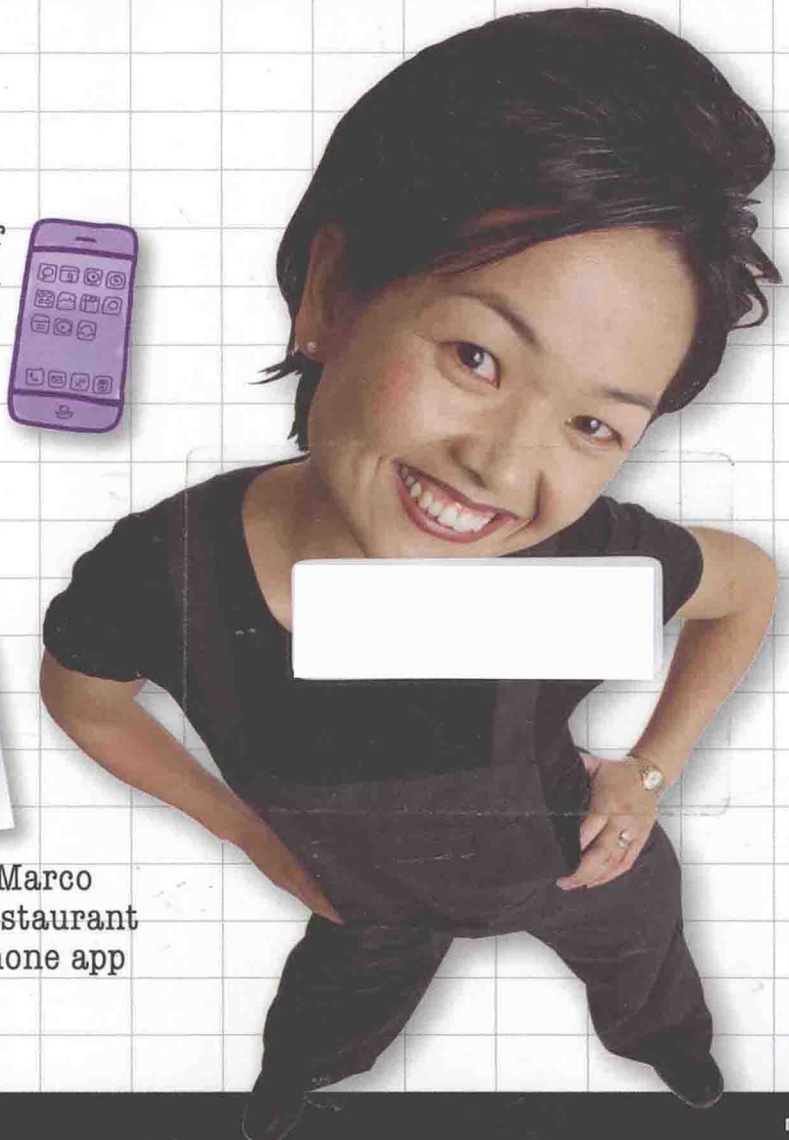
O'REILLY®

第3版
涵盖iOS 7
和Xcode 5

# Head First iPhone & iPad Development

## 深入浅出iPhone和iPad开发（影印版）



Master iOS design principles

Create top-selling apps

Tap into the iPhone's GPS and camera

See how Marco saved his restaurant with an iPhone app

Easily manage big data in your apps

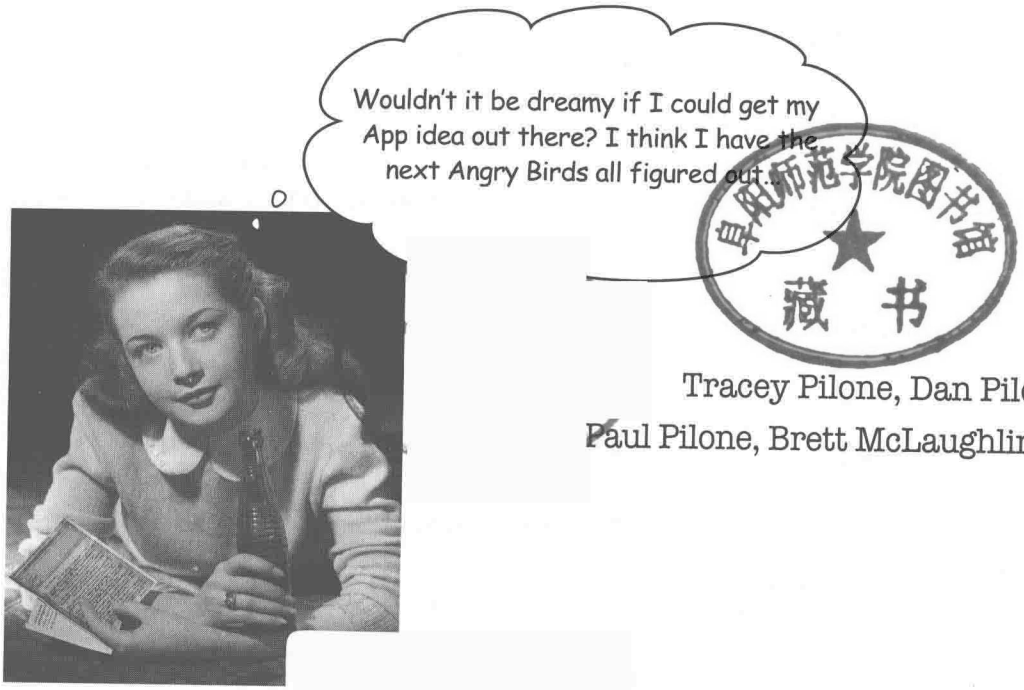
Tracey Pilone, Dan Pilone,
Paul Pilone, Brett McLaughlin 著

東南大學 出版社

# 深入浅出iPhone和iPad开发（影印版）

# Head First iPhone & iPad Development

第3版






Tracey Pilone, Dan Pilone,
Paul Pilone, Brett McLaughlin 著

O'REILLY®

*Beijing • Cambridge • Farnham • Köln • Sebastopol • 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 iPhone 和 iPad 开发:第 3 版:英文/（美）皮隆（Pilone, T.）等著 . —影印本 . —南京：东南大学出版社，2014.10

书名原文：Head First iPhone and iPad development, 3E

ISBN 978-7-5641-5003-7

I. ① 深… II. ①皮… III. ①移动电话机－应用程序－程序设计－英文 ②便携式计算机－应用程序－程序设计－英文－ IV. ① TP929.53 ② TP368.32

中国版本图书馆 CIP 数据核字（2014）第 115563 号

江苏省版权局著作权合同登记

图字：10-2012-166 号



深入浅出 iPhone 和 iPad 开发 第 3 版（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：江建中
网　　址：http://www.seupress.com
电子邮件：press@seupress.com
印　　刷：常州市武进第三印刷有限公司
开　　本：787 毫米 × 980 毫米　12 开本
印　　张：30.5
字　　数：510 千字
版　　次：2014 年 10 月第 1 版
印　　次：2014 年 10 月第 1 次印刷
书　　号：ISBN 978-7-5641-5003-7
定　　价：98.00 元

本社图书若有印装质量问题，请直接与营销部联系。电话（传真）：025-83791830

# Authors of Head First iPhone and iPad Development


**Tracey Pilone** is co-founder (along with Dan Pilone) and operations director at Element 84, a high value outsourcing and consulting company specializing in highly scalable web and mobile software development . In addition to handling the business development work for the company, she works with Element 84's agile development teams to manage and deliver projects to customers.

Prior to starting Element 84, she spent several years working in and around the Washington, D.C., area for two of Engineering News Record's top 20 contractors as a construction manager in commercial construction. This is her fourth *Head First* book, including the two earlier editions of this book and *Head First Algebra*.

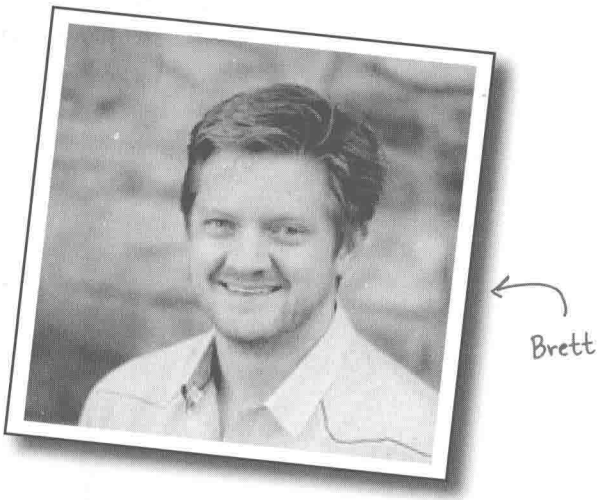
She has a civil engineering degree from Virginia Tech and a Master's of Education from the University of Virginia, and holds a professional engineer's license in Virginia. You can follow Tracey on Twitter: @traceypilone.


**Dan Pilone** is the founder and Managing Partner of Element 84. He has designed and implemented systems for NASA, Hughes, ARINC, UPS, and the Naval Research Laboratory. He currently serves as technical lead for projects with NASA as well as all of Element 84's projects. He speaks frequently in the community most recently at ESIP, AGU, and the DC Ruby Users Group.

He has taught project management, software design, and software engineering at The Catholic University in Washington, D.C. Dan has been an instructor for the D.C. iPhone Bootcamp and has written several books on software development, including *Head First Software Development*, *UML 2.0 in a Nutshell*, and *UML 2.0 Pocket Reference*. You can follow Dan on Twitter: @danpilone.

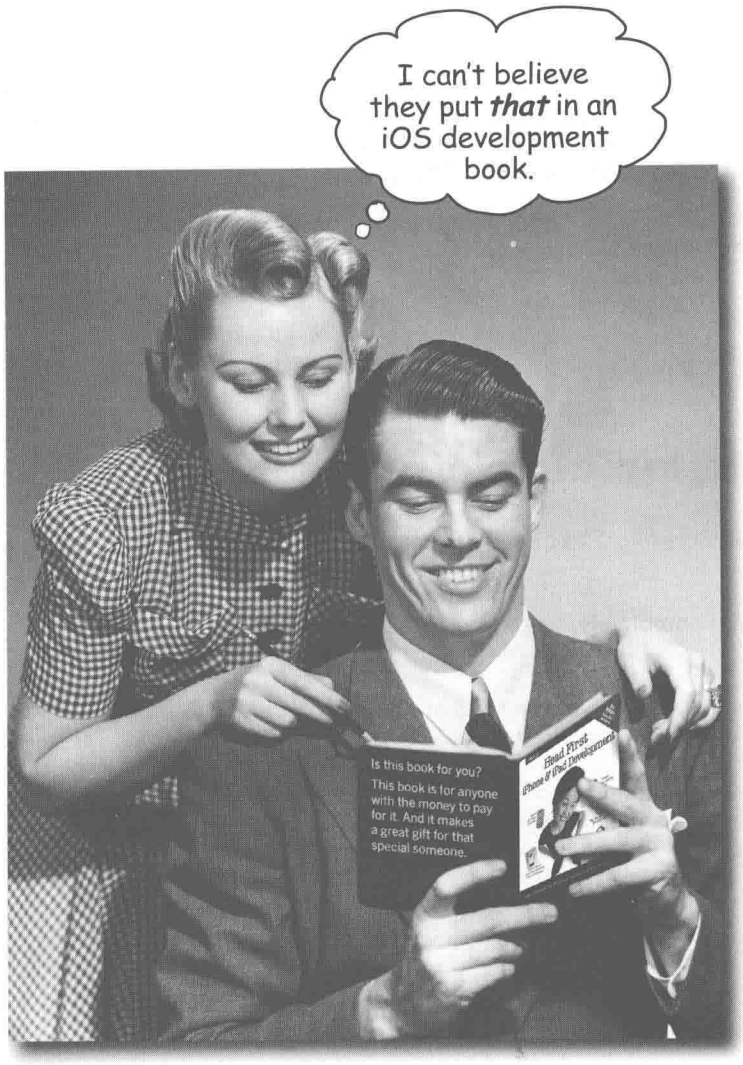
# Coauthors of Head First iPhone and iPad Development


**Paul Pilone** is an iOS and Rails developer with Element 84. He's the author of iHomework, an iPhone, iPad, and Mac app for managing homework assignments. Paul has developed software for the Naval Research Labs, Lockheed Martin, NASA, and Cengage Learning. You can follow Paul on Twitter: @paulpilone.




**Brett McLaughlin** is a software developer at Element84. He's also a developer who's got his hands into cognitive theory. That means that he sees HTML5, CSS, JavaScript, Java, and Rails as the means to tell interesting stories to users rather than just a load of boring technology and protocols. He's also really interested in the next generation of communication technologies, ranging from ePub to ebooks to mobile devices. And in his free time (what free time?), he's usually playing with video and writing projects and playing guitar... really expensive acoustic ones, if he can manage.

# Intro




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

# Who is this book for?

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

1. Do you have **previous development experience**?
2. Do you want to **learn, understand, remember,** and **apply** important iOS design and development concepts so that you can write your own iOS apps?
3. Do you prefer **stimulating dinner party conversation** to dry, dull, academic lectures?

It definitely helps if you've already got some object-oriented chops, too. Experience with Mac development is helpful, but certainly not required.

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 software development?
2. Are you already developing iOS apps and looking for a **reference book** on Objective-C?
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 there's a TARDIS mentioned in it?

Check out Head First Programming for an excellent introduction to object-oriented development, and then come back and join us in iPhoneville.

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 iOS 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.*

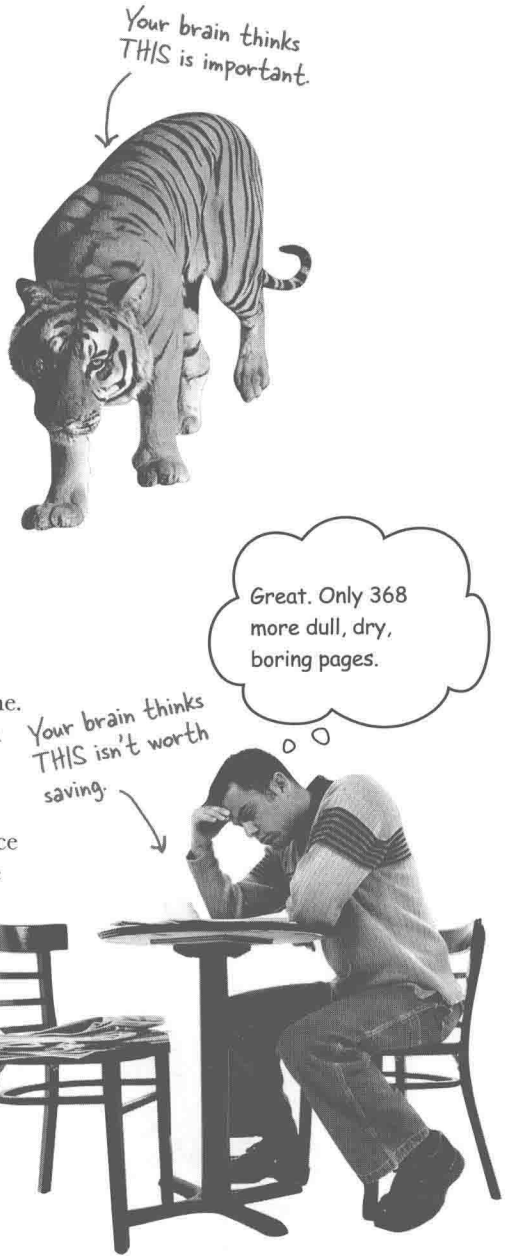
And that's how your brain knows...

**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 how you should never have posted those party photos on your Facebook page.

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.

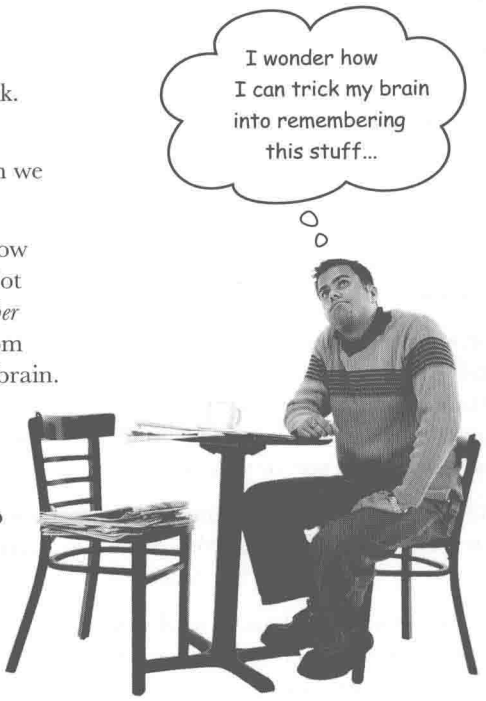
**Use a conversational and personalized style.** In recent studies, students performed up to 40% better on post-learning tests if the content spoke directly to the reader, using a first-person, conversational style rather than taking a formal tone. Tell stories instead of lecturing. Use casual language. 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 design user-friendly websites. 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 iOS development like it was a hungry tiger?

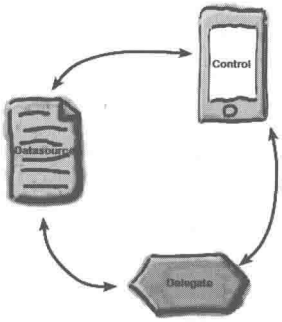
There's the slow, tedious way, or the faster, more effective way. The slow way is about sheer repetition. You obviously know that you *are* able to learn and remember even the dullest of topics if you keep pounding the same thing into your brain. With enough repetition, your brain says, "This doesn't *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:



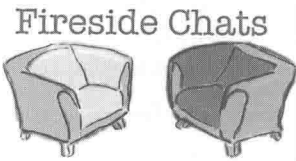

We used ***pictures***, because your brain is tuned for visuals, not text. As far as your brain's concerned, a picture really *is* worth a thousand words. And when text and pictures work together, we embedded the text *in* the pictures because your brain works more effectively when the text is *within*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

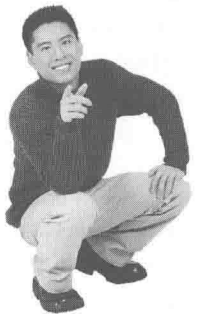
We included more than 80 ***activities***, because your brain is tuned to learn and remember more when you ***do*** things than when you *read* about things. And we made the exercises challenging-yet-do-able, because that's what most people prefer.




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.

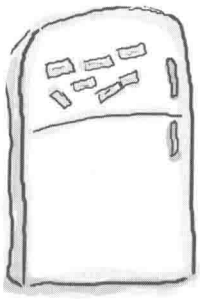
We include content for ***both sides of your brain***, because the more of your brain you engage, the more likely you are to learn and remember, and the longer you can stay focused. Since working one side of the brain often means giving the other side a chance to rest, you can be more productive at learning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



Cut this out and stick it on your refrigerator.

# Here's what YOU can do to bend your brain into submission

So, we did our part. The rest is up to you. These tips are a starting point; listen to your brain and figure out what works for you and what doesn't. Try new things.

1. **Slow down. The more you understand, the less you have to memorize.**

   Don't just *read*. Stop and think. When the book asks you a question, don't just skip to the answer. Imagine that someone really *is* asking the question. The more deeply you force your brain to think, the better chance you have of learning and remembering.

2. **Do the exercises. Write your own notes.**

   We put them in, but if we did them for you, that would be like having someone else do your workouts for you. And don't just *look* at the exercises. **Use a pencil.** There's plenty of evidence that physical activity *while* learning can increase the learning.

3. **Read the "There are No Dumb Questions" sections.**

   That means all of them. They're not optional sidebars, ***they're part of the core content!*** Don't skip them.

4. **Make this the last thing you read before bed. Or at least the last challenging thing.**

   Part of the learning (especially the transfer to long-term memory) happens *after* you put the book down. Your brain needs time on its own, to do more processing. If you put in something new during that processing time, some of what you just learned will be lost.

5. **Talk about it. Out loud.**

   Speaking activates a different part of the brain. If you're trying to understand something, or increase your chance of remembering it later, say it out loud. Better still, try to explain it out loud to someone else. You'll learn more quickly, and you might uncover ideas you hadn't known were there when you were reading about it.

6. **Drink water. Lots of it.**

   Your brain works best in a nice bath of fluid. Dehydration (which can happen before you ever feel thirsty) decreases cognitive function.

7. **Listen to your brain.**

   Pay attention to whether your brain is getting overloaded. If you find yourself starting to skim the surface or forget what you just read, it's time for a break. Once you go past a certain point, you won't learn faster by trying to shove more in, and you might even hurt the process.

8. **Feel something.**

   Your brain needs to know that this *matters*. Get involved with the stories. Make up your own captions for the photos. Groaning over a bad joke is *still* better than feeling nothing at all.

9. **Write a lot of code!**

   There's only one way to learn to program: **writing a lot of code**. And that's what you're going to do throughout this book. Coding is a skill, and the only way to get good at it is to practice. We're going to give you a lot of practice: every chapter has exercises that pose a problem for you to solve. Don't just skip over them—a lot of the learning happens when you solve the exercises. We included a solution to each exercise—don't be afraid to **peek at the solution** if you get stuck! (It's easy to get snagged on something small.) But try to solve the problem before you look at the solution. And definitely get it working before you move on to the next part of the book.

# Read Me

This is a learning experience, not a reference book. We deliberately stripped out everything that might get in the way of learning whatever it is we're working on at that point in the book. And the first time through, you need to begin at the beginning, because the book makes assumptions about what you've already seen and learned.

### We begin by modifying a completed iOS app by pulling code right from GitHub.

While this book is focused on iOS development, part of what we're hoping to teach is how to use the tools that work not only with iOS development, but with software development in general. So to kick it off, we're going to drop you into completed code that needs some changes, not starting from scratch.

### We don't get into app submission.

We used to. But there are two things going on that make that tough. First, once you get into the Apple Developer Program, there are significant chunks that are under NDA, and second, iOS development has gotten more advanced over time. This book is geared toward getting basic knowledge under your belt. You'll need more to get an app ready for submission.

### We focus on what you can build and test on the simulator.

The iPhone software development kit (SDK) comes with a great (and free!) tool for testing your apps on your computer. The simulator lets you try out your code without having to worry about getting it on the App Store or on a real device. But it also has its limits. There's some cool iOS stuff you just can't test on the simulator, like the accelerometer and compass. So we don't cover those kinds of things in very much detail in this book since we want to make sure you're creating and testing apps quickly and easily.

### The activities are NOT optional.

The exercises and activities are not add-ons; they're part of the core content of the book. Some of them are to help with memory, some are for understanding, and some will help you apply what you've learned. ***Don't skip the exercises.*** The crossword puzzles are the only thing you don't *have* to do, but they're good for giving your brain a chance to think about the words and terms you've been learning in a different context.

### The redundancy is intentional and important.

One distinct difference in a Head First book is that we want you to *really* get it. And we want you to finish the book remembering what you've learned. Most reference books don't have retention and recall as a goal, but this book is about *learning*, so you'll see some of the same concepts come up more than once.

### The examples are as lean as possible.

Our readers tell us that it's frustrating to wade through 200 lines of an example looking for the two lines they need to understand. Most examples in this book are shown within the smallest possible context, so that the part you're trying to learn is clear and simple. Don't expect all of the examples to be robust, or even complete—they are written specifically for learning, and aren't always fully functional.

We've placed the code on GitHub so you can copy see the full application and all of the code when you need it. The code is available here:

***https://github.com/dpilone/Head-First-iPhone-iPad-Development-3rd-Edition***

### The Brain Power exercises don't have answers.

For some of them, there is no right answer, and for others, part of the learning experience of the Brain Power activities is for you to decide if and when your answers are right. In some of the Brain Power exercises, you will find hints to point you in the right direction.

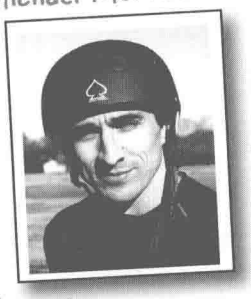
## System requirements

To develop for the iPhone and iPad, you need an Intel-based Mac, period. We wrote this book using OS X version 10.8.5 and Xcode 5.0. If you are running an older version of Xcode, most differences you will see are look and feel based. For some of the more advanced capabilities, like the accelerometer and the camera, you'll need an actual device and to be a registered developer.

In Chapter 1, we point you in the direction to get the SDK and Apple documentation, so don't worry about that for now.

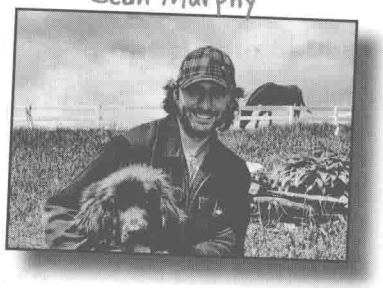
# The technical review team

Michael Morrison



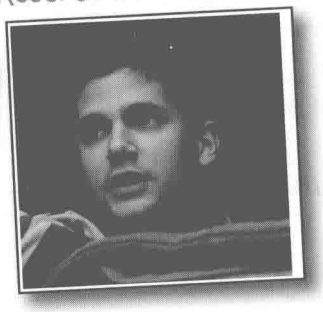
Joe Heck

Sean Murphy

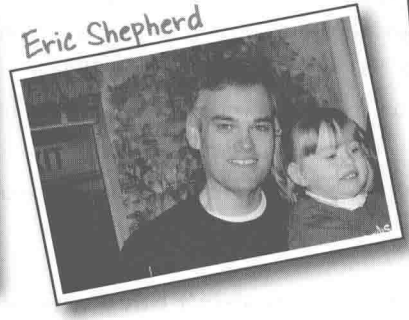


Rene Janssen Morrison

Roberto Luis



Eric Shepherd



Rich Rosen

**Rich Rosen** is one of the co-authors of *Mac OS X for Unix Geeks*. He also collaborated with Leon Shklar on *Web Application Architecture: Principles, Protocols & Practices*, a textbook on advanced web application development.

**Sean Murphy** has been a Cocoa aficionado for almost 10 years, contributes to open source projects such as Camino, and works as an independent iOS designer and developer.

**Joe Heck** is a software developer, technology manager, author, and instructor who's been involved with computing for 25 years and developing for the iPhone platform since the first beta release. He's the founder of the Seattle Xcoders developer group, and the author of SeattleBus, an iPhone app that provides real-time arrival and departure times of Seattle public transportation.

**Eric Shepherd** got started programming at age nine and never looked back. He's been a technical writer, writing developer documentation since 1997, and is currently the developer documentation lead at Mozilla.

**Michael Morrison** is a writer, developer, and author of *Head First JavaScript*, *Head First PHP & MySQL*, and even a few books that don't have squiggly arrows, stick figures, and magnets. Michael is the founder of Stalefish Labs (*www.stalefishlabs.com*), an edutainment company specializing in games, toys, and interactive media, including a few iPhone apps.

**Roberto Luis** is a young programmer who loves learning new languages and tools. He's a Computer Science Engineer from Autonoma de Madrid University in Spain, during his career has been involved in desktop, mobile and cloud development.

**René Janssen** is a multimedia designer from The Netherlands and owner of Ducklord Studio. He started off as a typical graphic designer way, working with Indesign, Photoshop and Illustrator but wanted to learn more, from languages like HTML, CSS, PHP, MySQL to Actionscript. He's worked for years in Flash and Actionscript and evolved to iOS Development, Xcode and Objective-C.

# Acknowledgments

***Our editor:***

Thanks to **Courtney Nash**, who has worked on all three editions of this book (since 2009!), from the beginning to the end. We've had to work around Apple's release cycles and this round was tough to get out the door—thanks to Courtney, we finally did!

Courtney Nash

***The O'Reilly team:***

To the talented crew over at O'Reilly who prettied up our files after we were done with them and is always there for reachback when we need help with the process and our learners, too.

***Our intern:***

Thanks to **Jayanth Prathipati**, Element 84's first intern who stepped in to pinch hit and help us finish up with screen shots that had to be updated for iOS7.

***Our friends and family:***

To all of the **Pilones** and **Chadwicks**, who have helped us with the kids and were understanding of us busting out laptops at various inappropriate times to get this one done.

To all our friends at **Element 84** who ended up helping out here and there when we needed opinions.

To **Paul Pilone**, who helped us write the code for the book and ran through a mess of iOS7 updates pretty quick.

To **Brett McLaughlin**, who worked through storyboarding with us and gave us someone else to help bear the InDesign load.

To **Vinny** and **Nick**, who have been practicing baseball and taekwondo while parts of this book were written, and who we hope are going to be able to help us write the next one!

# Safari Books Online


Safari Books Online is an on-demand digital library that delivers expert in both book and video form from the world's leading authors in technology and business.

Technology professionals, software developers, web designers, and business and creative professionals use Safari Books Online as their primary resource for research, problem solving, learning, and certification training.

Safari Books Online offers a range of and pricing programs for organizations, government, and individuals. Subscribers have access to thousands of books, training videos, and prepublication manuscripts in one fully searchable database from publishers like O'Reilly Media, Prentice Hall Professional, Addison-Wesley Professional, Microsoft Press, Sams, Que, Peachpit Press, Focal Press, Cisco Press, John Wiley & Sons, Syngress, Morgan Kaufmann, IBM Redbooks, Packt, Adobe Press, FT Press, Apress, Manning, New Riders, McGraw-Hill, Jones & Bartlett, Course Technology, and dozens more. For more information about Safari Books Online, please visit us online.