

Table of Contents

Introduction	1.1
简单介绍Python	1.2
搭建Python开发环境	1.3
三个demo不同程度体验Python	1.4
初步使用：一行代码的开始-Hello World	1.4.1
感受精简：五行代码的面试题-分小球	1.4.2
感受高效：七行代码的后台代码-Flask	1.4.3

Introduction

面向群体

面向对Python感兴趣的任何朋友

内容简介

简单介绍Python

搭建Python开发环境（在windows系统配置最朴素开发环境，非小白可跳过）

三个demo不同程度体验Python

简单介绍Python

Python在我们身边

- Python(美 ['paɪθən]英 ['paɪθ(ə)n]), 一门比在日渐火爆的语言, 火爆到有的老美朋友都只知道它是一门编程语言而不知道其原意为蟒蛇。
- Python比Java还要大4岁,有不同于Java的强壮的生态体系、丰富的模块,近年来成为许多IT从业者、编程爱好者、数据痴迷者甚至科研工作人员的宠儿。
- 如果你对编程感兴趣或者想了解编程,这个以易上手而闻名的语言是你的不二之选。
- 另外,Python已经在部分省份列为高中必修课甚至小学必修课,至少Python是后浪们推前浪的必备技能。

Python功能强大、丰富

- 豆瓣、知乎等国内知名网站选择Python作为后端主体语言;
- supervisor、fabric、ansible等众多高效运维工具是使用Python编写的;
- 网络爬虫(数据抓取)工作者往往选择Python开发他们的工具;
- 数据清洗、数据分析等数据处理工作者们也喜欢Python(这个范围内的工作者中有相当一部分是科研人员或者有科研经验、或者是金融分析、商务咨询从业者跨界);
- 压轴的还是高大上的人工智能,Python的tensorflow框架能够编程实现机器学习算法,是当之无愧的AI宠儿。

Python语言简单易学

Python上手成本非常之低,读完本文,抽出半个小时到一个小时动手操作,可以初步享受使用Python的乐趣。
Python简单是为了优雅、高效,并不代表它哪里差.人生苦短,我用Python!

环境搭建（编程前的准备）

说明：

- mac自带Python2，可以无差别实现下一章的三个demo（自行安装pip或用其他方法安装flask模块即可），配置Python3开发环境、使用Python都很简单，下面给出安装包链接，在此不做详细介绍。
- 在windows配置Python开发环境有两种方法比下文介绍的朴素方法快的多：Subsystem、Docker。此处只提出方法名，没有使用经验的话就按照朴素方法开始吧。
- 最朴素的安装方法使用者最多，教程也最乱，这里给出精准的详细步骤，咱们不走弯路。

安装包下载





全部需要的文件 Python安装包(根据系统选择链接下载当前最新Python3安装包)

- [windows x64](#)
- [mac x64](#)

windows 安装Python3

此处会安装好Python3以及包管理工具pip

1.选择windows需要的安装包点击开始安装

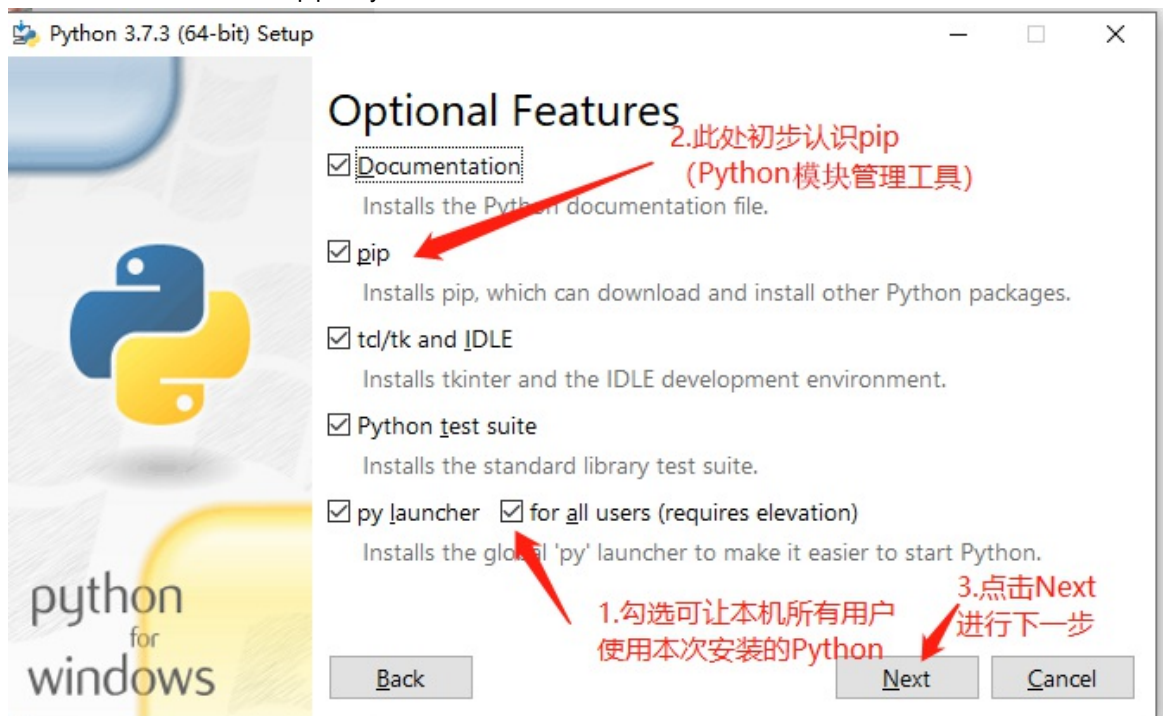
	pycharm-professional-2019.1.1.dmg	5/5/2019 下午 10:11	DMG 文件	421,276 KB
	pycharm-professional-2019.1.1.exe	5/5/2019 下午 10:03	应用程序	306,198 KB
	python-3.7.3-amd64.exe	5/5/2019 下午 9:58	应用程序	25,578 KB
	python-3.7.3-macosx10.9.pkg	5/5/2019 下午 10:11	PKG 文件	27,188 KB

windows x64安装包

1. 自动配置环境变量，以及选择自己配置安装（主要为了能让本机所有用户使用此次安装的Python）

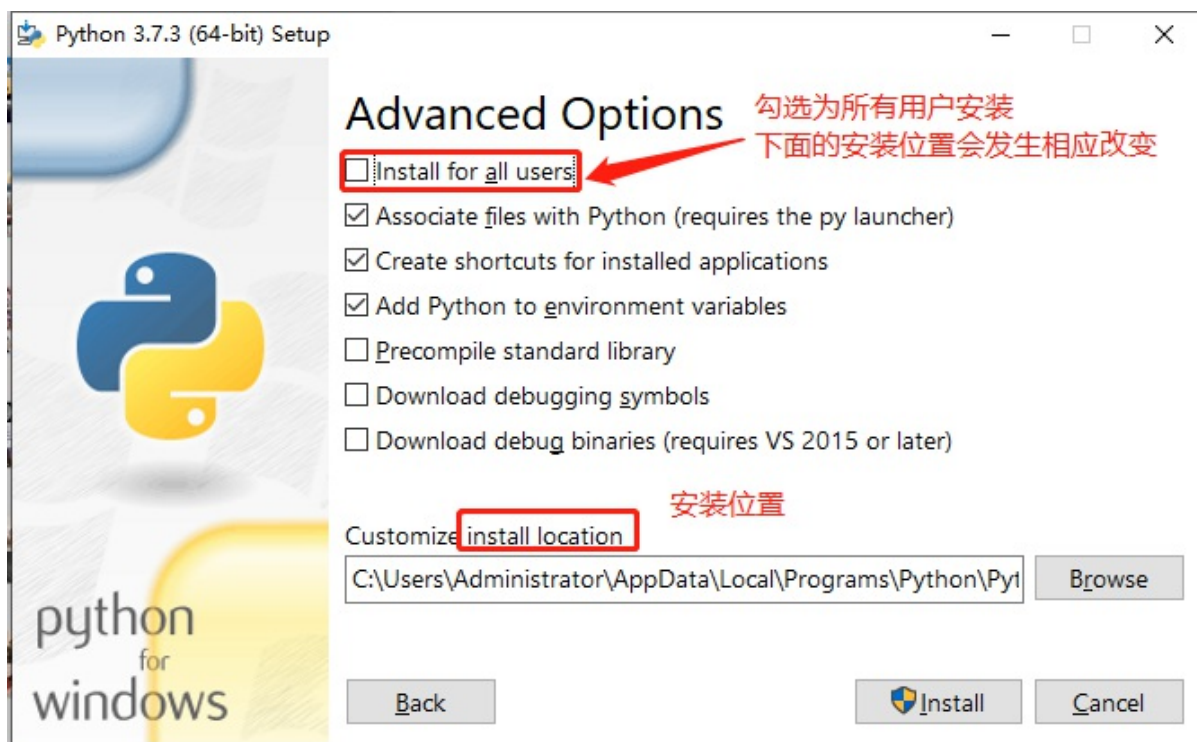


2. 进入下一步，再点击Next（pip为Python的模块管理工具）



3. 勾选为所有用户安装，安装位置发生改变，初步配置完成，点击Install

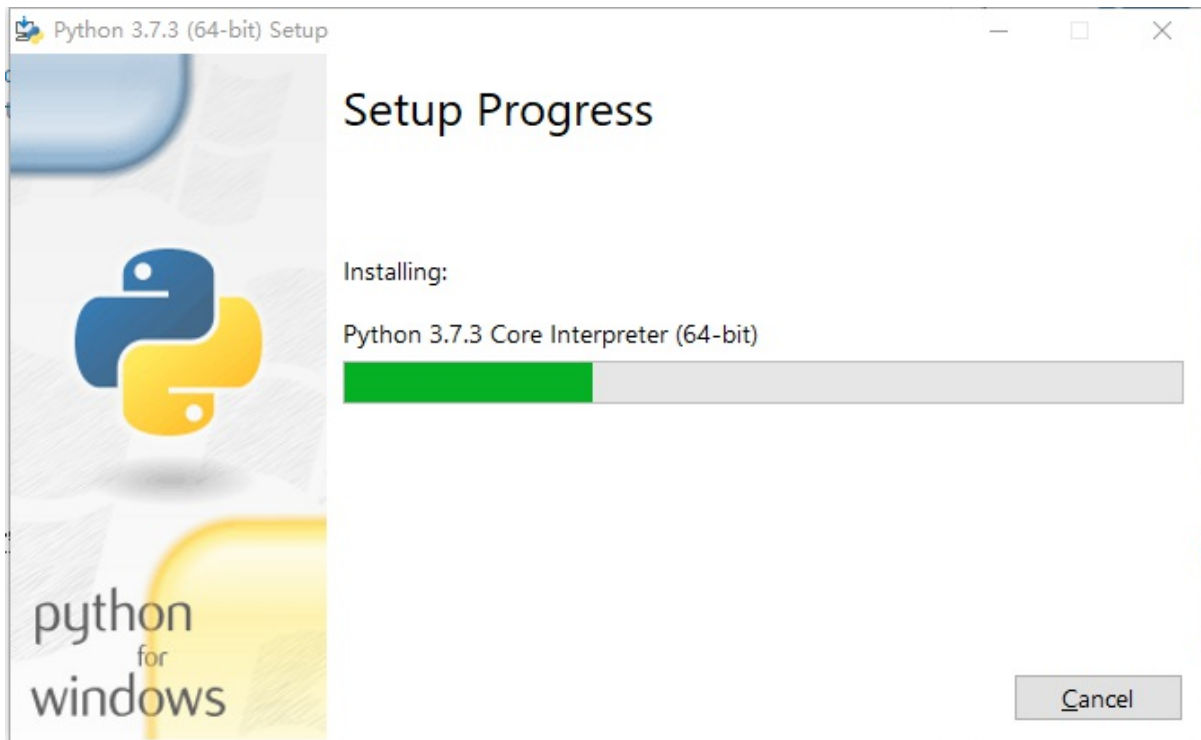
4. 勾选前



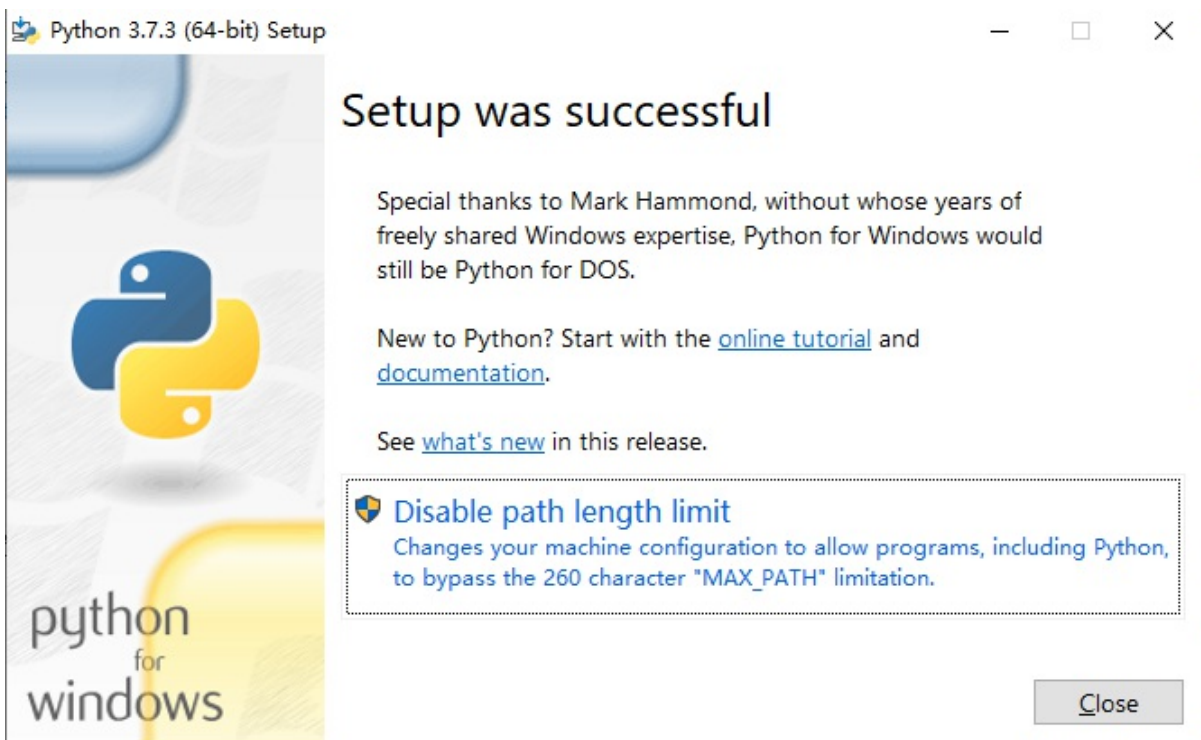
- 勾选后



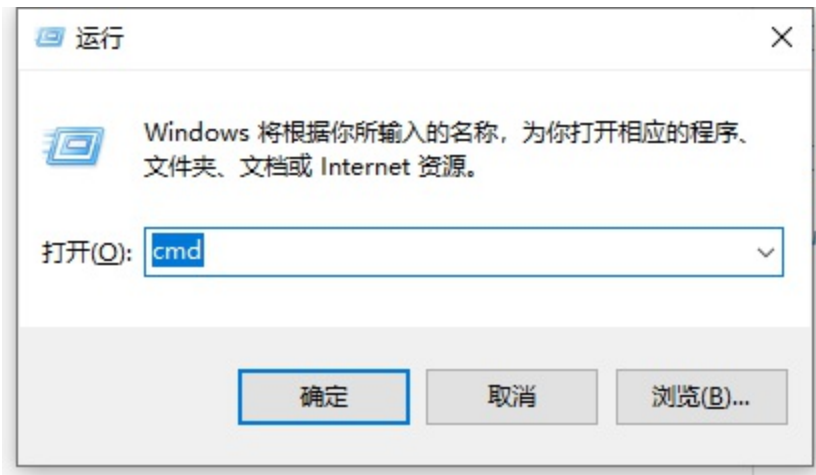
- 安装中



- 安装完成



1. 验证安装
2. `win+r` 打开 运行 ， 输入 `cmd` ， 按 `enter` (回车) 打开 命令行窗口 (下文简称命令行)



- 输入 `python -V`、`pip -V`（注意大写V）得到python版本、pip版本，安装完成

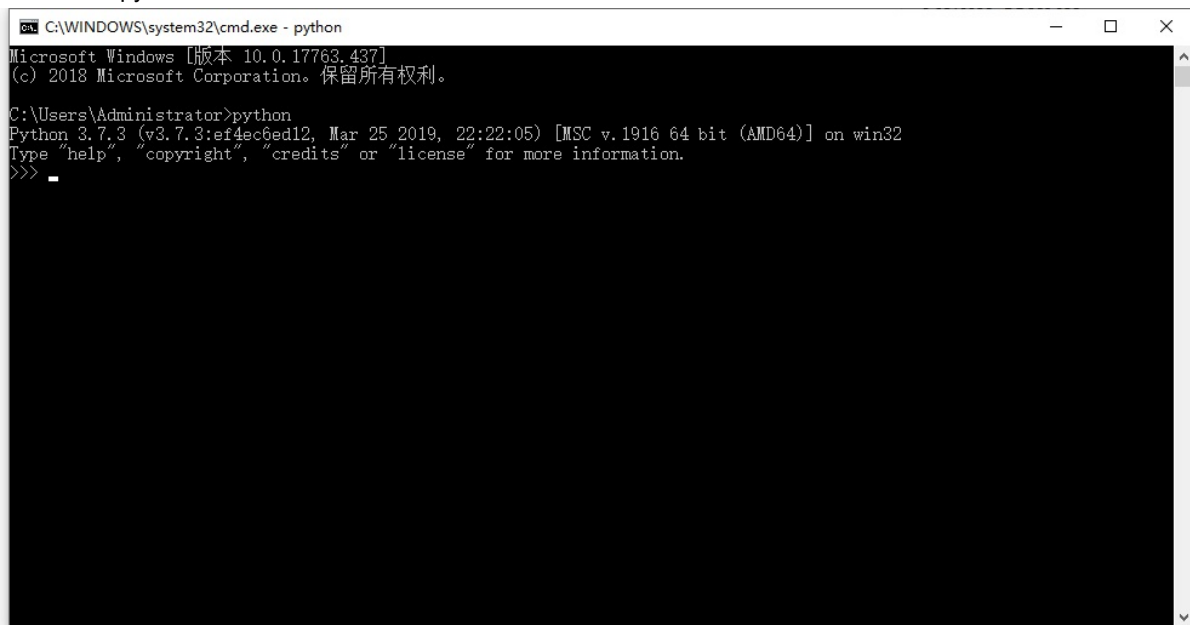
A screenshot of a Windows Command Prompt window. The title bar shows the path 'C:\WINDOWS\system32\cmd.exe'. The window contains the following text:

```
Microsoft Windows [版本 10.0.17763.437]  
(c) 2018 Microsoft Corporation。保留所有权利。  
  
C:\Users\Administrator>python -V  
Python 3.7.3  
  
C:\Users\Administrator>pip -V  
pip 19.0.3 from c:\program files\python37\lib\site-packages\pip (python 3.7)  
  
C:\Users\Administrator>
```


三个demo不同程度体验Python

开始编程

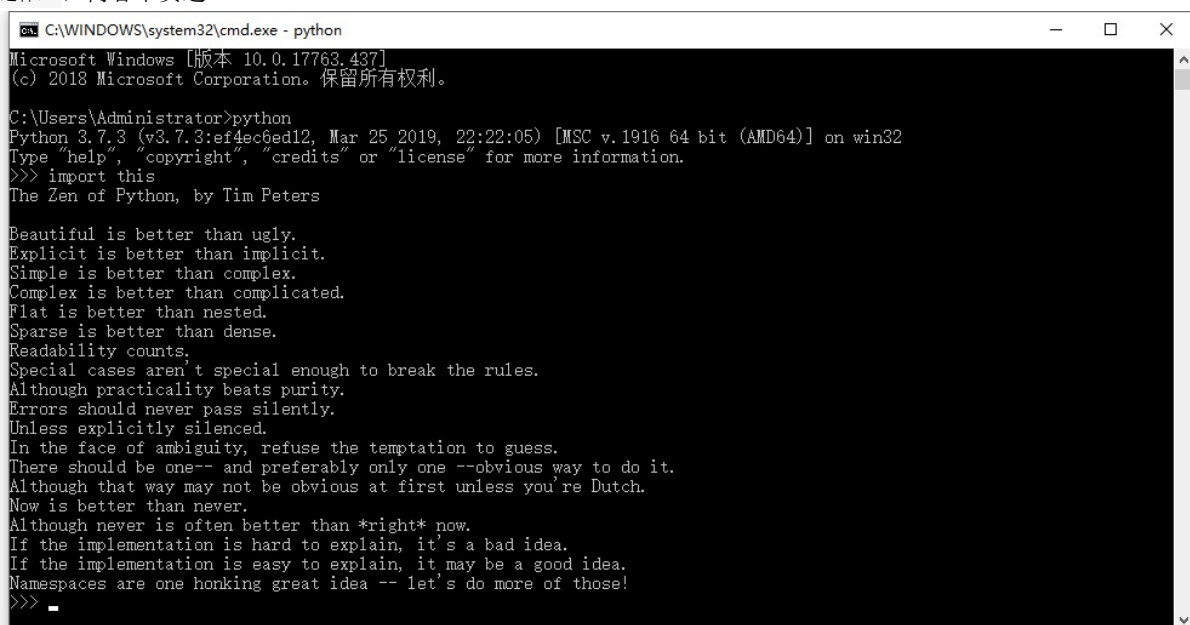
命令行输入python开始编程



```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [版本 10.0.17763.437]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>python
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> _
```

此时可以开始练习后面的demo，不过在开始使用python之前，建议输入 `import this` 查看 The Zen of Python (Python 之禅)，内容不赘述。



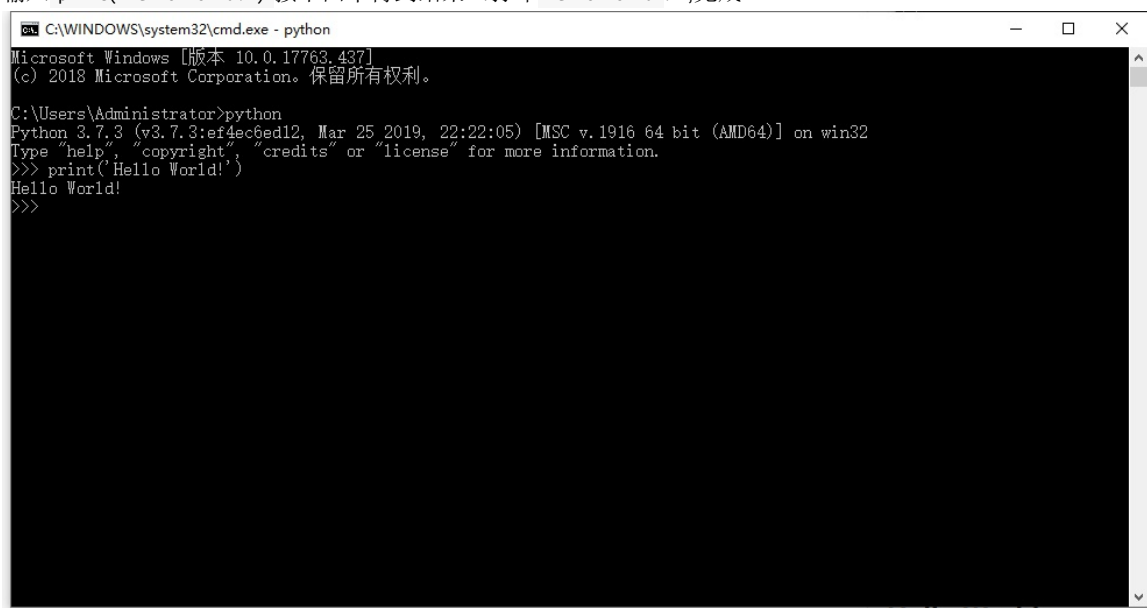
```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [版本 10.0.17763.437]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>python
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
>>> _
```

初步使用：一行代码的开始-Hello World

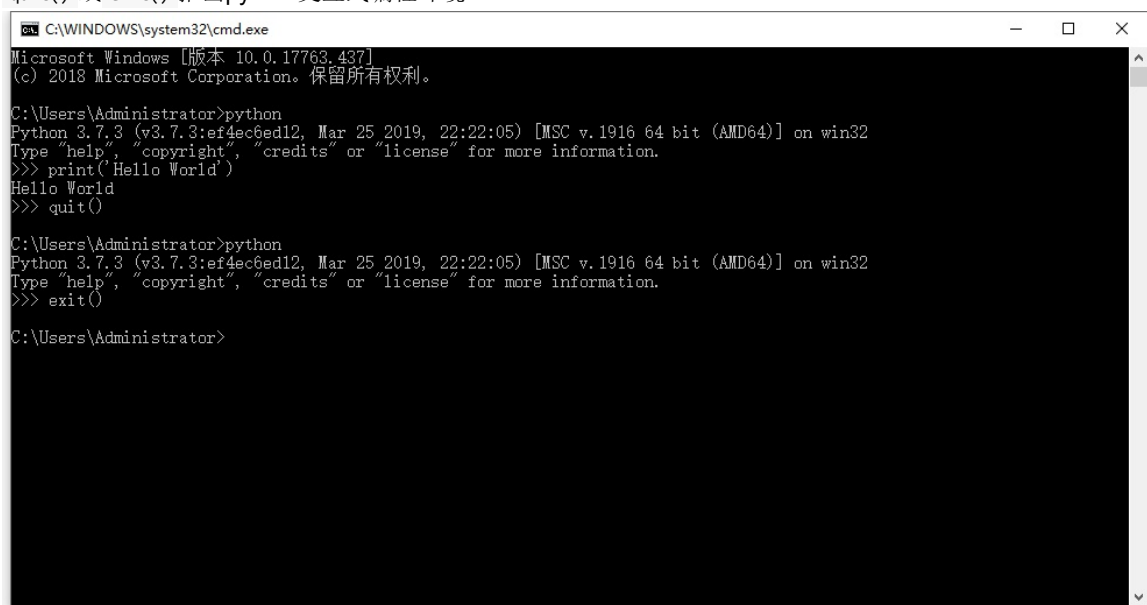
- 输入 `print('Hello World!')` 按下回车得到结果（打印 Hello World ），完成。



```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [版本 10.0.17763.437]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>python
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('Hello World!')
Hello World!
>>>
```

- `quit()` 或 `exit()` 推出python交互式编程环境



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.17763.437]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>python
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('Hello World')
Hello World
>>> quit()

C:\Users\Administrator>python
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> exit()

C:\Users\Administrator>
```

打印Hello World是一门编程语言最朴素的实现。Python打印Hello World，仅需要“真·一行代码”，可见一斑。接下来看第二个demo，用Python标准库和最基础的语法实现去解决一个面试题，感受其精简，思考其特点。

感受精简：五行代码的面试题-分小球

通过一个常见的面试题，感受Python的精简。

req:有10个球分别3红、3蓝、4白，现需要将这10个球放入这3个盒子，要求每个盒子至少有一个白球。

```
import random # 导入random模块
lst = [['w'], ['w'], ['w'],]
for i in ['w'] + ['r'] * 3 + ['b'] * 3:
    random.choice(lst).append(i)

print(lst) # 打印结果
```

效果

是不是很酷炫呢？

习惯其他语言的程序员看到这里可能为其精简诧异，可能为其“随意”担忧，不要紧，对Python稍有了解之后就会明白其中的奥秘。

接下来我们看第三个demo,七行代码启动一个web后台，感受python基于强大模块的高效。

感受高效：七行代码的后台代码-Flask

通过简单使用著名的精简型Python Web后台框架Flask，来感受其高效。

使用pip安装flask模块

- 管理员打开命令行



- `python -m pip install --upgrade pip` 升级pip版本
- `pip install flask` 安装flask模块

编写程序，保存为flask_demo.py（后缀必须为.py）

内容如下：

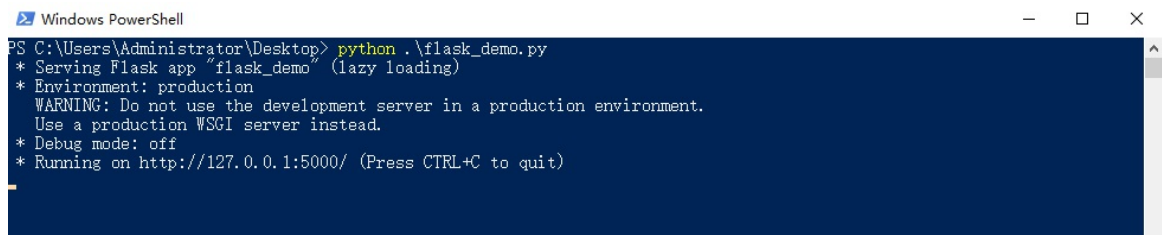
```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    app.run()
```

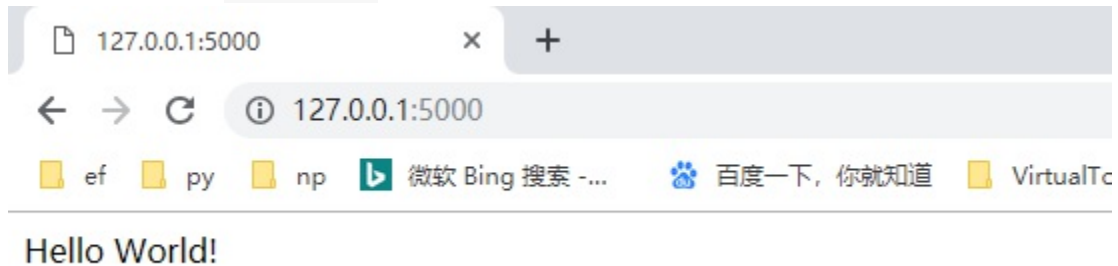
在当前目录重开一个命令行

- 按住`shift`右键点击程序所在目录窗口的空白处，左键 在此处打开powershell窗口
- 输入`python falsk_demo.py`开启后台服务

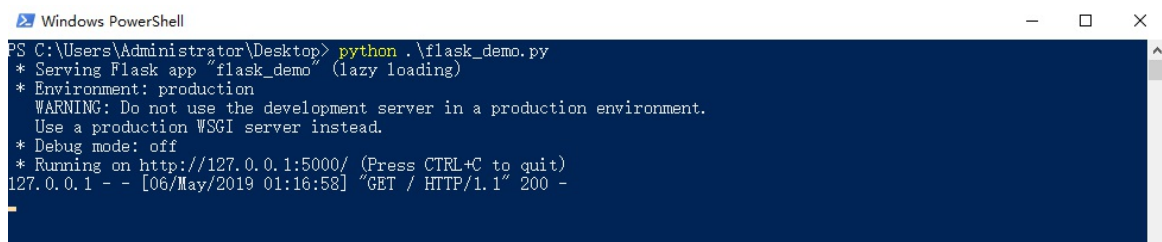


```
Windows PowerShell
PS C:\Users\Administrator\Desktop> python .\falsk_demo.py
* Serving Flask app "falsk_demo" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

- 在浏览器地址栏输入 `127.0.0.1:5000` 查看效果，实现完成



- 可以在后台看到访问记录



```
Windows PowerShell
PS C:\Users\Administrator\Desktop> python .\falsk_demo.py
* Serving Flask app "falsk_demo" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [06/May/2019 01:16:58] "GET / HTTP/1.1" 200 -
```