaaS/SaaS平台，一般支持各种计算运行环境（conda/Spark/TensorFlow等），用户无需关注环境部署等问题，如AWS SageMaker、 Google Conlab、 Azure Notebook等
- 非托管接口：
  - 支持通过Notebook访问计算平台，但不Host Jupyter环境，如阿里云PyODPS, Azure HDInsight.
- NB增强：
  - 除了环境支持增强外，对NB做了较大交互增强甚至重写，如Google Conlab、PyODPS。
- 阅读分享NB：
  - 强化了展示与分享功能，如mybinder, NBViewer、 Google Conlab，Azure Notebook等。

AWS SageMaker

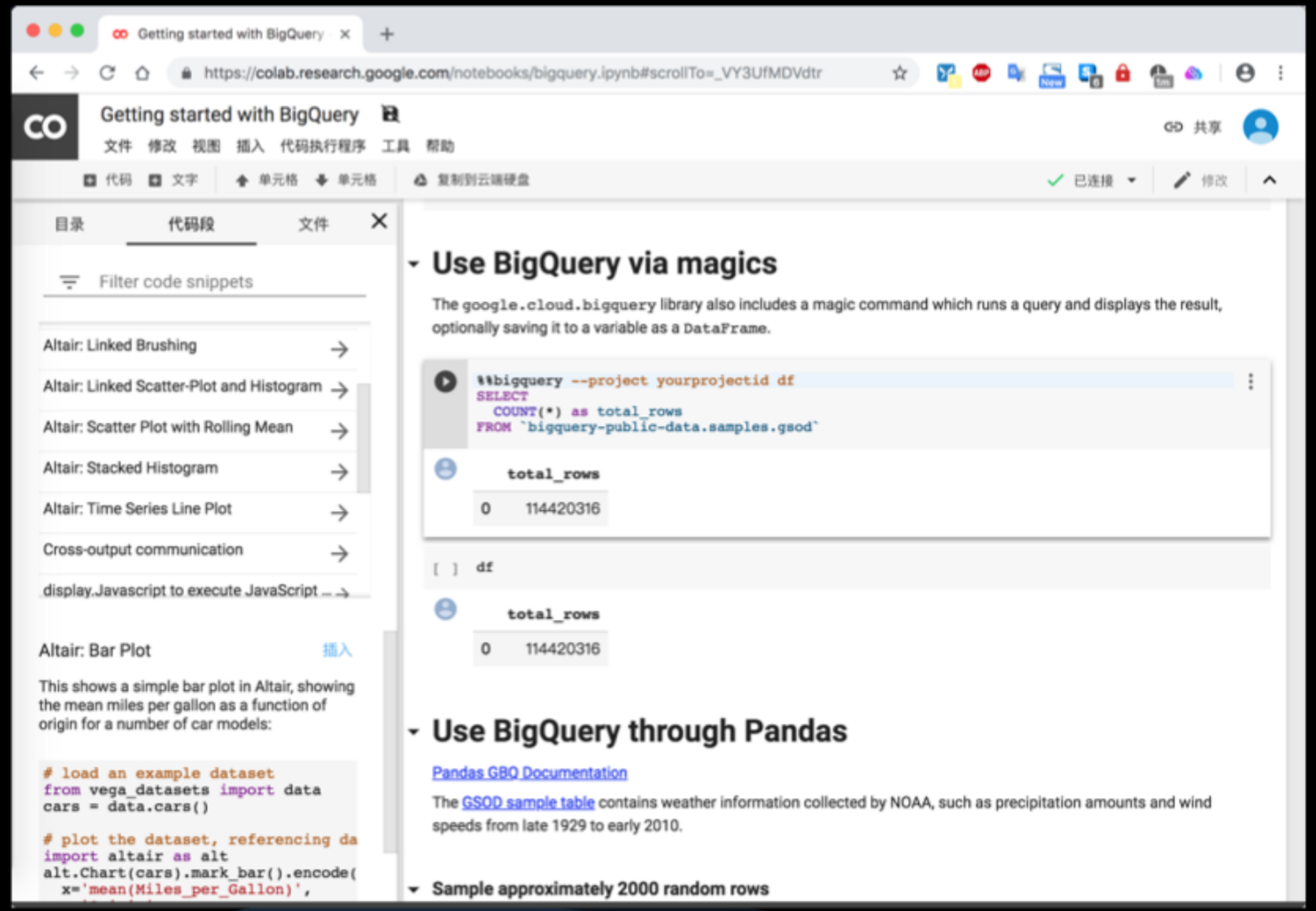
PYCON CHINA 2018

- 全面支持Jupyter的多种科学计算环境
  - PySpark、Spark、SparkR、Tensorflow、MxNet、Conda等
- 支持**基于以上集群的离线任务调度**



Google Conlab

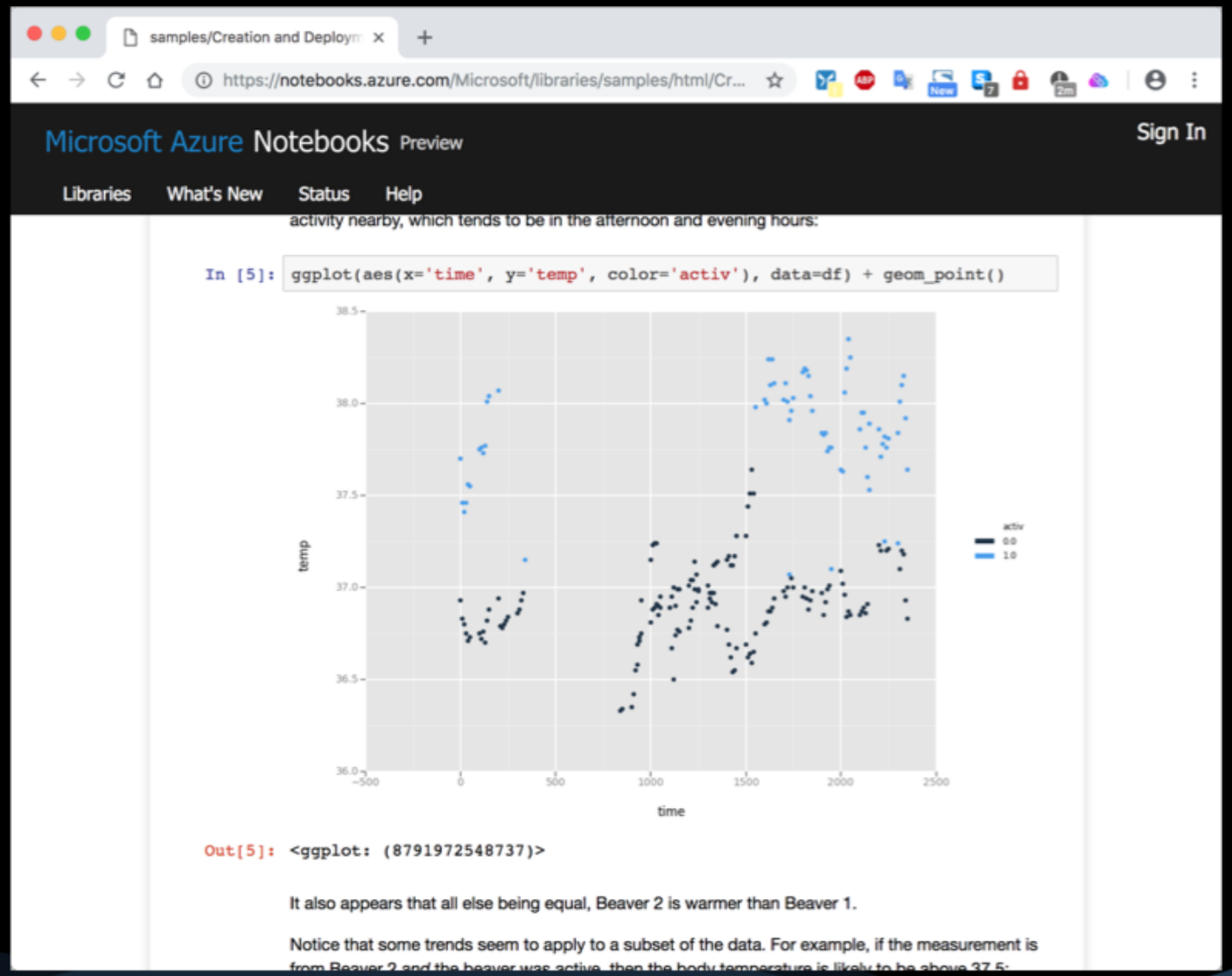
- 提供Notebook的科学计算环境
  - 主要TensorFlow、Conda等
- 重写了Jupyter前端
  - **支持多人编辑、表单**
- **运行时间不可靠，但支持本地运行。**
- 存储绑定云盘使用等



Azure Notebook (Preview)

PYCON CHINA 2018

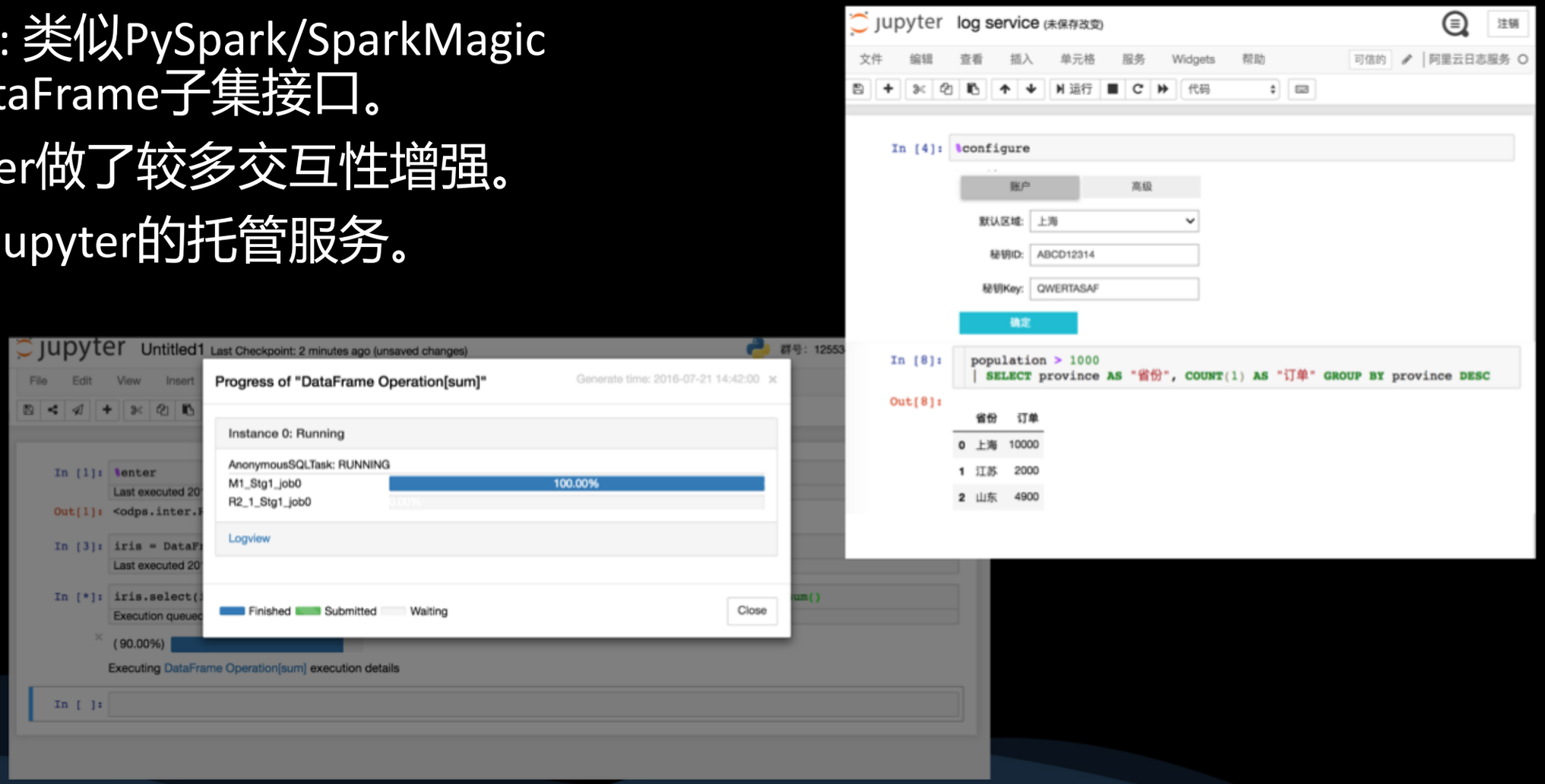
- Jupyter as Service平台
  - 主要Conda、 R和F#
  - **免费4GB内存+1GB存储**
- 内置JupyterLab、 RISE等



Alicloud PyODPS/Log Service

PYCON CHINA 2018

- PyODPS: 类似PySpark/SparkMagic 提供DataFrame子集接口。
- 对Jupyter做了较多交互性增强。
- 不提供Jupyter的托管服务。

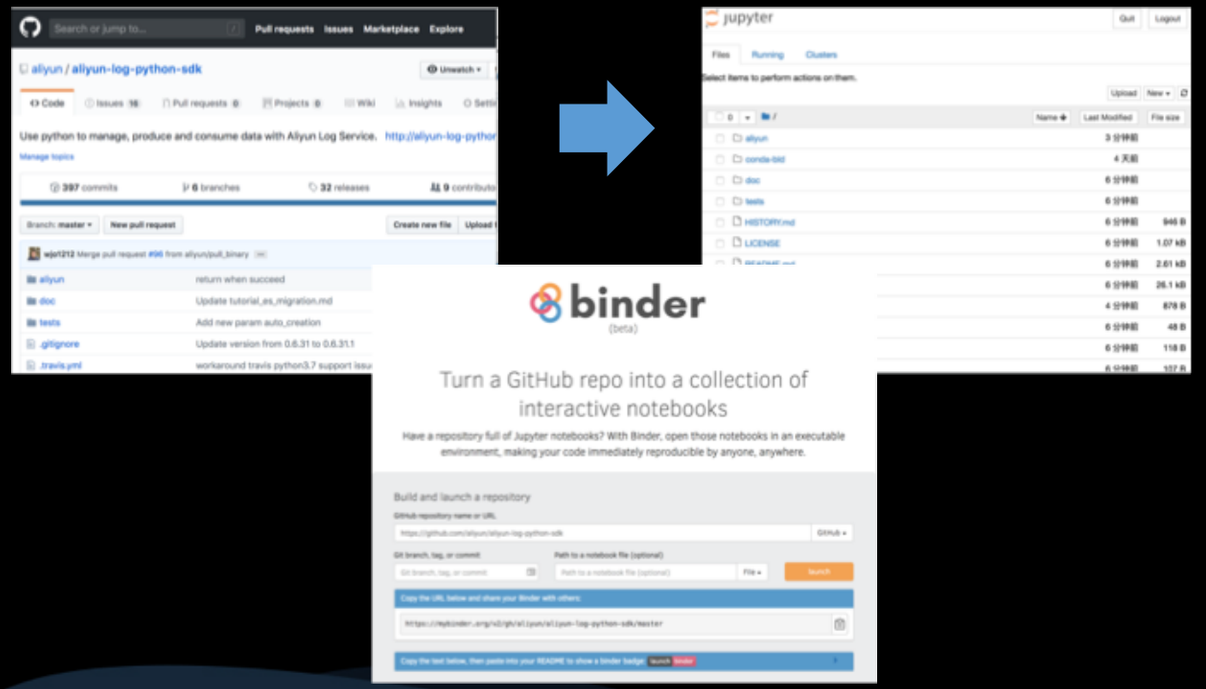


MyBinder/NBViewer

PYCON CHINA 2018

MyBinder：提供将Github项目转化为一个Docker环境下可运行的Jupyter项目

NBViewer：提供URL到Notebook展示的服务。



4

Demo

演示扩展Jupyter、赋能云服务的一些场景

PYCON CHINA 2018

THANK YOU

本人微信

Github下载PPT

日志服务钉钉群