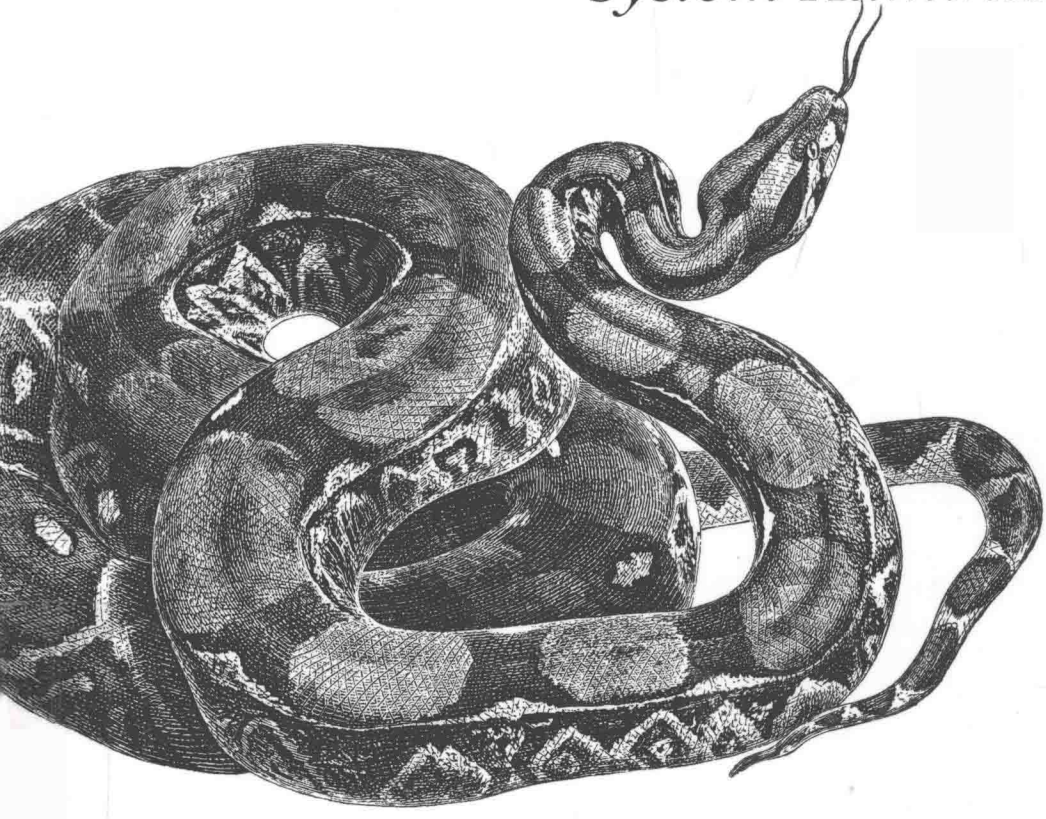
*Python*在*Unix*和*Linux*系统管理中的应用（影印版）

Python

for Unix and Linux System Administration



O'REILLY®
开明出版社

Noah Gift & Jeremy M. Jones 著

Python在Unix和Linux系统管理中的应用(影印版)

Python for Unix and Linux System Administration

Noah Gift & Jeremy M. Jones

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Taipei • Tokyo

O'Reilly Media, Inc.授权北京凤凰天下文化发展有限公司、开明出版社出版发行

开明出版社

图书在版编目（CIP）数据

Python在Unix、Linux系统管理中的应用：英文 /（美）琼斯（Jones，J.），（美）吉夫特（Gift，N.）著.—影印本.—北京：开明出版社，2009.3
ISBN 978-7-80205-738-8

Ⅰ. U… Ⅱ.琼… Ⅲ.①软件工具—应用—UNIX操作系统—程序设计—英文②软件工具—应用—Linux操作系统—程序设计—英文 Ⅳ.TP316.8 TP311.56

中国版本图书馆CIP数据核字（2009）第035513号

江苏省版权局著作权合同登记

图字：10-2009-085 号

书名：Python 在 Unix 和 Linux 系统管理中的应用（影印版）
出版：开明出版社出版（北京海淀区西三环北路 19 号 邮编 100089）
经销：全国新华书店
印刷：北京市梦宇印务有限公司（北京市通州区张家湾镇张辛庄村）
开本：787×1092 1 / 16
印张：28.75
字数：483 千字
版次：2009 年 4 月 北京第 1 版
印次：2009 年 4 月 北京第 1 次印刷
定价：64.00 元

印刷、装订质量问题，出版社负责调换货 联系电话：（010）88817647

Foreword

I was excited to preview this book on using Python for system administration. I remembered how I felt when I discovered Python after many years of programming in other languages: it was like a breath of spring air and the warmth of the sun after a long winter indoors. Code was suddenly easy and fun to write again, and I finished programs much more quickly than before.

As a system administrator, most of my own Python use is for system and network management tasks. I already knew how useful a good book focusing on system administration with Python would be. I am happy to say that this is that book. Overall, Noah and Jeremy have created an interesting, intelligent work on Python that is planted firmly in the system administration sphere. I found the book both very useful and enjoyable to read.

The two opening chapters are a great introduction to Python for system administrators (and others) who are new to Python. I consider myself an intermediate Python programmer, and I learned a lot from the book. I suspect even Python wizards will come across a few new tricks in here. I can especially recommend the chapters on networking and managing network services, SNMP, and management of heterogeneous systems as particularly useful and well focused on nontrivial, real-world tasks that system administrators face every day.

—Æleen Frisch, July 2008

Preface

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates new terms, URLs, email addresses, filenames, and file extensions.

`Constant width`
: Used for program listings, in text to refer to program elements, such as variable or function names, databases, data types, environment variables, statements, utilities, keywords, utilities, and modules.

`Constant width bold`
: Shows commands or other text that should be typed literally by the user.

`Constant width italic`
: Shows text that should be replaced with user-supplied values or by values determined by context.

This icon signifies a tip, suggestion, or general note.

This icon indicates a warning or caution.

Using Code Examples

This book is here to help you get your job done. In general, you may use the code that is included in this book in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission; selling or distributing a CD-ROM of examples from O'Reilly

books does require permission. Answering a question by citing this book and quoting example code does not require permission; incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN, for example: "*Python for Unix and Linux System Administration* by Noah Gift and Jeremy M. Jones. Copyright 2008 Noah Gift and Jeremy M. Jones, 978-0-596-51582-9."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at *permissions@oreilly.com*.

Safari® Books Online

When you see a Safari® Books Online icon on the cover of your favorite technology book, that means the book is available online through the O'Reilly Network Safari Bookshelf.

Safari offers a solution that's better than e-books. It's a virtual library that lets you easily search thousands of top tech books, cut and paste code samples, download chapters, and find quick answers when you need the most accurate, current information. Try it for free at *http://safari.oreilly.com*.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at:

http://www.oreilly.com/9780596515829

To comment or ask technical questions about this book, send email to:

bookquestions@oreilly.com

For more information about our books, conferences, Resource Centers, and the O'Reilly Network, see our website at:

http://www.oreilly.com

Acknowledgments

Noah's Acknowledgments

As I sit writing an acknowledgment for this book, I have to first mention Dr. Joseph E. Bogen, because he made the single largest impact on me, at a time that it mattered the most. I met Dr. Bogen while I was working at Caltech, and he opened my eyes to another world giving me advice on life, psychology, neuroscience, math, the scientific study of consciousness, and much more. He was the smartest person I ever met, and was someone I loved. I am going to write a book about this experience someday, and I am saddened that he won't be there to read it, his death was a big loss.

I want to thank my wife, Leah, who has been one of the best things to happen to me, ever. Without your love and support, I never could have written this book. You have the patience of a saint. I am looking forward to going where this journey takes us, and I love you. I also want to thank my son, Liam, who is one and a half, for being patient with me while I wrote this book. I had to cut many of our guitar, piano, and pushup lessons short, so I owe you payback times two, little goat.

To my mom, I love you, and thank you for encouraging me throughout life.

Of course, I want to thank Jeremy M. Jones, my coauthor, for agreeing to write this book with me. I think we were a great team with different, but complementary styles, and we wrote a great book. You have taught me a lot about Python, and have been a good partner and friend. Thanks!

Titus Brown, whom I suppose I have to call Dr. Brown now, was the person that got me interested in Python to begin with, when I met him at Caltech. He is another example of how one person can make a difference, and I am glad to consider him an "old" friend, the kind money can't buy. He kept asking me, "Why don't you use Python?" And then one day I did. If it wasn't for Titus, I would certainly have continued down the Java and Perl path. You can read his blog here: *http://ivory.idyll.org/blog*.

Shannon Behrens has a heart of solid gold, a mind as sharp as a razor, and a knowledge of Python that is truly scary. I first met Shannon through Titus, ironic again, but he and I became quick friends. Shannon is the real deal in every sense of the word, and has taught me a tremendous amount about Python, in fact, staggering would be a better word. His help with Python, and editing this book has been incredible, and I owe him tremendously. I shudder to think of what it would have looked like without him. I can't ever imagine a company being foolish enough to let him get away, and I look forward to helping him with his first book. Finally, he is just an incredible technical reviewer. You can read his blog here: *http://jjinux.blogspot.com/*.

Doug Hellmann was our other star technical reviewer and was exceptionally productive and helpful. Jeremy and I are extremely fortunate to get someone of his caliber to review the book. He went above and beyond his call of duty, and is truly a force of efficiency

to reckon with. He was also a great source of motivation while we worked together at Racemi. You can read his blog here: *http://blog.doughellmann.com/*.

Thanks to Scott Leerseen for reviewing our book and giving us good advice along the way. I also especially enjoyed our code review battles. Just remember, I am always right.

Thanks to Alfredo Deza for the work on making an Ubuntu virtual machine for the book, your expertise was greatly appreciated.

A very large thanks to Liza Daly, for providing good feedback on some really early, and rough, parts of our book. This was tremendously helpful.

Special thanks to Jeff Rush for his advice and reference material on Buildout, Eggs, and Virtualenv.

Thanks to Aaron Hillegass who has given me some great advice and help along the way, and who has a great training company, Big Nerd Ranch. He is a special person, who I am lucky to have met. Thanks to Mark Lutz, who I had the pleasure of taking a Python training course from, and who has written some great books on Python.

Thanks to the people in the Python community in Atlanta, and the members of PyAtl: *http://pyatl.org*; you have all taught me a great deal. Rick Copeland, Rick Thomas, Brandon Rhodes, Derek Richardson, Jonathan La Cour, a.k.a Mr. Metaclass, Drew Smathers, Cary Hull, Bernard Matthews, Michael Langford, and many more I have forgotten to mention. Brandon and Rick Copeland in particular have been very helpful and are awesome Python programmers. You can read Brandon's blog at *http://rhodesmill.org/brandon/*.

Thanks to Grig Gheorghiu for giving us expert sysadmin and testing advice and for giving us a kick in the butt when we needed one.

Thanks to my former employer Racemi, and the CTO/Founder, Charles Watt. I learned a lot from you and was glad you knew which competitive buttons to push. Just remember I will kick your butt at writing code, a 26-mile run, or a 200-mile bike ride any day, just tell me where and when.

Thanks to Dr. Nanda Ganesan, who was a great mentor in graduate school at CSULA. You taught me a lot about information technology and life and encouraged me to think big.

Thanks to Dr. Cindy Heiss, who was my professor for my undergraduate degree in nutritional science. You got me started on web development, encouraged me to believe in myself, and ultimately made an impact on my life, thanks!

Thanks to Sheldon Blockburger, who let me try out for Division I decathlon as a walk-on at Cal Poly SLO. Even though I didn't make the team, you showed me how to be a fierce competitor and warrior, and taught me the self-discipline to run 200-meter intervals by myself. I believe weekly 200-meter interval workouts make me a better software engineer.

There were many other people who helped tremendously along the way, including Jennifer Davis, yet another friend from Caltech, who gave us some great feedback; some of my friends and coworkers at Turner; Doug Wake, Wayne Blanchard, Sam Allgood, Don Voravong; some of my friends and coworkers from Disney Feature animation, including Sean Someroff, Greg Neagle, and Bobby Lea. Greg Neagle in particular taught me a lot about OS X. Also, thanks to J.F. Panisset, who I met at Sony Imageworks, for teaching me quite a bit about engineering in general. Although he is now a CTO, he is another rare catch for any company.

I would like to thank a few others who made some important contributions: Mike Wagner, Chris McDowell, and Shaun Smoot.

Thanks to Bruce J. Bell, who I worked with at Caltech. He taught me quite a bit about Unix and programming over the years, and I owe him greatly for it. You can read his material here: *http://www.ugcs.caltech.edu/~bruce/*.

Also thanks to Alberto Valez, my boss at Sony Imageworks, for being possibly the best boss I ever had and giving me the chance to completely automate my job. Thanks to film editor Ed Fuller, who helped with advice on the book, and was a good friend during this process.

Thanks to many people in the Python community. First, thanks to Guido van Rossum for writing a great language, for being a great leader, and for being patient with me when I asked for advice on the book. There are so many rock stars in the Python community who crank out useful tools that I use everyday. They include Ian Bicking, Fernando Perez and Villi Vainio, Mike Bayer, Gustavo Niemeyer, etc. Thanks! Thanks to the great book by David Beazely, and his fantastic tutorial at PyCon 2008 on Generators. Thanks to other writers about Python and systems administration as well. You can find links to their work here: *http://wiki.python.org/moin/systems_administration*. Thanks also to the Repoze crew: Tres Seaver and Chris McDonough (*http://repoze.org/index.html*).

Special thanks to the great tools, advice, and tolerance from Phillip J. Eby on the setuptools section. Also, thanks to Jim Fulton who tolerated my questions about ZODB and buildout, with a crazy schedule. Additional thanks to Martijn Fassen, who taught me about ZODB and Grok. If you want to see the future of Python web development, check out Grok: *http://grok.zope.org/*.

Thanks to *Red Hat Magazine* staff, Julie Bryce, Jessica Gerber, Bascha Harris, and Ruth Suehle, for letting me try out ideas we used in this book in the form of articles. Also, thanks to Mike McCrary at IBM Developerworks, for letting me write articles to try out ideas we used in this book.

I want to thank the multitudes of people who told me at one point in my life that I couldn't do something. At almost every step, I have met discouraging people who told me everything from I would never get into the college I wanted to to I would never learn to program. Thank you for giving me the extra motivation to succeed at my dreams.

Humans can create their own reality if they truly believe in themselves, and I would encourage everyone to give themselves a chance to do what they truly want to do.

Finally, thanks to O'Reilly and Tatiana Apandi, for believing in my original pitch for a book on Python and Systems Administration. You took a chance and believed in me and Jeremy, and I thank you for that. Although Tatiana left O'Reilly near the end of our book to pursue her dreams, her impact was still felt. I also want to thank our new editor Julie Steele, who has been supportive and helpful every step of the way. You have really provided a sea of calm that I personally appreciated greatly. I look forward to hearing great things in the future from Julie, and I'm excited to work with her again.

Jeremy's Acknowledgments

After reading Noah's list of thanks, it makes me feel both ungrateful, because I know my list won't be that long, and at a loss, because I think he covered nearly everyone that I wanted to thank.

First, I must thank my God, through Whom I can do all things and without Whom, I can do nothing.

First in an earthly sense, I thank my wife, Debra. You kept the children engaged with other activities while I worked on the book. You enforced the so-often reapeated rule "Don't bother Daddy while he's working on his book." You encouraged me when I needed it, and you also gave me a lot of space, which is what I needed most. Thank you. I love you. I could not have written this book without you.

I also must thank my sweet children, Zane and Justus, for their patience through the process of my writing this book. I missed out on a lot of trips to Stone Mountain with you both. I still put one of you to bed most nights, but I missed out on staying in there long enough to fall asleep with you, like I used to. I missed out on the last several weeks of Kid's Rock on Wednesday nights. I missed out on so much, but you bore it patiently. So, thank you for your patience. And thank you for your excitement as you hear that I'm almost done with the book. I love you both.

I want to thank my parents, Charles and Lynda Jones, for their support through the course of my writing this book. But more than that, I want to thank them for being a living example of a strong work ethic, of earning everything you have, of working hard to better yourself, and of spending money wisely. Those are lessons I hope to pass on to Zane and Justus.

Thank you to Noah Gift, my coauthor, for getting me into this mess. It has been hard, harder than I thought and definitely one of the hardest things I've ever done in my life. I think it says a lot about a person when you work on something like this with him and at the end, you can still think of him as your friend. Thanks, Noah. This book would not have begun if not for you.

I want to thank our team of reviewers. I think that Noah has already thanked all of you, but I want to thank everyone that I can: Doug Hellman, Jennifer Davis, Shannon JJ Behrens, Chris McDowell, Titus Brown, and Scott Leerseen. You guys were awesome. There were times when I thought that I had something spot-on and you readjusted my thinking. Or you just brought a completely different perspective to the book and helped me see my work through a different set of eyes. (That was mostly you, Jennifer. If the text processing chapter is useful for sysadmins, it's mostly because of you.) Thank you all.

I also want to thank our editors, Tatiana Apandi and Julie Steele. You guys handled the hard stuff, freeing us up to work on the book. You both eased our burden along the way. Thank you.

I'd also like to thank Fernando Perez and Ville Vainio for your amazing feedback. I hope I've done IPython justice. And thank you for IPython. I feel like I couldn't live without it.

Thank you Duncan McGreggor, for helping me get the Twisted code in better shape. Your comments were extemely helpful. And thank you for working on Twisted. It is an amazing framework. I hope to use it more, real soon now.

I thank Bram Moolenaar and everyone who has ever worked on the Vim editor. Almost every word and XML tag that I wrote flowed through capabilities Vim. I picked up a few tricks along the way that I'll incorporate into my daily editing habits. Vim made me more productive. Thank you.

I also want to thank Linus Torvalds, the Debian folks, the Ubuntu folks, and anyone else who has ever worked on Linux. Almost every word that I typed was done on Linux. You made it incredibly simple to set up new environments and test different things. Thank you.

Finally, but by no means least, I want to thank Guido van Rossum and everyone who has ever done any work on Python. I have been benefitting from your work for a number of years now. I was hired for my last two jobs because of Python. Python, the language, and Python, the community, have been both a great joy for me since I started working with it sometime around 2001–2002. Thank you. Python has been very good to me.

Table of Contents

CHAPTER 1
Introduction

Why Python?

If you are a system administrator, it is likely that you have encountered Perl, Bash, ksh, or some other scripting language. You may have even used one or more yourself. Scripting languages are often used to do repetitive, tedious work at a rate and with an accuracy that far surpass what you could accomplish without them. All languages are tools. They are simply a means to get work done. They have value only insofar as they help you get your job done better. We believe that Python is a valuable tool, specifically because it enables you to get your work done efficiently.

So is Python better than Perl, Bash, Ruby, or any other language? It's really difficult to put that sort of qualitative label on a programming language, since the tool is so closely tied to the thought process of the programmer who is using it. Programming is a subjective, deeply personal activity. For the language to be excellent, it must fit the person using it. So we're not going to argue that Python is better, but we will explain the reasons that we believe Python can be an excellent choice. We'll also explain why it is a great fit for performing sysadmin tasks.

The first reason that we think that Python is excellent is that it is easy to learn. If a language can't help you become productive pretty quickly, the lure of that language is severely diminished. Why would you want to spend weeks or months studying a language before you are able to write a program that does something useful? This is especially the case for sysadmins. If you are a sysadmin, your work can pile up faster than you can unpile it. With Python, you can start writing useful scripts literally in hours rather than in days or weeks. If you can't learn a language quickly enough to start writing scripts with it almost immediately, you should strongly question whether you should be learning it.

However, a language that is easy to learn but doesn't allow you to do fairly complex tasks isn't worth much either. So the second reason that we consider Python to be an excellent programming language is that, while it lets you start simply, it also allows you to perform tasks that are as complex as you can imagine. Do you need to read through a logfile line by line and pull out some pretty basic information? Python can handle

that. Or do you need to parse through a logfile, extract every piece of information that it provides, compare usage from each IP address in this logfile to usage in each logfile (which are stored in a relational database, by the way) from the past three months, and then store the results to a relational database? Sure, Python can do that as well. Python is being used on some pretty complex problems, such as analysis of genomic sequences, multithreaded web servers, and heavy duty statistical analysis. You may never have to work on anything like that, but it's nice to know that when you need to do complex things, the language is able to work with you.

Additionally, if you are able to perform complex operations, but the maintainability of your code suffers along the way, that isn't a good thing. Python doesn't prevent code maintenance problems, but it does allow you to express complex ideas with simple language constructs. Simplicity is a huge factor in writing code that is easy to maintain later. Python has made it pretty simple for us to go back over our own code and work on it after we haven't touched it in months. It has also been pretty simple for us to work on code that we haven't seen before. So the language, that is the language's syntax and common idioms, are clear and concise and easy to work with over long periods of time.

The next reason we consider Python to be an excellent language is its readability. Python relies on whitespace to determine where code blocks begin and end. The indentation helps your eyes quickly follow the flow of a program. Python also tends to be "word-based." By that we mean that while Python uses its share of special characters, features are often implemented as keywords or with libraries. The emphasis on words rather than special characters helps the reading and comprehension of code.

Now that we've outlined a few of Python's benefits, we'll show some comparisons of code examples in Python, Perl, and Bash. Along the way, we'll also look at a few more of Python's benefits. Here is a simple example, in Bash, of showing all the combinations of 1, 2 and a, b:

```
#!/bin/bash

for a in 1 2; do
    for b in a b; do
        echo "$a $b"
    done
done
```

And here is a comparable piece of Perl:

```
#!/usr/bin/perl

foreach $a ('1', '2') {
    foreach $b ('a', 'b') {
        print "$a $b\n";
    }
}
```

This is a pretty simple nested loop. Let's compare these looping mechanisms with a `for` loop in Python: