PERFECTING INTERFACE DESIGN IN MOBILE APPS

ESSENTIAL MOBILE INTERACTION DESIGN

Cameron BANGA
Josh WEINHOLD

Praise for *Essential Mobile Interaction Design*

"In *Essential Mobile Interaction Design,* Banga and Weinhold do a great job of explaining what it takes to make a good-looking and easy-to-use app. The accessible language and visual examples of quality work combine to make this book a great resource for those looking to get into app design, or to take their craft to the next level."

—**Jon Becker**
boom. reactive.

"*Essential Mobile Interaction Design* is not merely a book full of pictures and design concepts or one of straight technical drivel. Instead, it is a guidebook for creating human-based interfaces that feature simplicity, functionality, and value. Whether you have questions about how mobile design is different from traditional desktop design, how to work with a developer, or even what tools to use for the creation process, *Essential Mobile Interaction Design* demonstrates the answer for that."

—**Phil Dutson**
Lead UX and Mobile Developer, ICON Health & Fitness

"Filled with nuggets of useful information, this book is a solid resource for the many aspects of designing a mobile app. I've found many recommendations in this book that we can use in our apps."

—**Lucius Kwok**
CEO, Felt Tip, Inc.

"A well-rounded, easy-to-read book that provides a good grounding in mobile design and how to keep all those small details in mind so that your apps will really shine."

—**Dave Verwer**
Shiny Development and iOS Dev Weekly

This page intentionally left blank

Essential Mobile Interaction Design

This page intentionally left blank

Essential Mobile Interaction Design

Perfecting Interface Design in Mobile Apps

Cameron Banga
Josh Weinhold

Addison-Wesley

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Cataloging-in-Publication Data is on file with the Library of Congress

ISBN-13: 978-0-321-96157-0
ISBN-10: 0-321-96157-9
Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana.
First printing, March 2014

Editor-in-Chief
Mark L. Taub

Executive Editor
Laura Lewin

Development Editor
Michael Thurston

Managing Editor
John Fuller

Project Editor
Elizabeth Ryan

Packager
Anna Popick

Copy Editor
Melinda Rankin

Indexer
Jack Lewis

Proofreader
Anna Popick

Technical Reviewers
Jonathon Becker
Victor Lara
Francisco Velazquez

Editorial Assistant
Olivia Basegio

Cover Designer
Chuti Prasertsith

Compositor
Shepherd, Inc.

I dedicate this book to Gavin. Although we've only just met, I couldn't be more excited to be your uncle. Hopefully, you'll be interested in interface design one day so that you can read through this and remind me how silly and archaic our phones and computers once were.
—Cameron

I dedicate this book to Mallory. Thank you for always encouraging me to take on new challenges and pushing me to always aim higher.
—Josh

This page intentionally left blank

Contents

This page intentionally left blank

PREFACE

This text offers an introduction to and general overview of interaction design for all mobile computing platforms, with a particular emphasis on Google's Android and Apple's iOS platforms.

Mobile apps should feel natural and intuitive. Users should quickly and easily understand them. This means that effective interaction and interface design is crucial to the success of any mobile app. Few mobile app developers (or even designers), however, have had adequate training in these areas.

Touchscreen-focused, mass-market mobile applications are a type of technology that's only been possible to create since 2008, and the industry has seen monumental shifts and growth in the six years between the introduction of the "app economy" and the publication of this work.

This book aims to help put you in a place to succeed as a designer in today's app market by teaching proven principles and techniques that you can use in your next app, no matter what mobile platform, targeted device, form factor, or user base you're targeting.

In short, the tutorial style used here aims to help you master the mindset, processes, and vocabulary of mobile interaction design so that you can start making better choices right away. This book guides you through the entire process of app design, demystifying many of the tasks and issues that arise during the many stages of developing, releasing, and improving a mobile app.

Cameron Banga has been working in mobile application development since 2009, and in the nearly five years since releasing his first app he's contributed as a designer or as an adviser to more than 100 applications for iPhone, iPad, Android, and OS X. In that time, he's seen firsthand the growth and change experienced in the mobile industry and has worked to meet client and customer expectations throughout the many evolutions of mobile platforms.

This book aims to serve as a central resource for programmers or designers looking to best determine how to establish themselves in today's modern mobile landscape by offering advice formulated and acquired through Cameron's experiences over the past several years as a pioneer of the mobile app economy.

Topics chosen for this book were selected carefully by Cameron with advice from and in coordination with several successful and award-winning industry colleagues. The goal of each chapter was to focus on a particular strong primary skill required of any successful designer, breaking that skill down into a few key components that any novice could focus on and, with some strong advice and clear guidance, work to master quickly.

No programming knowledge and only basic design knowledge is required to understand this book, as it's been carefully crafted to be universally readable and helpful. In situations in which it does dive into extremely specific terminology or a topic for which prior information would be required, breakout boxes offer context and suggestions as to where the reader can look for further information that's beyond the scope of this book.

Only basic design tools were used to create the example work included in this book, and the software or hardware that was used is detailed where relevant. Much of this book focuses on general theories and somewhat universal design practices that can be slightly modified and fine-tuned to the reader's specific circumstances. Additional resources that are required or may be helpful have been posted online at http://cameronbanga.com/EMIDbook.

We hope you enjoy this book and that it helps you make progress toward your goal of becoming an outstanding mobile app designer. If you would like to share your thoughts or if you have a question, feel free to contact the authors at book@cameronbanga.com.

ACKNOWLEDGMENTS

This book would not exist without the dedicated work of Laura Lewin (our gracious and ever-helpful executive editor) and Olivia Basegio (our wonderful editorial assistant, always willing to lend a hand and keep us on track). We'd like to give a huge thank you to the entire Addison-Wesley editorial team, especially Michael Thurston and Melinda Rankin.

Special thanks go out as well to our technical reviewers: Jonathon Becker, Victor Lara, and Francisco Velazquez. The book is significantly better due to their recommendations, nitpicking, and fact-checking.

Cameron

I am extremely thankful for all of the people who worked with me and tolerated me throughout the lengthy process of book development, particularly my mother, father, brother, and sister-in-law. I can't explain the gratitude I have for my family and their ability to put up with the absences, stress, and limited schedule that comes with writing a book. Thanks to the entire 9magnets team: Dillon Carter, Nathan Feldsien, Michael Phelps, and Francisco Velazquez; they were amazing coworkers and even better friends throughout the entire process. I would also like to give special thanks to all the amazing people that I've had the pleasure to meet in the independent app development community that serve as my biggest source of inspiration every day. Cabel Sasser, I hope that this makes you proud.

Josh

I am incredibly grateful for the patience and understanding of my wife, Mallory, who was amazingly focused on planning our wedding while I was off working on this book. I'm also thankful for my parents, who never stopped encouraging me to work hard and see what followed, and for the grandparents, aunts, and uncles who feigned amazement at the not-so-creative writing of my childhood. I'm also indebted to the many teachers and editors over the years who offered invaluable lessons and priceless opportunities, especially Jeff Burton, Greg Halling, Linda Jegerlehner, Marshall King, Pat Milhizer, Paul Oren, Mark Schwehn, and Susan Swift.

ABOUT THE AUTHORS

Cameron Banga is the lead designer at a company he cofounded, 9magnets, LLC. He has worked on more than 100 mobile apps for clients ranging from professional sports teams, to educators, to large corporations. His first application, Battery Go!, quickly became an iPhone best-seller, and his apps have been recommended by the *New York Times*, *Fox Business News*, *Macworld*, *PC Magazine*, and many other media outlets. Cameron holds a B.A. in economics from Valparaiso University.

When not writing, Cameron is an avid photographer, novice runner, and coffee connoisseur in training. Cameron can be most easily reached via Twitter, at @CameronBanga.

Josh Weinhold is the assistant editor of the *Chicago Daily Law Bulletin* newspaper and *Chicago Lawyer* magazine. He spent five years as a political reporter and has written hundreds of news articles and feature stories published in the *Daily Law Bulletin*, the *Elkhart Truth*, the *Dubuque Telegraph Herald,* and on msnbc.com. He shared a National Press Club Online Journalism Award with other members of an msnbc.com and *Elkhart Truth* reporting team, and won The Chicago Bar Association's Herman Kogan Award for legal beat reporting.

He spends his free time slowly working through the long list of movies and TV shows he's been dying to see and fanatically following many real-life and fantasy sports teams.

This page intentionally left blank

CHAPTER 4

FIRST SKETCHES OF AN APP

You won't need a hammer or a screwdriver; maybe you'll need a tape measure—though preferably one in digital form on the top and side of your computer screen. Like any job, there's an established set of tools that most interaction and interface designers use to create their projects. Programs such as Photoshop, Balsamiq, xScope, and others are critical components of the interface-building process. In this chapter, you'll find a general strategic outlay for planning the design of a mobile application. Using the steps and techniques presented, you'll be prepared for the different phases a design evolves through during its infancy, before a programmer writes the first lines of code.

What Tools Do You Need?

A mechanic is only as good as his or her tools, or so the saying goes. The ones that care about their work the most are the ones that most significantly invest in their tools. It's true for auto body shops, and it's true for design shops as well.

Before making a serious effort to create an app, designers need to make sure they have the best equipment available at all times. When starting out, this can be a bit difficult, as new computers and professional software are often quite expensive. To avoid wasting money on improper tools, it's important to get the best bang for your tech buck.

Many of the tools and tips recommended in this chapter developed from labors of love: fondness and expertise forged over a couple of years and a hundred apps worth of experience. But it's important to note that there are no one-size-fits-all solutions when it comes to choosing tools or selecting a process to draft a design. The following recommendations come from a process that has led to the creation of several successful apps, but if you come across a piece of software better suited to your task at hand don't be afraid of going your own route. Likewise, the tools available to designers grow and evolve at a lightning-quick pace, and new products are constantly hitting the market that make design faster, easier, and more efficient. It's always worth giving new products a try, as any learning curve involved may pay off significantly down the road.

The first tool needed in a designer's supply kit is one that's essential to everyone from elementary school students to rocket scientists: a quality notebook, journal, or word processor. Being a successful interaction designer requires taking notes consistently and excessively. Everything from trends in the industry seen in other apps to thoughts on personal work should be documented for future reference.

note

Remember that the tool suggestions in this chapter are just that: suggestions. If you currently have a workflow that functions better than what's recommended, feel free to diverge (or, even better, share your setup with other designers online). The goal is to do apps well, regardless of the tools and methods used.

Interaction design focuses on the constant development of a product in order to increase usability and value, so there's always room to improve a work. As is also true for painters and comedians, inspiration doesn't always strike at the most convenient moments. Some of a designer's best ideas will come when he or she isn't working; they'll arrive while walking down the street or in the middle of the night. Always having a notebook or phone-based word processor handy is a great way to quickly jot down thoughts as soon as genius strikes. Try Field

Notes by Draplin Design Company and Coudal Partners. These handy, portable notebooks come in a standard size that's roughly the same aspect ratio, width, and height as a smartphone screen and, likewise, work well to approximately portray a scaled-down tablet screen. They fit well in a pants pocket or purse, and they're great for scrawling out quick ideas or sketching out design prototypes.

When it comes to computer hardware for a designer's utility belt, it's tough to suggest anything other than an Apple laptop running OS X, preferably the most recent version available so there are no issues with compatibility for Apple's development applications. There's no denying that all iOS development and interface design implementation and most Android development takes place on computers running OS X. Access to Windows is required, however, for Windows phone app development, so designers planning on taking that route will need to keep that in mind.

If you don't plan on doing any coding at all and most of your work will be focused on creating visual designs, you could be perfectly fine with a Windows PC. Do consider using a machine, however, that will allow you to commit code for the projects you plan on contributing to, even if you don't see yourself as the programming type. It can be very valuable for designers to have access to source code for modifying art files or making basic code changes, typography selections, or color choices. If you plan on developing for iOS, it will be well worth your while to have an Apple laptop or desktop so that you won't be limited in case you want to tinker with code in the future.

The most frequently recommended computer for mobile design is Apple's MacBook Pro, ideally one with a Retina display. The benefits of the mobility a laptop provides far outweigh the added power provided by a desktop. Apple's most recent laptops with Retina display are great for designing work that looks fantastic on the high-resolution displays found in most phones and tablets. Designing on a low-density display can be difficult, because in some cases you may not be able to preview app designs from Photoshop or a similar program in full resolution. If cost is an issue, the MacBook Air is an excellent laptop, but steer clear of the 11-inch model; such a small screen size will make design work difficult.

If a stationary computer is preferable based on your personal needs, a designer can't go wrong with an iMac, either. These need to be capable of professional-level functionality, so it's best to purchase the most well-equipped computer you can afford. If you're low on budget, Apple's Mac mini is a more than capable machine for design and development. The biggest and best system isn't always essential; for most practical purposes, Apple's recently redesigned Mac Pro is probably overkill for the type of work you'll be doing.

warning

UPGRADES CAN BE DIFFICULT It's important to note that for many Apple computers, specifically the MacBook Air, MacBook Pro, and iMac, it can be difficult or impossible to upgrade RAM or hard drive storage space after purchase. Carefully consider your spec decisions before ordering a computer.

Preferences for design software can vary greatly based on personal taste, but there are a few essential tools to look for in any program. First and foremost, designers need some sort of wireframe or mockup function that can take interaction ideas and translate them into a visual element programmers can use to begin their work.

Balsamiq by Balsamiq Tools LLC (see Figure 4.1) is the multiplatform industry standard for quickly creating visual wireframes. The application is built specifically for digital design work and comes equipped with many templates and styles that cater to building Web sites and mobile software. Balsamiq balances speed and style and also quickly visualizes interaction thoughts into something others can see, understand, and offer feedback on.

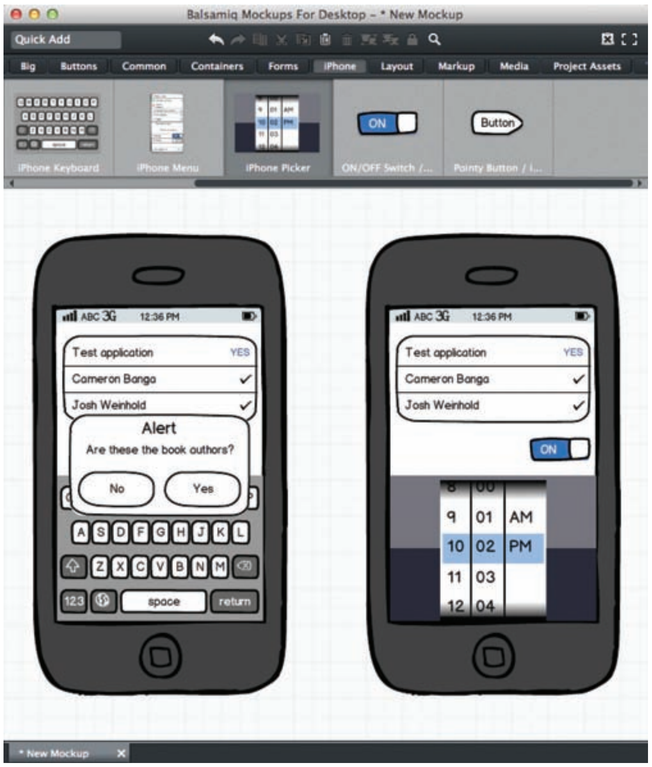


Figure 4.1 Creating attractive, quick wireframes with a tool like Balsamiq is rather simple, as designs take just minutes and can be extremely helpful in the visualization process.

Two other valuable sketching and early prototype applications of note are OmniGraffle by The OmniGroup and MindNode Pro by IdeasOnCanvas GmbH. OmniGraffle is a wireframing and digital prototype application in the same light as Balsamiq, but OmniGraffle focuses more heavily on creating work that looks close to reality. Such a feature offers output that's visually appealing for clients or stakeholders, but it does add time to the concept process. MindNode Pro, shown in Figure 4.2, is a mind-mapping application that's used for creating general text outlines. It's a fairly simple tool, but it allows a designer to take a simple idea, spread it out into actual words and thoughts, and then transform those thoughts into patterns that outline a more complete thought process.

MindNode Pro is a favorite tool of designers due to its ability to easily and quickly visualize ideas. It's also useful for a variety of non-app-related tasks. For one example, look no farther than this very page; MindNode Pro for iPad and OS X was used to visualize and outline each chapter of this book.

When it comes to rendering anything in pixels, meanwhile, Adobe's Photoshop is far and away the most popular choice for computer graphics creation and editing, and it's a piece of software that's used heavily in interaction and interface design. If it's something visual and not something done in code, odds are it'll need to be done in Photoshop. Adobe recently made a major model shift to its Creative Cloud platform, which is basically an all-you-can-eat buffet for their products. For a monthly fee ($50 currently), users have access to Adobe's entire Creative Suite. This is a great shift for designers who previously found the high cost of each Adobe program prohibitive, as they are no longer limited to one program but can instead now use other Adobe products when creating software, such as Illustrator for vector art creation or Audition for audio editing.

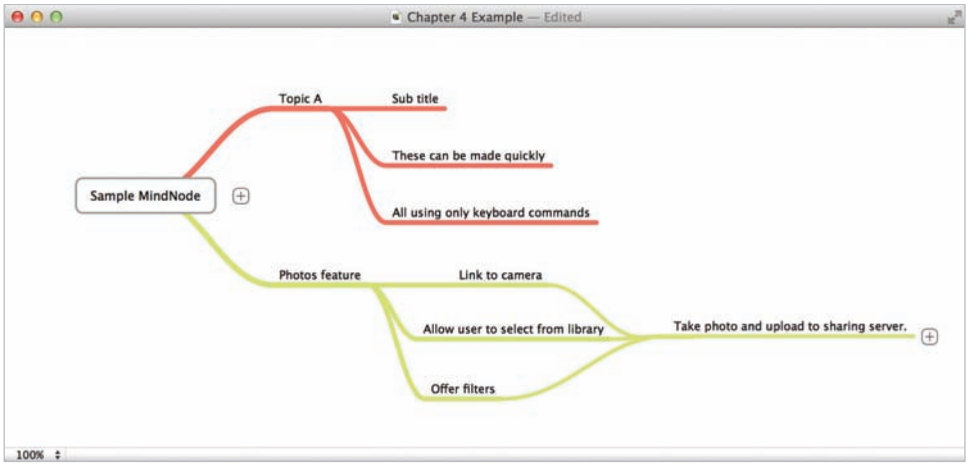

Chapter 4 Example — Edited
Sample MindNode
Topic A
Sub title
These can be made quickly
All using only keyboard commands
Photos feature
Link to camera
Allow user to select from library
Offer filters
Take photo and upload to sharing server.
100%

Figure 4.2 MindNode Pro offers a simple, clean interface for creating mind maps.

Still, there are other options available for small app-creation teams who find that monthly expense to be a creative barrier or for designers who simply want a product not offered by Adobe. Several strong—and extremely affordable—competitors have emerged recently, including Pixelmator by Pixelmator Team Ltd. and Sketch by Bohemian Coding. The downside of going with less popular products, though, is losing out on the wisdom of the crowds. Countless online tutorials, books, and instructional videos have been developed to walk users through basic and advanced Photoshop techniques, so individuals not experienced in visual design may have some trouble instantly mastering alternative programs.

Finally, a software gem that's absolutely imperative to have in a designer's tool kit is xScope by The Iconfactory. It's essentially the Swiss Army knife for interaction designers, offering a variety of magnification and pixel-measuring tools to use when analyzing an application on the iOS simulator or an Android virtual device. The tool is priceless because designers sweat to make sure every pixel is in exactly the right place while debugging and testing software.

It's somewhat difficult to describe what xScope does (or appreciate how well it does it) without using it. Essentially, the application makes it simple to measure a variety of important on-screen metrics when designing and developing apps. In Figure 4.3, you'll see an on-screen ruler and magnification loupe being used to inspect the visuals of a Web site.

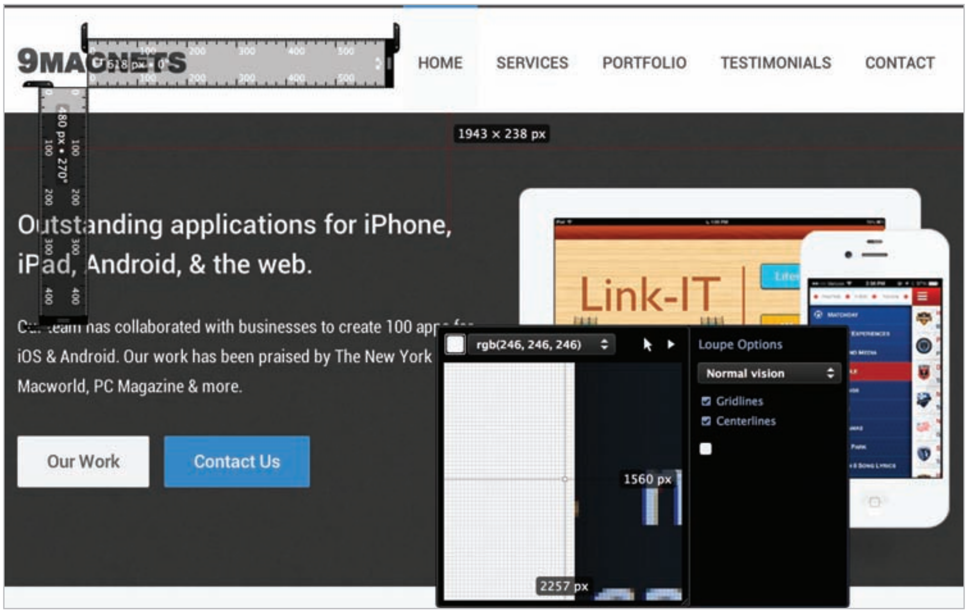

9MAGNETS
HOME
SERVICES
PORTFOLIO
TESTIMONIALS
CONTACT
618 px
480 px • 270°
1943 × 238 px
Outstanding applications for iPhone, iPad, Android, & the web.
Our team has collaborated with businesses to create 100 apps for iOS & Android. Our work has been praised by The New York Macworld, PC Magazine & more.
Our Work
Contact Us
Link-IT
rgb(246, 246, 246)
Loupe Options
Normal vision
Gridlines
Centerlines
1560 px
2257 px

Figure 4.3 xScope helps designers measure a variety of on-screen metrics.

Becoming a Designer

Once your design utility belt is firmly buckled and your tool kit fully equipped, it's time to determine how to actually tackle the job of becoming a designer. There's no certification exam to pass or credentials to acquire, but there are many classes to enroll in (for a fee) and even some universities offering to teach the trade of design; but are they worth your time and money?

If you're young and either in college or about to head off to it, by all means enter a program that's focused on design and product creation, even if it's not specifically geared toward the development of mobile apps. There's a great deal of benefit to be derived from going through a full college or trade-school program on how to become highly proficient in computer software creation.

But if your college years have passed you by or the cost or time required for a full course load is daunting, there's a nearly endless supply of free information available on the Internet that can help you become more adept at this craft.

Currently, most mobile app designers are either self-taught or have some background in computer engineering or another traditional visual design field. Eventually, though, the leaders of the industry five, ten, or twenty years from now will have gone through some post-high school program focused on software development.

Another way to hone your skills or grow your knowledge base is something often discussed by those looking to get into the world of app development: conferences. There's certainly no shortage of events, ranging anywhere from a day to a week in length, vying for developers' time and dollars. These sessions are often quite expensive, but they remain one of the only ways a programmer or designer can spend hours upon hours listening to or talking with titans of the industry.

Based on personal experience, conferences offer the opportunity to draw from a wealth of knowledge and enjoy a healthy dose of much-needed social interaction and networking. The face-to-face benefits of a conference cannot be understated, especially in the tech industry. Many mobile developers work alone at home or at small companies of two to three people and each one is perhaps the only person in town with such a hobby or profession. Thus, conferences offer a valuable opportunity to foster camaraderie between people with similar interests, providing both inspiration and motivation.

With many conferences making audio and video of lectures and roundtable discussions available via the Internet, though, any technical expertise gained by attending in person becomes less and less valuable. If you're paying your own way to an event, aim for the ones that are most affordable—something in line with what you'd spend on a short and cheap weekend getaway. Look at conferences as the entree of establishing friendships and interacting with individuals with shared passions that just happens to come with a side dish of learning. Don't break the bank on pricey conferences, and you won't be saddled with overeater's remorse the day after.

tip

For a good list of various conferences that anyone can attend regardless of their mobile platform of choice, check out http://lanyrd.com. The site is dedicated to helping connect users to different professional conferences.

Planning for a Specific Platform

Once the basic wireframes of an application are drawn up, it's time to move on and begin preparing for the intricacies of a specific platform. Now, you should start thinking about how an application will look and feel on one specific mobile device or another.

First, it's essential to find the developer documentation for the appropriate platform. The human-interface guidelines will be the most important document for a designer, along with any other design-specific documentation available from the platform's developer center. For iOS or Android, Apple and Google frequently update documents on human interfaces much like they do for API and other technical processes. Major mobile platform developers also have other documents available detailing how to implement specific looks and feels for common interface features, and they often update these style bibles after a major operating system update. Even if you're comfortable with a platform's interface guidelines, it's always important to check back with the developer's recommendations to see if anything has changed. Human-interface guidelines are most definitely a living document, sometimes even more so than the platform programming guides themselves.

While reading over a platform's documentation, it's valuable to make sure that one or more test devices are available at a designer's disposal. Ideally, a minimum of one fairly new and up-to-date physical device should be handy, and some virtual devices should be installed and loaded. These can be things such as an iOS simulator that's prepackaged with Xcode, an emulated Android device from the Android SDK, or something similar that allows a designer to run test applications on a computer. These vary by platform, so visit the manufacturer's developer resources page to learn what options are available.

tip

A good rule of thumb is to always have three devices for testing: one device that's new and uses the most recent technology, one that's old and contains the least powerful technology that you plan to support, and one dedicated for use in offline or other edge-case scenarios.

Once you get your hands on a device, ensure that you're comfortable using it. One recommended strategy during app development is to carry the targeted device as your primary phone or tablet for at least a week. After using it for several consecutive days, you'll become familiar with its common interaction practices and begin to appreciate how users work with the device in professional and personal settings.

New designers often make the mistake of using only screenshots or the human-interface guidelines document to draft their interface work for a new platform. Interaction design, though, is less about the look of an application and more about the feel and flow of how an application works. It's impossible to accurately judge what feels natural on a platform based solely on screenshots, and the guidelines document is written using colloquial terms that only make sense after using a device for a couple of days.

In Apple's human interface guidelines for iPhone and iPad, for example, the author uses the following sentence: "And, although people might not be aware of human interface design principles such as direct manipulation or consistency, they can tell when apps follow them and when they don't." Users will know if an application feels out of place, and there's no way for a designer to know if he or she has implemented principles correctly unless the operating system has been used personally for day-to-day tasks.

Once a device is in your pocket or backpack and you've studied up on an interface's official documentation, take a look at some third-party development resources such as books or blog tutorials on the design for your target platform. Although the interface guidelines will be your rulebook going forward, and the device itself will help you experience how to use the platform, advice and instruction from leading developers is one of the best ways to learn about real-world user expectations. Follow some top developers and designers on Twitter or RSS feeds to get a constant flow of information on how platform design changes daily. The mobile development community is still a very tight-knit group, and many bloggers or book authors are approachable and more than willing to discuss your interaction plan online or over lunch at a conference, so don't hesitate to reach out and ask for help.

And again, don't underestimate the power of social interaction. Find local groups or meetups with like-minded mobile developers; most medium- to large-sized cities have clubs with monthly meetings to discuss trends and evolutions in the industry. Getting together to chat and interact is crucial for members of an industry known for having its fair share of people working alone at home. You're designing for some of the hottest platforms in the world, and people love offering up their opinions, so go out and be social.

Starting with a Workflow

Now that you've got a device and have a plan of attack for learning about a specific platform and staying up to date on its news and trends, it's time to begin the real work: developing an application's interaction design. The best place to start is by composing a wireframe and building a general overview of the application's workflow.

If the world of interaction design is like a house, an application's workflow is the cement foundation, and the wireframe is the wood that supports the walls. It's not time yet to pick what type of doors to install or what color to paint the living room, but a number of important decisions are made at this early stage that will influence how an application works—decisions that will be very difficult to change once you progress further. It's crucial to make sure these choices are thought out well and thoroughly evaluated.

Begin this process by writing down or drawing a graph of a basic plan for what users will experience when first launching the application and then how they will move through it to accomplish a certain task. This initial phase of the workflow should be extremely abstract initially. The general purpose of this exercise is to understand the reasons why users will download this app, what their first impressions of it will be, and how information can be presented to them as quickly and efficiently as possible. You can conduct this process by using an application such as the previously mentioned MindNode Pro or you can simply use a large piece of paper with boxes, lines, and text that describes the setup and flow. You're essentially developing an advanced connect-the-dots process while also working to remove as many dots from the system as possible.

Figure 4.4 shows a relatively simple workflow design, starting with the user entering the application and ending with the user's purchase of a pair of jeans. The goal with the design is to minimize the steps needed to reach that end result and properly identify places in the interface at which a user might find interaction difficult. That helps a designer determine where to devote the most time during the development process. The key at this stage is simplicity, making rapid iteration easy as you see the need for changes while working on your ideas.

In this example, we realized it might be difficult to present a way for the user to quickly and easily pick out the exact size of jeans they want. We've got a couple of strategies in mind that might solve that problem, but we're not sure which one we like best yet, so we've jotted down a few notes to return to later.

Even at this early stage of working with a wireframe, it's not too soon to begin gauging user experience. One of the most common ways to evaluate an application is to calculate how many

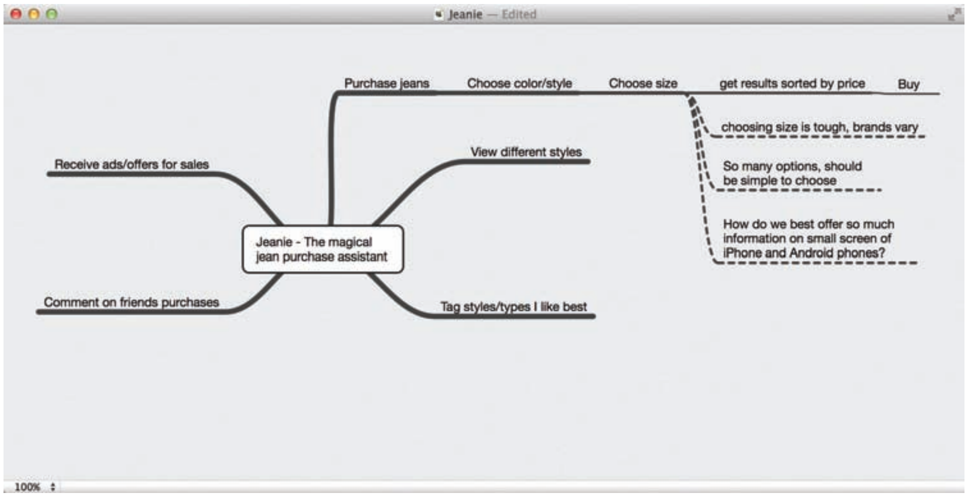

Jeanie — Edited
Jeanie - The magical jean purchase assistant
Purchase jeans
Choose color/style
Choose size
get results sorted by price
Buy
choosing size is tough, brands vary
So many options, should be simple to choose
How do we best offer so much information on small screen of iPhone and Android phones?
View different styles
Receive ads/offers for sales
Comment on friends purchases
Tag styles/types I like best
100%

Figure 4.4 MindNode Pro can be used to build a plan for a hypothetical application designed to help a user pick out a pair of jeans at a store.

taps the user must make or the number of page transitions required to go from launching an application to completing the desired task. As designers, we have two factors that are very much in our control: how easy it is for users to understand where to make input decisions on an interface and how many screen-to-screen transitions they must go through. Decisions made about those elements directly influence how much time a user spends moving through an application, and you're unlikely to find someone who enjoys an application that fruitlessly wastes his or her time. Respect the user, and always find the quickest way to get from Point A to Point B.

Ideally, designers should strive to strip away as much complexity and as many obstacles from an application as possible, removing until they can't remove anymore. Interaction design is all about creating an optimal experience for users, and for many reasons apps are optimal when they are the most simple. They're used on the go and on small screens, so complex experiences often do nothing but frustrate the average person. Designers should always be aware of those factors and aim to avoid complexities when designing a workflow.

tip

Another valuable metric—aside from measuring the number of taps or screens required to get to a solution—is measuring how much time it takes the user to complete tasks after entering the app. The faster the experience, the happier the user.

Meeting Design Expectations

Once a general workflow is laid out—something that looks like a combination of general word associations and a connect-the-dots puzzle—it's time to move on to a generalized wireframe.

At this phase of the design process, it's time to imagine and render every single interface piece and interaction method that will be replicated inside the application. Now, decisions will have to be made about whether to use elements such as integrated voice commands or advanced uncommon system gestures. The documentation created here will also be the primary way to communicate design concepts and philosophies to involved parties—programmers, managers, marketers, or other stakeholders—that might be involved in the app production process.

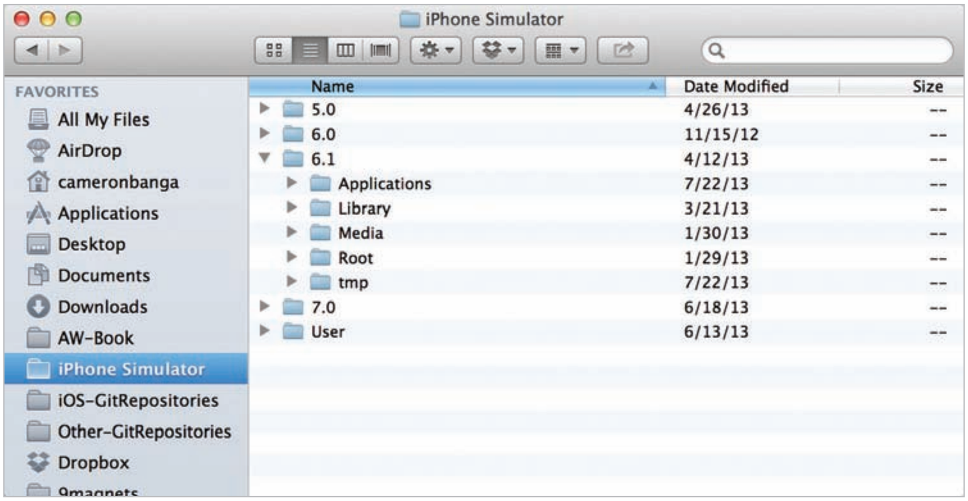
While wading into these waters, it's the perfect time to research how other applications with similar functions and features implement interaction design, especially ones developed and designed by the maker of the operating system itself. Keeping in mind prime examples of software on a platform helps guide how your own application should look and feel. It also allows you to spot flaws or problem areas in competitor apps, presenting an opportunity for your app to offer something different that helps it stand out in the marketplace.

tip

Constantly peruse the "Top Apps" lists for all major platforms that you plan to support and take note of how they handle complex interaction challenges. Hundreds of new applications are released each day, and the best of the best often tackle difficult problems in unique ways.

Once you've gained a good understanding of the way other applications address the problems your app might face and you've analyzed the best work on the platform by the people who made it, you can begin crafting a voice for your application. Will it be one that truly fits in with the rest of the apps on a user's phone, or will it be something that boasts a truly unique design and aims to stand out from the crowd? There are positives and negatives to both approaches, and now is the time to thoughtfully consider where your application will fit. Because most operating systems have a rather coherent universal design philosophy, it's important to remain cognizant of what breaking from that pattern will mean. If a user is aware of that design outlook, they know it by name—or at least by sight—and expect applications to look a certain way. Offering something strikingly different can be eye-catching, but it can also sometimes be unsettling to a user.

An operating system design philosophy is a deep concept that permeates an entire platform. Just look at Apple's Aqua visual interface design in OS X, unveiled in January 2000 but still in use today. Aqua (shown in Figure 4.5) is easily recognizable in the standard OS X window, with polished metal chrome capping off the top edge of the view; bright, glass-styled red, yellow,



Name	Date Modified	Size
5.0	4/26/13	--
6.0	11/15/12	--
6.1	4/12/13	--
Applications	7/22/13	--
Library	3/21/13	--
Media	1/30/13	--
Root	1/29/13	--
tmp	7/22/13	--
7.0	6/18/13	--
User	6/13/13	--

Figure 4.5 If you've worked with OS X before, you'll recognize Apple's Aqua visual interface design by its steel window appearance and blue tint.

and green buttons in the top-left corner; rounded rectangle buttons; and bright blue that highlights selected items.

Apple, though, sees Aqua as something much more than a basic visual look. To the company's designers, Aqua represents the foundation of the operating system's entire graphical user interface; it presents elements with a goal of "incorporating color, depth, translucence, and complex textures into a visually appealing interface," according to Apple's OS X Human Interface Guidelines.

Apple's plan was to use these principles in combination with an animation system that appeared to be as fluid as water itself to create designs that looked so great that (especially in the early versions, which far outpaced competitors at the time) you can understand what the writer is talking about and how that design principle set a standard for every single application on the operating system.

Clearly, Aqua is not just a visual style; it represents a design goal, one that Apple makes easy for developers to achieve in their own work. Aqua is also a great example of iterative design. The style was introduced nearly 15 years ago and has gotten better through 10 (and counting) major releases. The design language contained within it has evolved, but Aqua's core design philosophy remains unchanged.

As with Aqua, current mobile application platforms also have their own sets of design goals. It's extremely important for designers to understand the intentions and aims of these design goals and not just view them as a visual style to occasionally abide by. Currently, Google recommends that developers design to its Holo style, a system designed to unify applications' appearance, color scheme, and typography after a period in which Android software offered a very mixed bag when it came to interface design. The style has been hugely successful, creating a standard for applications that lets the platform appear distinct while also allowing for a design that is scalable and usable on multiple types of devices.

It's also worth noting that, because Android is open source, hardware manufacturers are free to customize the experience and make modifications to the standard interface design. Popular examples of this are Samsung's TouchWiz and HTC's Sense. As long as you design applications that conform to Google's Holo design standards, your interface should have no issues with being displayed on these manufacturer-specific designs.

Apple, meanwhile, is in a major transition phase, dropping its instantly recognizable iOS 6 look and feel in favor of a radically reimagined, visually simplified aesthetic and design functionality in iOS 7.

Well-designed interaction and interface designs often share an important trait: The designs are consistent across all applications on the operating system, and the user can easily predict how a common button, gesture, or interface structure will respond to interaction. There's much to be learned from using these operating systems repeatedly; you gain an understanding of the feel of the system, but you also develop a sense of what the creator's design expectation is. Through that, a designer can decide if it's wiser to stick to the platform norms or venture out onto a creative new path.

In most situations, especially for novice or inexperienced designers, it's best to stick to the platform's design conventions. As a result, you're less likely to make a serious design mistake or create an interaction design method that is confusing and discouraging to users. The interaction methods that are baked into an operating system are tried and true, tested with a multitude of usability drills and established as the common (and often best) practices on a system. By venturing out and attempting to create a new interface and interaction style, a designer risks stepping too far away from a user's comfort zone.

tip

New designers may see sticking to standard user-interface conventions as boring, opting instead to get wild and pursue their own creative ideas. But remember, the first goal of interaction design is to create something that works, not something that breaks the mold. Simple and boring trumps complex and confusing every time.

That's not to say there aren't wonderful examples of designers taking risks and reaping big rewards as a result. Look at Loren Brichter, a digital designer renowned for his creation of the pull-to-refresh interaction method, now common among thousands of apps on multiple platforms. Brichter took an action that was fairly common—scrolling up and down to view content—and used an "excessive pull" gesture at the top of a page to launch a screen-refresh function. The design is beautiful, immediately discoverable, easily comprehensible, and visually hypnotizing, and it in no way interferes with the rest of the application's interface.

Keep in mind, though, that Brichter was originally an Apple designer who worked on software for the first iPhone. When he created this new interface technique, he was already an accomplished expert in the field and understood the ramifications of what he was building. His story presents an important lesson on attempting to create new application interaction types: When choosing to throw caution to the wind and ignore pre-existing conventions, a designer had better know full well what he or she is doing.

Wrapping Up Design Documentation

Once the relatively primitive sketch of an app's general look and feel is complete—and thought has been put into its interaction and usability—it's almost time to move to the next big phase of the process and begin turning your design ideas into pixels and programming code. Work in the wireframe and early design stages shouldn't be brushed aside, however, as it's important to create as detailed a preliminary document as possible. Often, a designer will be tempted to jump quickly to Photoshop files or other advanced design work, but devoting extra time to these early steps will pay off down the road. An extra hour or two spent creating documentation can save dozens of hours further along in the project.

Keep this key guideline in mind: If you can't describe an animation, gesture, or other piece of interaction implementation to a programmer or teammate in a simple sentence or two, it probably needs additional refinement or further thought. The documentation you create at this stage of development will be the foundation for everything else on the project, from programming, to art, to testing. Be as direct and explicit as possible when discussing and writing directions for implementation.

Software development can often be like the "telephone" game that kids play in which a phrase is whispered from one person, to another, to another. Usually, the sentence the last person in the chain says out loud is far different from the one the originator first uttered. In app development, the lead designer is much like the person at the start of that game. If the direction and plan isn't clear, concise, and simple, it's likely that the vision will get misinterpreted somewhere along the development chain, resulting in an application that's much different than the one the designer intended.

Creating Pixel-Perfect Digital Mockups

After fully documenting the application in a wireframe or sketches, it's time to start creating art assets for the implementation of the software's design. Some designers may only be working on interaction design and concepts—such as interface feature implementation or gesture utilization—and as a result won't be implementing the actual visual design of the project. If that's the case for you, feel free to skip this section.

Creating art designs for a project is often one of the most difficult concepts to teach in any field of design, not just app development. Although interaction is very much an objective concept, and a platform's descriptive documentation clearly outlines when to use gestures, aesthetic design is much more subjective. Ahead are general tips and recommendations for creating app art, but for those looking to get additional help on aesthetic design, consider taking an art class or reviewing books on visual design principles.

Most of the example work described in this section is performed using Adobe's Photoshop, an industry-standard tool for anyone in the creative arts. The program allows designers to use a variety of visual tools such as brushes, shapes, and erasers to create nearly anything imaginable. Developing your own style and skills in Photoshop is something that takes time to master; the best artists in the industry often have a dozen or more years of experience and are still learning and growing.

If you're a new designer and uncomfortable with Photoshop, consider searching for tutorials online that best mimic the specific visual style you hope to achieve in your app and then use blog posts, podcasts, or videos to walk you through the steps needed to create this look. There are thousands upon thousands of Photoshop instruction sources online, and they can be an invaluable resource. Likewise, there are many books on Photoshop that can help you tap into the potential of every tool the program has to offer.

note

Photoshop talents and design skills require a lot of practice and plenty of trial and error to develop. Do you see an icon design or app style that you like? Practice by trying to recreate the look in Photoshop. Your first few attempts will be difficult, but with repetition you'll quickly learn how to create similar designs.

A great visual design is a very important component of interaction design; if a designer can't fully represent how to interact with an app via simple text, iconography, and interactive features users won't be able to understand the software, and in turn the app won't see much success.

Visual cues create a path for users, helping them find safe ground so they don't fall astray. That's a reason why it's often wise for a project's interaction designer to also be the visual designer, because a uniform thought process by a single individual helps maintain coherence between interaction intention and visual implementation. Imagine interaction design as the artistic idea, Photoshop or programmed code as the paintbrush, and the visualization of the app as the canvas. It's much easier to bring a work of art to life if only one person is in control of the paintbrush from start to finish.

The amount of art required for an app can fluctuate from project to project based on the technical requirements of the software, the desired visual aesthetics, and the platform, so there's no straight answer for how much art will be needed every time. A thorough discussion with the project's lead programmer is the best way to determine how to bring a wireframe to life with both art and code. Once again, this is where that extra time spent developing a detailed wireframe and application design document comes in handy; a programmer can review these and instruct the designer how they want the project's visual assets to be created.

On iOS and Android, most art will be produced in the PNG file format and will be imported and referred to in code to make the visuals appear on screen. It's best to create as little art as possible in Photoshop or another image-based program, instead implementing elements with code for native interface design pieces. Code is typically more nimble and able to be altered more easily while also being rendered on screen more quickly. Applications will thus be more responsive and require less work while also being less likely to break down when the operating system changes in the future. Each programmer has a different opinion, however, on what they prefer to do in code and what they want to do using other assets, and so constant communication with the app programmer is required.

Reiterating Before It's Too Late

One of the primary goals of interaction design is to be constantly iterating on an implementation in an effort to improve upon the original work. Although you've already done this for a wireframe, you now have actual art assets in PNG or another format along with full Photoshop design files that will aid in a more complete analysis.

> note
>
> Remember, iterating on a design is the thoughtful and intentional process of taking original work, questioning decisions, and potentially revising and recreating parts of the project in an effort to improve its design.

Now is the perfect time to sit down and review your design work with every stakeholder in the project, from the client who's funding it to the programmer that's implementing the design. The following five questions are often simple and easy to answer when working only with concept art designs, but they'll grow more difficult and expensive to resolve once the application becomes actual lines of code, so it's best to address them now:

1. **Does the app look like it will fit in with the platform?**

   It's a designer's prerogative whether he or she wants the work to blend in on the platform or not, so this question can often be answered either positively or negatively and still be OK, depending on the person's intentions. What's most important, though, is creating a coherent design that looks and acts like mobile software.

2. **Will users be able to use the application with no guidance?**

   Long gone are the days in which each piece of software came with a hefty instruction manual. Mobile apps must be capable of remaining useful while standing alone, because the production team won't be there to guide the user along the way. This question can often be answered by showing the Photoshop work to another tech-minded person who can offer an outside perspective while also understanding what the designs in the program are intended to represent.

3. **Can the programmer implement the design with art assets and design documentation only?**

   Most likely, a designer will be working hand-in-hand with the programmers on a project, who will hopefully be able to ask questions about why something was designed the way it was and how it should be implemented. This isn't always the case, though, and designers should be prepared for that possibility. Once design documentation is handed off to a manager or programmer, they should be able to deduce the designer's intentions and planned interaction design without being required to check in with concerns every five minutes. If a programmer can't create a design with only the documentation provided, more work is likely.

4. **Will the design age gracefully?**

   Age can wear heavily on things, and mobile apps are no different. There's an adage frequently quoted when creating logos for corporations or businesses that says that the goal should be to design something that would have looked outstanding 100 years ago, would look outstanding today, and will look outstanding 100 years in the future. Strive for a general style and brand that will remain tasteful and visually appealing as an operating system or platform changes over time. This can vary in difficulty based on platform, but it's wise to avoid trendy "flavor-of-the-week" design practices that will fall out of fashion quickly. Instead opt for classic, traditional, platform-friendly looks.

5. **Does the design meet future project goals?**

   A lot of new designers get tripped up in their development by creating a great first version of an app but failing to allow space for future feature improvements that will be necessary in subsequent releases. Designs shouldn't be handcuffed by hypothetical "maybes," but it is important to consider how an application's design might evolve after another six or twelve months of work. Take a journaling application, for example, in which a designer uses a swipe interaction to open various menus but fails to recognize that the app's 2.0 release will include a photo-adding function in which users will swipe to move through pictures. This will cause an interaction conflict, leading to a complete design overhaul a few months after launch, complete user confusion, a completely unreliable design, or some other terrible cocktail of those poor-design consequences.

note

Identify each stakeholder in a project long before work begins, especially if your project is for a client. Stakeholders will be managers, bosses, programmers, and anyone else who has a vested interest in a project. Moving too far along with the design before receiving stakeholder approval on a feature or style may result in work disapproval and unsatisfied clients.

As you get more and more involved with the world of software, you may run into an increasingly common programming approach called agile software design that runs in direct contradiction to the strategy just outlined. Agile software design involves design and programming team members working out basic features and plans to constantly add more to the product based on user response, testing, and developer experience. It's a great strategy, but only for those who are quite familiar with software development and are comfortable with how to handle a product constantly in motion. If you're new to design and programming, hesitate before adopting this strategy; you may be better off thoroughly thinking through a design prior to the start of programming.

Preparing for the Next Stage

Although the app remains in its infancy at this phase of the development process, it's not too soon to take formal opinions of it from potential users or colleagues. As designers, it's often our job to dictate or direct the development path and make important decisions about the project, but we're by no means dictators. Don't let yourself be above healthy discussion or critiques of your work from *anybody*. Each and every voice that responds to your output can be valuable, so don't jump to dismiss the opinions of others who don't have design experience. Apps aren't just for the tech elite; everyone from mothers to bank tellers to baristas are potential users too. Inevitably, someone in such a position will point out a bad design idea.

Ultimately, if you can't defend your own design it's probably not that great of a design to begin with. Be willing to question and evaluate your own ideas with the goal of an improved user experience in mind. Developing apps is very much a team sport, and your work will inevitably grow with thoughtful critique and constant iteration.

IN-DEPTH

In order to build apps, you need apps. A few great software-development tools were briefly mentioned in this chapter, but here's a more detailed look at these favorites, along with a few additional gems that you might come to depend on. These pieces of OS X software will give you a leg up on the competition when designing your app.

- **Adobe Photoshop** is the gold standard by which all other design tools are judged. Originally released by Adobe in 1990, the software has been used for a variety of graphic design purposes and has permeated society to such a point that "to Photoshop" has become a household verb. Regardless of the platform you're designing for or the platform you're building with, Photoshop will be an indispensable tool.
- **Skitch** is a simple application currently owned and managed by Evernote that allows for simple text and graphical markup on screenshots captured by a computer. It's very quick to use; just take a screenshot and then mark up the image with arrows to point out errors, text to explain intention, or simple shapes to note where content should be. When building work in a preproduction emulator, Skitch is a useful tool for quickly noting where interface errors exist so that programmers can fix bugs and improve the app.
- **Balsamiq** is a multiplatform tool that allows for rapid creation of a basic software wireframe, which is used to show stakeholders and programmers how an interaction design works when programmed. Clear communication is a necessary skill for any interface designer, and Balsamiq is a great tool to graphically indicate design intentions to the people who will be coding your work.
- **xScope** is the Swiss Army knife of interaction design, with a variety of invaluable tools to help improve and iterate on an interface. The application includes various measurement utilities, color indicators, and magnification tools that allow a designer to zoom in and view tiny details easily. xScope—created by Iconfactory, a team known for building some of the most beautiful interfaces available on OS X and iOS—provides a great way to double-check that you've properly placed all of your interface pieces.
- **Pixelmator** is renowned as a worthy competitor to the almighty Photoshop. It's an extraordinary digital art enhancement tool currently available for much less than a single monthly subscription payment to Photoshop. If you're a novice designer looking to get your feet wet with as little cash overhead as possible, Pixelmator is your soulmate.

Conclusion

A carpenter doesn't build a house the first time he or she picks up a hammer. A writer doesn't crank out a great novel the first time he or she sits down at the keyboard. Likewise, it takes time and development for designers to become comfortable with—much less master—using the tools of their trade. You've just begun to crack the surface of the work involved in creating an app by learning about the early steps of design. With a solid foundation laid, you can now begin building the framework of your original mobile creation.

This page intentionally left blank

This page intentionally left blank

INDEX