

book reviews



ELIZABETH ZWICKY, WITH JASON
DUSEK, SAM F. STOVER, AND RIK
FARROW

THE CARTOON GUIDE TO STATISTICS

Larry Gonick and Woollcott Smith

Collins Reference, 2005. 223 pages.
ISBN 978-0-06-273102-9

HEAD FIRST STATISTICS

Dawn Griffiths

O'Reilly, 2008. 655 pages.
ISBN 978-0-596-52758-7

THE MANGA GUIDE TO STATISTICS

Shin Takahashi

No Starch, 2008. 211 pages.
ISBN 978-1-59327-189-3

Who would have thought that the graphic introduction to statistics was a genre? But in my search for appropriate ways to introduce system administrators to statistics, I came across no fewer than three books that are trying to help out with pictures.

Here's a quick reference to the three. *The Cartoon Guide* is a classic statistics book, embellished with lots of pictures. It covers a full statistics class, it's moderately easy to read, and there's fun history included. *Head First Statistics* is the best of the lot, fully using the power of pictures to cover most of the same territory (minus the history) much more palatably. *The Manga Guide* covers the least, but boy does it have that authentic manga flavor, which is pretty amusing.

The Cartoon Guide is the hardest going of the three. (Probably not coincidentally, it's also the oldest.) It's the only one that touches at all on history of statistics. If you're good with math and would like it minimally spiced up, this is the way to go.

The Manga Guide is ideal if you're disappointed by the lack of romance in your technical books. It does have a romantic subplot. A slightly creepy romantic subplot, if you ask me, since it revolves around a high school girl with a crush on one of her father's co-workers, but as manga goes, it's not bad. It would help if you had some familiarity with the visual conventions of manga. Again, it's not outrageous, but if you're still prone to be bewildered by characters who change size and shape to reflect the way they feel, you're going to spend as much time trying to puzzle out the illustrations as trying to puzzle out the statistics. It does flow the reader right through the content, and it has a handy guide to using Excel to do statistics, which none of the other books does. Its real weak point is a lack of coverage for quartiles and box-and-whisker plots; quartiles are extremely handy for working with a lot of non-normally distributed data that you see in computer systems.

Head First Statistics covers all that stuff, and it does it slowly enough that anybody with basic mathematical competence should be able to cope. It's a thorough, palatable introduction to statistics and would be my choice for most people.

THE DAEMON, THE GNU, AND THE PENGUIN

Peter H. Salus

Reed Media Services, 2008. 177 pages.
ISBN 978-0-9790342-3-7

This is a small history of free and open software. It is admiring without being sycophantic about any of the players, and it balances as much as possible among the often warring factions involved. I learned parts of the history that I didn't know, and I enjoyed the ride. This would be a great gift for somebody young who's just coming into the world of open source and would like a brief historical tour that doesn't have too many axes to grind.

THE BACK OF THE NAPKIN

Dan Roam

Portfolio, 2008. 272 pages.
ISBN 978-1-59184-199-9

Here's a book that doesn't even aim for fair and balanced. This is a passionate defense of the diagram as a way of thinking and talking about problems, with a set of clear procedures that should make it accessible to anybody (since drawing ability is not required). I enjoyed it greatly.

There are two situations at which this book takes particular aim. The first is selling an idea: trying to convince people that your solution is a good one. This is something we all do, whether or not we are

salespeople, and if it's important to you, there's a great deal of insight here.

The second situation is creating solutions: getting your head around a pile of information and turning it into an understanding of the situation. The book offers some tools for doing that, alone or in a group.

I have two related misgivings. First, did I mention this is a passionate defense of a particular procedure? It's clearly laid out, and comprehensive, and probably not an exact fit for most people. I nodded my head all the way through, but I haven't actually tried it, because I need to figure out what parts I'm going to work into my process, and how I'm going to adapt it to my style. I'm not going to use it completely.

Second, the piece of it that I understood best is also the piece I liked least, which is never a good sign. As an illustration of why his diagrams are good for just about everything, the author talks about diagramming out how a software package works and says, "To improve security, we'd need to enhance protection around those areas where the most information enters and leaves the system." Given that the hypothetical product is said to be relatively secure to start with, this is staggeringly unlikely. Those are the most obviously security-related parts of the system, and somebody has probably already waved some sort of security wand at them. The weakest link is almost certainly somewhere else. Now, I don't expect the author to understand security engineering, but it's a good illustration of how drawing out the system only helps if you're bringing deep understanding to the surface or sorting out something you already know something about. It won't substitute for expertise in bringing you insights into unfamiliar territory.

ADOBE PHOTOSHOP CS4 ONE-ON-ONE

Deke McClelland

O'Reilly, 2008. 499 pages (includes DVD).
ISBN 978-0-596-52189-9

THE PHOTOSHOP CS4 COMPANION FOR PHOTOGRAPHERS

Derrick Story

O'Reilly, 2008. 177 pages.
ISBN 978-0-596-52193-6

Photoshop may seem like an odd topic for people who're used to mostly open solutions, but it's oddly familiar in many ways. Rob Pike is reputed to have said "UNIX is user friendly. It's just selective about who its friends are." Photoshop is much the same. Not only that, but it takes a Perl-like "let many

flowers bloom" approach. I'm no Photoshop expert, but I know three different ways to fill an entire document with a pattern, and I'm not at all certain what's different about two of them. In the first video for *CS4 One-on-One*, Deke McClelland demonstrates at least five different ways to zoom in and out. Couple this with the documentation, which is, as the Australians say, "very ordinary" (i.e., terrible in all the usual ways), and you have a powerful but intimidating program. Early adventures in Photoshop tend to involve a lot of murmuring "Why did that happen?" and "Surely there's a way to do it. I wonder if it's on a panel somewhere?"

This of course means that there are 4 million Photoshop books available. I'm looking at CS4 books because CS4 just came out, and this therefore cuts down the crowd considerably. It's also interesting because CS4 introduces some serious changes to how Photoshop is best used.

The CS4 Companion is a nice, succinct introduction to Photoshop for photographers, and it takes full advantage of the new features in CS4. It's a great starting point if you're interested in using Photoshop for seriously managing photos. I have a few quibbles—photographic workflows are pretty idiosyncratic, and I disagree about some of them (e.g., why on earth would you reformat your memory card all the time?)—but mostly I find it a direct and useful reference that pushes a CS4-based process. This is important, and it's going to be a problem for most CS4 books based on CS3 books, because the best way of doing things has changed pretty dramatically. All the things you might be used to are still possible, but they're generally no longer ideal.

The main issue here, if you're familiar with Photoshop, has to do with adjustments, for things such as Levels or Curves. It used to be that the real hotshots used these with masks, which involved interesting tricks, and everybody else used the History brush. Now there are adjustment layers, which make it easy to do nondestructive changes you can always back out of without damaging your image, and just using Levels or Curves is wantonly destroying data for no good reason.

These changes are an issue for *One-on-One*, which does cover all the cool new features but introduces more traditional ways of working first. This makes some sense, because the book is clearly intended to teach people who want to be graphic artists and need to be able to integrate with other Photoshop users. But if you're learning for personal use, you're going to learn techniques and then discard them later.

As a tutorial for people who're committed to learning Photoshop, I liked *One-on-One* a lot. I think it uses video well and I like the way it works through examples to show you when techniques are useful. (The videos teach things that are hard to convey in text, but they're not essential to the experience.) It is really a class in a book, though. It's not for you if you want to pick up a technique at a time. For instance, I knew that CS4 had added a new automatic layering trick I wanted to try, and I looked at both books to figure out how to do it. Both books do cover it, but *The CS4 Companion* gave me a brief recipe on the most straightforward way of using it, while *One-on-One* gave me a sidebar about it, which did explain how to do it and showed a really interesting use of it, but was much harder for me to interpret in the moment.

MAVEN: THE DEFINITIVE GUIDE

Sonatype Company

September 2008. 468 pages.
ISBN 978-0-596-51733-5

REVIEWED BY JASON DUSEK

The book introduces Maven, its operation, its core configuration, a wealth of plug-ins, and the elements of customization. Maven, a build and release tool, models projects with the Project Object Model (POM). The copious examples are enough to remind one of how XML's verbosity can be made up for by its readability—for the POM is XML all the way down.

More than offering a view into Maven, this book offers a glimpse into the degree to which build and release management processes have been automated, codified, and standardized (which certainly came as a pleasant surprise to this reviewer, who still does all this stuff with Make). Maven offers a rich collection of tools for not only build management but dependency visualization, plug-in extension, and, to no small degree, Web publishing. The number of markup languages and template engines supported by Maven is no small credit to the project, and it is all to the good that I need only browse through a few pages of this book to find out about them.

One avowed goal of the book is to make whole the uneven documentation of Maven and its plug-ins. The authors are nothing if not thorough in this regard. Nearly every other page offers a figure, example code, or sample configuration. Where a model relationship is mentioned, a diagram is offered. The authors do not even assume knowledge of octal permissions—a link is offered to clarify.

The richness of Maven and more generally of Java technology perhaps makes a fault of the authors' diligence—one cannot escape a discussion of Inversion of Control containers, the multitude of JVM scripting languages with which one may extend Maven, or integration with Eclipse and the attendant “materialization.” A very small section of the book is devoted to walk-throughs of build specification; the vast majority is given to concepts, features, and customization, in short vignettes unrelated to one another. This is more of a “how to do it to Maven” book than it is a “how to do it with Maven” book.

In consequence, the reviewer is left with a disappointment: the one thing he really wanted to know, “how to build something that is not Java with Maven,” is handled by a plug-in not covered in this book.

DESIGN CONCEPTS IN PROGRAMMING LANGUAGES

Franklyn A. Turbak and David K. Gifford

MIT Press. 2008. 1200 pages.
ISBN 978-0262201759

REVIEWED BY JASON DUSEK

In comparison to texts such as *Structure and Interpretation of Computer Programs* or *Concepts, Techniques, and Models of Computer Programming*, this text is much more mathematically oriented, guided by an interest in language design and specification as opposed to program design and specification. As such, it is an encyclopedic reference of languages and language features as opposed to a deep exploration of one particular language and the means of programming with it.

For those of us who have nurtured an abiding interest in programming language theory with research papers and blog posts, this book offers a thorough and disciplined approach to terminology as well as a way to round out the inevitably lopsided knowledge of the auto-didact. The expository text and numerous citations place each development in context, historically as well as functionally.

Each design concept is accompanied by a mini-language, all of which are presented in a uniform, Scheme-like syntax. This uniformity dovetails with the small scope of the languages and makes translation from the grammar (provided in full for each language) to the concrete language a simple exercise. The text covers a variety of paradigms, including stack-based programming, purely functional programming with static and dynamic types, type inference, effect systems, and object-oriented pro-

programming languages. A listing of languages and their feature sets is provided at the end of the text.

Programming content is actually pretty light—most (all?) of the numerous exercises can be worked with paper and pencil. Some time is given to garbage collection and compilation, but efficient implementation is not a focus of this text. The book's Web site provides links to programming problems from an associated MIT course.

This book is more edifying than enabling for most of us: Rarely is the working programmer called upon to implement a programming language. Just the same, mini-languages find application surprisingly often; the material presented is food for thought the next time you are deciding just how to arrange a batch job format or a simple template system. In recent years, some “esoteric” languages—Factor, Haskell, and Erlang come to mind—have become surprisingly usable; this text provides an excellent basis for appreciating their guiding design ideas.

THE CRAFT OF SYSTEM SECURITY

Sean Smith and John Marchesini

Addison-Wesley Professional. 2007. 592 pages.
ISBN 978-0321434838

REVIEWED BY SAM STOVER

I'm not all that sure where this book came from, but I wish it had been around about 10 years ago. What started out as a working text for a college course has ended up as a solid intro to a large number of topics in the information security field. I really can't recommend it enough for the technical guy or gal who wants to learn more about security.

The book is broken down into five parts: History, Security and the Modern Computing Landscape, Building Blocks for Secure Systems, Applications, and Emerging Tools.

The three chapters in the History section discuss, well, history. I breezed over this, to be honest, but I think the authors do a reasonably good job of introducing important, albeit historical, information in an academic yet readable manner. That was one thing that really struck me throughout the entire book: the adherence to scientific principles when discussing technology. I've found that sometimes that style of writing doesn't play well with the “craft” of security, but these authors do it well.

Security and the Modern Computing Landscape consists of three chapters, which tend to be the meat of most other security books. OS Security touches on process isolation, filesystem permissions, and common attack vectors such as keylog-

gers and rootkits. Network Security follows the same style but deals with things such as the TCP/IP stack, different protocols on the stack, and common attacks for those protocols. Implementation follows suit in the application arena, dealing with buffer overflows, fuzzing, malware, and other tools and attacks relevant to applications.

Building Blocks for Secure Systems starts out with Using Cryptography, followed by Subverting Cryptography. I think these two chapters alone are worth the price of admission—the pros and cons, as well as common pitfalls of the different cryptographic implementations, should be required reading for anyone deploying crypto as a security measure. It's all too easy to assume that encryption equals security, and these two chapters remove a lot of that mystique. The remaining chapters in this section deal with Authentication, Public Key Infrastructure and Standards, Compliance, and Testing. Again, this is more great, highly applicable reading for anyone doing any kind of secure system building.

The Applications section focuses on Web security, common office tools, digital cash, and securing different types of information. These topics aren't typically my cup of tea, but, as with all topics in the book, they are covered in depth, yet remain very readable. Emerging Tools goes into the weeds a bit, but it's still great reading, covering things such as hardware-based security, virtualization, armored CPUs, RFID, and even a little AI for good measure. Some of it is directly relevant; some of it is a bit more theoretical—but don't think of theoretical as a negative thing. Theories are good. The book wraps up with some “take-home” thoughts that, again, head out into the weeds, but if you're a math geek, it's all you. (They actually discuss some quantum mechanics as a means to break RSA.)

Overall, the book was well written, informative, pertinent to contemporary security issues, and just a solid all-around reference. I have no problem recommending it to anyone who wants to bone up on any of the topics discussed. If there's one thing I would wish, it would be that more security books were written with the attention to detail and topic coverage that this one has. Happy reading.

STATISTICS IN A NUTSHELL

Sarah Boslaugh and Paul Andrew Watters

O'Reilly, 2008. 452 pages.
ISBN 978-0-596-51049-7

REVIEWED BY RIK FARROW

I got this book because I had recently been faced with just how much I had forgotten about statis-

tics. I had taken more than one statistics class in college, but I found myself at a loss when attempting to interpret the statistics I encountered in CS papers.

Although the subtitle of this book is *A Desktop Quick Reference*, this nutshell book is really a textbook. It is written not just for learning about statistics but also for researchers who need to design their model data so that it can be properly analyzed. The target audience is really not those of us in CS, but practitioners in other sciences. Having said that, I have found that I will be using this book as my desktop reference, having read enough of it that I am immediately interested in.

I also got *The Manga Guide to Statistics*, which Elizabeth reviewed, but can't fairly compare them. The difference between the two is so large as to be incomparable. Whereas *The Manga Guide* does present equations, for example, *Nutshell* explains where the equations come from. I found that I was uncomfortable with complex equations that just appear in comic panels. When the *Nutshell* authors introduce equations, they do so in a context that makes the equations appear understandable. I found the same equations much less impenetrable—in fact, less complex—with this presentation.

The *Nutshell* book authors carefully explain the usage of different statistical measures, as well as potential pitfalls. Most of this in *The Manga Guide* gets covered in one chapter, but the *Nutshell* book's authors cover a complete course in statistics geared to research in crystal-clear text and in depth. There are also chapters on research design and critiquing statistics presented by others. Chapters 17–19 cover business, medical, epidemiological, educational, and psychological statistics—a bit outside the scope of my own interests.

If you want a quick romp in statistics, perhaps the graphics versions we reviewed will provide you

with the background you need to read papers. Perhaps. If you want to write your own papers and include statistical analysis of your data to prove your assertions, *Statistics in a Nutshell* is the book for you.

GETTING STARTED WITH ARDUINO

Massimo Banzi

O'Reilly, 2008. 117 pages.
ISBN 978-0-596-15551-3

REVIEWED BY RIK FARROW

You may have heard of Arduino, an open-source hardware project that uses a simple microprocessor, such as the Atmega168, and a handful of other parts that can be easily programmed. I had watched someone playing with an Arduino designed to be included in clothing (think Burning Man), so I wanted to check this out. O'Reilly sent me this book, and I ordered what turned out to be a slightly older model Arduino (the Diecimila) but it worked fine with this book.

The book is really aimed at nonprogrammers who want to play with physical computing. The Arduino allows you to do rapid prototyping, where the microprocessor coupled with the development environment does most of the work. Although you can accomplish some simple projects with nothing but your computer, a USB cable, the Arduino, and some simple electronics, getting a solderless breadboard for assembling resistors, photodetectors, other sensors, and LEDs is much handier.

If you are already a programmer, the online documentation may be all you need, but if you have been hesitating to leap into hardware, this book and Arduino hardware provide a painless way of doing so. Having the book is what pushed me over the edge, enabling me to discover just how easy it was to get started.