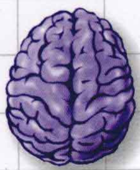
深入浅出Python（影印版）

Head First Python



Load important programming concepts directly into your brain

Model data with lists, sets, and dictionaries

Preserve your data in a pickle

Hook up with JSON, Android, and App Engine

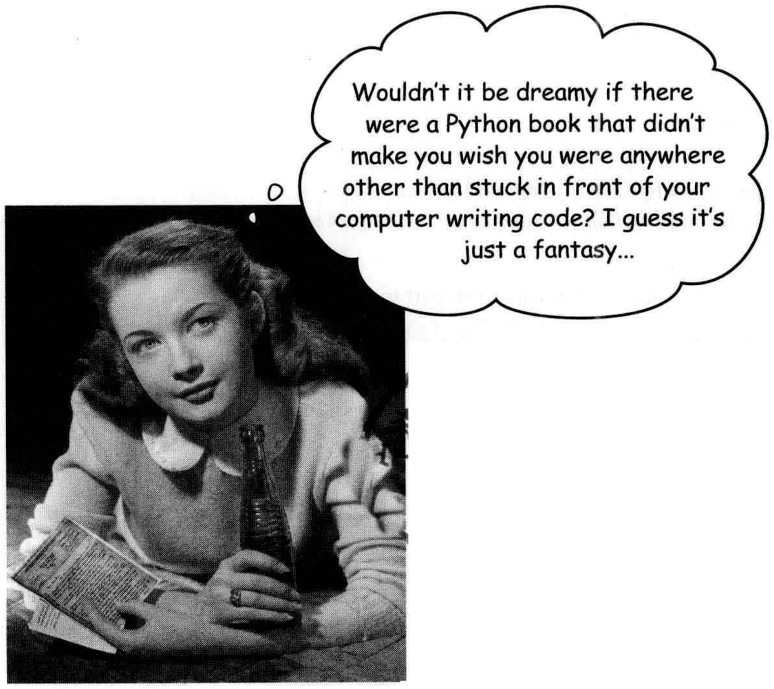
Move your custom app to the Web

Share your code with the world on PyPI

O'REILLY® 東南大學出版社

Paul Barry 著

深入浅出 Python（影印版）
Head First Python


Wouldn't it be dreamy if there were a Python book that didn't make you wish you were anywhere other than stuck in front of your computer writing code? I guess it's just a fantasy...

Paul Barry

O'REILLY®

Beijing* • *Cambridge* • *Köln* • *Sebastopol* • *Tokyo

O'Reilly Media, Inc. 授权东南大学出版社出版

东南大学出版社

图书在版编目（CIP）数据

深入浅出Python：英文 /（英）巴里（Barry, P.）著 .—影印本 .—南京：东南大学出版社，2011.5

书名原文：Head First Python

ISBN 978-7-5641-2675-9

I. ①深…　II. ①巴…　III. ①软件工具－程序设计－英文　IV. ① TP311.56

中国版本图书馆 CIP 数据核字（2011）第 042748 号

江苏省版权局著作权合同登记

图字：10-2010-439 号

深入浅出 Python（影印版）

出版发行：东南大学出版社
地　　址：南京四牌楼 2 号　　邮编：210096
出 版 人：江建中
网　　址：http://www.seupress.com
电子邮件：press@seupress.com
印　　刷：扬中市印刷有限公司
开　　本：787 毫米 × 980 毫米　12 开本
印　　张：41.5 印张
字　　数：694 千字
版　　次：2011 年 5 月第 1 版
印　　次：2011 年 5 月第 1 次印刷
书　　号：ISBN 978-7-5641-2675-9
定　　价：88.00 元（册）

本社图书若有印装质量问题，请直接与读者服务部联系。电话（传真）：025-83792328

Table of Contents (Summary)

Table of Contents (the real thing)

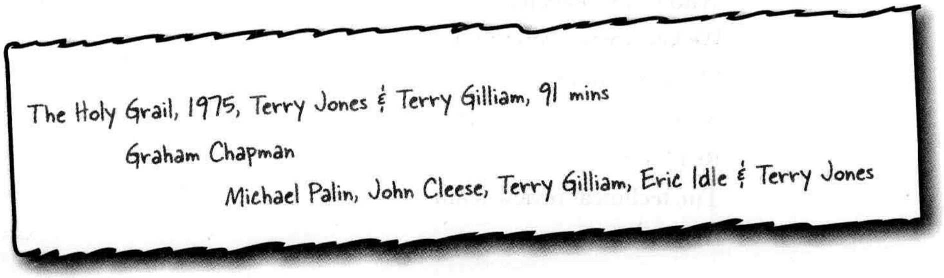
Intro

Your brain on Python. Here you are trying to learn something, while here your brain is doing you a favor by making sure the learning doesn't stick. Your brain's thinking, "Better leave room for more important things, like which wild animals to avoid and whether naked snowboarding is a bad idea." So how do you trick your brain into thinking that your life depends on knowing Python?

meet python

1 Everyone loves lists

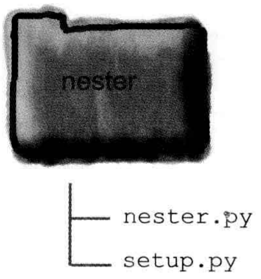
You're asking one question: "What makes Python different?" The short answer is: *lots of things*. The longer answers starts by stating that there's lots that's familiar, too. Python is a lot like any other *general-purpose* programming language, with **statements**, **expressions**, **operators**, **functions**, **modules**, **methods**, and **classes**. All the *usual stuff*, really. And then there's the other stuff Python provides that makes the programmer's life—*your* life—that little bit easier. You'll start your tour of Python by learning about **lists**. But, before getting to that, there's another important question that needs answering…


The Holy Grail, 1975, Terry Jones & Terry Gilliam, 91 mins
Graham Chapman
Michael Palin, John Cleese, Terry Gilliam, Eric Idle & Terry Jones

2 sharing your code
Modules of functions

Reusable code is great, but a shareable module is better.

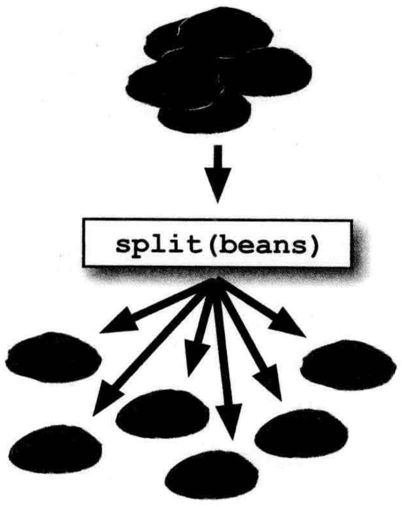
By sharing your code as a Python module, you open up your code to the entire Python community...and it's always good to share, isn't it? In this chapter, you'll learn how to create, install, and distribute your own shareable modules. You'll then load your module onto Python's software sharing site on the Web, so that *everyone* can benefit from your work. Along the way, you'll pick up a few new tricks relating to Python's functions, too.



3 files and exceptions

Dealing with errors

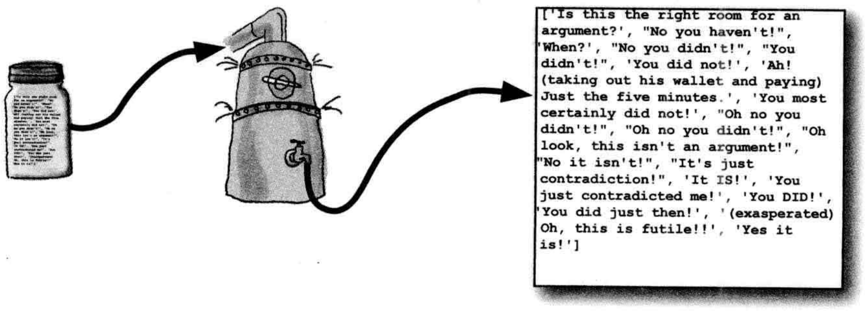
It's simply not enough to process your list data in your code. You need to be able to get your data *into* your programs with ease, too. It's no surprise then that Python makes reading data from **files** easy. Which is great, until you consider what can go *wrong* when interacting with data *external* to your programs… and there are lots of things waiting to trip you up! When bad stuff happens, you need a strategy for getting out of trouble, and one such strategy is to deal with any exceptional situations using Python's **exception handling** mechanism showcased in this chapter.



4 persistence
Saving data to files

It is truly great to be able to process your file-based data.

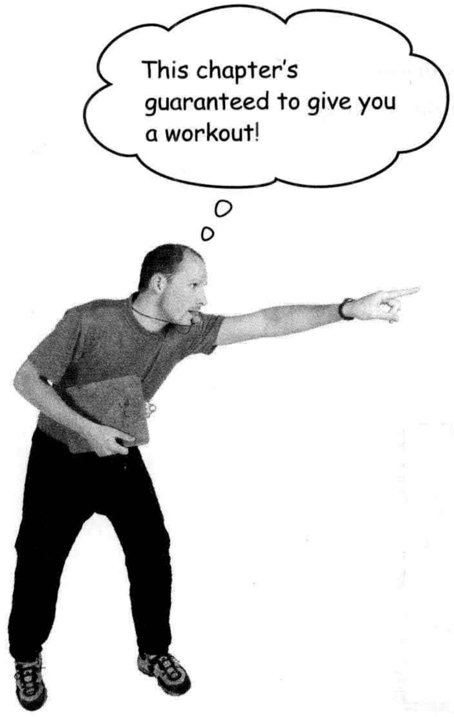
But what happens to your data when you're done? Of course, it's best to save your data to a disk file, which allows you to use it again at some later date and time. Taking your memory-based data and storing it to disk is what **persistence** is all about. Python supports all the usual tools for writing to files and also provides some cool facilities for *efficiently* storing Python data.



```
['Is this the right room for an
argument?', "No you haven't!",
'When?', "No you didn't!", "You
didn't!", 'You did not!', 'Ah!
(taking out his wallet and paying)
Just the five minutes.', 'You most
certainly did not!', "Oh no you
didn't!", "Oh no you didn't!", "Oh
look, this isn't an argument!",
"No it isn't!", "It's just
contradiction!", 'It IS!', 'You
just contradicted me!', 'You DID!',
'You did just then!', '(exasperated)
Oh, this is futile!!', 'Yes it
is!']
```

5 comprehending data
Work that data!

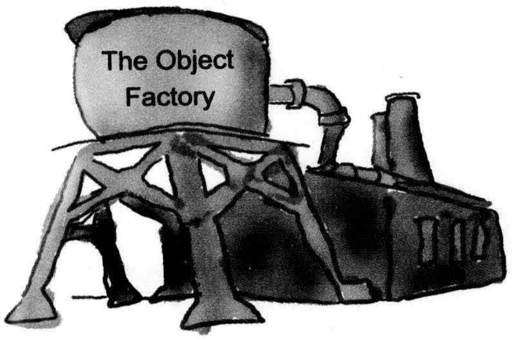
Data comes in all shapes and sizes, formats and encodings. To work effectively with your data, you often have to manipulate and transform it into a common format to allow for efficient processing, sorting, and storage. In this chapter, you'll explore Python goodies that help you work your data up into a sweat, allowing you to achieve data-munging greatness.



6 custom data objects

Bundling code with data

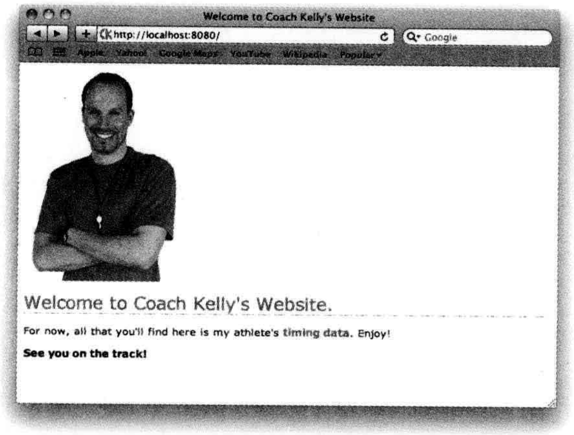
It's important to match your data structure choice to your data. And that choice can make a big difference to the complexity of your code. In Python, although really useful, lists and sets aren't the only game in town. The Python **dictionary** lets you organize your data for speedy lookup by *associating your data with names*, not numbers. And when Python's built-in data structures don't quite cut it, the Python **class** statement lets you define your own. This chapter shows you how.



7 web development

Putting it all together

Sooner or later, you'll want to share your app with lots of people. You have many options for doing this. Pop your code on PyPI, send out lots of emails, put your code on a CD or USB, or simply install your app manually on the computers of those people who need it. Sounds like a lot of work...not to mention boring. Also, what happens when you produce the next best version of your code? What happens then? How do you manage the update? Let's face it: it's such a pain that you'll think up really creative excuses not to. Luckily, you don't have to do any of this: just create a webapp instead. And, as this chapter demonstrates, using Python for web development is a breeze.


Welcome to Coach Kelly's Website
http://localhost:8080/
Google
Apple Yahoo! Google Maps YouTube Wikipedia Popular

Welcome to Coach Kelly's Website.

For now, all that you'll find here is my athlete's timing data. Enjoy!

See you on the track!

8 mobile app development
Small devices

Putting your data on the Web opens up all types of possibilities. Not only can anyone from anywhere interact with your webapp, but they are increasingly doing so from a collection of diverse computing devices: PCs, laptops, tablets, palmtops, and even mobile phones. And it's not just humans interacting with your webapp that you have to support and worry about: *bots* are small programs that can automate web interactions and typically want your data, not your human-friendly HTML. In this chapter, you exploit Python on Coach Kelly's mobile phone to write an app that interacts with your webapp's data.

9 manage your data
Handling input

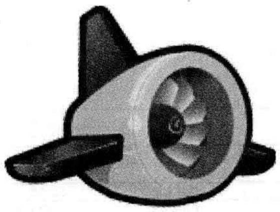
The Web and your phone are not just great ways to display data. They are also great tools to for accepting input from your users. Of course, once your webapp accepts data, it needs to put it somewhere, and the choices you make when deciding what and where this "somewhere" is are often the difference between a webapp that's easy to grow and extend and one that isn't. In this chapter, you'll extend your webapp to accept data from the Web (via a browser or from an Android phone), as well as look at and enhance your back-end data-management services.

10

scaling your webapp

Getting real

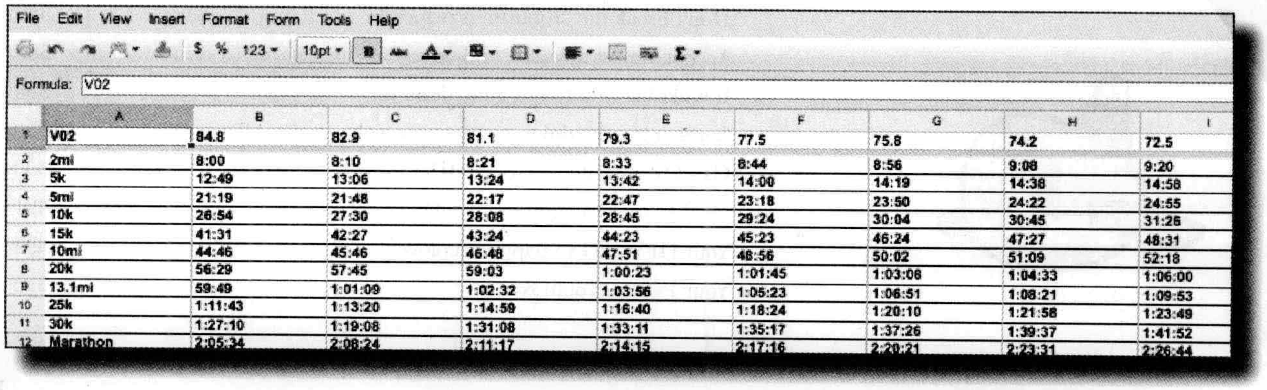
The Web is a great place to host your app...until things get real. Sooner or later, you'll hit the jackpot and your webapp will be *wildly successful*. When that happens, your webapp goes from a handful of hits a day to thousands, possibly ten of thousands, *or even more*. Will you be ready? Will your web server handle the **load**? How will you know? What will it **cost**? *Who will pay?* Can your data model scale to millions upon millions of data items without *slowing to a crawl*? Getting a webapp up and running is easy with Python and now, thanks to Google App Engine, **scaling** a Python webapp is achievable, too.



11 dealing with complexity

Data wrangling

It's great when you can apply Python to a specific domain area. Whether it's *web development*, *database management*, or *mobile apps*, Python helps you **get the job done** by *not* getting in the way of you coding your solution. And then there's the other types of problems: the ones you can't categorize or attach to a domain. Problems that are in themselves so *unique* you have to look at them in a different, highly specific way. Creating **bespoke** software solutions to these type of problems is an area where Python *excels*. In this, your final chapter, you'll *stretch* your Python skills to the limit and solve problems along the way.



Formula: V02

	A	B	C	D	E	F	G	H	I
1	V02	84.8	82.9	81.1	79.3	77.5	75.8	74.2	72.5
2	2mi	8:00	8:10	8:21	8:33	8:44	8:56	9:08	9:20
3	5k	12:49	13:06	13:24	13:42	14:00	14:19	14:38	14:58
4	5mi	21:19	21:48	22:17	22:47	23:18	23:50	24:22	24:55
5	10k	26:54	27:30	28:08	28:45	29:24	30:04	30:45	31:26
6	15k	41:31	42:27	43:24	44:23	45:23	46:24	47:27	48:31
7	10mi	44:46	45:46	46:48	47:51	48:56	50:02	51:09	52:18
8	20k	56:29	57:45	59:03	1:00:23	1:01:45	1:03:08	1:04:33	1:06:00
9	13.1mi	59:49	1:01:09	1:02:32	1:03:56	1:05:23	1:06:51	1:08:21	1:09:53
10	25k	1:11:43	1:13:20	1:14:59	1:16:40	1:18:24	1:20:10	1:21:58	1:23:49
11	30k	1:27:10	1:19:08	1:31:08	1:33:11	1:35:17	1:37:26	1:39:37	1:41:52
12	Marathon	2:05:34	2:08:24	2:11:17	2:14:15	2:17:16	2:20:21	2:23:31	2:26:44

1 meet python

Everyone loves lists

Yes, yes...we have lots of Pythons in stock... I'll just make a quick list.

You're asking one question: "What makes Python different?"

The short answer is: *lots of things*. The longer answers starts by stating that there's lots that's familiar, too. Python is a lot like any other *general-purpose* programming language, with **statements**, **expressions**, **operators**, **functions**, **modules**, **methods**, and **classes**. All the *usual stuff*, really. And then there's the other stuff Python provides that makes the programmer's life—*your* life—that little bit easier. You'll start your tour of Python by learning about **lists**. But, before getting to that, there's another important question that needs answering…

What's to like about Python?

Lots. Rather than tell you, this book's goal is to *show* you the greatness that is Python.

Yeah... I need something that I can deploy on PCs, Macs, handhelds, phones,the Web, on big servers and small clients...and it has to let me build GUIs quickly and painlessly... OK, yes, yeah, I'm listening... What?!? You're kidding! Python can do all that?
python

Before diving head first into Python, let's get a bit of housekeeping out of the way.

To work with and execute the Python code in this book, you need a copy of the **Python 3 interpreter** on your computer. Like a lot of things to do with Python, it's not difficult to install the interpreter. Assuming, of course, it's not already there…