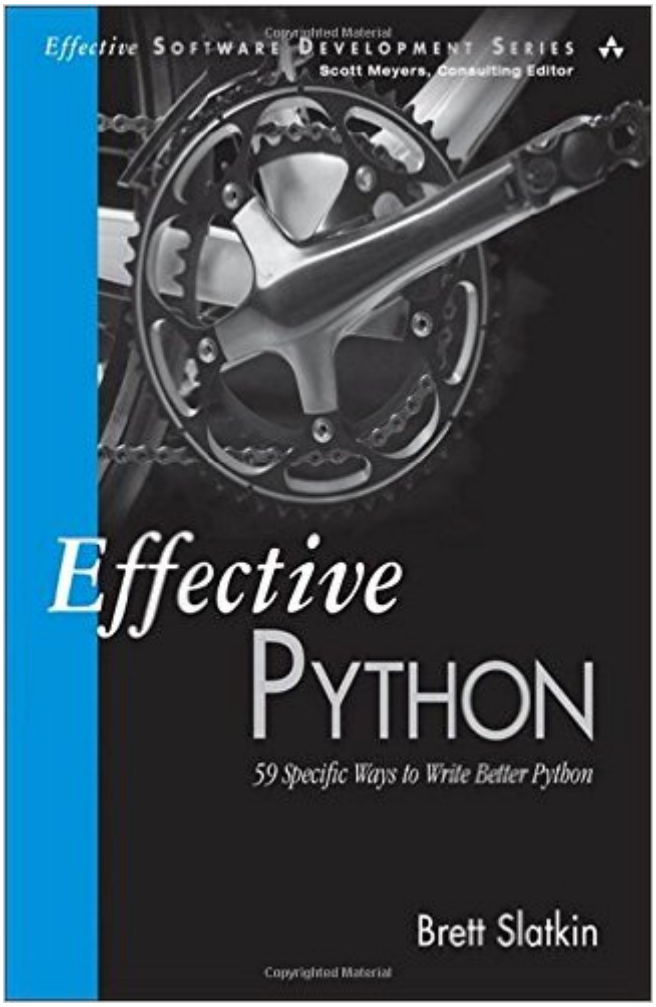
Effective Python: 59 Specific Ways To Write Better Python (Effective Software Development Series)



Synopsis

It’s easy to start writing code with Python: that’s why the language is so immensely popular. However, Python has unique strengths, charms, and expressivity that can be hard to grasp at first -- as well as hidden pitfalls that can easily trip you up if you aren’t aware of them. Effective Python will help you harness the full power of Python to write exceptionally robust, efficient, maintainable, and well-performing code. Utilizing the concise, scenario-driven style pioneered in Scott Meyers’s best-selling Effective C++, Brett Slatkin brings together 59 Python best practices, tips, shortcuts, and realistic code examples from expert programmers. Through realistic examples, Slatkin uncovers little-known Python quirks, intricacies, and idioms that powerfully impact code behavior and performance. You’ll learn how to choose the most efficient and effective way to accomplish key tasks when multiple options exist, and how to write code that’s easier to understand, maintain, and improve. Drawing on his deep understanding of Python’s capabilities, Slatkin offers practical advice for each major area of development with both Python 3.x and Python 2.x. Coverage includes: Algorithms Objects Concurrency Collaboration Built-in modules Production techniques And more Each section contains specific, actionable guidelines organized into items, each with carefully worded advice supported by detailed technical arguments and illuminating examples. Using Effective Python, you can systematically improve all the Python code you write: not by blindly following rules or mimicking incomprehensible idioms, but by gaining a deep understanding of the technical reasons why they make sense.

Book Information

Series: Effective Software Development Series
Paperback: 256 pages
Publisher: Addison-Wesley Professional; 1 edition (March 8, 2015)
Language: English
ISBN-10: 0134034287
ISBN-13: 978-0134034287
Product Dimensions: 7 x 0.7 x 9 inches
Shipping Weight: 1 pounds (View shipping rates and policies)
Average Customer Review: 4.7 out of 5 starsÂ Â See all reviewsÂ (64 customer reviews)
Best Sellers Rank: #56,792 in Books (See Top 100 in Books) #72 inÂ Books > Computers & Technology > Programming > Languages & Tools > Python #166 inÂ Books > Computers & Technology > Programming > Web Programming #242 inÂ Books > Textbooks > Computer

Science > Programming Languages

Customer Reviews

Most Python developers are familiar with the widespread mantra that "[t]here should be one-- and preferably only one --obvious way to do it.Ã¢Â Â• In Python itself, it is often the case that there is one obvious and well-established One Way To Do ItÃ¢Â„Â¢. But as a community Ã¢Â Â“ and even as organizations and teams Ã¢Â Â“ we frequently fail to make use of customary One WayÃ¢Â„Â¢ approaches. We contribute code in a variety of styles, and to varying degrees of re-inventing Python's included wheels.This is hardly a fault of the language development community, which is mindful of the One WayÃ¢Â„Â¢ principle in design discussions. Nor is it due to lack of community-established practices, which are spread around in PEPs, high-impact open source projects, mailing list advice, and the blogs of prominent community members.As developers, we perhaps collectively tend to neglect reading code, following our communities, improving our fundamentals, and thinking about how we improve our approach to building software. And perhaps it's because these activities seem to draw away from our favorite activity: building stuff.But, of course, the activity of writing code is just one component of building stuff. Knowledge of our tools and how to best use them helps us to build more stuff in less time, spend less time debugging and fixing our old stuff, and work more easily with others.Effective Python is a time-efficient way to learn Ã¢Â Â“ or remind yourself Ã¢Â Â“ what the best practices are and why we use them. ItÃ¢Â Â™s a concise book of practical techniques to write maintainable, performant and robust code using practices widely accepted in the community.In the style of Scott Meyer's "Effective" series, the book is split into 59 sections (referred to as "items"), each covering an important topic conducive to high-quality Python code. These items connect to a larger theme within each chapter, such as "Pythonic Thinking", "Metaclasses and Attributes", or "Concurrency and ParallelismÃ¢Â Â•.Each item introduces a common scenario in Python development, often followed by a simple way to address the problem at hand. The book frequently starts out with simplistic code that developers may be temped to use to solve a particular problem, points out issues with it, and incrementally improves the code to incorporate best practices. The alternative solution(s) proposed are often both simpler and more robust than the seemingly straightforward, but flawed, initial approach. A few sections substantiate the preferred approach with simple explanations of what Python is doing under the hood.The broad set of topics addressed include readability ("Prefer Helper Classes Over Bookkeeping with Dictionaries and Tuples"), protecting memory ("Use tracemalloc to Understand Memory Usage and Leaks"), writing defensive code ("Know the Differences Between bytes, str, and

unicode"), creating powerful classes ("Annotate Class Attributes with Metaclasses"), using parallelism ("Consider Coroutines to Run Many Functions Concurrently"), and development practices ("Profile Before Optimizing").Unlike many books on programming languages, Effective Python leaves installation instructions and a basic language tutorial to books focused on Python beginners, and assumes that the reader has cursory knowledge of the language Ã¢Â Â” or is, occasionally, willing to look something up.To help the reader understand some of the issues in a broader context, many of the items refer to other sections for explanations of out-of-scope but related topics. The contained nature of the items makes them equally suitable for random access (skipping from one section to another) as for back-to-back reading.This book distills the best practices spread around the Ã¢Â Âˆnet. Reading it is a time-effective way to make sure youÃ¢Â Â™re up to speed with writing great Python code.

Very good book. Starts with basics, but it quickly goes into ’magic’ territory, so whether you’re a veteran or just wanting to see what’s special about Python, you will find some new approaches.The examples are clear. They are not some ’dry’ Class X, method Y sort of examples, they tend to use examples that better illustrate the scenario; so if you at first don’t get the author’s explanation, you might see what’s going on because of the connotations that come with sensible examples.The formatting and layout are fairly good, especially with syntax highlighting code. There are few goofy spots where text talks about code that’s displayed on the next page, so to relate one to the other you need to flip pages back’n’forth, and some tables/boxes spill over to the next page for like two lines. But that’s just minor nit-picking.My biggest ’complaint’ is about what’s NOT in the book. There is only a slight mention of itertools, or any functional programming concepts, that are so neatly baked into Python. There is also very slim amount of information on testing, which I would like to read more about, as it’s become very commonplace. I would love to see a second edition of this book with extended sections on the aforementioned topics.

If you browse Ã¢Â Â™s catalog for Python books, you end up with several pages of titles that introduce you to the language, those that get you from hello world up to classes and a little taste of the standard library. What a struggle for those that already know what a closure is to find something worth reading! BrettÃ¢Â Â™s book is one of those that experienced developers hope to see showing up as on top of the results instead: a title that covers those lesser used features that teach you to write better, robust code.This thin book, 250 pages only, is split into 59 recipes, short and concise reviews of real life scenarios that every Python developer faces. Each focuses on a specific

problem and starts with a brief overview . Next, we find an inefficient, not Pythonic solution that most of the intermediate programmers would come up with to solve it. What follows is a discussion that gets the reader, step by step, from this initial solution to an elegant and robust one. Brett clearly explains the benefits and the issues raised by each intermediate solution. Finally, he sails us to the Pythonic way introducing features and techniques.As stated, the book is not meant to be read by a beginner. If the reader does not have a strong Python knowledge, he will struggle. Concepts like comprehension lists, closures and decorators must be well understood already.The small size of the book could fool the reader. ItÃ¢Â Â™s thin, but intense, plenty of things to learn. If the reader doesnÃ¢Â Â™t get lost, by the time he gets to the back cover, he will certainly start developing better code.A real pearl. Every Python developer should jealously own a copy.Suggested readings:Learning Python Design PatternsLearning Cython ProgrammingAs usual, you can find more reviews on my personal blog: http://books.lostinmalloc.com Feel free to pass by and share your thoughts!

Project. The only book you need to start coding in Python immediately Maya Python for Games and Film: A Complete Reference for Maya Python and the Maya Python API Python: Python Programming For Beginners - The Comprehensive Guide To Python Programming: Computer Programming, Computer Language, Computer Science (Machine Language) Deep Learning: Recurrent Neural Networks in Python: LSTM, GRU, and more RNN machine learning architectures in Python and Theano (Machine Learning in Python) Unsupervised Deep Learning in Python: Master Data Science and Machine Learning with Modern Neural Networks written in Python and Theano (Machine Learning in Python)