

# Chapter 1. Python 初探

## Python 是什么？



要回答这个问题，首先要知道 Python 不是什么。Python 不是一个软件，它没有图形界面。Stata、SAS 乃至 Eviews 是由专业公司开发的，以盈利为目的而售卖的软件，它们虽然可以用写代码的方式进行操作，但本质上还是一个软件，许多用代码可以完成的，都可以在图形界面用点击鼠标的方式完成。也正因为它们是应用软件，所以能做的事情非常局限。

Python 是一门通用高级编程语言，由 Guido van Rossum 在 1989 年发明。Python 这个单词本身的意思是巨蛇、大蟒。

所谓通用，在于其万能。虽然也可以处理数据、跑计量模型，但 Python 的诞生，绝不只是为了给实证研究者玩耍。除了编写操作系统、编写大型应用软件、编写 APP（这些事一般交给 C、Java 等语言去做），Python 几乎可以做任何事，满足你对计算机的任何幻想。举个例子，Youtube 的网站是用 Python 写的；再举个例子，现在人工智能概念这么火，而大部分深度学习框架都是用 Python 做主力语言的。

所谓高级，是相对于汇编、C、Java 等语言。对于计算机而言，Python 太高级了，高级的东西是非常难懂的，而 C 语言就相对好懂得多。Python 代码的运行速度远不如 C，因为计算机看到 Python 代码需要消化一阵子，而消化起低级语言简直像喝粥一样轻松。但正因为高级，Python 使用者写代码的负担大大简化。同样实现一个计算，用 Python 的代码量为 10 行，用 Java 需要 40 行，用 C 可能就要 100 行。所以，虽然在速度上有明显劣势，让你选择一门语言，你会选择？

在最新的编程语言调查中，Python 的受欢迎程度已经跃居第一。Python 在数据科学中的使用率开始或已经超越 R。最近，一直以高素质实习生项目闻名的高盛集团发布了一份《2017 高盛调查报告》，针对全球 2500 名在高盛的夏季实习生调查，当问到你认为“哪个语言在未来会更重要”时，在被调查的全球 2500 名 80、90 后优秀年轻人中，72%选了 Python，而排第二的是我们的汉语。

## 为什么学 Python？

用一句话回答这个问题：人生苦短，学 Python 吧。而展开来说，Python 拥有太多优点。

首先它易学。Python 的难度比 SAS 低，比 Stata 难一点点。

第二，Python 的扩展模块（Module）多且质量高。简单的 import 可以实现 Python plus everything。社会科学研究者常用的文本分析、机器学习、爬虫等，都有功能强大的第三方模块做支持，而不用我们自己去造轮子。

第三，Python 的社区非常完善，这意味着，我们所遇到的问题，基本都能在网上找到解

决方案。Python 的官方网站上就有各种教学资源，满足从入门到精通的需求。另外，Stackoverflow 等知名社区不乏与 Python 相关的精彩问答。

第四，在对开发效率和运算效率做取舍时，大部分人都会选择 Python。我们宁愿让计算机多跑一小时，也不愿牺牲自己多写一千行代码。况且，在大部分应用场景中，运算速度的差距并非瓶颈。例如，在使用爬虫时，最耗时的部分在于网络数据传输，这对于任何编程语言，都是公平的。此外，尽管速度慢是 Python 的缺点，但在很多情况下，尤其是在使用高度优化的第三方模块时，表面上我们在用 Python，实际上底层运行的语言是 C。换句话说，聪明的开发者们倾向于用更快的语言去实现很多耗时耗电的计算，只不过最终封装成 Python 模块供我们使用。

最后，不得不提，Python 的语言哲学是简洁、规范、灵巧，由此衍生出的一个形容词是 Pythonic。它的优美需要我们在使用过程中体会。

## 怎么学 Python?

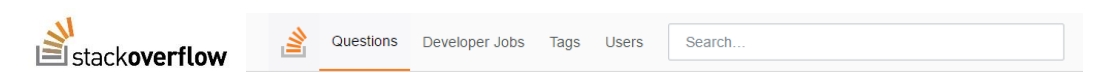
之前说了，Python 简单易学。入门 Python 是很快的，但和真正会用 Python 是两码事。我最早接触 Python 是在 edx 平台上，跟了 MIT 推出的一门课——6.00.1x: Introduction to Computer Science and Programming Using Python。在大概刷完这门网课之后，我完全不知道 Python 能用来做什么。直到有一天，我琢磨如何把 PDF 转化成 txt 格式，看了一波网上的现成代码以后，仿佛任督二脉被打通。这成了我学习 Python 的分水岭。

从我的经历来看，学 Python 要在干中学。在任务的驱动下，更能化抽象的符号为具象的目标。所以，在编写这个入门教程时，我希望在重新组织现有资源的基础之上，把我当时摸索 Python 的情景还原。尤其在编写最后几章时，我会尽可能地复现大家可能面临的任务场景。我也会把当初走了很多弯路才找到的“好东西”，直接呈现给大家。

当然，本人水平有限。干这事儿大半出于情怀，我觉得从零基础入门 Python 的视角出发，去编写一个 Python 教程，本身是一件很有意思、也很有意义的事情。如果要向专家们学习，有更好的资源。现在我把我所接触的优秀 Python 入门资源罗列如下：

类型	资源名	简评
网络课程	<a href="#">6.00.1x: Introduction to Computer Science and Programming Using Python</a>	MIT 在 edx 平台推出的课程。本质上是一门计算机科学入门课，讲了一些基础的 CS 概念和算法，在这个过程中顺带教了 Python。
书籍	Learn Python the Hard Way, 中文名 <a href="#">《“笨办法”学 Python》</a>	用非常笨的办法把 Python 的入门知识学烂。
官方文档	<a href="#">The Python Tutorial</a>	官方出品，所以面面俱到。
在线编程	<a href="#">Codecademy-Python Track</a>	在线编程，就当是打游戏了。
博客	<a href="#">廖雪峰的官方网站</a>	中文 Python 第一博客。

在学习的过程中，遇到问题或困难是不可避免的。请利用好各类搜索引擎。搜索的技巧是把 Python 列为第一个词，后面跟你想的关键词，如“python 列表排序”、“python how to install a module using pip”。推荐使用 Google 查询英文关键词。在这里隆重推荐 Stack Overflow (<https://stackoverflow.com/>)，这是一个 IT 问答社区，你想了解的任何问题(不局限于 Python)基本上都能在这儿找到解答。



最后，在学习心态上，千万不要害怕犯错。本人就是因为犯了无数次错误，把能踩的雷踩遍了，才最终成长为 debug 小能手。一般情况下，长度超过二十行的代码一次性运行成功的可能性微乎其微，总是需要各种调试。不夸张地说，如果一次性成功不报错，程序员的本能反应是“会不会哪里出 bug 了”。本教程将有专门的章节讲解这方面的内容。

## 安装 Python

安装 Python 之前，说说 Python 的版本问题。Python 2 和 Python 3 在基础语法上仅有细微差异，但在很多方面又像是完全不同的语言。它们是不能兼容的。现在主流的 Python 发行版是 Python 2.7 和 Python 3.5（及更新的版本）。Python 3 在很多方面都展现出了优越性，尤其是 Python 2 最让人头疼的字符编码问题。官方对于 Python 2 的支持，已进入倒计时阶段。毫无疑问，Python 3 是我们的首选。

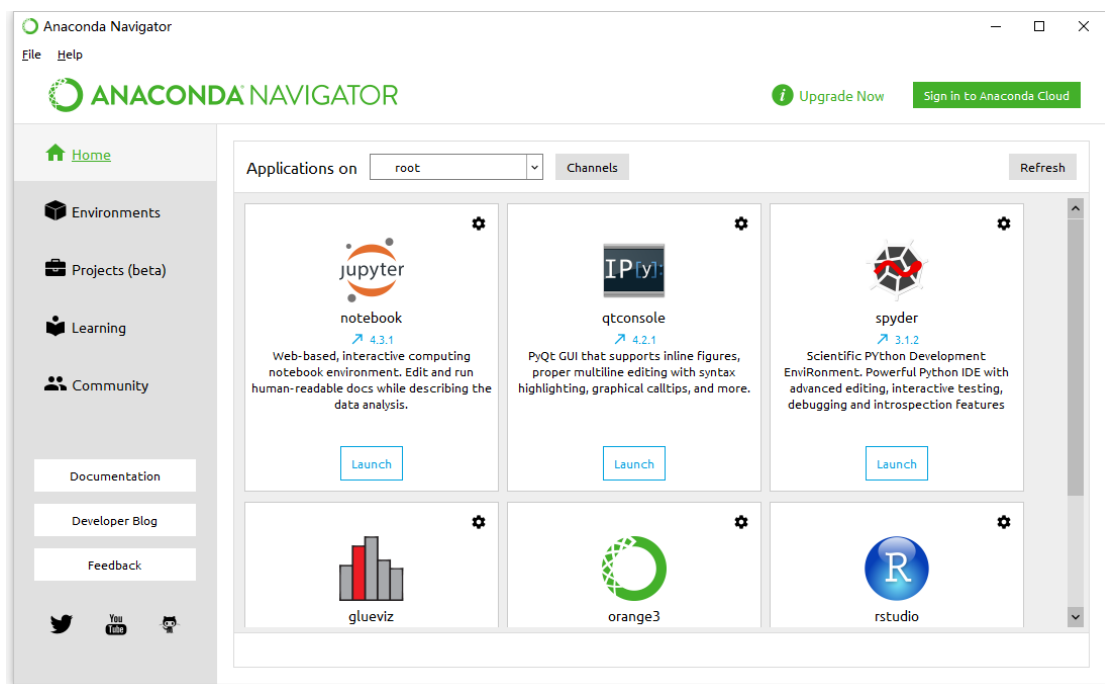
Python 本身的安装包应该只有三十兆左右，可以在 <https://www.python.org/downloads/> 下载到。在这里，我推荐大家安装 Anaconda。Anaconda 是围绕 Python 打造的科学计算环境。除了 Python，还集成了交互式网页编程环境 Jupyter Notebook 和著名 IDE（集成开发环境）Spyder。此外，集成了各种常用第三方模块，如著名的机器学习库 Scikit-Learn，免去逐个下载安装的烦恼。事实上，一些常用模块在纯手工安装的情况下，出错是大概率的事。

### Anaconda installer archive

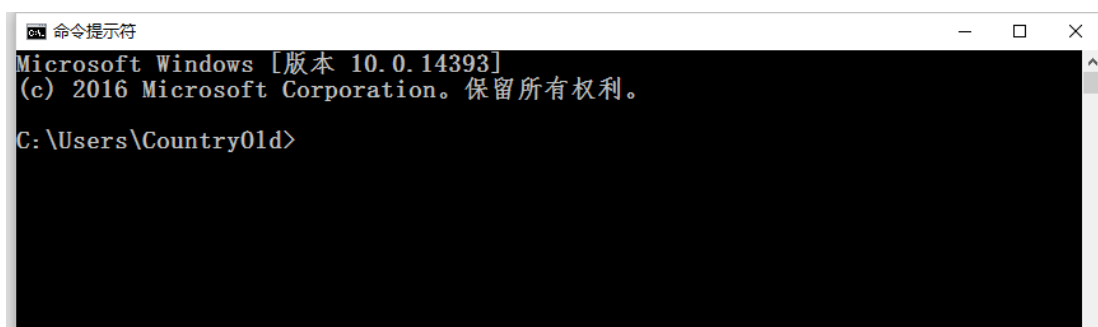
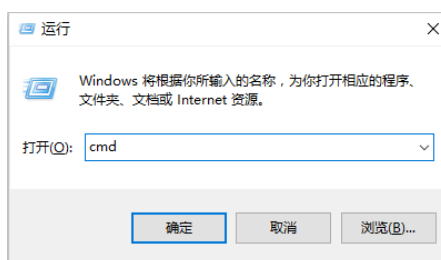
Filename	Size	Last Modified	MD5
<a href="#">Anaconda2-5.0.0-Linux-ppc64le.sh</a>	282.3M	2017-09-26 16:25:07	157890d591c61a9b511f8452476d6d19
<a href="#">Anaconda2-5.0.0-Linux-x86.sh</a>	411.4M	2017-09-26 14:48:02	a574e495c157d59bf4ec337fa4f72ddd
<a href="#">Anaconda2-5.0.0-Linux-x86_64.sh</a>	505.7M	2017-09-26 14:37:21	2272857fcf773fc75a1bc49f6d507a48
<a href="#">Anaconda2-5.0.0-MacOSX-x86_64.pkg</a>	561.3M	2017-09-26 16:25:08	8a2bbf7eb66290eb0bc82963056fb96c
<a href="#">Anaconda2-5.0.0-MacOSX-x86_64.sh</a>	485.3M	2017-09-26 16:25:09	b8d555fae2b4994f1094c2da85c7e9a4
<a href="#">Anaconda2-5.0.0-Windows-x86.exe</a>	402.2M	2017-09-26 16:25:09	bd3ed48229db828cef4c6b371a8759d1
<a href="#">Anaconda2-5.0.0-Windows-x86_64.exe</a>	498.2M	2017-09-26 14:30:49	8323b1d5f0b1c3fdb5b85efbb099beb0
<a href="#">Anaconda3-5.0.0-Linux-x86.sh</a>	429.3M	2017-09-26 14:48:02	8120fcd072916e4a28d0179be8d29053
<a href="#">Anaconda3-5.0.0-Linux-x86_64.sh</a>	523.4M	2017-09-26 14:37:22	bb2656314d22aeca6af243dabbfb32c
<a href="#">Anaconda3-5.0.0-MacOSX-x86_64.pkg</a>	567.2M	2017-09-26 16:25:10	de004893c4d5714e06d4903e0780aabd
<a href="#">Anaconda3-5.0.0-MacOSX-x86_64.sh</a>	489.9M	2017-09-26 16:25:11	a72e7b22c29f0b4e05579cb8453f89fa
<a href="#">Anaconda3-5.0.0-Windows-x86.exe</a>	415.8M	2017-09-26 16:25:12	4a48ded89f15b4a2e39ffa69f3532df2
<a href="#">Anaconda3-5.0.0-Windows-x86_64.exe</a>	510.0M	2017-09-26 14:14:53	fee3fad608d0006afa5c7bca4de3d02b
<a href="#">Anaconda3-5.0.0-Linux-ppc64le.sh</a>	296.3M	2017-09-25 14:39:31	8fe5b29ca5be3ff11411621f79babfc2

Anaconda 的下载地址在此，<https://repo.continuum.io/archive/>，页面内容如图所示。我用的版本是 Anaconda3-4.3.1-Windows-x86\_64.exe，其中 64 代表 64 位系统。这个版本的安装程序在网盘里有，不过还是推荐下载最新版。请务必找到对应于系统环境的版本。我的使用环境是 Windows，所以在本教程中未必能兼顾到 MacOSX 的所有需求。

下载完成后安装，在安装之前，强烈推荐把之前安装的所有 Python 都卸载干净。安装过程中除了路径，基本不用改别的。安装完成后风平浪静，不会有额外的弹窗和程序自启动。这时我们可以在开始菜单搜索 anaconda，进入 Anaconda Navigator。下图展示了 AN 的界面。在 Home 一栏我们可以看到，Anaconda 集成了之前提到过的 Jupyter Notebook 和 Spyder。我们可以在这个界面，对各个子应用进行安装、升级。Environments 一栏展示了 Anaconda 集成的 Python 第三方模块。在这里我们可以检索、升级已安装的模块，也可以安装新模块。不过，本教程很少涉及第三方模块的安装，此功能的具体使用还请自行摸索。



如何百分百确定本地的 Python 安装成功了？Windows 用户可以按下 Windows 键+R 键，打开“运行”，输入 cmd 单击确定，进入命令提示符界面。



命令提示符长这样，在不同电脑上可能有差异。除了小时候没事干把开始菜单里所有能点开的东西都点一遍，这可能是你们为数不多的看到它的时候。概括而言，命令提示符就是在 Windows 图形界面出现之前的 DOS 系统。图中“C:\Users\CountryOld”是我电脑当前的工作路径（CountryOld 是计算机名）。此处我们召唤 cmd 的目的是检查 Python 是否正确安装。输入 python，轻按回车，如果看到了下图所示，那说明安装成功。在这里，我们实际上进入了 Python 编程界面。所以说，Python 本身是没有图形化操作界面的。Mac 系统与 cmd 对应的是 Terminal。找到 Terminal 以后，检验方法同理。

```
命令提示符 - python
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation. 保留所有权利。

C:\Users\Country0ld>python
Python 3.6.0 |Anaconda 4.3.1 (64-bit)| (default, Dec 23 2016, 11:57:41) [MSC
v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

## 你的第一个 Python 程序

所有的编程教科书都遵循一个套路——第一个程序必然是在屏幕上打印出“Hello, world!”——我也未能幸免。所以，请在方才打开的 Python 编程界面中输入 `print("Hello, world!")`。Literally，这就是你的第一个 Python 程序。print 是打印的意思，括号里面的内容是打印的对象，一切都是如此简单明了。而作为对比，一个书写规范的 Java 版 Hello World 占据了五行。

```
命令提示符 - python
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation. 保留所有权利。

C:\Users\Country0ld>python
Python 3.6.0 |Anaconda 4.3.1 (64-bit)| (default, Dec 23 2016, 11:57:41) [MSC
v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, world!")
Hello, world!
>>>
```

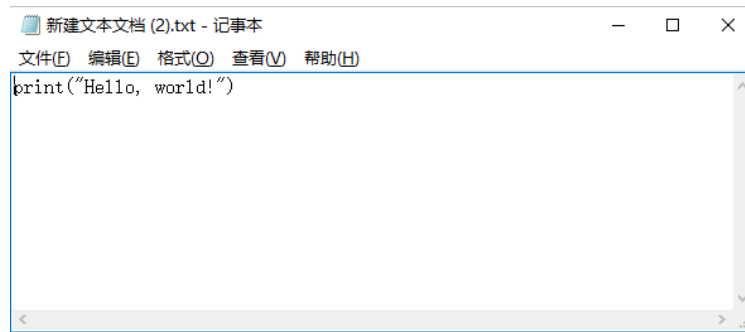
```
1 public class HelloWorld {
2     public static void main(String []args) {
3         System.out.println("Hello World");
4     }
5 }
```

但是，失落是难免的。用过 Stata 的都知道，display “Hello World” 也能实现相同的目的。于是，对于第一个 Python 程序，我们来换一种 Stata 玩不起的玩法。

```
C:\Users\Country0ld>cd desktop
C:\Users\Country0ld\Desktop>
```

首先，我们在 Python 界面中输入 `quit()` 以退出。然后，在命令提示符中输入 `cd desktop`。如果一切顺利，如上图所示，工作路径会切换到桌面文件夹。之后我们在桌面新建文本文档，双击进入以后（默认是用记事本打开），依旧输入 `print("Hello, world!")`。输入完毕后保存，并修改文件名为 `prog1.py`。注意，如果你的电脑默认不显示拓展名，请设置为显示，这样你才能看到原来的 `.txt` 后缀，修改文件名时，一定要把 `txt` 改成 `py`，以达到转换文件类型的目的。`py` 是 python 脚本（script）的拓展名，值得注意的是，修改后缀名后，其本质上依旧是一个文本，可以被包括记事本在内的各种文字处理工具打开、编辑。最后，我们回到命令提示符，

键入 `python prog1.py`。



如下图所示，我们发现，刚才保存在桌面上的 Python 脚本执行了！在这段输入中，“python”告诉计算机，要调用 Python 来处理 `prog1.py` 文件，而该文件的功能正是打印出“Hello, world!”。

```
C:\Users\Country01d\Desktop>python prog1.py
Hello, world!
```

事实上，我们可以把程序保存在任何地方，把工作路径切换至相应的文件夹目录，然后利用 `python + filename` 的方法执行。下图展示了利用 `cd` 和 `dir` 命令切换工作目录的方法。`cd ..` 代表返回上级目录。输入驱动器符号+`:`，如“`E:`”，可切换盘符。`dir` 的功能是列出当前目录下所有文件和文件夹，若是文件夹（下图显示为`<DIR>`的），我们可以通过 `cd+文件夹名` 进入。Mac 系统 Terminal 中 `cd` 功能相同，而与 `dir` 对应的是 `ls`，在此不做示范。



```
命令提示符
C:\Users\Country01d>cd desktop
C:\Users\Country01d\Desktop>cd pydofile
C:\Users\Country01d\Desktop\pydofile>cd ..
C:\Users\Country01d\Desktop>cd ..
C:\Users\Country01d>E:
E:\>dir
驱动器 E 中的卷是 Software
卷的序列号是 C08F-050C

E:\ 的目录
2017/06/29 00:10          21,885 2016101029-吴恺.csv
2017/06/29 15:04          21,288 75111aa2d027b7b5fd0250be6b50aee1_8741f94
6jw1f86a8m1v16j20fq066q38.jpg
2017/09/30 23:55          <DIR>          Anaconda3
2017/03/20 22:23       365,005,040 Anaconda3-4.3.1-Windows-x86.exe
2017/05/14 13:50       442,630,816 Anaconda3-4.3.1-Windows-x86_64.exe
2017/09/24 18:12       458,893,576 Anaconda3-4.4.0-Windows-x86_64.exe
2017/03/21 09:21          <DIR>          Kingsoft
2017/06/29 13:49          470,239 lxx.rar
2017/06/25 14:44          12,698 mydata.dta
2017/06/26 13:25           2,858 parameter.csv
2017/09/08 00:41          <DIR>          Program
2017/05/15 01:08          <DIR>          Program Files
2017/08/07 09:23          <DIR>          Program Files (x86)
2017/09/30 21:34          <DIR>          Python for Social Scientists
2017/06/26 12:22           7,682 replace.dta
2017/06/26 13:17          13,240 Rplot.png
2017/06/26 12:00          12,102 test.dta
2017/09/13 19:15          <DIR>          test1
2017/06/26 12:01          24,126 testdata.dta
2017/06/29 14:00          480,205 tly.rar
2017/01/15 10:00          <DIR>          Wind
                13 个文件 1,267,595,755 字节
                8 个目录 43,099,578,368 可用字节

E:\>cd Anaconda3
E:\Anaconda3>
```

接着，我们更进一步，写一段可交互的代码。请用记事本打开 prog1.py，把内容修改为：

```
print("Hello, world!")
words = input("Please type in your words:")
print(words)
```

保存后，回到命令提示符，输入 python prog1.py（请确保路径正确）。不出意外，cmd 先是打印出了“Hello, world!”，之后打印出了“Please type in your words:”并且光标在后头闪烁。这时，我们输入任意想说的话，按回车结束，刚才输入的内容就又打印了出来。

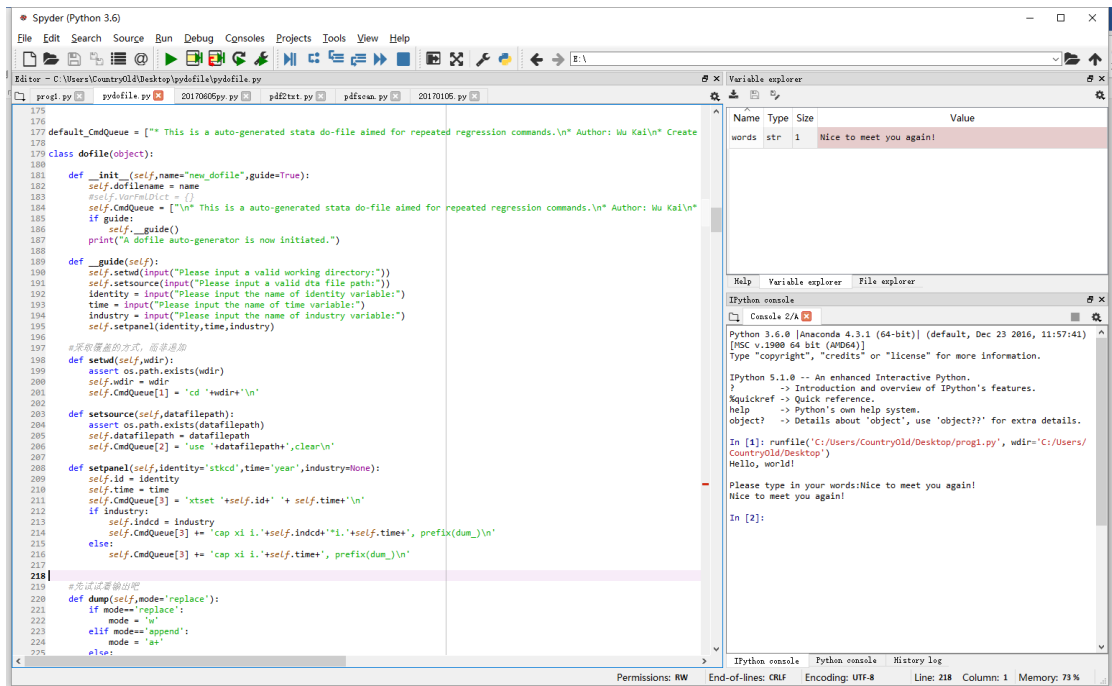
```
C:\Users\Country01d\Desktop>python prog1.py
Hello, world!
Please type in your words:We love Python!
We love Python!
```


回过头看这段程序。第一行依旧是打印“Hello, world!”。第二行使用了 input() 函数，然后把函数的结果赋值给一个叫 words 的变量。第三行把 words 变量打印出来。所以，input() 函数在这里起到的作用，是给计算机操作者抛出输入请求，而这里“Please type in your words:”是提示信息。程序运行到第二行后暂停，仅当我们完成输入操作，words 变量有了赋值以后，程序才继续执行。关于变量、函数等概念我们会在之后的章节讨论。

## 认识 Spyder

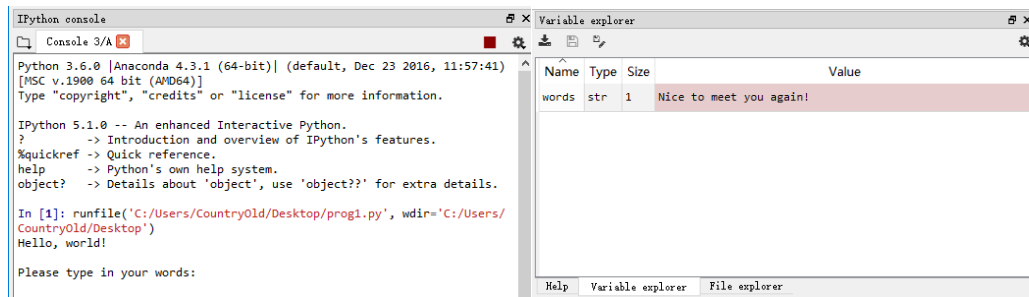
玩 Python，确实只需要用到系统命令提示符+文字编辑软件。这种狂拽酷炫吊炸天的操作模式，是很多神人的选择，但对于大部分人而言，非常不方便。可悲的是，本人在学完 Python 的很长一段时间内都是这样操作的。以为很能唬人，但实际上麻烦了自己。


在编程圈，有一种东西叫 IDE（Integrated Development Environments，集成开发环境）。我们可以在开始菜单搜索 Spyder，打开这款由 Anaconda 集成的 IDE（或者在 Anaconda Navigator 中直接启动）。下图展示了 Spyder 的默认界面布局。左侧是代码区，也是我们使用 Spyder 时的核心工作区。它相当于一个为 Python 优化的高级文本编辑器，其中代码的不同部分按不同颜色高亮显示。再看右侧，右上角默认情况下包含 Help（用于搜索 Python 帮助文件）、Variable explorer（展示当前所有已定义的变量）和 File explorer（用于浏览本地文件系统）。右侧下方默认包含了 Python console、IPython console 和 History log Python console 相当于之前在命令提示符中输入 python 后跳出的编程界面。IPython 是 Python 的增强版（An enhanced Interactive Python），在某些方面更友好，之后我们会对其有进一步的了解。History log 是日志，用来记录运行过的代码。

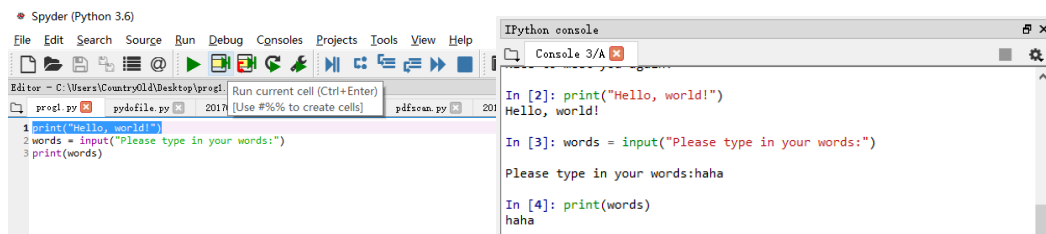


接下去我们通过菜单栏的 File--Open 打开之前在桌面创建好的 prog1.py。之后，我们在右下侧点击 IPython console，把 Spyder 默认的运行环境切换的 IPython。接着，我们点击 Spyder 上方工具栏的  按钮，这个按钮的功能是运行当前程序。如果一切顺利，右下方 IPython console 应当如下图所示。这表明 prog1.py 已提交给了 IPython 运行。可以输入任意文字（我输入的是 Nice to meet you again!），按回车以后，IPython 会把刚才输入的文字打印出来。这时，注意到右上窗口的 Variable explorer 多出了一行内容，这是因为，prog1.py 里的 words 变量成功赋值了。Name 是这个变量的名字，Type 是变量的类型（此处是 str，代表字符串），Value 是变量的值（在这里是一句话）。如果在一开始选择 Python console，那么，程序会在 Python console 中执行，你可以进行尝试。但从显示效果来看，IPython 更优，此外 IPython 还有很多的高级功能。





把编辑器、Python 控制台、变量窗口、文件浏览器、日志等整合到一起，这就是 IDE 中 Integrated 的含义。Spyder 并不是最好的 Python IDE，它在应用场景上于偏重于科学计算，而这恰恰最合社会科学工作者的胃口。使用 Spyder 还有一个好处：我们可以分段执行 python 脚本（尽管对于 Stata、SAS、Rstudio 等用户而言这应该是理所当然的事）。例如，在下图中，我们用鼠标选择第一行，再在工具栏中按  按钮（Run current cell）或者按快捷键 Ctrl+Enter，就可以只执行这一行。当然你也可以同时执行多行。请回忆上一部分，我们在命令提示符中执行 python prog1.py 时，该脚本的所有代码都被提交了。



## 认识 Jupyter Notebook

恭喜你，刚才你已经了解了非常 powerful 的集成开发环境 Spyder，它能满足我们大部分使用需求。但知道这些，还不够 Pythonic。现在，我们的认知将进一步刷新。我们要实现在浏览器里玩转 Python。

这里我们请到的明星级应用是 Jupyter Notebook。它原名 IPython Notebook，是开发 IPython 的那帮人发明的。在进入 Jupyter Notebook 之前，我们还是请出老朋友命令提示符。我们要在 D 盘根目录下创建一个名为 learnpython 的文件夹，并在其中新建 chapter1 文件夹（当然，你可以选择在 Windows 资源浏览器里完成这个简单的操作）。如下图所示，这里，我们先切换盘符，之后利用 mkdir 命令创建 learnpython 文件夹，再通过 cd 命令把工作目录切换到 D:\learnpython 下，接着还用 mkdir 创建新文件夹 chapter1。在完成了上述工作之后，我们在 cmd 输入 jupyter notebook（大小写不敏感），如果一切顺利，会显示一连串配置信息，之后我们的本地浏览器就自动打开了一个新窗口。MacOSX 的操作过程应该是一模一样的，在此不作赘述。在以下过程在，请保持命令提示符处于打开状态。

```
命令提示符 - jupyter notebook
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation。保留所有权利。

C:\Users\Country01d>D:

D:\>mkdir learnpython

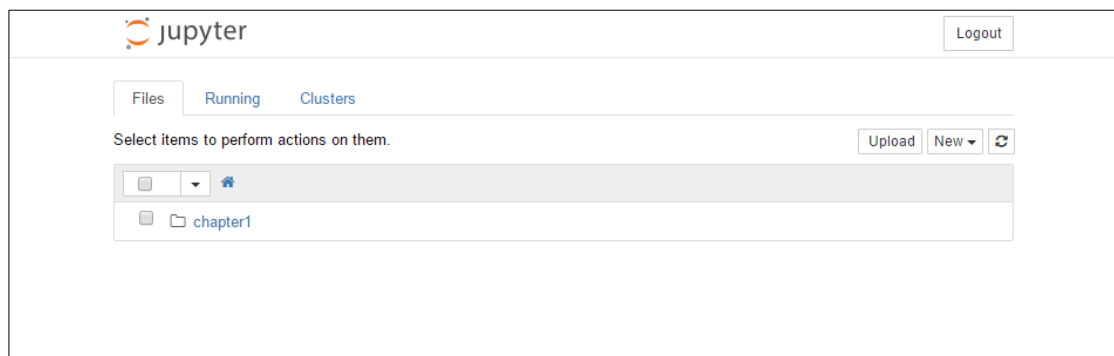
D:\>cd learnpython

D:\learnpython>mkdir chapter1

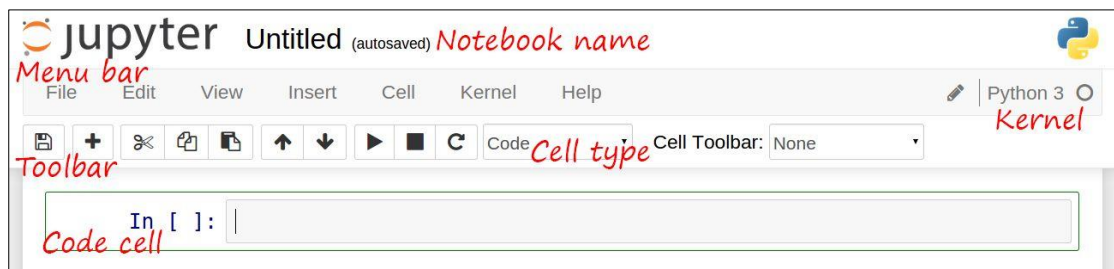
D:\learnpython>jupyter notebook
[I 12:51:56.499 NotebookApp] Serving notebooks from local directory: D:\learnpython
[I 12:51:56.499 NotebookApp] 0 active kernels
[I 12:51:56.500 NotebookApp] The Jupyter Notebook is running at: http://localhost:8888/?token=0fa3b6097ca9de07b5a31abbf853d66a64824d3a1092b0e6
[I 12:51:56.500 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 12:51:56.502 NotebookApp]

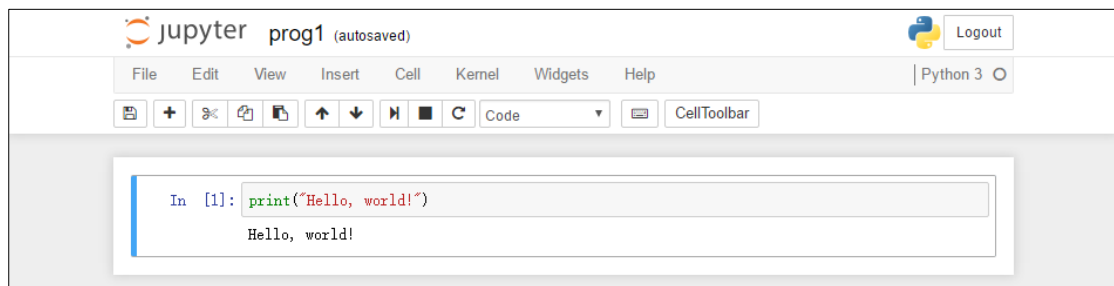
Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
http://localhost:8888/?token=0fa3b6097ca9de07b5a31abbf853d66a64824d3a1092b0e6
[I 12:51:57.463 NotebookApp] Accepting one-time-token-authenticated connection from ::1
```


如下图，这个窗口显示的是Jupyter 文件浏览器。因为我们是在 D:\learnpython 这个文件夹下进入 Jupyter 的，所以我们看到目录下有一个刚才创建的 chapter1。现在我们鼠标点击，进入这个文件夹。之后，点击右侧 **New** 按钮。从给出的选项中我们知道，在这个目录下可以创建文本文件（Text File）、目录（Folder）、Python 脚本（Python 3）、R 脚本（R），部分电脑上还可以创建 Terminal。



显然，我们要选择 Python 3。点击完毕，浏览器自动弹出新窗口，这个窗口（如下图）呈现的就是一个 notebook。最上方 Untitled 表示这个 notebook 尚未命名，我们可以点击命名（我命名为 prog1）。请注意最左侧为蓝色的框，这是 notebook 的主体，是用来跑代码的。按照惯例，我们还是输入 `print("Hello, world!")`，输入完毕后，按下 `Ctrl+Enter`，程序成功执行。代码框左侧也显示为 `In [1]:`，表示在这个 notebook 里，这个代码框是第一个执行的。`In [1]:` 这种记号是否眼熟？没错，在“认识 Spyder”小节中 IPython console 也是这样显示的。通俗地讲，Jupyter Notebook 启动后创建了一个本地服务器，这样我们就可以通过这个服务器登陆 Jupyter 挂在本地的网站，而这个网站联通了 IPython 等 kernel。





此时如果我们什么也不做，把浏览器窗口切换成刚才那个，点击右侧的  按钮刷新, 会看到 chapter1 目录下多出了一个 prog1.ipynb，并且图标是绿色的，表示正在运行。在这个窗口下我们可以关闭正在运行的 notebook（方法请自行探索）,但是并不推荐这样操作，因为大多数情况下需要保存 notebook 再关闭。我们回到第二个浏览器窗口，点击工具栏最左侧的保存按钮，然后在菜单栏选择 File - Close and Halt。这才是最正确的退出方式。



如果操作成功，第二个浏览器窗口会关闭。第一个窗口中的绿色图标也会变暗。但我们并没有真正退出 Jupyter Notebook，因为刚刚创建的本地服务器还在运行。我们回到命令提示符界面，连续按两下 Ctrl+C，如果看到 Interrupted 和 Shutting down kernels 等词出现，并且 cmd 显示了一开始的工作路径，这就表明 Jupyter Notebook 停止运行了。

我不打算在教程中详细写出 Jupyter Notebook 的基本操作，而是把它留成作业。Jupyter Notebook 提供了强大的交互式操作环境，这也提升了“干中学”的可行性。今后很大一部分教程都会以 notebook 的形式呈现给大家。

从编辑器+命令提示符，到 IDE，再到 Jupyter Notebook，本章带领大家把常用的工作环境都看了一遍。之后我们的大部分内容都会在 Jupyter Notebook 中呈现，这也是当前数据科学界的主流工作环境。

## 作业 0：熟悉 Jupyter Notebook

请将压缩包中的 pics 文件夹和 chapter1.ipynb 放入 D:\learnpython\chapter1 目录，然后进入 chapter1.ipynb（请回忆最后一节的内容）。chapter1.ipynb 的编写目的是为了快速提升大家对操作环境的熟悉程度。