

目录

第 1 章 入门准备

01 开篇词：你为什么要学 Python ？

02 我会怎样带你学 Python ？

03 让 Python 在你的电脑上安家落户

04 如何运行 Python 代码 ？

第 2 章 通用语言特性

05 数据的名字和种类—变量和类型

06 一串数据怎么存—列表和字符串

07 不只有一条路—分支和循环

08 将代码放进盒子—函数

09 知错能改—错误处理、异常机制

10 定制一个模子—类

11 更大的代码盒子—模块和包

12 练习—密码生成器

第 3 章 Python 进阶语言特性

13 这么多的数据结构（一）：列表、元祖、字符串

14 这么多的数据结构（二）：字典、集合

15 Python大法初体验：内置函数

16 深入理解下迭代器和生成器

17 生成器表达式和列表生成式

18 把盒子升级为豪宅：函数进阶

19 让你的模子更好用：类进阶

20 从小独栋升级为别墅区：函数式编程

25 Python的影分身之术：虚拟环境

更新时间：2019-10-23 10:36:14



“

今天应做的事没有做，明天再早也是耽误了。

”

——裴斯泰洛齐

安装第三方包

在之前的文章中，我们多次使用了 Python 标准库，如 `random`、`functools`。标准库直接被内置在 Python 中，无需安装。但标准库的功能也是有限的，有时并不满足我们的需求，这时可以考虑引入第三方包来解决。

Python 社区有大量的开发者和组织，贡献了数以万计的第三方包，它们包罗万象并且其中不乏高质量者，借助它们可以节省许多编码工作。Python 之所以开发效率高，一个原因就在于此。你可以在 [Python Package Index](#) 中查找所需要的包。

要安装第三方包可以使用命令行工具 `pip`，它将从 Python Package Index 中检索并下载安装所指定的包。

在命令行中使用如下命令：

```
pip3 install 包名
```

如安装 Python 下非常流行的 HTTP 客户端工具 `requests`：

```
pip3 install requests
```

→ pip3 install requests

<div>← 慕课专栏</div> <div>三 你的第一本Python基础入门书 / 25 Python的影分身之术：虚拟环境</div>	
目录	Summary: Python HTTP for Humans. Home-page: http://python-requests.org Author: Kenneth Reitz Author-email: me@kennethreitz.org License: Apache 2.0 Location: /Users/obsession/projects/venv/lib/python3.7/site-packages Requires: certifi, chardet, idna, urllib3 Required-by:
第 1 章 入门准备	
01 开篇词：你为什么要学 Python ？	
02 我会怎样带你学 Python ？	
03 让 Python 在你的电脑上安家落户	
04 如何运行 Python 代码 ？	
第 2 章 通用语言特性	虚拟环境
05 数据的名字和种类—变量和类型	我们之前说过，使用 <code>pip</code> 安装的第三方包被集中放置在系统中 Python 目录下的 <code>lib/python3.x/site-packages</code> 目录中，这是个公共路径，其中的包可以被多个项目共享使用。
06 一串数据怎么存—列表和字符串	包被共享使用也会带来一个问题，如果不同的项目需要不同版本的包，这时就产生了版本冲突。比如：项目 1 中需要使用 <code>requests=2.6.0</code> ，项目 2 中需要使用 <code>requests=2.22.0</code> ，但 <code>lib/python3.x/site-packages</code> 目录中只能同时存在一个 <code>requests</code> 包的版本，这就不好办了！
07 不只有一条路—分支和循环	同时，第三方包被集中放置也会带来一个问题，那就是如果系统中所安装的包数目非常大，对这些包的管理也会是个问题。
08 将代码放进盒子—函数	这就引入了虚拟环境的概念了。
09 知错能改—错误处理、异常机制	当我们拥有多个项目时，可以对这些项目分别创建单独的虚拟环境，每个虚拟环境就是一个专有的 Python 运行环境。各个项目运行在自己的虚拟环境中，被完全隔离开。当我们在一个虚拟环境中安装第三方包，这个包只能在当前虚拟环境中使用，其它虚拟环境中是无法看到它的，所以也就不会再有项目间版本冲突的问题。另外每个虚拟环境中只包含当前项目所使用到的包，不会包含大量无用的包。
10 定制一个模子—类	创建虚拟环境
11 更大的代码盒子—模块和包	Python 3 中自带虚拟环境模块，通过它即可创建虚拟环境，无需借助其它工具。
12 练习—密码生成器	创建虚拟环境使用命令：
第 3 章 Python 进阶语言特性	<pre>python3 -m venv 虚拟环境名称</pre>
13 这么多的数据结构（一）：列表、元祖、字符串	如：
14 这么多的数据结构（二）：字典、集合	<pre>python3 -m venv venv</pre>
15 Python大法初体验：内置函数	此时将在当前目录中创建出 <code>venv</code> 目录，这即是虚拟环境目录，其中包含 Python 运行环境。
16 深入理解下迭代器和生成器	激活虚拟环境
17 生成器表达式和列表生成式	要使用虚拟环境首先需要激活它，激活后当前命令行所执行的 Python 代码都将运行于该虚拟环境中。
18 把盒子升级为豪宅：函数进阶	
19 让你的模子更好用：类进阶	
20 从小独栋升级为别墅区：函数式编程	

← 慕课专栏	☰ 你的第一本Python基础入门书 / 25 Python的影分身之术：虚拟环境
目录	<div>cd venv\Scripts activate.bat</div>
第 1 章 入门准备	Linux 和 MacOS 中使用命令：
01 开篇词：你为什么要学 Python ？	<div>source venv/bin/activate</div>
02 我会怎样带你学 Python ？	上述命令执行过后，命令行的提示符前将显示 (venv) ，如下：
03 让 Python 在你的电脑上安家落户	<div>(venv) →</div>
04 如何运行 Python 代码 ？	这表示当前已处于虚拟环境中。此后在该命令行中所执行的所有 Python 代码都运行于该虚拟环境中。
第 2 章 通用语言特性	退出虚拟环境
05 数据的名字和种类—变量和类型	当我们不要使用虚拟环境时，可以退出当前虚拟环境。
06 一串数据怎么存—列表和字符串	Windows 操作系统中使用命令：
07 不只有一条路—分支和循环	<div>cd venv\Scripts deactivate.bat</div>
08 将代码放进盒子—函数	Linux 和 MacOS 中使用命令：
09 知错能改—错误处理、异常机制	<div>deactivate</div>
10 定制一个模子—类	此后命令行的提示符将恢复回原本状态。
11 更大的代码盒子—模块和包	<div>← 24 让你的代码更灵活：进程和线程</div> <div>26 更加 Python 的 Python 代码风格 →</div>
12 练习—密码生成器	精选留言 0
第 3 章 Python 进阶语言特性	欢迎在这里发表留言，作者筛选后可公开显示
13 这么多的数据结构（一）：列表、元祖、字符串	
14 这么多的数据结构（二）：字典、集合	
15 Python大法初体验：内置函数	
16 深入理解下迭代器和生成器	
17 生成器表达式和列表生成式	
18 把盒子升级为豪宅：函数进阶	
19 让你的模子更好用：类进阶	
20 从小独栋升级为别墅区：函数式编程	

目录
第 1 章 入门准备
01 开篇词：你为什么要学 Python ？
02 我会怎样带你学 Python ？
03 让 Python 在你的电脑上安家落户
04 如何运行 Python 代码 ？
第 2 章 通用语言特性
05 数据的名字和种类—变量和类型
06 一串数据怎么存—列表和字符串
07 不只有一条路—分支和循环
08 将代码放进盒子—函数
09 知错能改—错误处理、异常机制
10 定制一个模子—类
11 更大的代码盒子—模块和包
12 练习—密码生成器
第 3 章 Python 进阶语言特性
13 这么多的数据结构（一）：列表、元祖、字符串
14 这么多的数据结构（二）：字典、集合
15 Python大法初体验：内置函数
16 深入理解下迭代器和生成器
17 生成器表达式和列表生成式
18 把盒子升级为豪宅：函数进阶
19 让你的模子更好用：类进阶
20 从小独栋升级为别墅区：函数式编