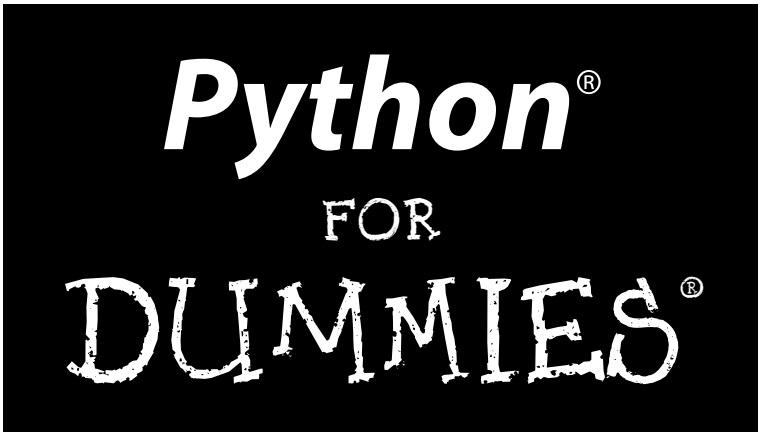
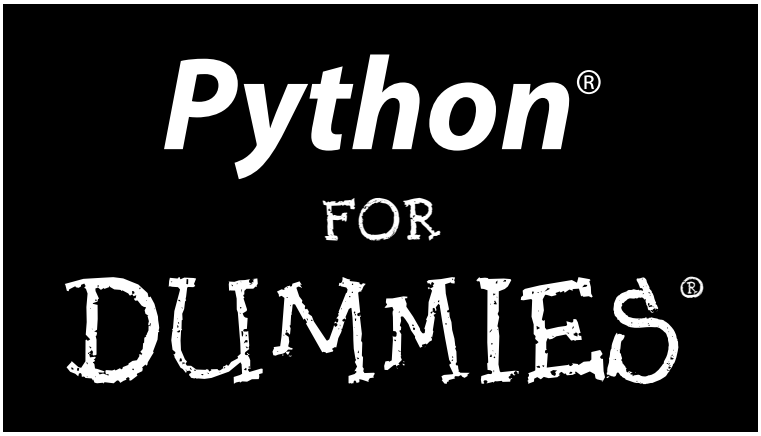


by Stef Maruch and Aahz Maruch

WILEY
Wiley Publishing, Inc.



by Stef Maruch and Aahz Maruch

WILEY
Wiley Publishing, Inc.

Python® For Dummies®
Published by
Wiley Publishing, Inc.
111 River Street
Hoboken, NJ 07030-5774
www.wiley.com

Published by Wiley Publishing, Inc., Indianapolis, Indiana

Published simultaneously in Canada

Library of Congress Control Number: 2006924031

ISBN-13: 978-0-471-77864-6

ISBN-10: 0-471-77864-8

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

1O/ST/QY/QW/IN

About the Authors

Stef Maruch got her hands on an original 128K Mac in 1984 and has been writing about computers ever since. She has over fifteen years' experience in instructional design, writing, and editing end-user computer manuals, including tutorials and user's guides for Apple Newton, HyperCard and HyperTalk, and DVD Studio Pro.

Aahz Maruch is a writer, trainer, and consultant who has been using Python for more than seven years. He has been using computers professionally for 20 years, and his background includes stints of high-end tech support, systems administration, and programming. Aahz is currently working as a programmer for a company with a Web-based application.

The authors can be reached at `authors@pythonfood.com`.

Dedication

Stef: I dedicate this book to my parents, Don and Betty Jones. You have always believed in me, even at times when I was quite improbable.

Aahz: I dedicate this book to the Python community. I hated programming until I learned Python (yes, for more than 20 years). I hope this book brings the joy of Python to many people.

This book is also dedicated to the Flying Spaghetti Monster.

Authors' Acknowledgments

Many people have helped us and supported us in writing this book. There are too many to mention all of them by name, so we want to start by thanking all the people we don't name here — all the family and friends and community who have sustained us.

Paula Anderson, Naomi Tilsen, Joyce Wermont, and Maggie Young provided much appreciated emotional support.

Thanks to our editors at Wiley:

- Acquisition editors Terri Varveris, Tiffany Franklin, and Kyle Looper, who shepherded these first-time *For Dummies* authors with just the right balance of patience and whip-cracking.
- Project editor Pat O'Brien, who provided invaluable assistance in e-mails that were often time-stamped with hours well past the time any less-dedicated person would have been in bed.
- Copy editor Andy Hollandbeck, who improved the book with his keen grasp of the beginner's mind, light-hearted prose, and Monty Python quotations.

High praise also to the production staff at Wiley, who are doing such great work with an extraordinarily complex and flexible book design.

Our technical editor, David Goodger, vastly improved our book with his edits and suggestions.

We feel fortunate to have our agents, David Fugate, who supported us expertly and patiently through the lengthy acquisition process, and Carole McClendon, who provided support at a critical juncture. Thanks also to the efficient staff at Waterside Productions.

Many people gave us advice and help while we were writing:

- Don and Betty Jones provided invaluable advice from the point of view of programming beginners
- Aahz's coworkers at Printra (`http://printra.net/`), especially Tony Lownds

The community of Python programmers on `comp.lang.python` and `tutor@python.org` not only helped Stef learn Python but also tirelessly work every day to promote Python and help make it accessible. This book wouldn't be possible without them.

The folks who maintain `www.python.org` and run the Python Software Foundation provide a critical service without which Python would be poorer.

Millions of people volunteer their time and efforts to make the Open Source movement a powerful force for good in the computer industry.

We also want to thank each other. Living together and writing a book is stressful, but we're glad we did this.

And, of course, none of it would be possible without Guido.

Publisher's Acknowledgments

We're proud of this book; please send us your comments through our online registration form located at www.dummies.com/register/.

Some of the people who helped bring this book to market include the following:

Acquisitions, Editorial, and Media Development

Project Editor: Pat O'Brien

Acquisitions Editor: Kyle Looper

Copy Editor: Andy Hollandbeck

Technical Editor: David Goodger

Editorial Manager: Kevin Kirschner

Media Development Specialists: Angela Denny, Kate Jenkins, Steven Kudirka, Kit Malone

Media Development Coordinator: Laura Atkinson

Media Project Supervisor: Laura Moss

Media Development Manager: Laura VanWinkle

Editorial Assistant: Amanda Foxworth

Sr. Editorial Assistant: Cherie Case

Cartoons: Rich Tennant (www.the5thwave.com)

Composition Services

Project Coordinator: Tera Knapp

Layout and Graphics: Claudia Bell, Denny Hager, Jake Mansfield, Barbara Moore, Barry Offringa, Heather Ryan

Proofreaders: Laura Albert, Susan Moritz, Techbooks

Indexer: Techbooks

Publishing and Editorial for Technology Dummies

Richard Swadley, Vice President and Executive Group Publisher

Andy Cummings, Vice President and Publisher

Mary Bednarek, Executive Acquisitions Director

Mary C. Corder, Editorial Director

Publishing for Consumer Dummies

Diane Graves Steele, Vice President and Publisher

Joyce Pepple, Acquisitions Director

Composition Services

Gerry Fahey, Vice President of Production Services

Debbie Stailey, Director of Composition Services

Contents at a Glance

Table of Contents

Introduction

Congratulations! You're ready to discover the easiest-to-read powerful programming language — or maybe the most powerful, easy-to-read programming language. That's Python, of course.

With *Python For Dummies*, you can ferret out just a little or a lot. And with Python, you can write a little program that picks a random quote from a file, or you can write a set of programs that runs a complex business.

This book is for you whether you're a student, you're a hobbyist, you need to understand more about what your programmer co-workers are talking about, or you're taking the first steps on a new career path.

Python For Dummies gives you everything you need to get to an advanced-beginner level of Python programming. And it points you to other resources so you can take your Python programming skills even further.

About This Book

Python For Dummies is a reference book, which means you can read it in any order, and you don't have to read every chapter or section. However, to some extent, later chapters about more complex Python features rely on information introduced in earlier chapters. So if you don't understand something you see in a later chapter, go to Chapter 3, or go to the chapter on that feature to find out more. You can also look in the index to find a term or feature you want to know more about.

Conventions Used in This Book

This book contains Python code examples. All code examples are in `monospaced font` so they are easy to recognize. Anything that you need to type is also indicated in `monospaced font` so you know exactly which commas should be typed and which commas are part of the surrounding sentence.

Python interactive mode examples include this prompt: >>>. If you don't see the prompt, you can assume the code was written in a text editor.

Foolish Assumptions

We make the following assumptions about readers of this book:

- **You know how to use your computer and its operating system.**

  It's helpful but not necessary to know how to set environment variables on your computer. It's also helpful to have a Web browser with access to the Internet.

- **You have and know how to use a text editor that can produce plain ASCII text or files that end with the `.txt` extension.**

  If you don't have a text editor that can do this, we include instructions for setting up Python's IDLE programming environment to work with the examples in this book.

- **You have had a minimal amount of exposure to programming.**

  We really do mean *minimal*. If you had a programming class in high school, or wrote a few BASIC programs at one time, or even if you have used HTML tags, that counts.

  If you have absolutely no experience with programming, you can still find out plenty from this book, but we recommend that you also look at a book or Web tutorial designed to introduce programming to beginners. You'll benefit from the extended explanations of some concepts that we don't have the space to discuss in detail here.

- **You might have done some programming in another language.**

  Programming knowledge is not required for this book, but people who have programmed in other languages have their own sets of issues when transitioning to Python, and we provide some material for such people.

- **You know little to nothing about Python.**

  If you know Python, this book will still be helpful as a reference or a source of tips and tricks you may not be aware of.

How This Book Is Organized

This book gives you an overview of Python; the lowdown about all of its major parts, structures, and libraries; and a glimpse into some more advanced features. You also find out where to go to discover more.

Part I: Getting Started

In this part, we introduce Python and situate it among the myriad other programming languages available. Python is good for some things and not for others; you find out which is which. We provide a hands-on introduction to some of Python's abilities, using its helpful interactive mode and its IDLE programming environment. We briefly describe each of Python's basic building blocks and show how all these blocks come together by dissecting a working program. We sketch an overview of how professional programmers design programs and debug code and show you how to put these practices to work to make your own programming life easier.

Part II: Building Blocks

Python has six basic data types and many ways to work with each of them. In this part, we describe how to work with strings (chunks of text), numbers, lists and tuples (both of which store multiple data elements), dictionaries (which associate one element with another), and sets (which always contain unique elements, never duplicates).

Part III: Structures

Python code usually comes in chunks, both small and big, and each chunk does a particular thing. This part also includes a brief introduction to some advanced features and the new features of Python 2.5.

Part IV: Libraries

Python comes with everything you need to write a very powerful program, and other people have already solved lots of programming conundrums for you. Its libraries include primary services such as communication with the operating system, text processing tools, various ways of reading and writing information to disk, and Internet access methods.

Part V: The Part of Tens

All *For Dummies* books include The Part of Tens. In this part, we give you ten useful but not-so-obvious programming idioms and ten resources where you can find out more about Python.

Part VI: Appendixes

Here you find instructions on how to install Python and its documentation, as well as a list of new features introduced with each new version of Python since 2.0.

Icons Used in This Book

Icons appear throughout the book to indicate special material. Here's what they mean:

A Tip explains how to do something a little bit more easily and efficiently.

A Warning gives you a heads-up about tricky stuff or common mistakes that might cause data loss or some other sort of headache. It's best to read Warnings to make sure a tricky feature doesn't "getcha."

A Technical Stuff icon flags text that's of interest to readers who like to know about the inner workings or history of a subject. You don't need to read Technical Stuff material. After you've internalized a little about a subject, reading this text might help you understand it from a different angle.

Remember icons highlight important concepts or pieces of information to keep in mind.

Where to Go from Here

If you want an overview of Python's history and what it can do, go to Chapter 1. If you're new to Python and want to start working with it right away, go to Chapter 2. If you want a brief overview of all of Python's building blocks, go to Chapter 3. If you know some Python and you want a refresher or additional info on some of its tools, go to the specific chapters you're interested in.

Part I
Getting Started

The 5th Wave By Rich Tennant

"The engineers lived on Jolt and cheese sticks putting this product together, but if you wanted to just use 'cola and cheese sticks' in the Users Documentation, that's okay too. We're pretty loose around here."

In this part . . .

You get an overview of the Python programming language, an introduction to its interactive and developer environment, and a walkthrough of the building blocks that make up Python programs.

Chapter 1 describes the history of Python and all the exciting things it's being used for today. You find out why computers are both the fastest and dumbest things around. Best of all, you discover why it's called *Python* anyway.

Chapter 2 lets you talk to Python via its interactive mode and IDLE environment. You write a few basic programs and find out how to get Python to carry out commands for you, how to get Python to tell you things, and how to import tools that let you do even more.

Chapter 3 introduces you to Python's data types and code blocks, the chunks you use to build programs.

Chapter 4 shows you a working program. You see how all the chunks of a Python program talk to each other, and you find out something about the design philosophies behind Python programs.

Chapter 5 lets you try on a programmer's hat to understand how programmers work and why they make the design decisions they do. (Unfortunately, it doesn't explain the relevance of caffeinated sodas to this process — you'll have to figure that out for yourself.) There's also a very useful section on strategies for debugging programs, which is a huge part of every programmer's job.

Chapter 1

Introducing Python

In This Chapter

- The history of Python
- What people use Python for
- Useful concepts for Python programming

Welcome to Python! If you're the type of person who wants to know what you're getting into, this chapter is for you. We give you a quick history of Python and its community of developers. You find out what Python is and isn't good for (the "is" section is much longer than the "isn't" section) and the most important principles of good Python programming. If you're new to programming, you'll see how it's very similar to a task you're probably familiar with.

The Right Tool for the Job

Python is a general-purpose, high-level language that can be extended and embedded (included in applications as a tool for writing macros). That makes Python a smart choice for many programming problems, both small and large, and not so good for a couple of computing tasks.

Good uses of Python

Python is ideal for projects that require quick development. It supports multiple programming philosophies, so it's good for programs that require flexibility. The many packages and modules already written for Python provide versatility and save you time.

The story of Python

Guido van Rossum created Python and is affectionately bestowed with the title "Benevolent Dictator For Life" by the Python community. In the late 1980s, Guido liked features of several programming languages, but none of them had all the features he wanted. Specifically, he wanted a language that had the following features:

- **Scripting language:** A *script* is a program that controls other programs. Scripting languages are good for quick development and prototyping because they're good at passing messages from one component to another and at handling fiddly stuff like memory management so that the programmer doesn't have to. Python has grown beyond scripting languages, which are used mostly for small applications. The Python community prefers to call Python a *dynamic programming language*.
- **Indentation for statement grouping:** Python specifies that several statements are part of a single group by indenting them. The indented group is called a *code block*. Other languages use different syntax or punctuation for statement grouping. For example, the C programming language uses `{` to begin an instruction and `}` to end it. Indentation is considered good practice in other languages also, but Python was one of the first to *enforce* indentation. Indentation makes code easier to read, and code blocks set off with indentation have fewer begin/end words and punctuation to accidentally leave out (which means fewer bugs).
- **High-level data types:** Computers store everything in 1s and 0s, but humans need to work with data in more complex forms, such as text. A language that supports such complex data is said to have *high-level data types.* A high-level data type is easy to manipulate. For example, Python strings can be searched, sliced, joined, split, set to upper- or lowercase, or have white space removed. High-level data types in Python, such as lists and dicts (which can store other data types), encompass much more functionality than in other languages.
- **Extensibility:** An extensible programming language can be added to. These languages are very powerful because additions make them suitable for multiple applications and operating systems. Extensions can add data types or concepts, modules, and plug-ins. Python is extensible in several ways. A core group of programmers works on modifying and improving the language, while hundreds of other programmers write modules for specific purposes.
- **Interpreted:** Interpreted languages run directly from source code that humans generate (whereas programs written in *compiled languages,* like C++, must be translated to machine code before they can run). Interpreted languages run more slowly because the translation takes place on the fly, but development and debugging is faster because you don't have to wait for the compiler. Interpreted languages are easier to run on multiple operating systems. In the case of Python, it's easy to write code that works on multiple operating systems — with no need to make modifications.

  People argue over whether Python is an interpreted or compiled language. Although Python works like an interpreted language in many ways, its code is compiled before execution (like Java), and many of its capabilities run at full machine speed because they're written in C — leaving you free to focus on making your application work.