Book Reviews

ELIZABETH ZWICKY, WITH RIK FARROW

Illustrated Guide to Home Forensic Science Experiments: All Lab, No Lecture

Robert Bruce Thompson and Barbara Fritchman Thompson

O'Reilly Media, July 2012. 428 pages.

ISBN 978-1-449-33451-2

This is a careful, practical guide to doing real forensic science at home. Kind of like those detective kits they try to sell kids, except with practical advice on really lifting fingerprints, including superglue fuming techniques. (In case you were wondering, the reason the detective kit didn't work is that dusting for fingerprints successfully is hard.) It's not at all like television; there's actual work involved.

On the one hand, it is just as cool as it sounds. On the other hand, did I mention it's not like television? I'd be prepared for the cool factor to wear off pretty quickly for the younger set. Doing science—forensic or not—requires a certain amount of meticulous attention to detail, accurate performance of repetitive processes, and a willingness to embrace "Yes, those are the same shade of orange" as a fabulous result.

But if you're willing to pay careful attention to detail, you too can do gel electrophoresis of DNA at home using primarily things available at your local supermarket (well, my local supermarket, which sells chopsticks and agar as well as Tupperware and D batteries). I have to admit I didn't do the experiments, but I did find myself with a sudden desire for a microscope so that I could.

The text is interesting and neither undersells nor oversells the experiments. It honestly lays out the authors' experiences with the practical issues. My only gripe is that some of the images are poor, and in many of them the authors' hands are shown ungloved when the text calls for gloves.

OS X Mountain Lion: The Missing Manual

David Pogue

O'Reilly Media, July 2012. 834 pages.

ISBN 978-1449-33027-9

OS X Mountain Lion Pocket Guide

Chris Seibold

O'Reilly Media, July 2012. 236 pages.

ISBN 978-1-449-33032-3

I have just upgraded to Mountain Lion, making a dizzying leap from running Snow Leopard on the home machine and Leopard on the work machine to Mountain Lion and Lion, so it seemed like a good moment to check out some Mountain Lion books. Despite the hoopla about Lion, as a very long-time Mac and iPad user, I didn't actually find anything particularly earth-shattering about it, or about Mountain Lion, either.

Then I picked up *Mountain Lion: The Missing Manual* and discovered that this was because I just didn't know about the exciting new bits. I'm still not convinced that they will change my life, but I am intrigued at the possibility that gestures will finally make full screen mode and spaces work for me. And despite my functional knowledge of Macintoshes, I learned useful things (Spotlight will do calculations for you, which gives you a keystroke that brings up a calculator that doesn't obscure the numbers you wanted to add—sadly, it is then impossible to cut-and-paste the answer). *The Missing Manual* also is smoothly readable and mildly funny (which, trust me, is all the funny you are likely to put up with for 834 pages). I'm not sure that I would be motivated to read it if I weren't reviewing it, but that's not its fault; I just don't have any driving desire to know that much all at once about my operating system. And it's obsessed with keyboard shortcuts, which I have a very limited amount of use for. If you want to know all the details instead of guessing, and/or you hate using a mouse, it's an excellent choice.

The *Mountain Lion Pocket Guide* is more my speed; the keyboard shortcuts are in sidebars and their own chapter instead of spread throughout, and it's brief. On the other hand, it's also slower to get to the cool stuff, or at least the stuff I think is cool. Its 10 cool new Mountain Lion features, although they are new to Mountain Lion, fail to pass my coolness bar reliably, and oddly it disagrees with *The Missing Manual* about one new feature; *The Missing Manual* says Mountain Lion now does full-screen separately on multiple screens, *Pocket Guide* says it does not. Sadly, *Pocket Guide* is right. Either one will let me have one app full screen on either monitor, while the other monitor displays a picture of linen apparently designed to convey blankness while letting you know the power is still on.

If you aren't particularly feeling the lack of a manual, you probably won't love either book. If you are feeling the lack of a manual in a nagging way, you probably want the *Pocket Guide.* If you'd like to cuddle up with a manual and have the interfaces to everything lovingly but gently explained to you, go for *The Missing Manual*. (This is also a good choice if you would like to feel clever and superior at work, where people are actually quite impressed by what I picked up.)

Think Like a Programmer: An Introduction to Creative Problem Solving

V. Anton Sproul

No Starch Press, 2012. 227 pages.

ISBN 978-1-59327-424-5

This is a valiant effort at teaching somebody who is intelligent and motivated how you do the essential parts of programming. Not what "if . . . then . . . else" means, or how you write syntactically correct code in some language, but how to program. As a reviewer, I'm handicapped by the fact that I've always known how to do that. I have, however, spent time teaching people to solve problems, with some success.

I think on the whole this is a worthy approach; it tries to demystify the process and break it down into pieces, which is very useful for learners. In fact, merely existing is a good first step, because many people—including learners—have a firm belief that these things can't be taught, which doesn't help.

I have two quarrels with the book. First, I think it moves over a lot of territory too fast. Yes, there are exercises, and people are strongly encouraged to do them. But exercises by themselves are not enough for most people learning to solve problems; they need smaller bites at a time. Additionally, some of these exercises don't appear to me to be possible as stated. For instance, using only two output statements, one that outputs the hash mark and one that outputs an end-of-line, writes a program that produces the following shape:

```
########
 ######
  ####
   ##
```

Where did those initial spaces at the beginning of the line come from? We don't get to output spaces. Is there some C++ trick here I don't know about? If so, why are we depending on it in the first chapter where we actually write programs? This just seems mean, one way or another.

Second, it's not clear to me why the author thinks C++ is a good choice. Sure, that allows you to spend a lot of time thinking about linked lists and low-level programming abstractions, but it doesn't allow you to spend a lot of time thinking about appropriate data structures for your problem and high-level programming abstractions other than classes. Linked lists are not actually a good answer to most programs these days. Use a database. Don't need a database? Use a hash. I am not convinced that knowing how to do linked lists and think about pointers will teach a junior programmer how to avoid linked lists when they are not needed, which is just about always.

Essential Scrum: A Practical Guide to the Most Popular Agile Process

Kenneth S. Rubin

Pearson Education, July 2012. 426 pages.

ISBN 978-0-13-704329-3

This lives up to its title; it lays out, clearly, the basics of Scrum, from reasoning through mechanics. It's a fairly minimal version of Scrum; it doesn't insist on pair programming or user stories, for instance, and it doesn't try to deal with edge cases. There's a paragraph or two on Scrum in interrupt-driven groups (summary: don't), but nothing about other common issues (Scrum surrounded by other theories, groups divided by distance). If you are coming into a solidly Scrum-based organization, this would be a good way to get a feel for the bones of Scrum, separate from organizational issues and peculiarities. If you're struggling with a transition to Scrum, *The Scrum Field Guide* (previously reviewed) is a better choice. It's more dogmatic about some particular Scrum features, but much stronger on the nitty-gritty of transitioning to Scrum and running Scrum in edge cases. If you have patience for two books, they make a good pair and illuminate some of the internal variation in what Scrum practitioners find important.

—Elizabeth Zwicky

LED Lighting: A Primer for Lighting the Future

Sal Cangeloso

O'Reilly Maker Press, 2012. 58 pages.

ISBN 978-1-449-33476-5

As the time has come to phase out the terribly simple and horribly inefficient 100-watt incandescent bulbs, I wanted to learn more about LED-based lighting. I've already done most of the home improvements possible for a home: extra insulation in the attic, reflective barriers, extra insulation added to outside walls, 3.2 kW of solar panels, double-pane windows, and no incandescent bulbs in my house. Since I've already embraced fluorescents, what do I need to know about LEDs?

Cangeloso informs us of issues involving four forms of lighting: incandescents, halogen, fluorescents, and LEDs. I had already bought a couple of LED bulbs as experiments, and they were pretty disappointing: dim and off-color. And what's with the enormous heatsinks found in floodlight-type LEDs?

Lighting is heavily regulated, both to prevent fires and to provide standards for consumers. Whereas we once bought incandescent bulbs based on wattage, this is a poor method when it comes to more efficient bulbs measured in lumens. I still find myself comparing wattage to lumens. And it turns out that the old incandescent bulbs have another endearing feature: they produce a fuller spectrum of warm lighting. This isn't much of a surprise when you have a filament that glows white hot inside a protective enclosure: you get everything from ultraviolet to lots of infrared light. Cangeloso explains color temperatures that helped me understand the "cool" and "warm" terms.

LEDs, and incandescents, have another problem, and that is color accuracy when used for lighting. The warm, full spectrum is closer to what our eyes have evolved to use, but LED bulbs, which actually do use phosphors to produce most of their light, have difficulty producing anything close to a full spectrum. The color rendering index (CRI) for an incandescent bulb is 100, 80 for Philips' most popular A19 (the familiar screw-in base) bulb, and 92 for its award-winning (and very expensive) bulb. Color temperature is going to be more important for most people, but not for artists and other people who expect or require their colors to appear true.

What this book lacks are tables. I found myself paging back through the book for the watts-to-lumens comparisons, and just wished there was at least one table in this short book.

And those heatsinks? It turns out the LEDs produce less light (and don't last as long) when they get hot, and while incandescents discard lots of heat as infrared light, LEDs need other forms of cooling. That, and because LED bulbs also have electronics in their bases that provide both regulated power and reduced power if overheating is detected, makes LED bulbs truly high-tech compared to the glowing filaments that are now being hoarded by some. I can suggest reading this book if you want to learn more, see cutaway photos of LED bulbs, and want to prepare for the future.

—Rik Farrow