Learning Python from Anaconda

Author: wangfeicheng

Mail: [wang\_feicheng@163.com](mailto:wang_feicheng@163.com) or [wangfeichengzjhw@huawei.com](mailto:wangfeichengzjhw@huawei.com)

# The zen of Python

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

…..

# Anaconda介绍、安装及使用教程

## 参考资料网址：

<https://zhuanlan.zhihu.com/p/32925500>

## 〇、序

Python是一种面向对象的解释型计算机程序设计语言，其使用，具有跨平台的特点，可以在Linux、macOS以及Windows系统中搭建环境并使用，其编写的代码在不同平台上运行时，几乎不需要做较大的改动，使用者无不受益于它的便捷性。

此外，Python的强大之处在于它的应用领域范围之广，遍及人工智能、科学计算、Web开发、系统运维、大数据及云计算、金融、游戏开发等。实现其强大功能的前提，就是Python具有数量庞大且功能相对完善的标准库和第三方库。通过对库的引用，能够实现对不同领域业务的开发。然而，正是由于库的数量庞大，对于管理这些库以及对库作及时的维护成为既重要但复杂度又高的事情。

## Anaconda 是什么？

Anaconda is a completely free Python distribution (including for commercial use and redistribution). It includes over 195 of the most popularPython packagesfor science, math, engineering, data analysis.

**1. 简介**

Anaconda（[官方网站](https://link.zhihu.com/?target=https%3A//www.anaconda.com/download/%23macos)）就是可以便捷获取包且对包能够进行管理，同时对环境可以统一管理的发行版本。Anaconda包含了conda、Python在内的超过180个科学包及其依赖项。

**2. 特点**

Anaconda具有如下特点：

▪ 开源

▪ 安装过程简单

▪ 高性能使用Python和R语言

▪ 免费的社区支持

其特点的实现主要基于Anaconda拥有的：

▪ conda包

▪ 环境管理器

▪ 1,000+开源库

如果日常工作或学习并不必要使用1,000多个库，那么可以考虑安装Miniconda（[下载界面请戳](https://link.zhihu.com/?target=https%3A//conda.io/miniconda.html)），这里不过多介绍Miniconda的安装及使用。

**3. Anaconda、conda、pip、virtualenv的区别**

**① Anaconda**

Anaconda是一个包含180+的科学包及其依赖项的发行版本。其包含的科学包包括：conda, numpy, scipy, ipython notebook等。

**② conda**

conda是包及其依赖项和环境的管理工具。

▪ 适用语言：Python, R, Ruby, Lua, Scala, Java, JavaScript, C/C++, FORTRAN。

▪ 适用平台：Windows, macOS, Linux

▪ 用途：

① 快速安装、运行和升级包及其依赖项。

② 在计算机中便捷地创建、保存、加载和切换环境。

如果你需要的包要求不同版本的Python，你无需切换到不同的环境，因为conda同样是一个环境管理器。仅需要几条命令，你可以创建一个完全独立的环境来运行不同的Python版本，同时继续在你常规的环境中使用你常用的Python版本。——[Conda官方网站](https://link.zhihu.com/?target=https%3A//conda.io/docs/)

▪ conda为Python项目而创造，但可适用于上述的多种语言。

▪ conda包和环境管理器包含于Anaconda的所有版本当中。

**③ pip**

pip是用于安装和管理软件包的包管理器。

▪ pip编写语言：Python。

▪ Python中默认安装的版本：

① Python 2.7.9及后续版本：默认安装，命令为 ***pip***

② Python 3.4及后续版本：默认安装，命令为 ***pip3***

▪ pip名称的由来：pip采用的是**递归缩写**进行命名的。其名字被普遍认为来源于2处：

① “Pip installs Packages”（“pip安装包”）

② “Pip installs Python”（“pip安装Python”）

**④ virtualenv**

virtualenv是用于创建一个**独立的**Python环境的工具。

▪ 解决问题：

1. 当一个程序需要使用Python 2.7版本，而另一个程序需要使用Python 3.6版本，如何同时使用这两个程序？如果将所有程序都安装在系统下的默认路径，如：***/usr/lib/python2.7/site-packages***，当不小心升级了本不该升级的程序时，将会对其他的程序造成影响。
2. 如果想要安装程序并在程序运行时对其库或库的版本进行修改，都会导致程序的中断。
3. 在共享主机时，无法在全局 ***site-packages*** 目录中安装包。

▪ virtualenv将会为它自己的安装目录创建一个环境，这并**不与**其他virtualenv环境共享库；同时也可以**选择性**地不连接已安装的全局库。

**⑤ pip 与 conda 比较**

**→ 依赖项检查**

▪ pip：

① **不一定**会展示所需其他依赖包。

② 安装包时**或许**会直接忽略依赖项而安装，仅在结果中提示错误。

▪ conda：

① 列出所需其他依赖包。

② 安装包时自动安装其依赖项。

③ 可以便捷地在包的不同版本中自由切换。

**→ 环境管理**

▪ pip：维护多个环境难度较大。

▪ conda：比较方便地在不同环境之间进行切换，环境管理较为简单。

**→ 对系统自带Python的影响**

▪ pip：在系统自带Python中包的更新/回退版本/卸载将影响其他程序。

▪ conda：不会影响系统自带Python。

**→ 适用语言**

▪ pip：仅适用于Python。

▪ conda：适用于Python, R, Ruby, Lua, Scala, Java, JavaScript, C/C++, FORTRAN。

**⑥ conda与pip、virtualenv的关系**

▪ conda**结合**了pip和virtualenv的功能。

## Anaconda的适用平台及安装条件

**1. 适用平台**

Anaconda可以在以下系统平台中安装和使用：

▪ Windows

▪ macOS

▪ Linux（x86 / Power8）

**2. 安装条件**

▪ 系统要求：32位或64位系统均可

▪ 下载文件大小：约500MB

▪ 所需空间大小：3GB空间大小（Miniconda仅需400MB空间即可）

## 三、Anaconda的安装步骤

### 1. macOS系统安装Anaconda

**① 图形界面安装**

1. 前往[官方下载页面](https://link.zhihu.com/?target=https%3A//www.anaconda.com/downloads%23macos)下载。有两个版本可供选择：Python 3.6 和 Python 2.7，我下载的是前者。选择版之后点击“64-Bit Graphical Installer”进行下载。

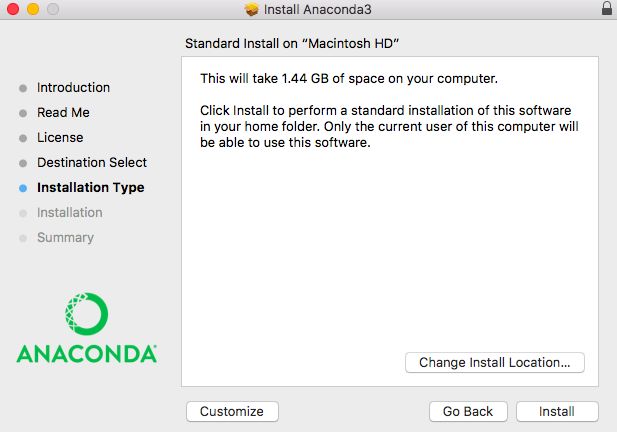
2. 完成下载之后，双击下载文件，在对话框中“Introduction”、“Read Me”、“License”部分可直接点击下一步

3. “Destination Select”部分选择“Install for me only”并点击下一步。

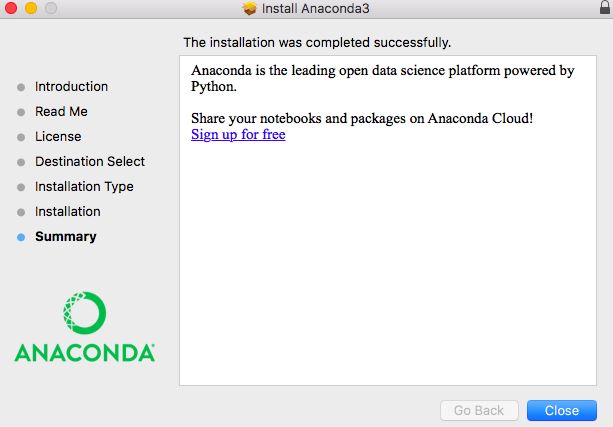
* 注意：若有错误提示信息“You cannot install Anaconda in this location”则重新选择“Install for me only”并点击下一步。



4. “Installation Type”部分，可以点击“Change Install Location”来改变安装位置。标准的安装路径是在用户的家目录下。在这一步我没有改变安装位置。若选择默认安装路径，则直接点击“Install”进行安装。



5. 等待“Installation”部分结束，在“Summary”部分若看到“The installation was completed successfully.”则安装成功，直接点击“Close”关闭对话框。

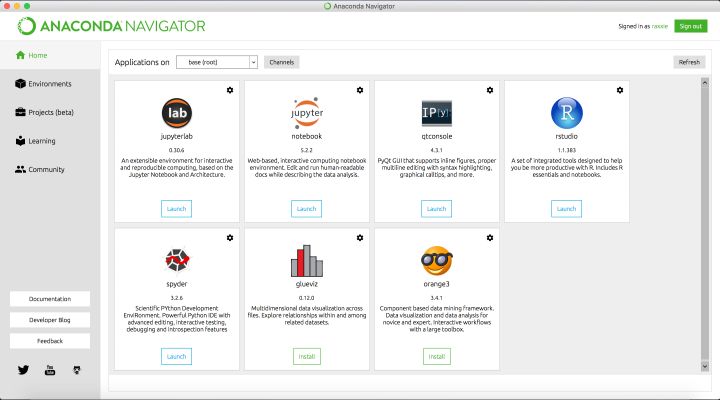


6. 在mac的Launchpad中可以找到名为“Anaconda-Navigator”的图标，点击打开。



7. 若“Anaconda-Navigator”成功启动，则说明真正成功地安装了Anaconda；如果未成功，请务必仔细检查以上安装步骤。

* 提示：“Anaconda-Navigator”中已经包含“Jupyter Notebook”、“Jupyterlab”、“Qtconsole”和“Spyder”。（图中的“Rstudio”是我后来安装的，但它默认出现在“Anaconda-Navigator”的启动界面，只需要点击“Install”便可安装。）
* Jupyter Notebook有助于我们编写代码、运行代码以及获取代码的运行结果，特点是可以令我们便捷地为代码及其运行结果添加文档的描述、解释和说明。无论是学习还是工作，Jupyter Notebook都是提高效率和学习、工作质量的利器。具体的使用，可以参见我的另一篇文章：

[豆豆：Jupyter Notebook介绍、安装及使用教程​zhuanlan.zhihu.com](https://zhuanlan.zhihu.com/p/33105153)

8. 完成安装。

**② 命令行安装**

1. 前往[官方下载页面](https://link.zhihu.com/?target=https%3A//www.anaconda.com/downloads%23macos)下载。有两个版本可供选择：Python 3.6 和 Python 2.7，我下载的是前者。选择版之后点击“64-Bit Command-Line Installer”进行下载。

2. 完成下载之后，在mac的Launchpad中找到“其他”并打开“终端”。

▫ 安装Python 3.6： ***bash ~/Downloads/Anaconda3-5.0.1-MacOSX-x86\_64.sh***

▫ 安装Python 2.7： ***bash ~/Downloads/Anaconda2-5.0.1-MacOSX-x86\_64.sh***

* 注意：

1. 首词bash也需要输入，无论是否用的Bash shell。
2. 如果你的下载路径是自定义的，那么把该步骤路径中的 ***~/Downloads*** 替换成你自己的下载路径。
3. 如果你将第1步下载的 ***.sh*** 文件重命名了，那么把该步骤路径中的 ***Anaconda3-5.0.1-MacOSX-x86\_64.sh*** 或 ***Anaconda2-5.0.1-MacOSX-x86\_64.sh*** 替换成你重命名后的文件名。

▫ 强烈建议：**不要**修改文件名。如果重命名，使用**英文**进行命名。

3. 安装过程中，看到提示“In order to continue the installation process, please review the license agreement.”（“请浏览许可证协议以便继续安装。”），点击“Enter”查看“许可证协议”。

4. 在“许可证协议”界面将屏幕滚动至底，输入“yes”表示同意许可证协议内容。然后进行下一步。

5. 安装过程中，提示“Press Enter to confirm the location, Press CTRL-C to cancel the installation or specify an alternate installation directory.”（“按回车键确认安装路径，按'CTRL-C'取消安装或者指定安装目录。”）如果接受默认安装路径，则会显示***PREFIX=/home/<user>/anaconda<2 or 3>*** 并且继续安装。安装过程大约需要几分钟的时间。

* 建议：直接接受默认安装路径。

6. 安装器若提示“Do you wish the installer to prepend the Anaconda install location to PATH in your /home/<user>/.bash\_profile ?”（“你希望安装器添加Anaconda安装路径在***/home/<user>/.bash\_profile*** 文件中吗？”），建议输入“yes”。

* 注意：

① 路径 ***/home/<user>/.bash\_profile*** 中 ***<user>*** 即进入到家目录后你的目录名。

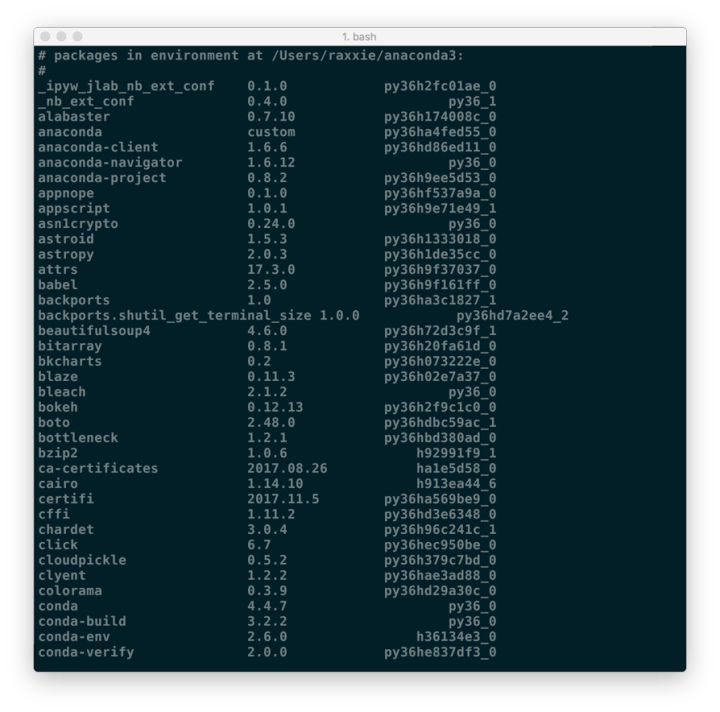
② 如果输入“no”，则需要手动添加路径。添加 ***export PATH="/<path to anaconda>/bin:$PATH"*** 在 ***.bashrc*** 或者 ***.bash\_profile*** 中。其中， ***<path to anaconda>***替换为你真实的Anaconda安装路径。

7. 当看到“Thank you for installing Anaconda!”则说明已经成功完成安装。

8. 关闭终端，然后再打开终端以使安装后的Anaconda启动。

9. 验证安装结果。可选用以下任意一种方法：

① 在终端中输入命令 ***condal list*** ，如果Anaconda被成功安装，则会显示已经安装的包名和版本号。



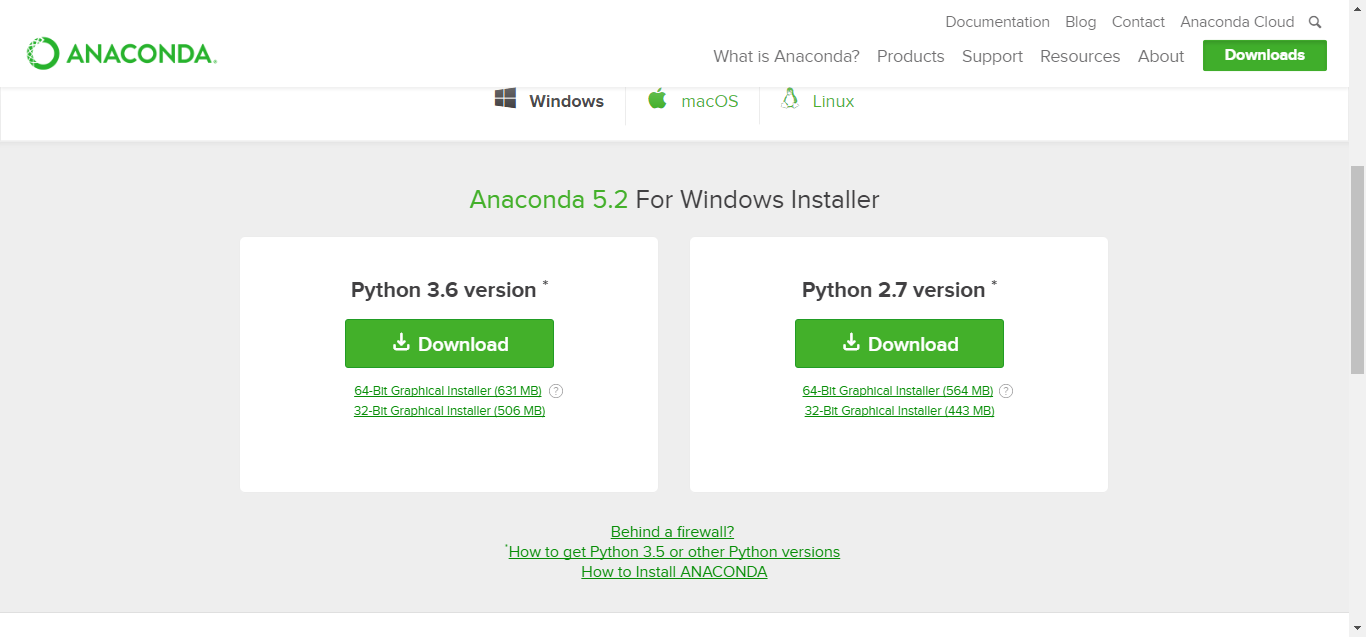
② 在终端中输入 ***python*** 。这条命令将会启动Python交互界面，如果Anaconda被成功安装并且可以运行，则将会在Python版本号的右边显示“Anaconda custom (64-bit)”。退出Python交互界面则输入 ***exit()*** 或 ***quit()*** 即可。



③ 在终端中输入 ***anaconda-navigator*** 。如果Anaconda被成功安装，则Anaconda Navigator的图形界面将会被启动。

### 2. Windows系统安装Anaconda

1. 前往[官方下载页面](https://link.zhihu.com/?target=https%3A//docs.anaconda.com/anaconda/install/windows)下载。有两个版本可供选择：Python 3.6 和 Python 2.7，选择版之后根据自己操作系统的情况点击“64-Bit Graphical Installer”或“32-Bit Graphical Installer”进行下载。



2. 完成下载之后，双击下载文件，启动安装程序。

* 注意：

① 如果在安装过程中遇到任何问题，那么暂时地关闭杀毒软件，并在安装程序完成之后再打开。

② 如果在安装时选择了“为所有用户安装”，则卸载Anaconda然后重新安装，只为“我这个用户”安装。

3. 选择“Next”。

4. 阅读许可证协议条款，然后勾选“I Agree”并进行下一步。

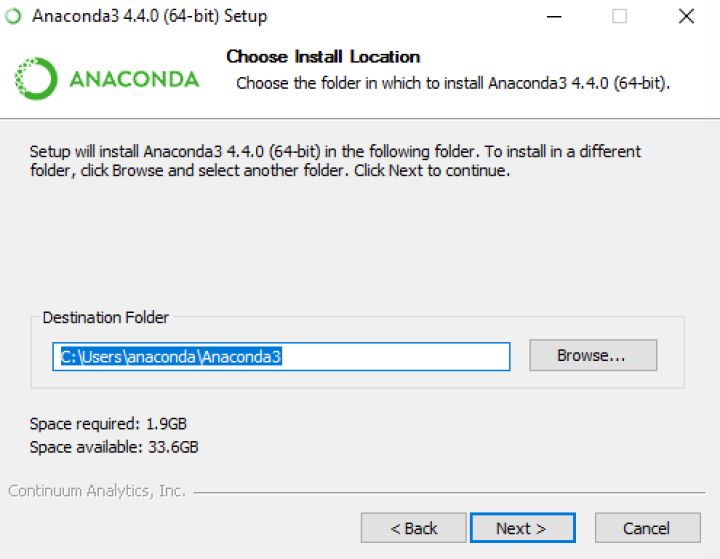
5. 除非是以管理员身份为所有用户安装，否则仅勾选“Just Me”并点击“Next”。

6. 在“Choose Install Location”界面中选择安装Anaconda的目标路径，然后点击“Next”。

* 注意：

① 目标路径中**不能**含有**空格**，同时不能是**“unicode”**编码。

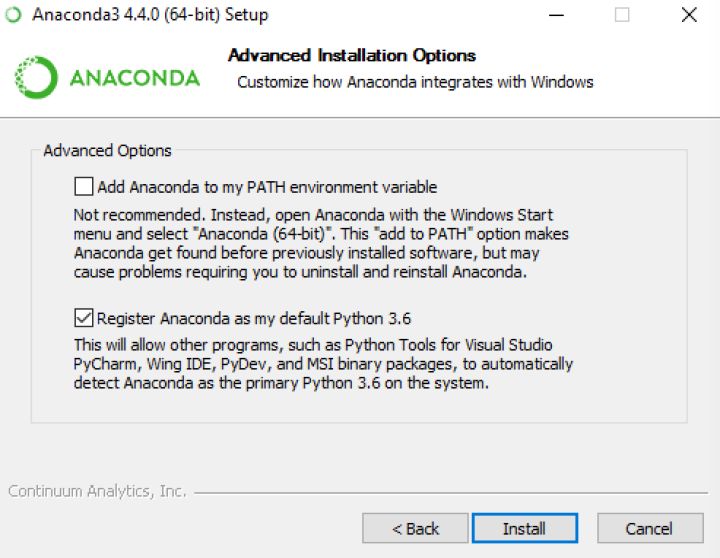
② 除非被要求以管理员权限安装，否则不要以管理员身份安装。



7. 在“Advanced Installation Options”中**不要**勾选“Add Anaconda to my PATH environment variable.”（“添加Anaconda至我的环境变量。”）。因为如果勾选，则将会影响其他程序的使用。如果使用Anaconda，则通过打开Anaconda Navigator或者在开始菜单中的“Anaconda Prompt”（类似macOS中的“终端”）中进行使用。

除非你打算使用多个版本的Anaconda或者多个版本的Python，否则便勾选“Register Anaconda as my default Python 3.6”。

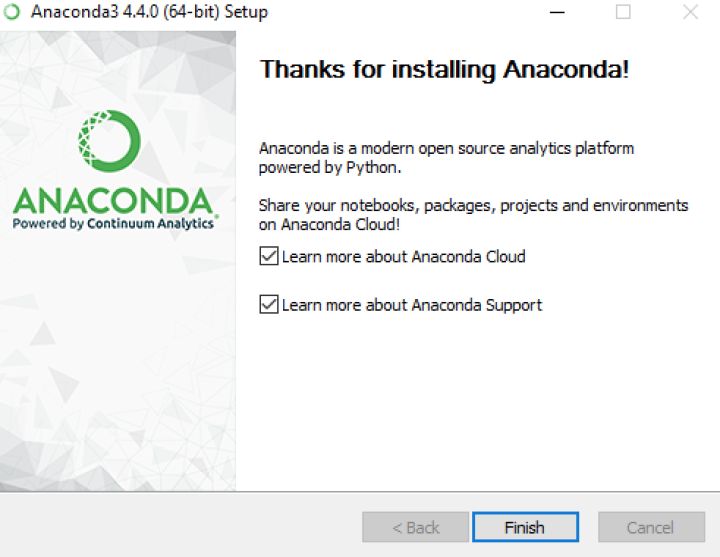
然后点击“Install”开始安装。如果想要查看安装细节，则可以点击“Show Details”。



8. 点击“Next”。

9. 进入“Thanks for installing Anaconda!”界面则意味着安装成功，点击“Finish”完成安装。

* 注意：如果你不想了解“Anaconda云”和“Anaconda支持”，则可以**不勾选**“Learn more about Anaconda Cloud”和“Learn more about Anaconda Support”。



10. 验证安装结果。可选以下任意方法：

① “开始 → Anaconda3（64-bit）→ Anaconda Navigator”，若可以成功启动Anaconda Navigator则说明安装成功。

② “开始 → Anaconda3（64-bit）→ 右键点击Anaconda Prompt → 以管理员身份运行”，在Anaconda Prompt中输入 ***conda list*** ，可以查看已经安装的包名和版本号。若结果可以正常显示，则说明安装成功。

### 3. Linux系统安装Anaconda

1. 前往[官方下载页面](https://link.zhihu.com/?target=https%3A//www.anaconda.com/download/%23linux)下载。有两个版本可供选择：Python 3.6 和 Python 2.7。

2. 启动终端，在终端中输入命令 ***md5sum /path/filename*** 或 ***sha256sum /path/filename***

* 注意：将该步骤命令中的 ***/path/filename*** 替换为文件的实际下载路径和文件名。其中，path是路径，filename为文件名。
* 强烈建议：

① 路径和文件名中不要出现空格或其他特殊字符。

② 路径和文件名最好以英文命名，不要以中文或其他特殊字符命名。

3. 根据Python版本的不同有选择性地在终端输入命令：

▫ Python 3.6： ***bash ~/Downloads/Anaconda3-5.0.1-Linux-x86\_64.sh***

▫ Python 2.7： ***bash ~/Downloads/Anaconda2-5.0.1-Linux-x86\_64.sh***

* 注意：

① 首词bash也需要输入，无论是否用的Bash shell。

② 如果你的下载路径是自定义的，那么把该步骤路径中的 ***~/Downloads*** 替换成你自己的下载路径。

③ 除非被要求使用root权限，否则均选择“Install Anaconda as a user”。

4. 安装过程中，看到提示“In order to continue the installation process, please review the license agreement.”（“请浏览许可证协议以便继续安装。”），点击“Enter”查看“许可证协议”。

5. 在“许可证协议”界面将屏幕滚动至底，输入“yes”表示同意许可证协议内容。然后进行下一步。

6. 安装过程中，提示“Press Enter to accept the default install location, CTRL-C to cancel the installation or specify an alternate installation directory.”（“按回车键确认安装路径，按'CTRL-C'取消安装或者指定安装目录。”）如果接受默认安装路径，则会显示***PREFIX=/home/<user>/anaconda<2 or 3>*** 并且继续安装。安装过程大约需要几分钟的时间。

* 建议：直接接受默认安装路径。

7. 安装器若提示“Do you wish the installer to prepend the Anaconda<2 or 3> install location to PATH in your /home/<user>/.bashrc ?”（“你希望安装器添加Anaconda安装路径在 ***/home/<user>/.bashrc*** 文件中吗？”），建议输入“yes”。

* 注意：

① 路径 ***/home/<user>/.bash\_rc*** 中 ***“<user>”*** 即进入到家目录后你的目录名。

② 如果输入“no”，则需要手动添加路径，否则conda将无法正常运行。

8. 当看到“Thank you for installing Anaconda<2 or 3>!”则说明已经成功完成安装。

9. 关闭终端，然后再打开终端以使安装后的Anaconda启动。或者直接在终端中输入 ***source ~/.bashrc*** 也可完成启动。

10. 验证安装结果。可选用以下任意一种方法：

① 在终端中输入命令 ***condal list*** ，如果Anaconda被成功安装，则会显示已经安装的包名和版本号。

② 在终端中输入 ***python*** 。这条命令将会启动Python交互界面，如果Anaconda被成功安装并且可以运行，则将会在Python版本号的右边显示“Anaconda custom (64-bit)”。退出Python交互界面则输入 ***exit()*** 或 ***quit()*** 即可。

③ 在终端中输入 ***anaconda-navigator*** 。如果Anaconda被成功安装，则Anaconda Navigator将会被启动。

## 四、管理conda

**0. 写在前面**

接下来均是以命令行模式进行介绍，Windows用户请打开“Anaconda Prompt”；macOS和Linux用户请打开“Terminal”（“终端”）进行操作。

**1. 验证conda已被安装**

conda --version

终端上将会以 ***conda 版本号*** 的形式显示当前安装conda的版本号。如： ***conda 3.11.0***

* 注意：如果出现错误信息，则需核实是否出现以下情况：

① 使用的用户是否是安装Anaconda时的账户。

② 是否在安装Anaconda之后重启了终端。

**2. 更新conda至最新版本**

conda update conda

执行命令后，conda将会对版本进行比较并列出可以升级的版本。同时，也会告知用户其他相关包也会升级到相应版本。

当较新的版本可以用于升级时，终端会显示 ***Proceed ([y]/n)?*** ，此时输入 ***y*** 即可进行升级。

**3. 查看conda帮助信息**

conda --help

或

conda -h

**4. 卸载conda**

**① Linux 或 macOS**

rm -rf ~/anaconda2

或

rm -rf ~/anaconda3

即删除Anaconda的安装目录。根据安装的Anaconda版本选择相应的卸载命令。

**② Windows**

控制面板 → 添加或删除程序 → 选择“Python X.X (Anaconda)” → 点击“删除程序”

* 注意：

① Python X.X：即Python的版本，如：Python 3.6。

② Windows 10的删除有所不同。

## 五、管理环境

**0. 写在前面**

接下来均是以命令行模式进行介绍，Windows用户请打开“Anaconda Prompt”；macOS和Linux用户请打开“Terminal”（“终端”）进行操作。

**1. 创建新环境**

conda create --name <env\_name> <package\_names>

* 注意：

▫ ***<env\_name>*** 即创建的环境名。建议以英文命名，且不加空格，名称两边不加尖括号“<>”。

▫ ***<package\_names>***即安装在环境中的包名。名称两边不加尖括号“<>”。

① 如果要安装指定的版本号，则只需要在包名后面以 ***=*** 和版本号的形式执行。如： ***conda create --name python2 python=2.7***，即创建一个名为“python2”的环境，环境中安装版本为2.7的python。

② 如果要在新创建的环境中创建多个包，则直接在 ***<package\_names>*** 后以**空格**隔开，添加多个包名即可。如： ***conda create -n python3 python=3.5 numpy pandas*** ，即创建一个名为“python3”的环境，环境中安装版本为3.5的python，同时也安装了numpy和pandas。

▫ ***--name*** 同样可以替换为 ***-n*** 。

* 提示：默认情况下，新创建的环境将会被保存在 ***/Users/<user\_name>/anaconda3/env*** 目录下，其中， ***<user\_name>*** 为当前用户的用户名。

**2. 切换环境**

**① Linux 或 macOS**

source activate <env\_name>

**② Windows**

activate <env\_name>

**③ 提示**

1. 如果创建环境后安装Python时没有指定Python的版本，那么将会安装与Anaconda版本相同的Python版本，即如果安装Anaconda第2版，则会自动安装Python 2.x；如果安装Anaconda第3版，则会自动安装Python 3.x。
2. 当成功切换环境之后，在该行行首将以“(env\_name)”或“[env\_name]”开头。其中，“env\_name”为切换到的环境名。如：在macOS系统中执行 ***source active python2*** ，即切换至名为“python2”的环境，则行首将会以(python2)开头。

**3. 退出环境至root**

**① Linux 或 macOS**

source deactivate

**② Windows**

deactivate

**③ 提示**

当执行退出当前环境，回到root环境命令后，原本行首以“(env\_name)”或“[env\_name]”开头的字符将不再显示。

**4. 显示已创建环境**

conda info --envs

或

conda info -e

或

conda env list

* 例如：



结果中星号“\*”所在行即为当前所在环境。macOS系统中默认创建的环境名为“base”。

**5. 复制环境**

conda create --name <new\_env\_name> --clone <copied\_env\_name>

* 注意：

①***<copied\_env\_name>*** 即为被复制/克隆环境名。环境名两边不加尖括号“<>”。

②***<new\_env\_name>*** 即为复制之后新环境的名称。环境名两边不加尖括号“<>”。

③***conda create --name py2 --clone python2*** ，即为克隆名为“python2”的环境，克隆后的新环境名为“py2”。此时，环境中将同时存在“python2”和“py2”环境，且两个环境的配置相同。

**6. 删除环境**

conda remove --name <env\_name> --all

* 注意： ***<env\_name>*** 为被删除环境的名称。环境名两边不加尖括号“<>”。

## 六、管理包

**1. 查找可供安装的包版本**

**① 精确查找**

conda search --full-name <package\_full\_name>

* 注意：

①***--full-name*** 为精确查找的参数。

②***<package\_full\_name>*** 是被查找包的**全名**。包名两边不加尖括号“<>”。

* 例如： ***conda search --full-name python*** 即查找全名为“python”的包有哪些版本可供安装。

**② 模糊查找**

conda search <text>

* 注意： ***<text>*** 是查找含有**此字段**的包名。此字段两边不加尖括号“<>”。
* 例如： ***conda search py*** 即查找含有“py”字段的包，有哪些版本可供安装。

**2. 获取当前环境中已安装的包信息**

conda list

执行上述命令后将在终端显示当前环境已安装包的包名及其版本号。

**3. 安装包**

**① 在指定环境中安装包**

conda install --name <env\_name> <package\_name>

* 注意：

**① <env\_name>** 即将包安装的指定环境名。环境名两边不加尖括号“<>”。

**② <package\_name>** 即要安装的包名。包名两边不加尖括号“<>”。

* 例如： ***conda install --name python2 pandas*** 即在名为“python2”的环境中安装pandas包。

**② 在当前环境中安装包**

conda install <package\_name>

* 注意：

①***<package\_name>*** 即要安装的包名。包名两边不加尖括号“<>”。

② 执行命令后在当前环境中安装包。

* 例如： ***conda install pandas*** 即在当前环境中安装pandas包。

**③ 使用pip安装包**

**→ 使用场景**

当使用 ***conda install*** 无法进行安装时，可以使用pip进行安装。例如：see包。

**→ 命令**

pip install <package\_name>

* 注意： ***<package\_name>*** 为指定安装包的名称。包名两边不加尖括号“<>”。
* 如： ***pip install see*** 即安装see包。

**→ 注意**

1. pip只是包管理器，无法对环境进行管理。因此如果想在指定环境中使用pip进行安装包，则需要先切换到指定环境中，再使用pip命令安装包。
2. pip无法更新python，因为pip并不将python视为包。
3. pip可以安装一些conda无法安装的包；conda也可以安装一些pip无法安装的包。因此当使用一种命令无法安装包时，可以尝试用另一种命令。

**④ 从**[http://**Anaconda.org**](https://link.zhihu.com/?target=http%3A//Anaconda.org)**安装包**

**→ 使用场景**

当使用 ***conda install*** 无法进行安装时，可以考虑从[http://Anaconda.org](https://link.zhihu.com/?target=http%3A//Anaconda.org)中获取安装包的命令，并进行安装。

**→ 注意**

1. 从[http://Anaconda.org](https://link.zhihu.com/?target=http%3A//Anaconda.org)安装包时，无需注册。

2. 在**当前环境**中安装来自于[http://Anaconda.org](https://link.zhihu.com/?target=http%3A//Anaconda.org)的包时，需要通过输入要安装的包在[http://Anaconda.org](https://link.zhihu.com/?target=http%3A//Anaconda.org)中的路径作为获取途径（channel）。查询路径的方式如下：

① 在浏览器中输入：[http://anaconda.org](https://link.zhihu.com/?target=http%3A//anaconda.org/)，或直接点击[Anaconda Cloud](https://link.zhihu.com/?target=http%3A//anaconda.org/)

② 在新页面“Anaconda Cloud”的上方搜索框中输入要安装的包名，然后点击右边“放大镜”标志。

## 七、参考资料

1. 知乎“初学python者自学anaconda的正确姿势是什么？？”[猴子的回答](https://www.zhihu.com/question/58033789/answer/254673663?utm_source=wechat_session&utm_medium=social)
2. [Anaconda Cheat Sheet](https://link.zhihu.com/?target=https%3A//docs.anaconda.com/_downloads/Anaconda-Starter-Guide-Cheat-Sheet.pdf)
3. [Anaconda官方网站](https://link.zhihu.com/?target=https%3A//www.anaconda.com/download/%23macos)
4. [conda官方网站](https://link.zhihu.com/?target=https%3A//conda.io/docs/)
5. [pip维基百科](https://link.zhihu.com/?target=https%3A//en.wikipedia.org/wiki/Pip_%28package_manager%29%23External_links)
6. [pip官方网站](https://link.zhihu.com/?target=https%3A//pip.pypa.io/en/stable/)
7. YouTube视频：[Pip vs Conda: Differences and Comparisons](https://link.zhihu.com/?target=https%3A//www.youtube.com/watch%3Fv%3DRXLRIHaabaI)
8. [virtualenv官方网站](https://link.zhihu.com/?target=https%3A//virtualenv.pypa.io/en/stable/)
9. [macOS系统安装Anaconda的官方教程](https://link.zhihu.com/?target=https%3A//docs.anaconda.com/anaconda/install/mac-os%23macos-graphical-install)
10. [Windows系统安装Anaconda的官方教程](https://link.zhihu.com/?target=https%3A//docs.anaconda.com/anaconda/install/windows)
11. [Linux系统安装Anaconda的官方教程](https://link.zhihu.com/?target=https%3A//docs.anaconda.com/anaconda/install/linux)
12. [conda官方使用指南](https://link.zhihu.com/?target=https%3A//conda.io/docs/user-guide/getting-started.html%23managing-conda)

# Jupyter Notebook介绍、安装及使用教程

参考资料网址：

<https://zhuanlan.zhihu.com/p/33105153>

## 一、什么是Jupyter Notebook？

### 1. 简介

Jupyter Notebook是基于网页的用于交互计算的应用程序。其可被应用于全过程计算：开发、文档编写、运行代码和展示结果。——[Jupyter Notebook官方介绍](https://link.zhihu.com/?target=https%3A//jupyter-notebook.readthedocs.io/en/stable/notebook.html)

简而言之，Jupyter Notebook是以网页的形式打开，可以在网页页面中**直接编写代码**和**运行代码**，代码的**运行结果**也会直接在代码块下显示的程序。如在编程过程中需要编写说明文档，可在同一个页面中直接编写，便于作及时的说明和解释。

### 2. 组成部分

① 网页应用

网页应用即基于网页形式的、结合了编写说明文档、数学公式、交互计算和其他富媒体形式的工具。**简言之，网页应用是可以实现各种功能的工具。**

② 文档

即Jupyter Notebook中所有交互计算、编写说明文档、数学公式、图片以及其他富媒体形式的输入和输出，都是以文档的形式体现的。

这些文档是保存为后缀名为.ipynb的JSON格式文件，不仅便于版本控制，也方便与他人共享。

此外，文档还可以导出为：HTML、LaTeX、PDF等格式。

### 3. Jupyter Notebook的主要特点

① 编程时具有**语法高亮**、*缩进*、*tab补全*的功能。

② 可直接通过浏览器运行代码，同时在代码块下方展示运行结果。

③ 以富媒体格式展示计算结果。富媒体格式包括：HTML，LaTeX，PNG，SVG等。

④ 对代码编写说明文档或语句时，支持Markdown语法。

⑤ 支持使用LaTeX编写数学性说明。

## 二、安装Jupyter Notebook

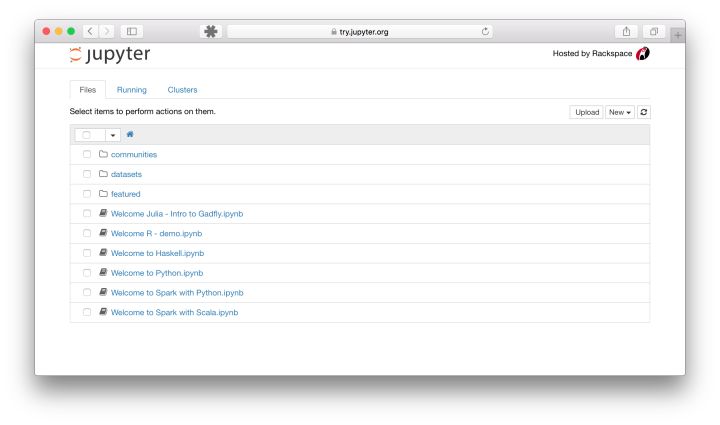
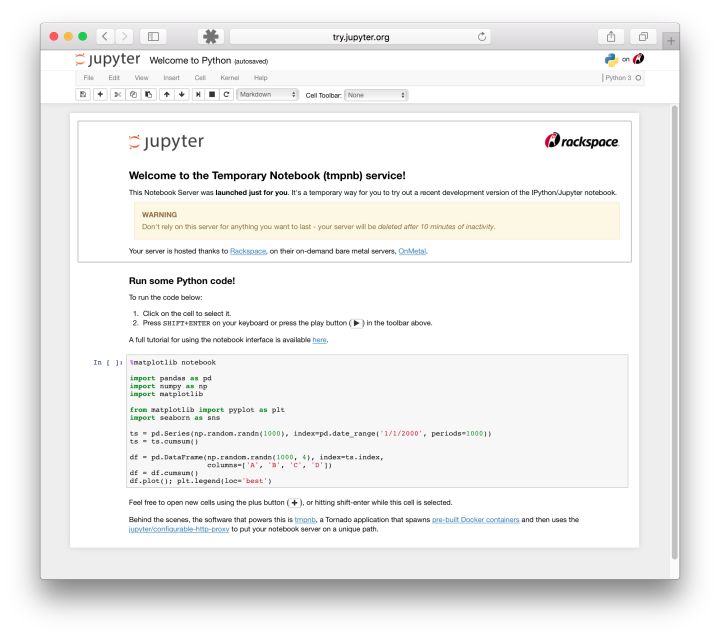
### 0. 先试用，再决定

如果看了以上对Jupyter Notebook的介绍你还是拿不定主意究竟是否适合你，那么不要担心，你可以先**免安装试用体验**一下，[戳这里](https://link.zhihu.com/?target=https%3A//try.jupyter.org/)，然后再做决定。

值得注意的是，官方提供的同时试用是有限的，如果你点击链接之后进入的页面如下图所示，那么不要着急，过会儿再试试看吧。

试用满线

如果你足够幸运，那么你将看到如下界面，就可以开始体验啦。

主界面编辑页面

### 1. 安装

#### ① 安装前提

安装Jupyter Notebook的前提是需要安装了Python（3.3版本及以上，或2.7版本）。

#### ② 使用Anaconda安装

如果你是小白，那么建议你通过安装Anaconda来解决Jupyter Notebook的安装问题，因为Anaconda已经自动为你安装了Jupter Notebook及其他工具，还有python中超过180个科学包及其依赖项。

你可以通过进入Anaconda的[官方下载页面](https://link.zhihu.com/?target=https%3A//www.anaconda.com/download/%23macos)自行选择下载；如果你对阅读**英文文档**感到头痛，或者对**安装步骤**一无所知，甚至也想快速了解一下**什么是Anaconda**，那么可以前往我的另一篇文章：

[豆豆：Anaconda介绍、安装及使用教程​zhuanlan.zhihu.com](https://zhuanlan.zhihu.com/p/32925500)

你想要的，都在里面！

常规来说，安装了Anaconda发行版时已经自动为你安装了Jupyter Notebook的，但如果没有自动安装，那么就在终端（Linux或macOS的“终端”，Windows的“Anaconda Prompt”，以下均简称“终端”）中输入以下命令安装：

conda install jupyter notebook

#### ③ 使用pip命令安装

如果你是有经验的Python玩家，想要尝试用pip命令来安装Jupyter  
Notebook，那么请看以下步骤吧！接下来的命令都输入在终端当中的噢！

**1. 把pip升级到最新版本**

* Python 3.x

pip3 install --upgrade pip

* Python 2.x

pip install --upgrade pip

* 注意：老版本的pip在安装Jupyter Notebook过程中或面临依赖项无法同步安装的问题。因此**强烈建议**先把pip升级到最新版本。

**2. 安装Jupyter Notebook**

* Python 3.x

pip3 install jupyter

* Python 2.x

pip install jupyter

## 三、运行Jupyter Notebook

### 0. 帮助

如果你有任何jupyter notebook命令的疑问，可以考虑查看官方帮助文档，命令如下：

jupyter notebook --help

或

jupyter notebook -h

### 1. 启动

#### ① 默认端口启动

在终端中输入以下命令：

jupyter notebook

执行命令之后，在终端中将会显示一系列notebook的服务器信息，同时浏览器将会自动启动Jupyter Notebook。

启动过程中终端显示内容如下：

$ jupyter notebook

[I 08:58:24.417 NotebookApp] Serving notebooks from local directory: /Users/catherine

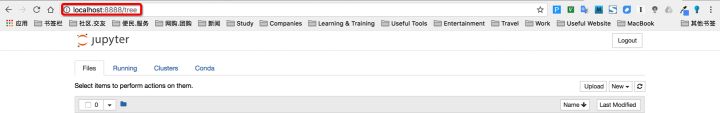
[I 08:58:24.417 NotebookApp] 0 active kernels

[I 08:58:24.417 NotebookApp] The Jupyter Notebook is running at: http://localhost:8888/

[I 08:58:24.417 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).

* 注意：之后在Jupyter Notebook的所有操作，都请保持终端**不要关闭**，因为一旦关闭终端，就会断开与本地服务器的链接，你将无法在Jupyter Notebook中进行其他操作啦。

浏览器地址栏中默认地将会显示：http://localhost:8888。其中，“localhost”指的是本机，“8888”则是端口号。

URL

如果你**同时**启动了多个Jupyter Notebook，由于默认端口“8888”被占用，因此地址栏中的数字将从“8888”起，每多启动一个Jupyter Notebook数字就加1，如“8889”、“8890”……

#### ② 指定端口启动

如果你想自定义端口号来启动Jupyter Notebook，可以在终端中输入以下命令：

jupyter notebook --port <port\_number>

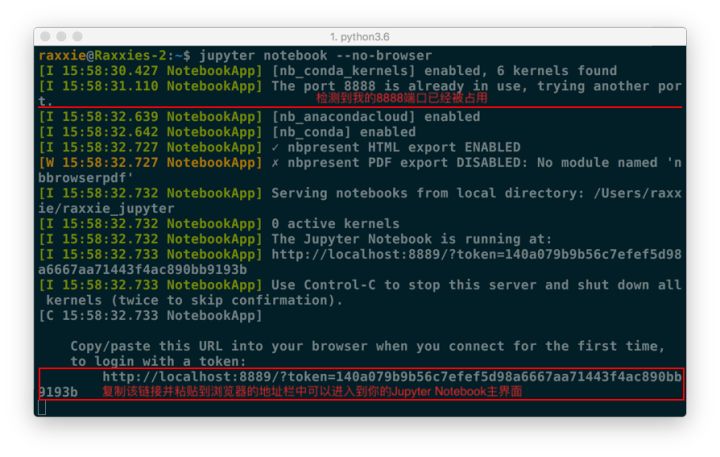
其中，“<port\_number>”是自定义端口号，直接以数字的形式写在命令当中，数字两边不加尖括号“<>”。如：jupyter notebook --port 9999，即在端口号为“9999”的服务器启动Jupyter Notebook。

#### ③ 启动服务器但不打开浏览器

如果你只是想启动Jupyter Notebook的服务器但不打算立刻进入到主页面，那么就无需立刻启动浏览器。在终端中输入：

jupyter notebook --no-browser

此时，将会在终端显示启动的服务器信息，并在服务器启动之后，显示出打开浏览器页面的链接。当你需要启动浏览器页面时，只需要复制链接，并粘贴在浏览器的地址栏中，轻按回车变转到了你的Jupyter Notebook页面。

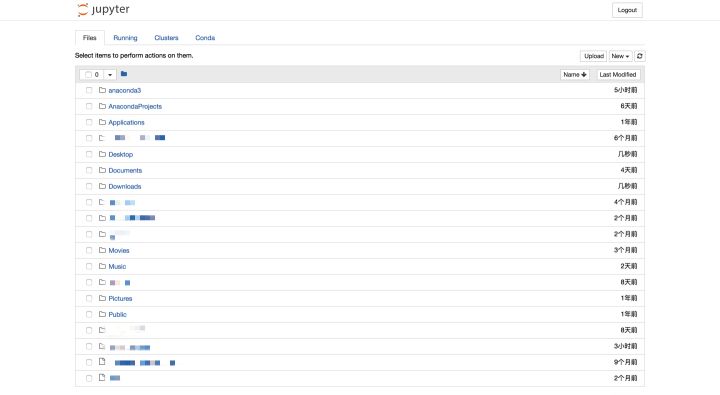


例图中由于在完成上面内容时我同时启动了多个Jupyter Notebook，因此显示我的“8888”端口号被占用，最终分配给我的是“8889”。

### 2. 主页面

#### ① 主页面内容

当执行完启动命令之后，浏览器将会进入到Notebook的主页面，如下图所示。

默认主页面

如果你的主页面里边的文件夹跟我的不同，或者你在疑惑为什么首次启动里边就已经有这么多文件夹，不要担心，这里边的文件夹全都是你的家目录里的目录文件。你可以在终端中执行以下2步来查看：

① cd 或 cd - 或 cd ~ 或cd /Users/<user\_name>

* 这个命令将会进入你的家目录。
* “<user\_name>” 是用户名。用户名两边不加尖括号“<>”。

② ls

* 这个命令将会展示你家目录下的文件。

#### ② 设置Jupyter Notebook文件存放位置

如果你不想把今后在Jupyter Notebook中编写的所有文档都直接保存在家目录下，那你需要修改Jupyter Notebook的文件存放路径。

##### ⑴ 创建文件夹/目录

* Windows用户在想要存放Jupyter Notebook文件的**磁盘**中**新建文件夹**并为该文件夹命名；双击进入该文件夹，然后复制地址栏中的路径。
* Linux/macOS用户在想要存放Jupyter Notebook文件的位置**创建目录**并为目录命名，命令为：mkdir <directory\_name>；进入目录，命令为：cd <directory\_name>；查看目录的路径，命令为：pwd；复制该路径。
* 注意：“<directory\_name>”是自定义的目录名。目录名两边不加尖括号“<>”。

##### ⑵ 配置文件路径

* 一个便捷获取配置文件所在路径的命令：

jupyter notebook --generate-config

* 注意： 这条命令虽然可以用于查看配置文件所在的路径，但主要用途是是否将这个路径下的配置文件**替换**为**默认配置文件**。 如果你是第一次查询，那么**或许**不会出现下图的提示；若文件已经存在或被修改，使用这个命令之后会出现询问“Overwrite /Users/raxxie/.jupyter/jupyter\_notebook\_config.py with default config? [y/N]”，即“用默认配置文件覆盖此路径下的文件吗？”，如果按“y”，则完成覆盖，那么之前所做的修改都将失效；如果只是为了查询路径，那么一定要输入“N”。

配置文件所在路径

常规的情况下，Windows和Linux/macOS的配置文件所在路径和配置文件名如下所述：

* Windows系统的配置文件路径：C:\Users\<user\_name>\.jupyter\
* Linux/macOS系统的配置文件路径：/Users/<user\_name>/.jupyter/ 或 ~/.jupyter/
* 配置文件名：jupyter\_notebook\_config.py

注意：

① “<user\_name>”为你的用户名。用户名两边不加尖括号“<>”。

② Windows和Linux/macOS系统的配置文件存放路径其实是相同的，只是系统不同，表现形式有所不同而已。

③ Windows和Linux/macOS系统的配置文件也是相同的。文件名以“.py”结尾，是Python的可执行文件。

④ 如果你不是通过一步到位的方式前往配置文件所在位置，而是一层一层进入文件夹/目录的，那么当你进入家目录后，用ls命令会发现找不到“.jupyter”文件夹/目录。这是因为凡是以“.”开头的目录都是隐藏文件，你可以通过ls -a命令查看当前位置下所有的隐藏文件。

##### ⑶ 修改配置文件

* Windows系统的用户可以使用文档编辑工具或IDE打开“jupyter\_notebook\_config.py”文件并进行编辑。常用的文档编辑工具和IDE有记事本、Notepad++、vim、Sublime  
  Text、PyCharm等。其中，vim是没有图形界面的，是一款学习曲线较为陡峭的编辑器，其他工具在此不做使用说明，因为上手相对简单。通过vim修改配置文件的方法请继续往下阅读。
* Linux/macOS系统的用户建议直接通过终端调用vim来对配置文件进行修改。具体操作步骤如下：

⒜ 打开配置文件

打开终端，输入命令：

vim ~/.jupyter/jupyter\_notebook\_config.py

命令详解

执行上述命令后便进入到配置文件当中了。

⒝ 查找关键词

进入配置文件后查找关键词“c.NotebookApp.notebook\_dir”。查找方法如下：

进入配置文件后不要按其他键，用**英文半角**直接输入/c.NotebookApp.notebook\_dir，这时搜索的关键词已在文档中高亮显示了，按回车，光标从底部切换到文档正文中被查找关键词的首字母。

⒞ 编辑配置文件

按**小写i**进入编辑模式，底部出现“--INSERT--”说明成功进入编辑模式。使用方向键把光标定位在第二个单引号上（光标定位在哪个字符，就在这个字符前开始输入），把“⑴ 创建文件夹/目录”步骤中复制的路径粘贴在此处。

⒟ 取消注释

把该行行首的**井号（#）**删除。因为配置文件是Python的可执行文件，在Python中，井号（#）表示注释，即在编译过程中不会执行该行命令，所以为了使修改生效，需要删除井号（#）。

⒠ 保存配置文件

先按esc键，从编辑模式退出，回到命令模式。

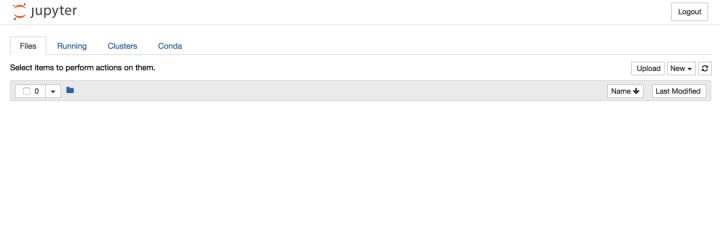
再用**英文半角**直接输入:wq，回车即成功保存且退出了配置文件。

注意：

* **冒号（:）** 一定要有，且也是**英文半角**。
* w：保存。
* q：退出。

⒡ 验证

在终端中输入命令jupyter notebook打开Jupyter Notebook，此时你会看到一个清爽的界面，恭喜！

配置后主页面

⒢ 注意

* 以上所有命令均以**英文半角**格式输入，若有报错，请严格检查这两个条件，**英文**且**半角**。
* 这里仅介绍了vim编辑器修改配置文件的方法，没有对vim编辑器的详细使用进行讲解，所以无需了解vim编辑器的具体使用方法，只需要按照上述步骤一定可以顺利完成修改！
* 推荐有时间和经历时学习一下vim编辑器的使用。这款强大的编辑器将会成为你未来工作中的利器。

## 四、Jupyter Notebook的基本使用

### 1. Files页面

Files页面

此时你的界面当中应该还没有“Conda”和“Nbextensions”类目。不要着急，这两个类目将分别在“五、拓展功能”中的“[1.关联Jupyter Notebook和conda的环境和包——‘nb\_conda’](https://zhuanlan.zhihu.com/p/33105153/edit#conda)”和“[2.Markdown生成目录](https://zhuanlan.zhihu.com/p/33105153/edit#nbextensions)”中安装。

Files页面是用于管理和创建文件相关的类目。

对于现有的文件，可以通过勾选文件的方式，对选中文件进行复制、重命名、移动、下载、查看、编辑和删除的操作。

同时，也可以根据需要，在“New”下拉列表中选择想要创建文件的环境，进行创建“ipynb”格式的笔记本、“txt”格式的文档、终端或文件夹。如果你创建的环境没有在下拉列表中显示，那么你需要依次前往“五、拓展功能”中的“[1.关联Jupyter Notebook和conda的环境和包——‘nb\_conda’](https://zhuanlan.zhihu.com/p/33105153/edit#conda)”和“[六、增加内核——‘ipykernel’](https://zhuanlan.zhihu.com/p/33105153/edit#ipykernel)”中解决该问题。

#### ① 笔记本的基本操作

笔记本页面注解图

上图展示的是笔记本的基本结构和功能。根据图中的注解已经可以解决绝大多数的使用问题了！

工具栏的使用如图中的注解一样直观，在此不过多解释。需要特别说明的是“单元格的状态”，有Code，Markdown，Heading，Raw NBconvert。其中，最常用的是前两个，分别是代码状态，Markdown编写状态。Jupyter Notebook已经取消了Heading状态，即标题单元格。取而代之的是Markdown的一级至六级标题。而Raw NBconvert目前极少用到，此处也不做过多讲解。

菜单栏涵盖了笔记本的所有功能，即便是工具栏的功能，也都可以在菜单栏的类目里找到。然而，并不是所有功能都是常用的，比如Widgets，Navigate。Kernel类目的使用，主要是对内核的操作，比如中断、重启、连接、关闭、切换内核等，由于我们在创建笔记本时已经选择了内核，因此切换内核的操作便于我们在使用笔记本时切换到我们想要的内核环境中去。由于其他的功能相对比较常规，根据图中的注解来尝试使用笔记本的功能已经非常便捷，因此不再做详细讲解。

#### ② 笔记本重命名的两种方式

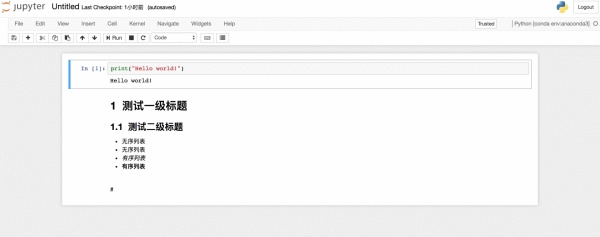
⑴ 笔记本内部重命名

在使用笔记本时，可以直接在其内部进行重命名。在左上方“Jupyter”的图标旁有程序默认的标题“Untitled”，点击“Untitled”然后在弹出的对话框中输入自拟的标题，点击“Rename”即完成了重命名。

⑵ 笔记本外部重命名

若在使用笔记本时忘记了重命名，且已经保存并退出至“Files”界面，则在“Files”界面勾选需要重命名的文件，点击“Rename”然后直接输入自拟的标题即可。

⑶ 演示



笔记本重命名演示图

### 2. Running页面

Running页面主要展示的是当前正在运行当中的终端和“ipynb”格式的笔记本。若想要关闭已经打开的终端和“ipynb”格式的笔记本，仅仅关闭其页面是无法彻底退出程序的，需要在Running页面点击其对应的“Shutdown”。更多关闭方法可以查阅“八、关闭和退出”中的“[1.关闭笔记本和终端](https://zhuanlan.zhihu.com/p/33105153/edit#quit)”。



Running页面功能演示图

### 3. Clusters页面

Clusters tab is now provided by IPython parallel. See '[IPython parallel](https://link.zhihu.com/?target=https%3A//github.com/ipython/ipyparallel)' for  
installation details.

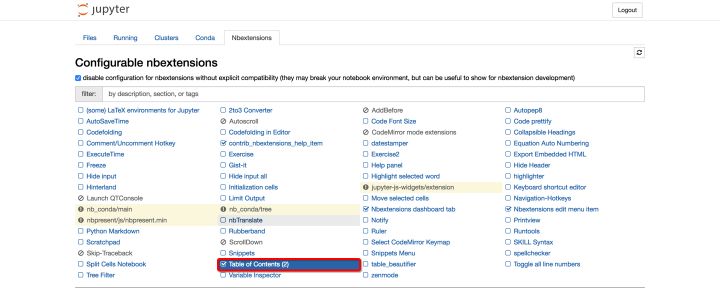
Clusters类目现在已由IPython parallel对接，且由于现阶段使用频率较低，因此在此不做详细说明，想要了解更多可以访问[IPython parallel的官方网站](https://link.zhihu.com/?target=https%3A//github.com/ipython/ipyparallel)。

### 4. Conda页面

Conda页面主要是Jupyter Notebook与Conda关联之后对Conda环境和包进行直接操作和管理的页面工具。详细信息请直接查阅“五、拓展功能”中的“[1.关联Jupyter Notebook和conda的环境和包——‘nb\_conda’](https://zhuanlan.zhihu.com/p/33105153/edit#conda)”。这是目前使用Jupyter Notebook的必备环节，因此请务必查阅。

### 5. Nbextensions页面

Nbextensions页面提供了多个Jupyter Notebook的插件，使其功能更加强大。该页面中主要使用的插件有nb\_conda，nb\_present，Table of Contents(2)。这些功能我们无需完全掌握，也无需安装所有的扩展功能，根据本文档提供的学习思路，我们只需要安装Talbe of Contents(2)即可，该功能可为Markdown文档提供目录导航，便于我们编写文档。该安装指导请查阅“五、拓展功能”中的“[2.Markdown生成目录](https://zhuanlan.zhihu.com/p/33105153/edit#nbextensions)”。

Nbextensions页面

## 五、拓展功能

### 1. 关联Jupyter Notebook和conda的环境和包——“nb\_conda”☆

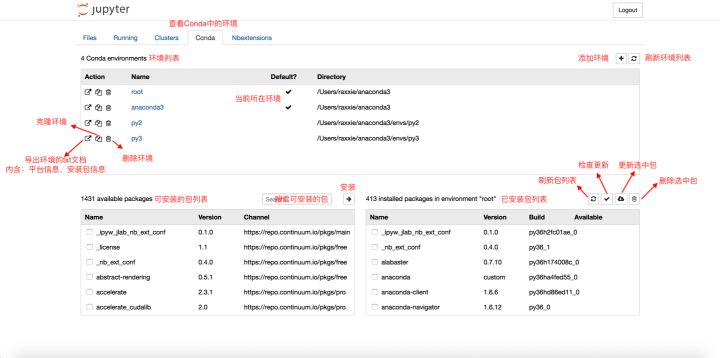
① 安装

conda install nb\_conda

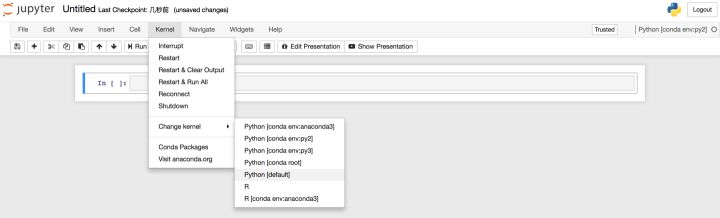
执行上述命令能够将你conda创建的环境与Jupyter Notebook相关联，便于你在Jupyter Notebook的使用中，在不同的环境下创建笔记本进行工作。

② 使用

* 可以在Conda类目下对conda环境和包进行一系列操作。

Conda页面注解图

* 可以在笔记本内的“Kernel”类目里的“Change  
  kernel”切换内核。

切换内核

③ 卸载

canda remove nb\_conda

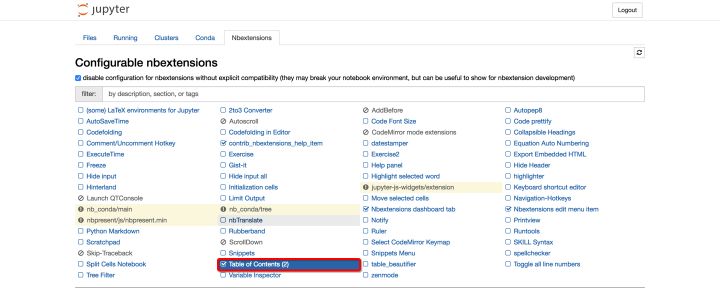
执行上述命令即可卸载nb\_conda包。

### 2. Markdown生成目录

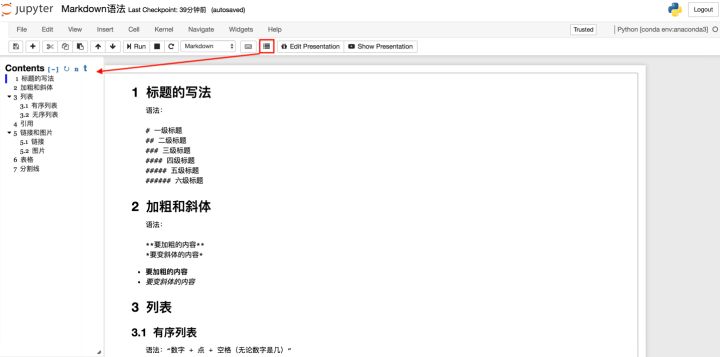
* 不同于有道云笔记的Markdown编译器，Jupyter Notebook无法为Markdown文档通过特定语法添加目录，因此需要通过安装扩展来实现目录的添加。

conda install -c conda-forge jupyter\_contrib\_nbextensions

* 执行上述命令后，启动Jupyter Notebook，你会发现导航栏多了“Nbextensions”的类目，点击“Nbextensions”，勾选“Table  
  of Contents ⑵”

Nbextensions页面

* 之后再在Jupyter Notebook中使用Markdown，点击下图的图标即可使用啦。

目录

### 3. Markdown在文中设置链接并定位

在使用Markdown编辑文档时，难免会遇到需要在文中设定链接，定位在文档中的其他位置便于查看。因为Markdown可以完美的兼容html语法，因此这种功能可以通过html语法当中“a标签”的索引用法来实现。

语法格式如下：

[添加链接的正文](#自定义索引词)

<a id=自定义索引词>跳转提示</a>

注意：

* 1. 语法格式当中所有的符号均是**英文半角**。
  2. “自定义索引词”最好是英文，较长的词可以用下划线连接。
  3. “a标签”出现在想要被跳转到的文章位置，html标签除了单标签外均要符合“有头（<a>）必有尾（</a>）”的原则。头尾之间的“跳转提示”是可有可无的。
  4. “a标签”中的“id”值即是为正文中添加链接时设定的“自定义索引值”，这里通过“id”的值实现从正文的链接跳转至指定位置的功能。

例：



有跳转提示语



无跳转提示语

### 4. 加载指定网页源代码

① 使用场景

想要在Jupyter Notebook中直接加载指定网站的源代码到笔记本中。

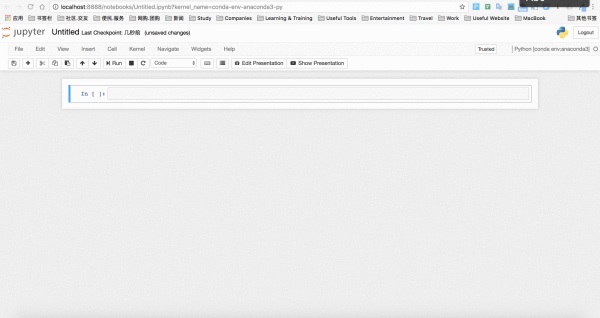
② 方法

执行以下命令:

%load URL

其中，URL为指定网站的地址。

③ 例



加载网络代码

### 5. 加载本地Python文件

① 使用场景

想在Jupyter Notebook中加载本地的Python文件并执行文件代码。

② 方法

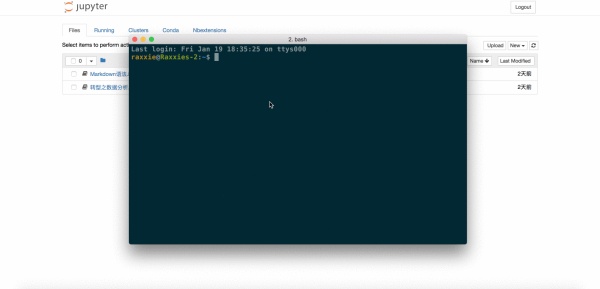
执行以下命令：

%load Python文件的绝对路径

③ 注意

1. Python文件的后缀为“.py”。
2. “%load”后跟的是Python文件的**绝对路径**。
3. 输入命令后，可以按CTRL 回车来执行命令。第一次执行，是将本地的Python文件内容加载到单元格内。此时，Jupyter Notebook会自动将“%load”命令注释掉（即在前边加井号“#”），以便在执行已加载的文件代码时不重复执行该命令；第二次执行，则是执行已加载文件的代码。

④ 例



加载本地Python文件

### 6. 直接运行本地Python文件

① 使用场景

不想在Jupyter Notebook的单元格中加载本地Python文件，想要直接运行。

② 方法

执行命令：

%run Python文件的绝对路径

或

!python3 Python文件的绝对路径

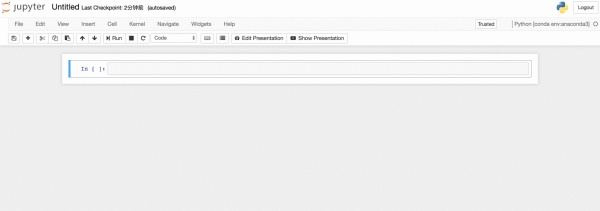
或

!python Python文件的绝对路径

③ 注意

1. Python文件的后缀为“.py”。
2. “%run”后跟的是Python文件的**绝对路径**。
3. “!python3”用于执行Python  
   3.x版本的代码。
4. “!python”用于执行Python  
   2.x版本的代码。
5. “!python3”和“!python”属于 !shell命令 语法的使用，即在Jupyter Notebook中执行shell命令的语法。
6. 输入命令后，可以按 control return 来执行命令，执行过程中将不显示本地Python文件的内容，直接显示运行结果。

④ 例



运行本地Python文件

### 7. 在Jupyter Notebook中获取当前位置

① 使用场景

想要在Jupyter Notebook中获取当前所在位置的**绝对路径。**

② 方法

%pwd

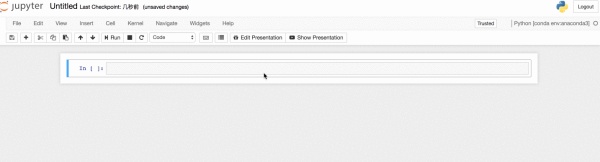
或

!pwd

③ 注意

1. 获取的位置是当前Jupyter Notebook中创建的笔记本所在位置，且该位置为**绝对路径**。
2. “!pwd”属于 !shell命令 语法的使用，即在Jupyter  
   Notebook中执行shell命令的语法。

④ 例



获取当前位置的绝对路径

### 8. 在Jupyter Notebook使用shell命令

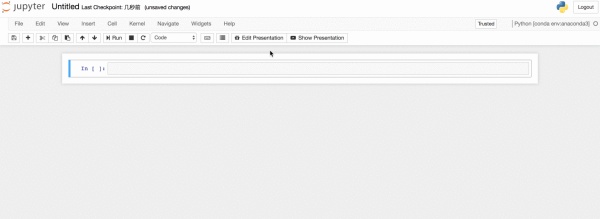
① 方法一——在笔记本的单元格中

⑴ 语法

!shell命令

* 在Jupyter Notebook中的笔记本单元格中用英文感叹号“!”后接shell命令即可执行shell命令。

⑵ 例



Shell命令的使用

② 方法二——在Jupyter Notebook中新建终端

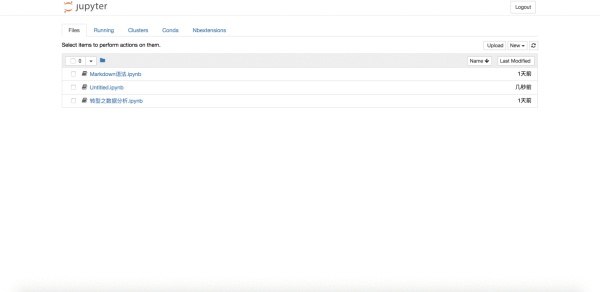
⑴ 启动方法

在Jupyter Notebook主界面，即“File”界面中点击“New”；在“New”下拉框中点击“Terminal”即新建了终端。此时终端位置是在你的家目录，可以通过pwd命令查询当前所在位置的绝对路径。

⑵ 关闭方法

在Jupyter Notebook的“Running”界面中的“Terminals”类目中可以看到正在运行的终端，点击后边的“Shutdown”即可关闭终端。

⑶ 例



笔记本中的终端使用

### 9. 隐藏笔记本输入单元格

① 使用场景

在Jupyter Notebook的笔记本中无论是编写文档还是编程，都有输入（In []）和输出（Out []）。当我们编写的代码或文档使用的单元格较多时，有时我们只想关注输出的内容而暂时不看输入的内容，这时就需要隐藏输入单元格而只显示输出单元格。

② 方法一

⑴ 代码

from IPython.display import display

from IPython.display import HTML

import IPython.core.display as di # Example: di.display\_html('<h3>%s:</h3>' % str, raw=True)

# 这行代码的作用是：当文档作为HTML格式输出时，将会默认隐藏输入单元格。

di.display\_html('<script>jQuery(function() {if (jQuery("body.notebook\_app").length == 0) { jQuery(".input\_area").toggle(); jQuery(".prompt").toggle();}});</script>', raw=True)

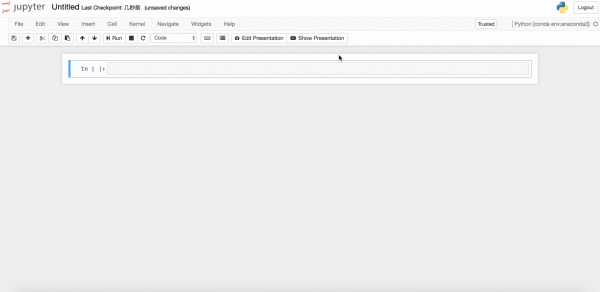
# 这行代码将会添加“Toggle code”按钮来切换“隐藏/显示”输入单元格。

di.display\_html('''<button onclick="jQuery('.input\_area').toggle(); jQuery('.prompt').toggle();">Toggle code</button>''', raw=True)

在笔记本第一个单元格中输入以上代码，然后执行，即可在该文档中使用“隐藏/显示”输入单元格功能。

* 缺陷：此方法不能很好的适用于Markdown单元格。

⑵ 例



方法一：隐藏/显示输入单元格

③ 方法二

⑴ 代码

from IPython.display import HTML

HTML('''<script>

code\_show=true;

function code\_toggle() {

if (code\_show){

$('div.input').hide();

} else {

$('div.input').show();

}

code\_show = !code\_show

}

$( document ).ready(code\_toggle);

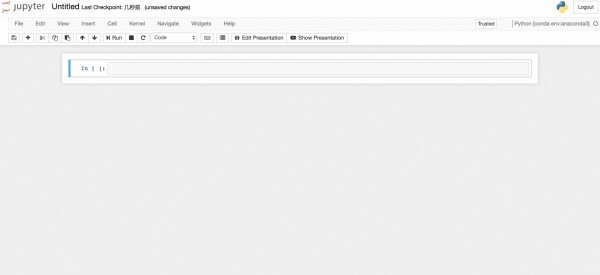
</script>

<form action="javascript:code\_toggle()"><input type="submit" value="Click here to toggle on/off the raw code."></form>''')

在笔记本第一个单元格中输入以上代码，然后执行，即可在该文档中使用“隐藏/显示”输入单元格功能。

* 缺陷：此方法不能很好的适用于Markdown单元格。

⑵ 例



方法二：隐藏/显示输入单元格

### 10. 魔术命令

由于目前暂时用不到过多的魔术命令，因此暂时先参考[官网的文档](https://link.zhihu.com/?target=http%3A//ipython.readthedocs.io/en/stable/interactive/magics.html)。

## 六、增加内核——“ipykernel” ☆

### 1. 使用场景

* 场景一：同时用不同版本的Python进行工作，在Jupyter Notebook中无法切换，即“New”的下拉菜单中无法使用需要的环境。
* 场景二：创建了不同的虚拟环境（或许具有相同的Python版本但安装的包不同），在Jupyter Notebook中无法切换，即“New”的下拉菜单中无法使用需要的环境。

接下来将分别用“命令行模式”和“图形界面模式”来解决以上两个场景的问题。顾名思义，“命令行模式”即在终端中通过执行命令来一步步解决问题；“图形界面模式”则是通过在Jupyter Notebook的网页中通过鼠标点击的方式解决上述问题。

其中，“图形界面模式”的解决方法相对比较简单快捷，如果对于急于解决问题，不需要知道运行原理的朋友，可以直接进入“3. 解决方法之图形界面模式”来阅读。

“命令行模式”看似比较复杂，且又划分了使用场景，但通过这种方式来解决问题可以更好的了解其中的工作原理，比如，每进行一步操作对应的命令是什么，而命令的执行是为了达到什么样的目的，这些可能都被封装在图形界面上的一个点击动作来完成了。对于想更深入了解其运作过程的朋友，可以接着向下阅读。

### 2. 解决方法之命令行模式

① 同时使用不同版本的Python

#### ⑴ 在Python 3中创建Python 2内核

⒜ pip安装

* 首先安装Python 2的ipykernel包。

python2 -m pip install ipykernel

* 再为**当前用户**安装Python 2的内核（ipykernel）。

python2 -m ipykernel install --user

* 注意：“--user”参数的意思是针对当前用户安装，而非系统范围内安装。

⒝ conda安装

* 首先创建Python版本为2.x且具有ipykernel的新环境，其中“<env\_name>”为自定义环境名，环境名两边不加尖括号“<>”。

conda create -n <env\_name> python=2 ipykernel

* 然后切换至新创建的环境。

Windows: activate <env\_name>

Linux/macOS: source activate <env\_name>

* 为**当前用户**安装Python 2的内核（ipykernel）。

python2 -m ipykernel install --user

* 注意：“--user”参数的意思是针对当前用户安装，而非系统范围内安装。

#### ⑵ 在Python 2中创建Python 3内核

⒜ pip安装

* 首先安装Python 3的ipykernel包。

python3 -m pip install ipykernel

* 再为**当前用户**安装Python 2的内核（ipykernel）。

python3 -m ipykernel install --user

* 注意：“--user”参数的意思是针对当前用户安装，而非系统范围内安装。

⒝ conda安装

* 首先创建Python版本为3.x且具有ipykernel的新环境，其中“<env\_name>”为自定义环境名，环境名两边不加尖括号“<>”。

conda create -n <env\_name> python=3 ipykernel

* 然后切换至新创建的环境。

Windows: activate <env\_name>

Linux/macOS: source activate <env\_name>

* 为**当前用户**安装Python 3的内核（ipykernel）。

python3 -m ipykernel install --user

* 注意：“--user”参数的意思是针对当前用户安装，而非系统范围内安装。

② 为不同环境创建内核

⑴ 切换至需安装内核的环境

Windows: activate <env\_name>

Linux/macOS: source activate <env\_name>

* 注意：“<env\_name>”是需要安装内核的环境名称，环境名两边不加尖括号“<>”。

⑵ 检查该环境是否安装了ipykernel包

conda list

执行上述命令查看当前环境下安装的包，若没有安装ipykernel包，则执行安装命令；否则进行下一步。

conda install ipykernel

⑶ 为当前环境下的当前用户安装Python内核

* 若该环境的Python版本为2.x，则执行命令：

python2 -m ipykernel install --user --name <env\_name> --display-name "<notebook\_name>"

* 若该环境的Python版本为3.x，则执行命令：

python3 -m ipykernel install --user --name <env\_name> --display-name "<notebook\_name>"

* 注意:

1. “<env\_name>”为当前环境的环境名称。环境名两边不加尖括号“<>”。

2. “<notebook\_name>”为自定义显示在Jupyter Notebook中的名称。名称两边不加尖括号“<>”，但**双引号必须加**。

3. “--name”参数的值，即“<env\_name>”是Jupyter内部使用的，其目录的存放路径为~/Library/Jupyter/kernels/。如果定义的名称在该路径已经存在，那么将自动覆盖该名称目录的内容。

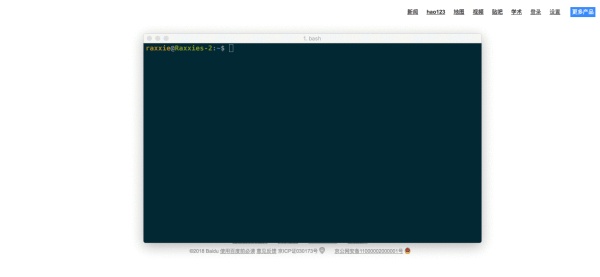
4. “--display-name”参数的值是显示在Jupyter Notebook的菜单中的名称。

⑷ 检验

使用命令jupyter notebook启动Jupyter Notebook；在“Files”下的“New”下拉框中即可找到你在第⑶步中的自定义名称，此时，你便可以尽情地在Jupyter Notebook中切换环境，在不同的环境中创建笔记本进行工作和学习啦！

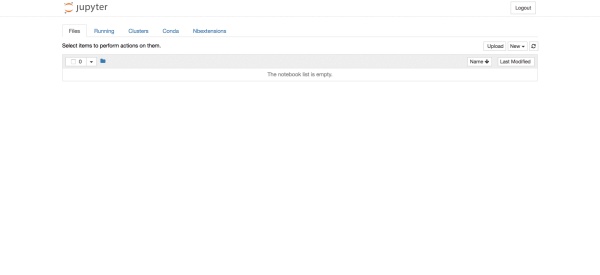
3. 解决方法之图形界面模式

① 你创建了一个新的环境，但却发现在Jupyter Notebook的“New”中找不到这个环境，无法在该环境中创建笔记本。



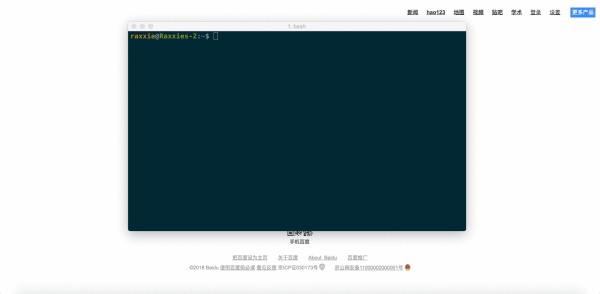
问题发现

② 进入Jupyter Notebook → Conda → 在“Conda  
environment”中点击你要添加ipykernel包的环境 → 左下方搜索框输入“ipykernel”  
→ 勾选“ipykernel” → 点击搜索框旁的“→”箭头 → 安装完毕 → 右下方框内找到“ipykernel”说明已经安装成功



解决方法

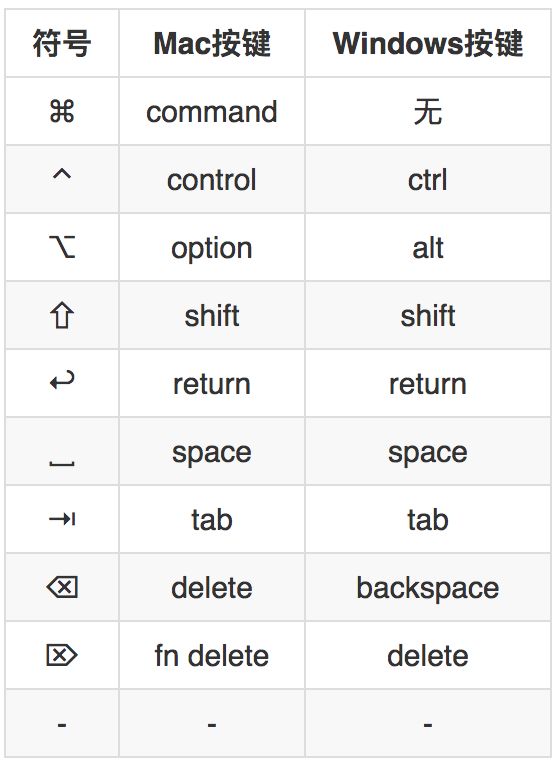
③ 在终端control c关闭Jupyter Notebook的服务器然后重启Jupyter Notebook，在“File”的“New”的下拉列表里就可以找到你的环境啦。



验证

## 七、Jupyter Notebook快捷键

### 1. Mac与Windows特殊按键对照表

Mac和Windows特殊按键对照表

### 2. Jupyter Notebook笔记本的两种模式

① 命令模式

* 命令模式将键盘命令与Jupyter Notebook笔记本命令相结合，可以通过键盘不同键的组合运行笔记本的命令。
* 按esc键进入命令模式。
* 命令模式下，单元格边框为灰色，且左侧边框线为蓝色粗线条。

命令模式

② 编辑模式

* 编辑模式使用户可以在单元格内编辑代码或文档。
* 按enter或return键进入编辑模式。
* 编辑模式下，单元格边框和左侧边框线均为绿色。

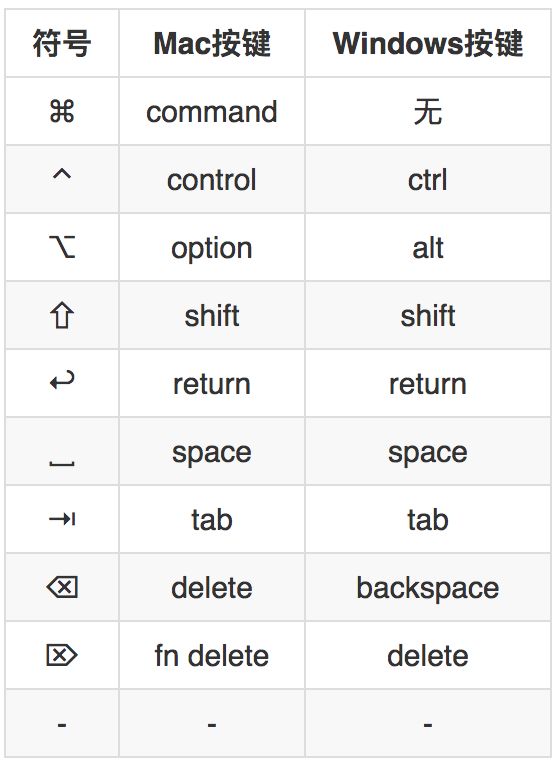
编辑模式

### 3. 两种模式的快捷键

① 命令模式

命令模式快捷键

② 编辑模式

编辑模式快捷键

### 4. 查看和编辑快捷键

① 查看快捷键

① 进入Jupyter Notebook主界面“File”中。

② 在“New”的下拉列表中选择环境创建一个笔记本。

③ 点击“Help”。

④ 点击“Keyboard Shortcuts”。

② 编辑快捷键

⑴ 方法一

① 进入Jupyter Notebook主界面“File”中。

② 在“New”的下拉列表中选择环境创建一个笔记本。

③ 点击“Help”。

④ 点击“Keyboard Shortcuts”。

⑤ 弹出的对话框中“Command Mode (press Esc to enable)”旁点击“Edit  
Shortcuts”按钮。

⑵ 方法二

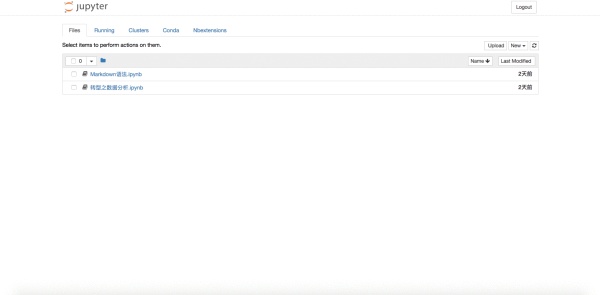
① 进入Jupyter Notebook主界面“File”中。

② 在“New”的下拉列表中选择环境创建一个笔记本。

③ 点击“Help”。

④ 点击“Edit Keyboard Shortcuts”。

③ 例



查看和编辑快捷键

## 八、关闭和退出

### 1. 关闭笔记本和终端

当我们在Jupyter Notebook中创建了终端或笔记本时，将会弹出新的窗口来运行终端或笔记本。当我们使用完毕想要退出终端或笔记本时，仅仅**关闭页面**是无法结束程序运行的，因此我们需要通过以下步骤将其完全关闭。

① 方法一

⑴ 进入“Files”页面。

⑵ 勾选想要关闭的“ipynb”笔记本。正在运行的笔记本其图标为绿色，且后边标有“Running”的字样；已经关闭的笔记本其图标为灰色。

⑶ 点击上方的黄色的“Shutdown”按钮。

⑷ 成功关闭笔记本。

* 注意：此方法只能关闭笔记本，无法关闭终端。

② 方法二

⑴ 进入“Running”页面。

⑵ 第一栏是“Terminals”，即所有正在运行的终端均会在此显示；第二栏是“Notebooks”，即所有正在运行的“ipynb”笔记本均会在此显示。

⑶ 点击想要关闭的终端或笔记本后黄色“Shutdown”按钮。

⑷ 成功关闭终端或笔记本。

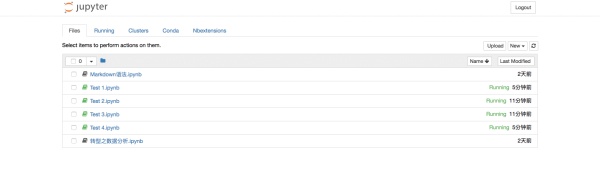
* 注意：此方法可以关闭任何正在运行的终端和笔记本。

③ 注意

⑴ 只有“ipynb”笔记本和终端需要通过上述方法才能使其结束运行。

⑵  
“txt”文档，即“New”下拉列表中的“Text  
File”，以及“Folder”只要关闭程序运行的页面即结束运行，无需通过上述步骤关闭。

④ 演示



关闭笔记本或终端程序

### 2. 退出Jupyter Notebook程序

如果你想退出Jupyter Notebook程序，仅仅通过关闭网页是无法退出的，因为当你打开Jupyter Notebook时，其实是启动了它的服务器。

你可以尝试关闭页面，并打开新的浏览器页面，把之前的地址输进地址栏，然后跳转页面，你会发现再次进入了刚才“关闭”的Jupyter Notebook页面。

如果你忘记了刚才关闭的页面地址，可以在启动Jupyter Notebook的终端中找到地址，复制并粘贴至新的浏览器页面的地址栏，会发现同样能够进入刚才关闭的页面。

因此，想要彻底退出Jupyter Notebook，需要关闭它的服务器。只需要在它启动的终端上按：

* Mac用户：control c
* Windows用户：ctrl c

然后在终端上会提示：“Shutdown this notebook server (y/[n])?”输入y即可关闭服务器，这才是彻底退出了Jupyter Notebook程序。此时，如果你想要通过输入刚才关闭网页的网址进行访问Jupyter Notebook便会看到报错页面。

## 九、参考资料

1.知乎：jupyter notebook 可以做哪些事情？[猴子的回答](https://www.zhihu.com/question/46309360/answer/254638807?utm_source=wechat_session&utm_medium=social)

2. [Jupyter Notebook官方介绍](https://link.zhihu.com/?target=https%3A//jupyter-notebook.readthedocs.io/en/stable/notebook.html)

3. [Anaconda官方下载页面](https://link.zhihu.com/?target=https%3A//www.anaconda.com/download/%23macos)

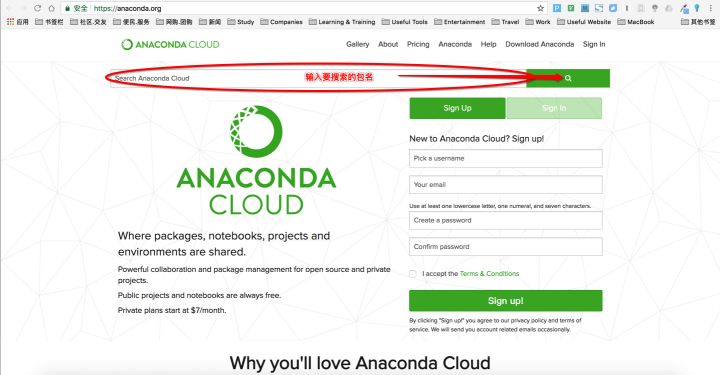
4. [Python·Jupyter Notebook各种使用方法记录](https://link.zhihu.com/?target=http%3A//blog.csdn.net/tina_ttl/article/details/51031113%2321-%25E6%2596%25B9%25E5%25BC%258F%25E4%25B8%2580)

5. [Stack Overflow中有关如何隐藏/显示输入单元格的问题](https://link.zhihu.com/?target=https%3A//stackoverflow.com/questions/27934885/how-to-hide-code-from-cells-in-ipython-notebook-visualized-with-nbviewer)

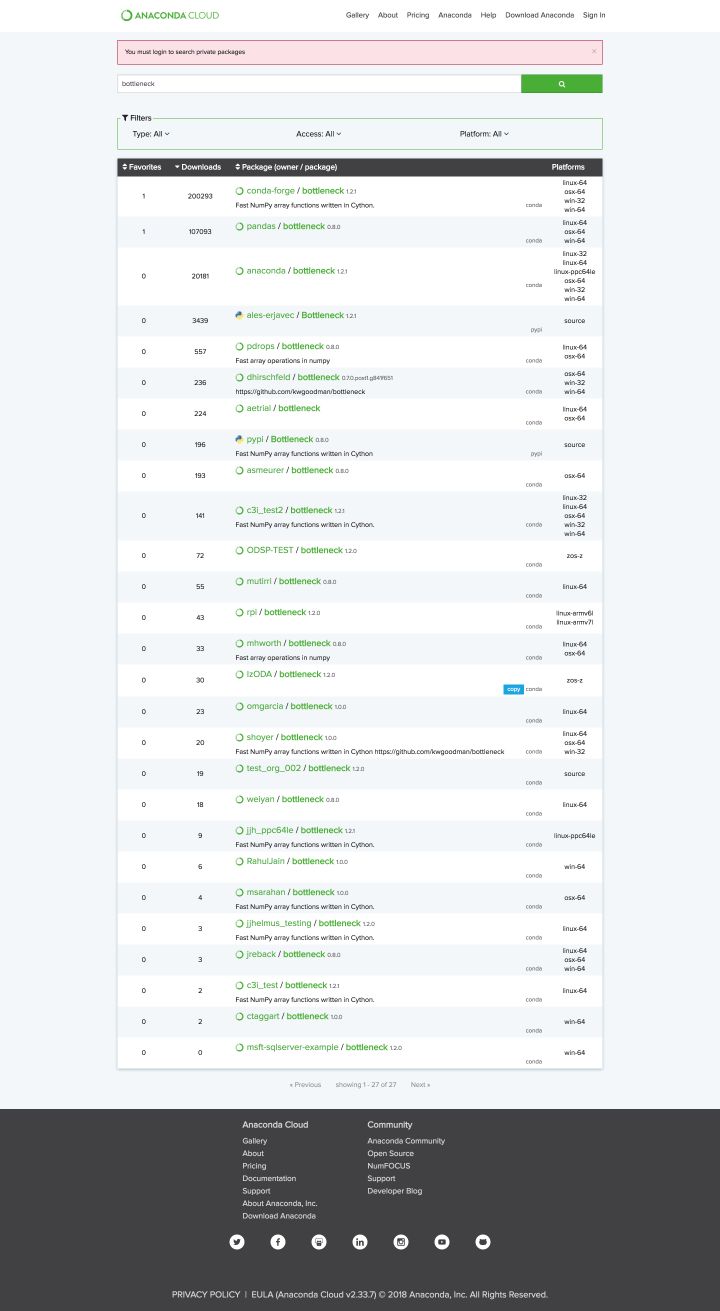
6. [魔术命令官方文档](https://link.zhihu.com/?target=http%3A//ipython.readthedocs.io/en/stable/interactive/magics.html)

7. [Jupyter Notebook 的快捷键](https://link.zhihu.com/?target=http%3A//blog.csdn.net/lawme/article/details/51034543)

8. [Jupyter Notebook官方文档](https://link.zhihu.com/?target=http%3A//jupyter.org/documentation)



3. 搜索结果中有数以千计的包可供选择，此时点击“Downloads”可根据下载量进行排序，最上面的为下载最多的包。（图中以搜索bottleneck包为例）



4. 选择满足需求的包或下载量最多的包，点击包名。

5. 复制“To install this package with conda run:”下方的命令，并粘贴在终端中执行。



6. 完成安装。

**4. 卸载包**

**① 卸载指定环境中的包**

conda remove --name <env\_name> <package\_name>

* 注意：

① ***<env\_name>*** 即卸载包所在指定环境的名称。环境名两边不加尖括号“<>”。

②***<package\_name>*** 即要卸载包的名称。包名两边不加尖括号“<>”。

* 例如： ***conda remove --name python2 pandas*** 即卸载名为“python2”中的pandas包。

**② 卸载当前环境中的包**

conda remove <package\_name>

* 注意：

①***<package\_name>*** 即要卸载包的名称。包名两边不加尖括号“<>”。

② 执行命令后即在当前环境中卸载指定包。

* 例如： ***conda remove pandas*** 即在当前环境中卸载pandas包。

**5. 更新包**

**① 更新所有包**

conda update --all

或

conda upgrade --all

* 建议：在安装Anaconda之后执行上述命令更新Anaconda中的所有包至最新版本，便于使用。

**② 更新指定包**

conda update <package\_name>

或

conda upgrade <package\_name>

* 注意：

①***<package\_name>*** 为指定更新的包名。包名两边不加尖括号“<>”。

② 更新多个指定包，则包名以**空格**隔开，向后排列。如： ***conda update pandas numpy matplotlib*** 即更新pandas、numpy、matplotlib包。

# Python 参考资料：

## 推荐学习网站（中文）：

廖雪峰老师的官方网站

<https://www.liaoxuefeng.com/wiki/0014316089557264a6b348958f449949df42a6d3a2e542c000>

## 官网（英文）

<https://docs.python.org/3/tutorial/index.html>

# Markdown 语言

参考文档：

<http://xianbai.me/learn-md/article/syntax/paragraphs-and-line-breaks.html>