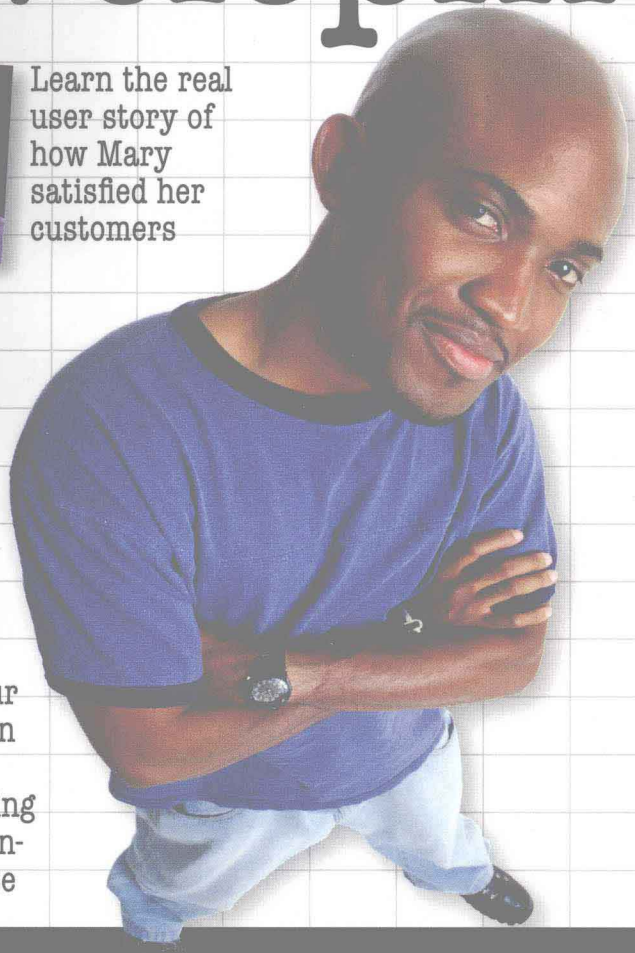
深入浅出软件开发（影印版）

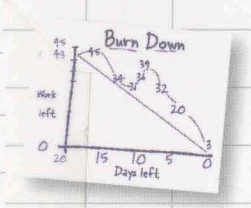
Head First Software Development

Learn the real user story of how Mary satisfied her customers

Score big by using velocity to figure out how fast your team can produce

Use test-driven development to avoid unsightly software disasters





Keep your project on schedule by tracking your burn-down rate

Master the techniques and tools of seasoned software developers

O'REILLY® 東南大學出版社

Dan Pilone & Russ Miles 著

深入浅出软件开发(影印版)

Head First Software Development

Dan Pilone
Russ Miles

O'REILLY®

Beijing • Cambridge • Köln • Paris • Sebastopol • Taipei • Tokyo

O'Reilly Media, Inc.授权东南大学出版社出版

南京　东南大学出版社

图书在版编目（CIP）数据

深入浅出软件开发：英文/（美）皮隆（Pilone, D.），（美）迈尔斯（Miles, R.）著. —影印本. —南京：东南大学出版社，2009.1

书名原文：Head First Software Development

ISBN 978-7-5641-1236-3

Ⅰ. 深… Ⅱ. ①皮… ②迈… Ⅲ. 软件开发－英文 Ⅳ. TP311.52

中国版本图书馆 CIP 数据核字（2008）第 104535 号

江苏省版权局著作权合同登记

图字：10-2008-124 号

深入浅出软件开发（影印版）

出版发行：东南大学出版社
地　　址：南京四牌楼 2 号　　邮编：210096
出 版 人：江　汉
网　　址：http://press.seu.edu.cn
电子邮件：press@seu.edu.cn
印　　刷：扬中市印刷有限公司
开　　本：787 毫米 × 980 毫米　12 开本
印　　张：41.5 印张
字　　数：523 千字
版　　次：2009 年 1 月第 1 版
印　　次：2009 年 1 月第 1 次印刷
书　　号：ISBN 978-7-5641-1236-3/TP · 205
印　　数：1~3000 册
定　　价：88.00 元（册）

本社图书若有印装质量问题，请直接与读者服务部联系。电话（传真）：025-83792328

O'Reilly Media, Inc.介绍

O'Reilly Media, Inc.是世界上在 UNIX、X、Internet 和其他开放系统图书领域具有领导地位的出版公司，同时是联机出版的先锋。

从最畅销的《The Whole Internet User's Guide & Catalog》（被纽约公共图书馆评为二十世纪最重要的50本书之一）到GNN（最早的Internet门户和商业网站），再到WebSite（第一个桌面PC的Web服务器软件），O'Reilly Media, Inc.一直处于Internet发展的最前沿。

许多书店的反馈表明，O'Reilly Media, Inc.是最稳定的计算机图书出版商——每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly Media, Inc.具有深厚的计算机专业背景，这使得O'Reilly Media, Inc.形成了一个非常不同于其他出版商的出版方针。O'Reilly Media, Inc.所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly Media, Inc.还有许多固定的作者群体——他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly Media, Inc.依靠他们及时地推出图书。因为O'Reilly Media, Inc.紧密地与计算机业界联系着，所以O'Reilly Media, Inc.知道市场上真正需要什么图书。

出版说明

随着计算机技术的成熟和广泛应用，人类正在步入一个技术迅猛发展的新时期。计算机技术的发展给人们的工业生产、商业活动和日常生活都带来了巨大的影响。然而，计算机领域的技术更新速度之快也是众所周知的，为了帮助国内技术人员在第一时间了解国外最新的技术，东南大学出版社和美国 O'Reilly Meida, Inc.达成协议，将陆续引进该公司的代表前沿技术或者在某专项领域享有盛名的著作，以影印版或者简体中文版的形式呈献给读者。其中，影印版书籍力求与国外图书“同步”出版，并且“原汁原味”展现给读者。

我们真诚地希望，所引进的书籍能对国内相关行业的技术人员、科研机构的研究人员和高校师生的学习和工作有所帮助，对国内计算机技术的发展有所促进。也衷心期望读者提出宝贵的意见和建议。

最新出版的影印版图书，包括：

- 《JavaScript: The Good Parts》（影印版）
- 《学习 ActionScript 3.0》（影印版）
- 《深入浅出 C#》（影印版）
- 《深入浅出软件开发》（影印版）
- 《Ruby 程序设计语言》（影印版）
- 《高性能网站》（影印版）
- 《学习 ASP.NET 2.0 与 Ajax》（影印版）
- 《ASP.NET 3.5 构建 Web 2.0 门户网站》（影印版）
- 《Mac OS X: The Missing Manual, Leopard Editon》（影印版）
- 《深入浅出 JavaScript》（影印版）
- 《敏捷开发艺术》（影印版）
- 《Intermediate Perl》（影印版）
- 《学习 Python 第三版》（影印版）
- 《ASP.NET AJAX 编程》（影印版）

Advance Praise for *Head First Software Development*

"*Head First Software Development* is a whimsical but very thoughtfully designed series of information diagrams and clever illustrations meant to accurately and clearly convey information directly into YOUR brain. It's a whole new kind of book."

— Scott Hanselman
Software Developer, Speaker, Author
Scott Hanselman's Computer Zen

"This is one of those books experienced developers wish they'd had back when they got started. I know, I'm one of them."

— Burk Hufnagel, Senior Software Architect

"I could have avoided a whole world of pain if I had read this book before my last project!"

— This developer asked to remain anonymous, so her last project's manager wouldn't be upset!

"*Head First Software Development* teaches many valuable lessons that will help anyone deliver quality software on time and on budget. Following the core principles taught in this book will help keep your project on track from start to finish. No matter how long you've been developing software, *Head First Software Development* will give you essential tools for developing successful projects from start to finish."

— Adam Z. Szymanski, Software Project Manager, Naval Research Laboratory

"The ideas in this book can be used by new and experienced managers to immediately improve their overall software development process."

— Dan Francis, Software Engineering Manager, Fortune 50 company

"A fresh new perspective on the software development process. A great introduction to managing a development team from requirements through delivery."

— McClellan Francis, Software Engineer

Praise for *Head First Object-Oriented Analysis and Design*

"*Head First Object-Oriented Analysis and Design* is a refreshing look at the subject of OOA&D. What sets this book apart is its focus on learning. There are too many books on the market that spend a lot of time telling you why, but do not actually enable the practitioner to start work on a project. Those books are very interesting, but not very practical. I strongly believe that the future of software development practice will focus on the practitioner. The authors have made the content of OOA&D accessible and usable for the practitioner "

— Ivar Jacobson, Ivar Jacobson Consulting

"I just finished reading *HF OOA&D*, and I loved it! The book manages to get across the essentials of object-oriented analysis and design with UML and use cases, and even several lectures on good software design, all in a fast-paced, easy to understand way. The thing I liked most about this book was its focus on why we do OOA&D—to write great software! By defining what great software is and showing how each step in the OOA&D process leads you towards that goal, it can teach even the most jaded Java programmer why OOA&D matters. This is a great 'first book' on design for anyone who is new to Java, or even for those who have been Java programmers for a while but have been scared off by the massive tomes on OO Analysis and Design."

— Kyle Brown, Distinguished Engineer, IBM

"Finally a book on OOA&D that recognizes that the UML is just a notation and that what matters when developing software is taking the time to think the issues through."

— Pete McBreen, Author, *Software Craftsmanship*

"The book does a good job of capturing that entertaining, visually oriented, 'Head First' writing style. But hidden behind the funny pictures and crazy fonts is a serious, intelligent, extremely well-crafted presentation of OO Analysis and Design. This book has a strong opinion of how to design programs, and communicates it effectively. I love the way it uses running examples to lead the reader through the various stages of the design process. As I read the book, I felt like I was looking over the shoulder of an expert designer who was explaining to me what issues were important at each step, and why."

— Edward Sciore, Associate Professor, Computer Science Department
Boston College

"This is a well-designed book that delivers what it promises to its readers: how to analyze, design, and write serious object-oriented software. Its contents flow effortlessly from using use cases for capturing requirements to analysis, design, implementation, testing, and iteration. Every step in the development of object-oriented software is presented in light of sound software engineering principles. The examples are clear and illustrative. This is a solid and refreshing book on object-oriented software development."

— Dung Zung Nguyen, Lecturer
Rice University

Praise for *Head First Design Patterns*

"I received the book yesterday and started to read it on the way home... and I couldn't stop. I took it to the gym and I expect people saw me smiling a lot while I was exercising and reading. This is tres 'cool'. It is fun but they cover a lot of ground and they are right to the point. I'm really impressed."

—Erich Gamma, IBM Distinguished Engineer, and co-author of Design Patterns

"'Head First Design Patterns' manages to mix fun, belly-laughs, insight, technical depth and great practical advice in one entertaining and thought provoking read. Whether you are new to design patterns, or have been using them for years, you are sure to get something from visiting Objectville."

—Richard Helm, coauthor of "Design Patterns" with rest of the Gang of Four—Erich Gamma, Ralph Johnson, and John Vlissides

"I feel like a thousand pounds of books have just been lifted off of my head."

—Ward Cunningham, inventor of the Wiki and founder of the Hillside Group

"This book is close to perfect, because of the way it combines expertise and readability. It speaks with authority and it reads beautifully. It's one of the very few software books I've ever read that strikes me as indispensable. (I'd put maybe 10 books in this category, at the outside.)"

—David Gelernter, Professor of Computer Science, Yale University and author of "Mirror Worlds" and "Machine Beauty"

"A Nose Dive into the realm of patterns, a land where complex things become simple, but where simple things can also become complex. I can think of no better tour guides than the Freemans."

—Miko Matsumura, Industry Analyst, The Middleware Company Former Chief Java Evangelist, Sun Microsystems

"I laughed, I cried, it moved me."

—Daniel Steinberg, Editor-in-Chief, java.net

"My first reaction was to roll on the floor laughing. After I picked myself up, I realized that not only is the book technically accurate, it is the easiest to understand introduction to design patterns that I have seen."

—Dr. Timothy A. Budd, Associate Professor of Computer Science at Oregon State University and author of more than a dozen books, including *C++ for Java Programmers*

"Jerry Rice runs patterns better than any receiver in the NFL, but the Freemans have out run him. Seriously...this is one of the funniest and smartest books on software design I've ever read."

—Aaron LaBerge, VP Technology, ESPN.com

Other related books from O'Reilly

Making Things Happen

Applied Software Project Management

Beautiful Code

Prefactoring

The Art of Agile Development

UML 2.0 In a Nutshell

Learning UML 2.0

Other books in O'Reilly's *Head First* series

Head First Java

Head First Object-Oriented Analysis and Design (OOA&D)

Head Rush Ajax

Head First HTML with CSS and XHTML

Head First Design Patterns

Head First Servlets and JSP

Head First EJB

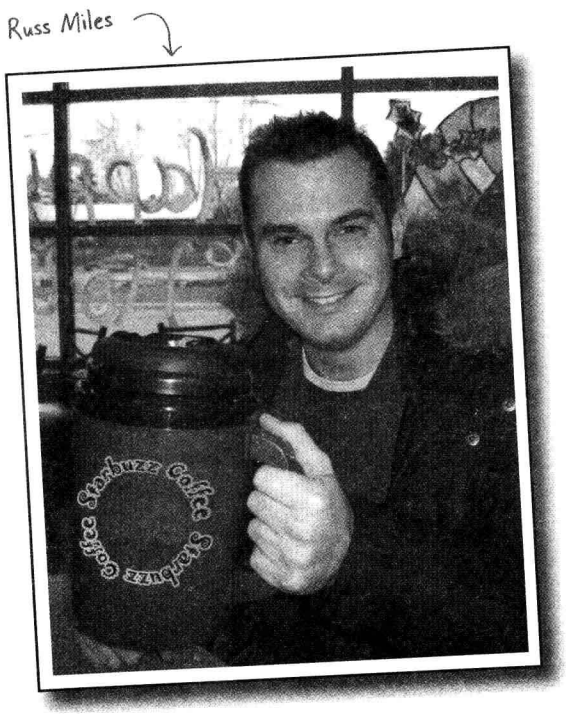
Head First PMP

Head First SQL

Head First JavaScript

To everyone who's worked on a project with us and told us where we've gone wrong, where we've gone right, and what books to read…here's our contribution back.

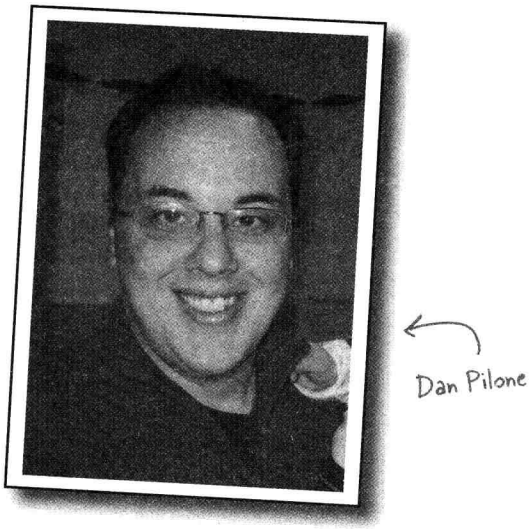
Author(s) of Head First Software Development



Russ is totally indebted to his fiancée, Corinne, for her complete love and support while writing this book. Oh, and he still can't believe she said yes to getting married next year, but I guess some guys have all the luck!

Russ has been writing for a long time and gets a huge kick out of demystifying technologies, tools, and techniques that shouldn't have been so mystified in the first place. After being a developer at various ranks for many years, Russ now keeps his days (and sometimes nights) busy by heading up a team of software developers working on super secret services for the music industry. He's also just finished up his Oxford Masters degree that only took him five years. He's looking forward to a bit of rest...but not for too long.

Russ is an avid guitar player and is relishing the spare time to get back to his guitars. The only thing he's missing is **Head First Guitar**...c'mon Brett, you know you want that one!

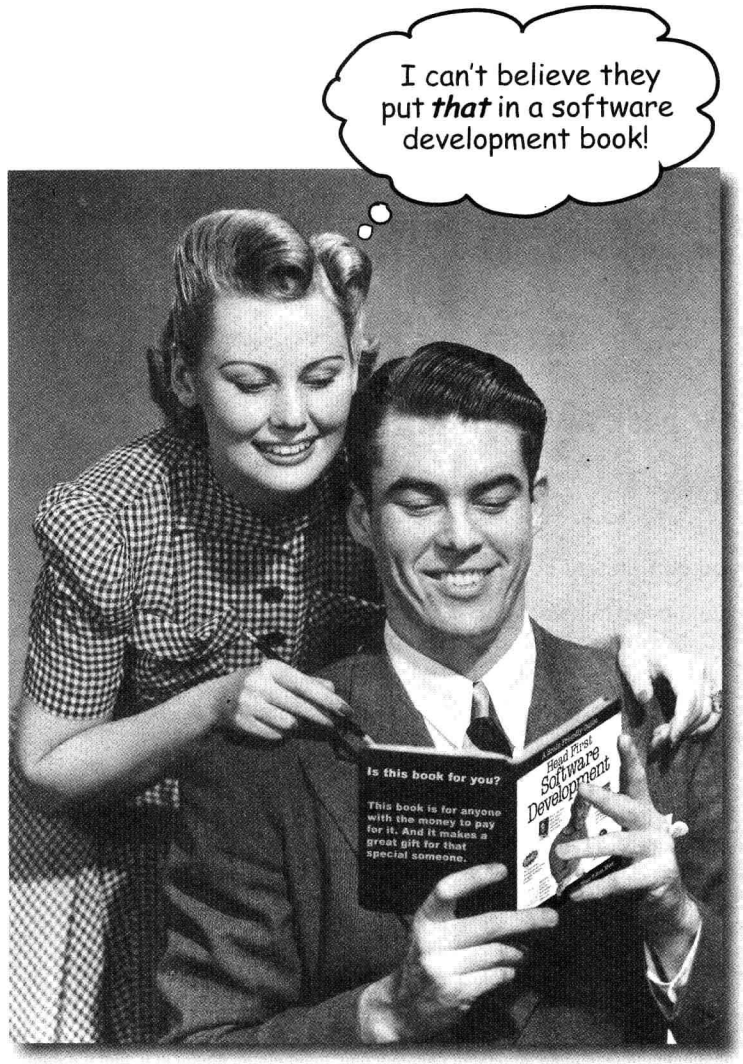


Dan is eternally grateful to his wife Tracey for letting him finish this book. Dan is a software architect for Vangent, Inc., and has led teams for the Naval Research Laboratory and NASA, building enterprise software. He's taught graduate and undergraduate Software Engineering at Catholic University in Washington, D.C. Some of his classes were interesting.

Dan started writing for O'Reilly by submitting a proposal for this book a little over five years ago. Three UML books, some quality time in Boulder, Colorado, with the O'Reilly Head First team, and a co-author later, he finally got a chance to put this book together.

While leading a team of software developers can be challenging, Dan is waiting patiently for someone to write **Head First Parenting** to help sort out seriously complex management problems.

Intro



In this section we answer the burning question:
"So why DID they put that in a software development book?"

Who is this book for?

If you can answer "yes" to all of these:

1. Do you have access to a computer and **some background in programming**?
2. Do you want to **learn techniques for building and delivering great software?** Do you want to **understand** the principles behind iterations and test-driven development?
3. Do you prefer **stimulating dinner party conversation** to **dry**, **dull**, **academic lectures**?

We use Java in the book, but you can squint and pretend it's C#. No amount of squinting will make you think it's Perl, though.

this book is for you.

Who should probably back away from this book?

If you can answer "yes" to any of these:

1. **Are you completely new to Java?**

   (You don't need to be advanced, and if you know C++ or C# you'll understand the code examples just fine.)

2. Are you a kick-butt development manager looking for a ***reference* book**?

3. Are you **afraid to try something different**? Would you rather have a root canal than mix stripes with plaid? Do you believe that a technical book can't be serious if iterations are anthropomorphized?

this book is not for you.

[Note from marketing: this book is for anyone with a credit card.]

We know what you're thinking

"How can *this* be a serious software development book?"

"What's with all the graphics?"

"Can I actually *learn* it this way?"

We know what your *brain* is thinking

Your brain craves novelty. It's always searching, scanning, *waiting* for something unusual. It was built that way, and it helps you stay alive.

So what does your brain do with all the routine, ordinary, normal things you encounter? Everything it *can* to stop them from interfering with the brain's *real* job—recording things that *matter*. It doesn't bother saving the boring things; they never make it past the "this is obviously not important" filter.

How does your brain *know* what's important? Suppose you're out for a day hike and a tiger jumps in front of you, what happens inside your head and body?

Neurons fire. Emotions crank up. *Chemicals surge.*

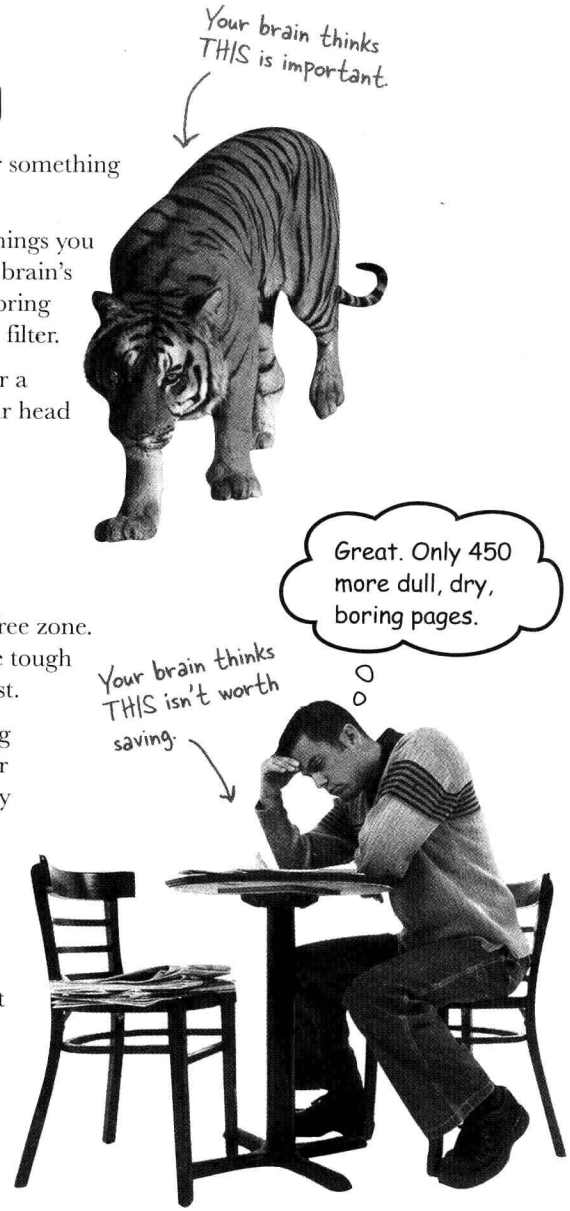
And that's how your brain knows...

This must be important! Don't forget it!

But imagine you're at home, or in a library. It's a safe, warm, tiger-free zone. You're studying. Getting ready for an exam. Or trying to learn some tough technical topic your boss thinks will take a week, ten days at the most.

Just one problem. Your brain's trying to do you a big favor. It's trying to make sure that this *obviously* non-important content doesn't clutter up scarce resources. Resources that are better spent storing the really *big* things. Like tigers. Like the danger of fire. Like the guy with the handle "BigDaddy" on MySpace probably isn't someone to meet with after 6 PM.

And there's no simple way to tell your brain, "Hey brain, thank you very much, but no matter how dull this book is, and how little I'm registering on the emotional Richter scale right now, I really *do* want you to keep this stuff around."



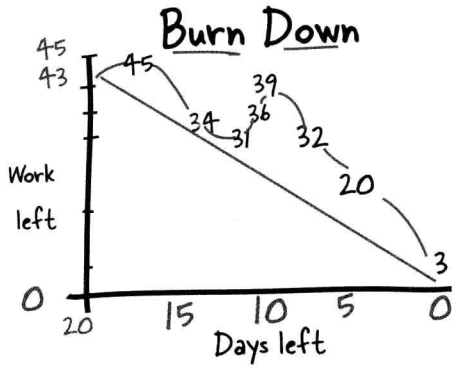
We think of a "Head First" reader as a learner.

So what does it take to *learn* something? First, you have to *get* it, then make sure you don't *forget* it. It's not about pushing facts into your head. Based on the latest research in cognitive science, neurobiology, and educational psychology, *learning* takes a lot more than text on a page. We know what turns your brain on.

Some of the Head First learning principles:

Make it visual. Images are far more memorable than words alone, and make learning much more effective (up to 89% improvement in recall and transfer studies). It also makes things more understandable. **Put the words within or near the graphics** they relate to, rather than on the bottom or on another page, and learners will be up to *twice* as likely to solve problems related to the content.

Use a conversational and personalized style. In recent studies, students performed up to 40% better on post-learning tests if the content spoke directly to the reader, using a first-person, conversational style rather than taking a formal tone. Tell stories instead of lecturing. Use casual language. Don't take yourself too seriously. Which would *you* pay more attention to: a stimulating dinner party companion, or a lecture?



Get the learner to think more deeply. In other words, unless you actively flex your neurons, nothing much happens in your head. A reader has to be motivated, engaged, curious, and inspired to solve problems, draw conclusions, and generate new knowledge. And for that, you need challenges, exercises, and thought-provoking questions, and activities that involve both sides of the brain and multiple senses.

Get—and keep—the reader's attention. We've all had the "I really want to learn this but I can't stay awake past page one" experience. Your brain pays attention to things that are out of the ordinary, interesting, strange, eye-catching, unexpected. Learning a new, tough, technical topic doesn't have to be boring. Your brain will learn much more quickly if it's not.

Touch their emotions. We now know that your ability to remember something is largely dependent on its emotional content. You remember what you care about. You remember when you *feel* something. No, we're not talking heart-wrenching stories about a boy and his dog. We're talking emotions like surprise, curiosity, fun, "what the...?", and the feeling of "I Rule!" that comes when you solve a puzzle, learn something everybody else thinks is hard, or realize you know something that "I'm more technical than thou" Bob from engineering *doesn't*.

Metacognition: thinking about thinking

If you really want to learn, and you want to learn more quickly and more deeply, pay attention to how you pay attention. Think about how you think. Learn how you learn.

Most of us did not take courses on metacognition or learning theory when we were growing up. We were *expected* to learn, but rarely *taught* to learn.

But we assume that if you're holding this book, you really want to learn how to really develop great software. And you probably don't want to spend a lot of time. If you want to use what you read in this book, you need to *remember* what you read. And for that, you've got to *understand* it. To get the most from this book, or *any* book or learning experience, take responsibility for your brain. Your brain on *this* content.

The trick is to get your brain to see the new material you're learning as Really Important. Crucial to your well-being. As important as a tiger. Otherwise, you're in for a constant battle, with your brain doing its best to keep the new content from sticking.

So just how *DO* you get your brain to treat software development like it was a hungry tiger?

There's the slow, tedious way, or the faster, more effective way. The slow way is about sheer repetition. You obviously know that you *are* able to learn and remember even the dullest of topics if you keep pounding the same thing into your brain. With enough repetition, your brain says, "This doesn't *feel* important to him, but he keeps looking at the same thing *over* and *over* and *over*, so I suppose it must be."

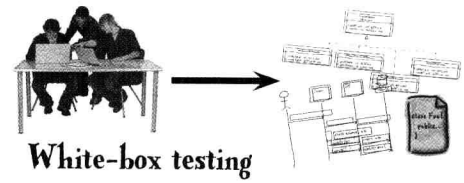
The faster way is to do ***anything that increases brain activity,*** especially different *types* of brain activity. The things on the previous page are a big part of the solution, and they're all things that have been proven to help your brain work in your favor. For example, studies show that putting words *within* the pictures they describe (as opposed to somewhere else in the page, like a caption or in the body text) causes your brain to try to makes sense of how the words and picture relate, and this causes more neurons to fire. More neurons firing = more chances for your brain to *get* that this is something worth paying attention to, and possibly recording.

A conversational style helps because people tend to pay more attention when they perceive that they're in a conversation, since they're expected to follow along and hold up their end. The amazing thing is, your brain doesn't necessarily *care* that the "conversation" is between you and a book! On the other hand, if the writing style is formal and dry, your brain perceives it the same way you experience being lectured to while sitting in a roomful of passive attendees. No need to stay awake.

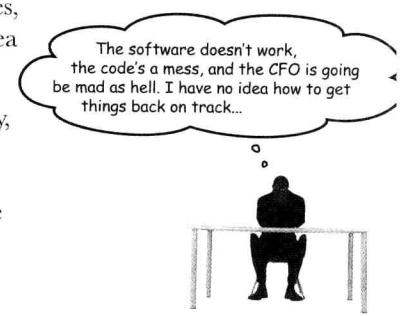
But pictures and conversational style are just the beginning...

Here's what WE did:

We used ***pictures***, because your brain is tuned for visuals, not text. As far as your brain's concerned, a picture really *is* worth a thousand words. And when text and pictures work together, we embedded the text *in* the pictures because your brain works more effectively when the text is *within* the thing the text refers to, as opposed to in a caption or buried in the text somewhere.

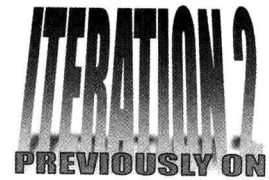


We used ***redundancy***, saying the same thing in *different* ways and with different media types, and *multiple senses*, to increase the chance that the content gets coded into more than one area of your brain.



We used concepts and pictures in ***unexpected*** ways because your brain is tuned for novelty, and we used pictures and ideas with at least *some* ***emotional*** *content*, because your brain is tuned to pay attention to the biochemistry of emotions. That which causes you to *feel* something is more likely to be remembered, even if that feeling is nothing more than a little ***humor***, ***surprise***, or ***interest.***

We used a personalized, ***conversational style***, because your brain is tuned to pay more attention when it believes you're in a conversation than if it thinks you're passively listening to a presentation. Your brain does this even when you're *reading*.



We included more than 80 ***activities***, because your brain is tuned to learn and remember more when you ***do*** things than when you *read* about things. And we made the exercises challenging-yet-do-able, because that's what most people prefer.

We used ***multiple learning styles***, because *you* might prefer step-by-step procedures, while someone else wants to understand the big picture first, and someone else just wants to see an example. But regardless of your own learning preference, *everyone* benefits from seeing the same content represented in multiple ways.

We include content for ***both sides of your brain***, because the more of your brain you engage, the more likely you are to learn and remember, and the longer you can stay focused. Since working one side of the brain often means giving the other side a chance to rest, you can be more productive at learning for a longer period of time.

And we included ***stories*** and exercises that present ***more than one point of view,*** because your brain is tuned to learn more deeply when it's forced to make evaluations and judgments.

We included ***challenges***, with exercises, and by asking ***questions*** that don't always have a straight answer, because your brain is tuned to learn and remember when it has to *work* at something. Think about it—you can't get your *body* in shape just by *watching* people at the gym. But we did our best to make sure that when you're working hard, it's on the *right* things. That ***you're not spending one extra dendrite*** processing a hard-to-understand example, or parsing difficult, jargon-laden, or overly terse text.

We used ***people***. In stories, examples, pictures, etc., because, well, because *you're* a person. And your brain pays more attention to *people* than it does to *things*.