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

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

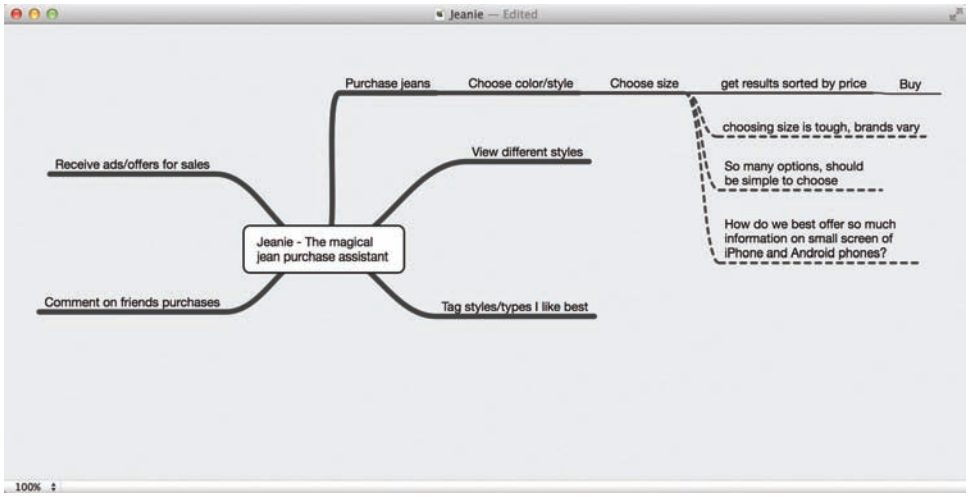

Figure 4.4 MindNode Pro can be used to build a plan for a hypothetical application designed to help a user pick out a pair of jeans at a store.

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,

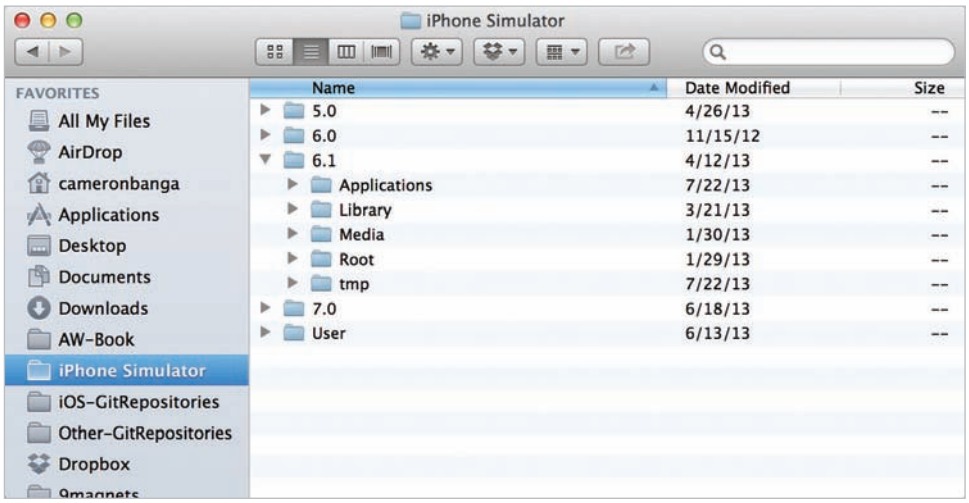

Figure 4.5 If you've worked with OS X before, you'll recognize Apple's Aqua visual interface design by its steel window appearance and blue tint.

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