7

Prototyping App Concepts

THE WORD PROTOTYPE comes from the Greek *protos*, which means "first," and *typos*, which means "impression." In the 1600s prototyping was used to describe the first impression from a printing press. Over time, its meaning has evolved to include the early forms of many things: automobiles, retail stores, home appliances. Perspectives on prototyping often differ depending on whom you ask—designer, developer, researcher. Regardless of the industry or discipline, I find it instructive to refer to Bill Moggridge's definition from *Designing Interactions*:[1]

> *A representation of a design, made before the final solution exists.*

This chapter looks at various iPhone prototyping approaches—paper, software, and video—and suggests how to choose the best approach for your app.

This chapter also includes case studies on prototyping at Dan4, Inc., and on the What's Shakin' iPhone app. Here you'll find insights into how the application design teams used prototyping to conceptualize their applications.

[1] Bill Moggridge, *Designing Interactions* (MIT Press, 2007).

Why Prototype?

Prototypes can help you solve design problems, evaluate designs, and communicate design ideas. These up-front activities can also expedite the development process, saving valuable time and money.

> *The most common estimate is that it's 100 times cheaper to make a change before any code has been written than it is to wait until after the implementation is complete.*
>
> —*Jakob Nielsen*[2]

SOLVE DESIGN PROBLEMS

Prototypes can be an efficient way to work through design problems before getting deep into coding. They can help address everything from higher-level conceptual issues to lower-level interactions. For example, imagine that you're creating a messaging app that will display a transition when users move messages from one folder to another: What is the optimal speed of the transition? What is the best form for the visual feedback? How do these two elements work together? Storyboards with directional arrows could illustrate the general concept, but an interactive prototype would be more effective at fine-tuning the solution.

EVALUATE DESIGN IDEAS

Prototypes are often used to evaluate design ideas—concepts, flows, and interactions—before investing development time. Evaluators may include the designer, design colleagues, and, of course, end users. Internal reviews can take a critique format or employ user-centered design methodologies such as heuristic evaluation, as discussed in Chapter 5, "Evaluating the Competition." Although internal reviews are tremendously valuable, they are no replacement for usability testing, which will be discussed in Chapter 8, "Usability-Testing App Concepts."

COMMUNICATE DESIGN IDEAS

Often prototypes are the only way to effectively communicate your app idea. In particular, apps that interact with the "real world"—location-aware apps, bar-code-scanning apps, voice recorders—must go beyond static screen designs to truly tell their stories. They need context: the people, places, and objects that are an integral part of the app.[3] Similarly, immersive apps such as musical

[2] Jakob Nielsen, "Paper Prototyping: Getting User Data Before You Code," www.useit.com/alertbox/20030414.html.

[3] Marion Buchenau and Jane Fulton Suri, "Experience Prototyping," *Proceedings of the 3rd Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques* (ACM, 2000).

instruments and games are considerably less compelling when presented in a static sketch format. Prototypes will take your app off the page and into a format that feels closer to the real thing. These may be presented within your company, shared with investors, or used to elicit feedback from users.

Common Questions

This section provides answers to common questions with regard to prototyping.

HOW MANY VARIATIONS SHOULD I PROTOTYPE?

Ideally, you should try to prototype a few divergent directions in the early design stages. As the design progresses, teams tend to get attached to one direction so it may be challenging to change course. Be sure to choose your prototype medium wisely—lower-fidelity options like paper make it easy to explore multiple directions, whereas some higher-fidelity tools tend to encourage incremental or superficial design changes.

HOW MUCH OF THE APP SHOULD I PROTOTYPE?

The scope of your prototype will depend on the design stage and your goals. At the onset of your project, it's beneficial to take a holistic approach to the prototype. This doesn't mean you must prototype every single screen and interaction, but you'll want to cover the primary scenarios identified in your up-front user research. In the middle to later design stages, you may return to prototyping to help resolve a particular flow or interaction issue. If you are creating a slideshow app, for example, you may want to fine-tune the transitions via a prototype.

WHAT IF THE DESIGNS AREN'T COMPLETELY WORKED OUT?

You don't have to wait until every aspect of the interface is completely worked out. It's fine to use *Wizard of Oz*[4] techniques to demonstrate certain aspects of the app. *Wizard of Oz* techniques require a human to simulate app interactions. For example, let's say a usability participant wants to search restaurant listings on your app but you haven't coded search yet. With a *Wizard of Oz* approach, you could ask the participant to wait a moment while the app "processes" the request. In the meantime, you or one of your colleagues could search for the information and provide the results on the fly. In addition to uncovering usability issues, this approach could help you refine requirements before coding begins.

[4] "Wizard of Oz Experiment," http://en.wikipedia.org/wiki/Wizard_of_Oz_experiment.

NOTE

The Lorem Ipsum web site (www.lipsum.com) has a generator to easily create filler text. The site also provides history on the origins of this practice.

WHAT IF MY SUPPORT CONTENT ISN'T FINALIZED?

Jared Spool recommends "Incredibly Intelligent Help"[5] to simulate a help system. Essentially, when users tap on help, the human "computer" responds to their questions. This approach provides a relatively seamless experience and can also identify areas of the interface that need design improvements or support content. On the other hand, be conservative with *greeking*, the practice of including *lorem ipsum* and other filler text instead of real text. If the text is central to the user experience, include draft copy and then iterate based on user feedback.

WHAT IS THE APPROPRIATE LEVEL OF FIDELITY?

The fidelity of your prototype should match the design challenge at hand as well as the stage in the design process. For example, paper prototypes (FIGURE 7.1) can typically uncover most flow and terminology issues but are less successful when it comes to low-level interaction issues. This doesn't mean that paper prototypes should always be your starting point. As discussed later in the chapter, some apps may require a higher-fidelity prototype earlier on.

If you plan to present to company executives or investors, you should assess their comfort level with low- versus high-fidelity prototypes. Some individuals may view low-fidelity prototypes in a negative light since they can be "rough" in appearance. That said, if you want to convince them otherwise, consider how Jane Fulton Suri, a partner and creative director at IDEO, assesses whether a prototype is effective: "If it is [a good experience], people get so involved in the experience that they forget about the limitations of the prototype."

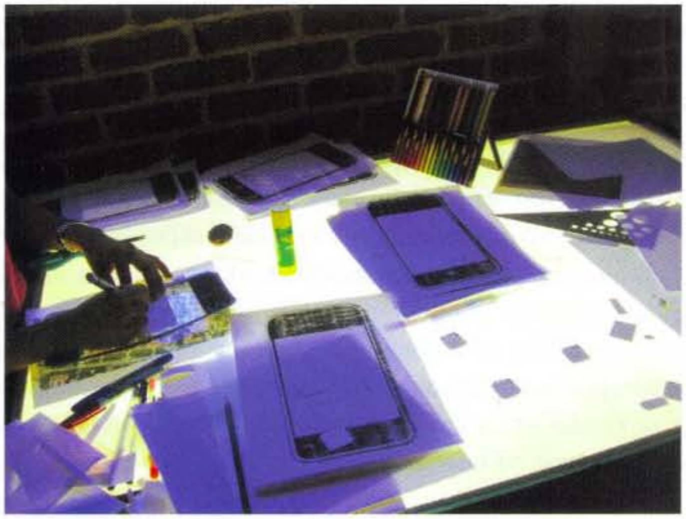

FIGURE 7.1 Paper prototype for a gaming app *(Courtesy of Dennis Paiz-Ramirez, photographer)*

[5] Quoted in Carolyn Snyder, *Paper Prototyping* (Morgan Kaufmann, 2003).

WHAT SHOULD I DO BEFORE I START PROTOTYPING?

Before you start prototyping, create app sketches, as discussed in Chapter 6, "Exploring App Concepts." If you haven't done so already, place these sketches in a high-level application flowchart. FIGURE 7.2 illustrates an application flowchart for a dictionary app; notice how the legend includes symbols for supported gestures. A flowchart provides a holistic view of your app and serves as a blueprint for your prototype. In the early design stages, focus on the "happy paths" that represent typical scenarios, not ones that generate unusual error conditions.[6] Edge cases can be added once you narrow down your concept.

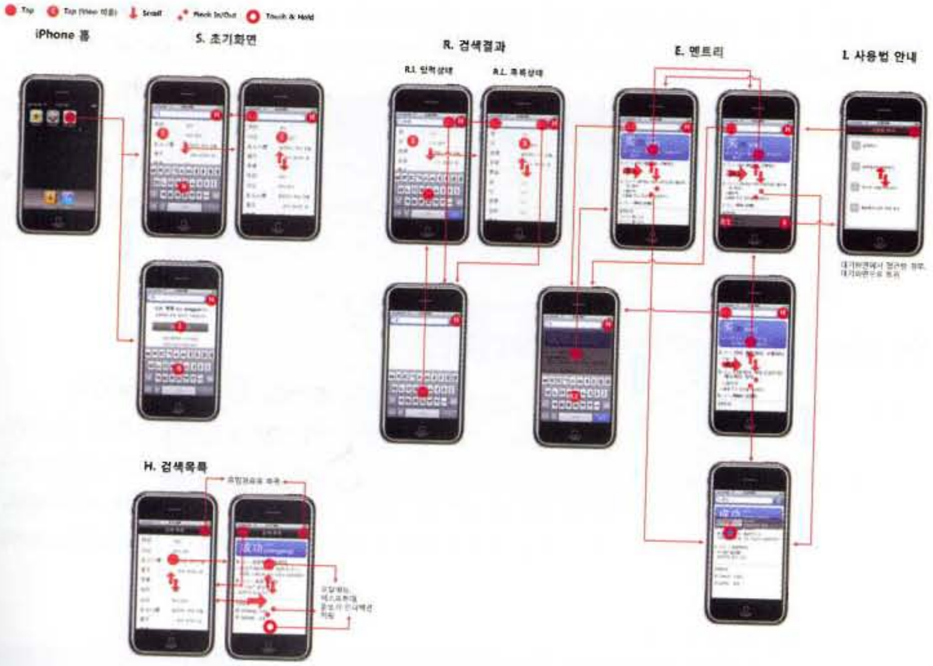

FIGURE 7.2 High-level application flowchart for a dictionary app *(Courtesy of Tony S. Kim)*

Other points to keep in mind when working on your app flows are the following:

- **Streamline, streamline, streamline.**

  As mentioned earlier, mobile users have limited time, so your app flows should be as succinct as possible without compromising usability. To that end, look for ways to combine or remove steps in multistep processes. For example, wizards are great for app setup and other linear processes (e.g., shopping checkout), but they can slow users down when used for frequent tasks, especially those with many optional items.

[6] "Happy Path," http://en.wikipedia.org/wiki/Happy_path.

- **Provide multiple ways to access information.**

  Users are often faced with dead ends, particularly when drilling down list views, but the app experience can be more fluid. For example, news article views could have cross-links to related articles, but many apps force users to navigate back to the original list view. Similarly, maps that contain points of interest (POI) should allow users to go directly to the POI, instead of requiring them to return to the corresponding list view.

- **Keep users within context.**

  As much as possible, try to keep users within your app. Leaving your app means users will require additional time and effort to reorient themselves, increasing the likelihood that they will not return. For example, many apps force users to visit their web site for help via Safari. Unfortunately, if users can't easily refer to their original problem, the external help may be useless. If users *must* leave your app (e.g., for map directions), at least provide a warning. When users return to your app, they should see the last screen visited, known as "saving state."

NOTE

For a broader discussion on prototyping (not iPhone-specific), consider reading Todd Zaki Warfel's *Prototyping: A Practitioner's Guide.*[7]

Prototyping Approaches

TABLE 7.1 summarizes five different prototyping approaches, from low-fidelity paper prototypes to the iPhone SDK. As the iPhone app space continues to evolve, you may find other approaches well suited to your application space. Be creative—adapt these as needed and formulate your own prototyping strategy. For example, audio can be incorporated into all of the options via a recording or live voice-over.

TABLE 7.1 **Alternative iPhone App Prototyping Approaches**

Prototype	Strengths	Weaknesses
Paper	Cheap and fast. Good for identifying conceptual, flow, and terminology issues.	Difficult to show low-level interactions; harder to simulate information-rich apps.
Static images on device	Incorporates iPhone form factor. Good for addressing visual issues, e.g., text size.	Limited interaction possible; essentially click through screen to screen.
Interactive on device	Incorporates iPhone form factor and some level of interactivity.	Achieving desired interactivity can require a significant amount of time.
Video	Storytelling approach that provides contextual information essential for location-aware and some immersive apps.	Can be time-consuming if many iterations are needed. Less suitable for usability testing.
iPhone SDK	Code may sometimes be used for the final app design.	Can be costly and less malleable for up-front iterative design.

[7] Todd Zaki Warfel, *Prototyping: A Practitioner's Guide* (Rosenfeld Media, 2009).

PAPER PROTOTYPES

Paper prototypes are essentially paper models of your iPhone apps (FIGURES 7.3–7.4). They can be used as a communication tool, but they are often developed for usability testing. In these situations the designer or developer plays "computer," hiding and showing user interface elements as needed. In contrast to electronic prototypes, Jared Spool, the founder of User Interface Engineering, describes paper prototypes the following way:[8]

> *We think of paper prototyping as the coarse-grain sandpaper and electronic-version testing as the fine grain. Once we've used the paper prototypes to validate what each screen contains and how it will work, we then move over to the electronic version to fine-tune the look and feel.*

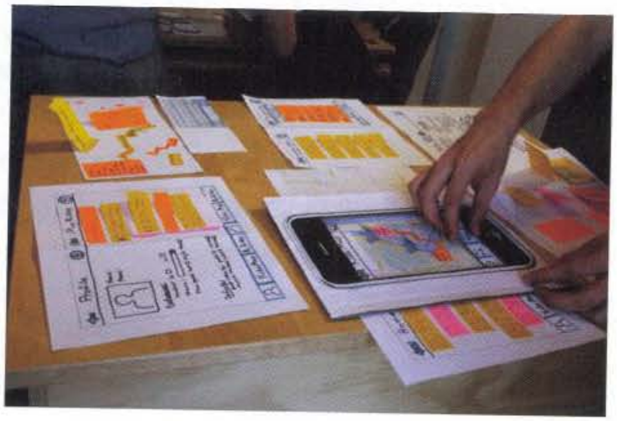

FIGURE 7.3 Paper prototype of a ride-sharing iPhone app *(Courtesy of Alex Jameson Braman, Joseph Lau, and Andreas Nomikos)*

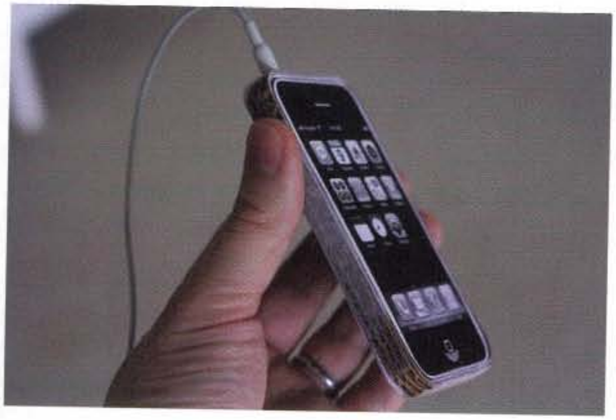

FIGURE 7.4 Paper prototype of an iPhone with the Home screen *(Courtesy of Steven Toomey)*

[8] Jared Spool, "Looking Back at 16 Years of Paper Prototyping," www.uie.com/articles/looking_back_on_paper_prototyping/ (July 2005).

Benefits

The benefits of paper prototypes range from quick iterations to improved collaboration:

- **Quick iterations**

  Paper prototypes enable you to rapidly iterate and experiment with many ideas. The modest time investment makes it easier to throw away less promising directions.

- **Inexpensive**

  Ordinary office supplies can be used for paper prototypes: Sharpies, Post-its, printer paper. Most important, these up-front paper iterations can reduce costly changes on the development end.

- **Collaborative**

  Paper prototypes do not require any technical skills; thus everyone (users, too!) can participate.

- **Easy to edit**

  You or your users can edit paper prototypes on the fly (e.g., change labels, add screens, add buttons).

Issues to Explore

User experience issues often explored with paper prototypes include

- **App concept**

  Do users understand your app's central concept?

- **Workflows**

  Is the overall navigation clear? Are there too many steps to complete tasks?

- **Information organization**

  Does the current organization match users' expectations?

- **Terminology**

  Are the labels on tabs, screens, and buttons clear?

- **Additional features**

  Over the course of evaluating your app, you may uncover additional features that users need. Users may vocalize these needs, or you may observe them trying to complete tasks not supported in the app. You may also learn which features users *don't* want, which could save valuable development time.

Benefits

The benefits of paper prototypes range from quick iterations to improved collaboration:

- **Quick iterations**

  Paper prototypes enable you to rapidly iterate and experiment with many ideas. The modest time investment makes it easier to throw away less promising directions.

- **Inexpensive**

  Ordinary office supplies can be used for paper prototypes: Sharpies, Post-its, printer paper. Most important, these up-front paper iterations can reduce costly changes on the development end.

- **Collaborative**

  Paper prototypes do not require any technical skills; thus everyone (users, too!) can participate.

- **Easy to edit**

  You or your users can edit paper prototypes on the fly (e.g., change labels, add screens, add buttons).

Issues to Explore

User experience issues often explored with paper prototypes include

- **App concept**

  Do users understand your app's central concept?

- **Workflows**

  Is the overall navigation clear? Are there too many steps to complete tasks?

- **Information organization**

  Does the current organization match users' expectations?

- **Terminology**

  Are the labels on tabs, screens, and buttons clear?

- **Additional features**

  Over the course of evaluating your app, you may uncover additional features that users need. Users may vocalize these needs, or you may observe them trying to complete tasks not supported in the app. You may also learn which features users *don't* want, which could save valuable development time.

Challenges

As mentioned previously, paper prototypes are less suitable for refining low-level interactions such as transitions, scrolling, and swiping. They may also be less useful for certain classes of apps, such as musical instruments, videos, and games.

HOW TO DO IT

iPhone paper prototypes typically include the "device" and some combination of screens, overlays, and controls. Steps for creating paper prototypes are summarized in this section.

Step 1: Gather materials.

In the previous chapter we listed office supplies that can be used when brainstorming and sketching your app designs; these items are also useful for paper prototyping. In addition, you may want to have the following materials on hand: cardboard, removable tape, glue, correction fluid, and scissors.

Step 2: Determine the form factor.

At some point your designs will have to match the iPhone screen dimensions—320 × 480 pixels (640 × 960 for iPhone 4)—but paper prototypes can be less exact. Using a larger form factor will make it easier for others to interact with the design (e.g., rearrange the layout and write in text fields).

Step 3: Create the prototype.

Your prototype will include a background, the screens, and the user interface controls. As you create the prototype, be sure your scenarios, as discussed in Chapter 4, and application flowchart are readily available.

Background

If your prototype is much larger than the iPhone, you may want to frame your designs with a cutout iPhone made with foam core or cardboard. This frame can help orient participants when usability-testing your app. Alternatively, if your prototype matches the iPhone dimensions, you can adhere it to the device, potentially making it "feel" closer to the real thing.

Screens

Your app screens can be hand-drawn or screenshots. Hand-drawn sketches tend to elicit high-level conceptual feedback, whereas screenshots may lead to low-level visual feedback. If possible, stick with one approach, not half hand-drawn screens and half screenshots. A few notable exceptions are photos, maps, and keyboards: Printing these out will save time, and they'll work fine when combined with hand-drawn sketches.

Prepare the Controls

This section includes tips on building standard controls for your paper prototype.

- **Tab bar**

  Create highlighted and non-highlighted versions of tab states (FIGURE 7.5). Use text if you haven't decided on the appropriate tab icon.

- **Keyboard**

  As mentioned earlier, you can use hand-drawn keyboard sketches or screenshots (FIGURE 7.6). It's not necessary to have the pressed state for each button, but pay attention to the default colors and special contextual keys such as those for web and email addresses.

- **Sliders**

  Sliders can be created with a tiny piece of construction paper folded over a narrow strip of paper (FIGURE 7.7). If you're short on time, you can provide verbal feedback as the user moves a finger back and forth across the slider. This verbal approach can also be applied to progress bars (e.g., "Downloading 1 of 10").

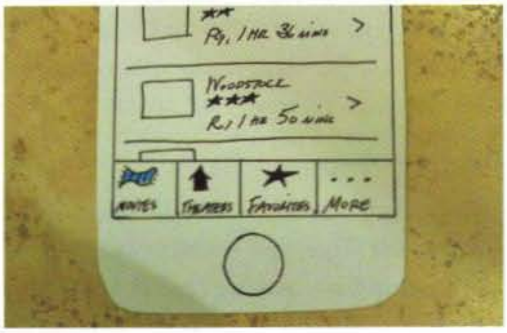

FIGURE 7.5 Paper prototype with a tab bar

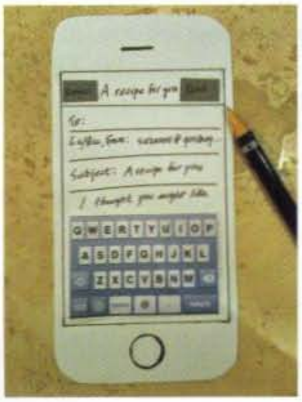

FIGURE 7.6 Sample sketch with a keyboard printout

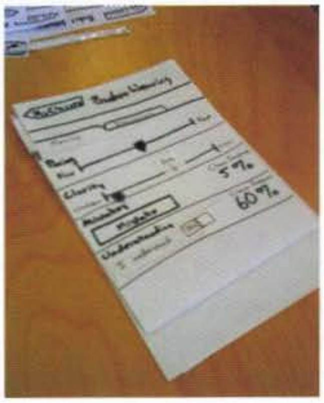

FIGURE 7.7 Example of a slider *(Courtesy of Angela Chiang, Andrew Hershberger, and Charles Naut)*

- **Text entry**

  For text entry, participants can write on Post-its or removable tape. Alternatively, they can use a pencil to write directly on the prototype. Be forewarned: Even with good erasing, if participants write too hard, your next user may see what the previous one wrote.

- **Pickers**

  Pickers provide essentially the same function as drop-down lists on web or desktop applications (FIGURE 7.8). Given that they can include a large number of items, you may need a long strip of paper to display all of the options. The strip can be folded and tucked away when the user is not interacting with the picker.

- **Highlight**

  Consider creating a highlight cutout that you can move up or down as the user makes selections (FIGURE 7.9). Another option is to buy transparent plastic sheets, which come in a variety of colors.

- **Alerts**

  Consider using a different background color for your alerts. Make sure they don't completely obscure content that should be visible underneath (FIGURE 7.10).

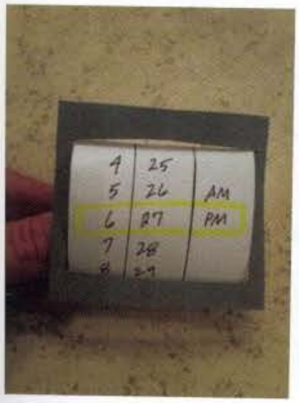

FIGURE 7.8 Example of a time picker

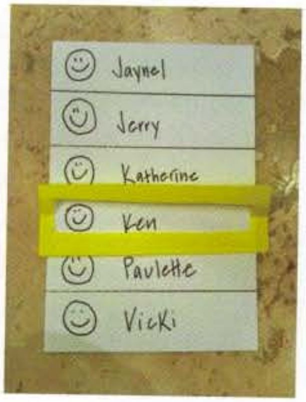

FIGURE 7.9 Example of a highlight

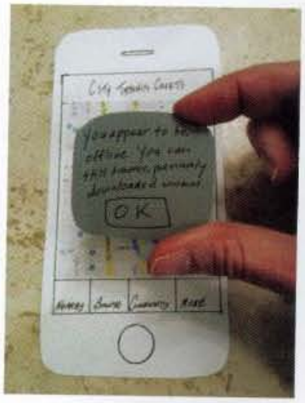

FIGURE 7.10 Example of an alert overlay

- **Segmented controls**

  Include different states of segmented controls, which are typically used for filters or sorts. Each state can show a different "segment" of the control highlighted. The segmented control in FIGURE 7.11 lets users sort the list by Popularity, Rating, and Title.

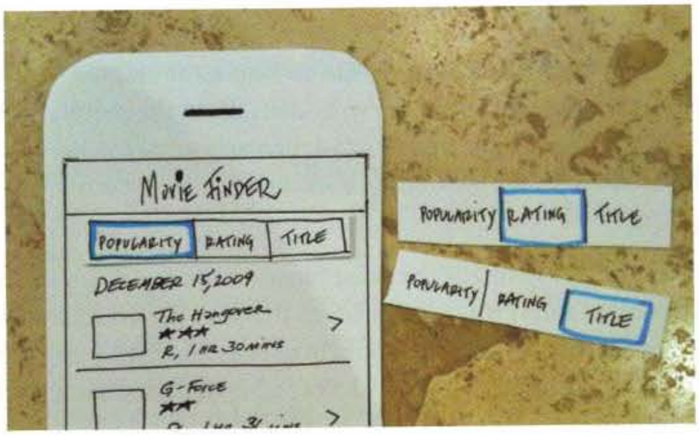

FIGURE 7.11 Segmented control example

NOTE

Apple's *iPhone Human Interface Guidelines* contain details on iPhone controls. In addition, some will be discussed in more detail in Chapter 9, "User Interface Design."

Here are some additional elements you may need to include:

- Loading page indicator
- On/off versions for switches that work like radio buttons
- Check mark for selected items

Word of Caution

As discussed earlier, paper prototypes can improve collaboration and reduce costs. These benefits can occur only if the approach supports rapid iterative design.

With that in mind, try not to go overboard with your paper prototypes. For example, cutting tiny bits of paper for each key on a keyboard probably isn't necessary. Users generally understand how the keyboard works, so you can say things like "Imagine that the keyboard would display the text you entered."

Similarly, if you are preparing a prototype for a study, don't worry if you haven't created every single interface element users may expect—the beauty of paper is that you can always sketch the widget and add it on the fly.

STATIC IMAGES ON THE DEVICE

Once you have refined your overall concept and flows, you may want to create screen captures of your designs and display them on the iPhone. If you link the images with "hot spots," you can offer a more exploratory user experience since no one needs to play the role of "computer" while switching out user interface elements. Moreover, the precise form factor may make it easier to refine visual design details such as type size, layout, and color.

HOW TO DO IT

To start, you'll want to create 320 × 480 images of your app screens. Many drawing programs have iPhone templates built into the software or available for download. Keep in mind that these templates do not have all of the user interface elements in the iPhone universe; developers often create controls that look and feel like standard controls, but they are actually custom-designed and custom-coded. If there's something you need that's not available, be prepared to sketch the desired solution. There are four drawing programs widely used by iPhone UI designers:

- OmniGraffle (see Graffletopia, www.graffletopia.com, for iPhone templates)
- Fireworks
- Photoshop
- Illustrator

NOTE

Screens can be hand-drawn, but it may be harder to fit them into the iPhone form factor.

I prefer OmniGraffle and Fireworks since the "page" framework translates well to iPhone screen design. Photoshop and Illustrator are excellent products, but the number of tools and options tends to overwhelm novice users. Three easy ways to view images on the iPhone are via the built-in slideshow, Safari, and LiveView.

Viewing via Built-in Slideshow

The downsides of the slideshow approach are the linearity—you can only swipe forward or back—and the presence of zoom and slideshow controls. On the plus side are speed and simplicity: Save your sketches in an acceptable iPhoto format, then add them to your iPhone photo collection. Be sure to include the status bar (battery and connection information) if it is part of your design.

Viewing via Safari

Another option is to create images with "hot spots" and display these images in Safari. With this approach, users can tap on rows and buttons just as they would with a real app. UNITiD design put together some scripts that enable you to disable zoom and view the images in full-screen mode. An easy-to-follow tutorial can be found on their web site.[9]

[9] Matthijs Collard, "Prototyping for the iPhone using Fireworks," http://unitid.nl/2009/04/prototyping-for-the-iphone-using-fireworks-cs3/ (April 2009).

LiveView

LiveView,[10] created at IDEO Labs, allows you to view desktop designs from your iPhone (FIGURE 7.12). This can help you evaluate the app layout, type sizes, and other visual elements, but it's not possible to interact with the prototype. Additionally, LiveView must be accessed over WiFi, so the geographical range of your testing may be limited.

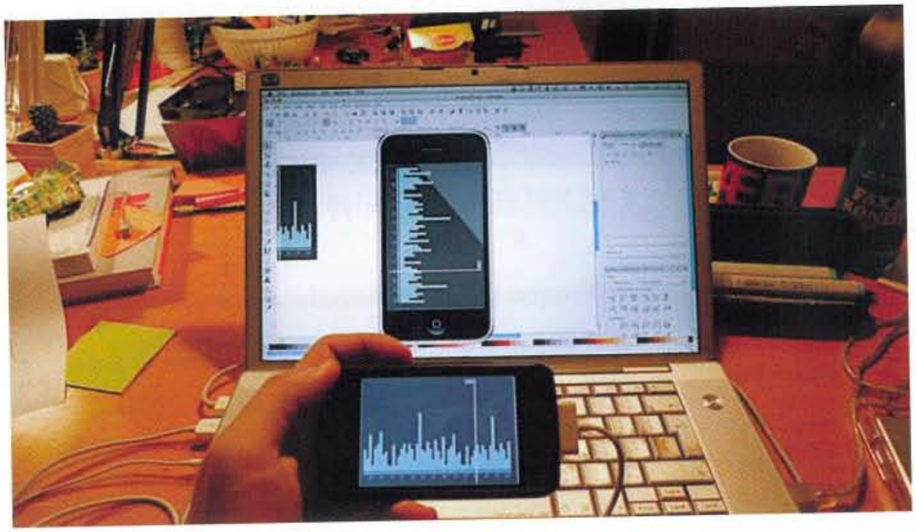

FIGURE 7.12 iPhone app viewed via LiveView, which was developed at IDEO *(Courtesy of Marcin Ignac)*

Word of Caution

Creating on-screen prototypes with static images isn't necessarily more effective than paper prototyping. Static images can't display alerts and control states, so you may need to create a hybrid prototype (e.g., on-screen plus paper overlays). Also, you can't simulate scrolling content with static images, but you can do so with paper.

[10] Gentry Underwood, "LiveView: An iPhone App for On-Screen Prototyping," http://labs.ideo.com/2009/01/20/liveview-an-iphone-app-for-on-screen-prototyping/ (January 2009).

INTERACTIVE ON THE DEVICE

Given the limitations of static image prototypes, you may prefer more interactive prototyping techniques. Before choosing this route, make sure you've evaluated all of the lower-fidelity options. If some aspects of your app, such as flows and layout, can be worked out on paper, start with paper before committing to an interactive prototype. I'm deliberately using the word *committing* because higher-fidelity prototypes have a tendency to become final designs.

NOTE

Flash is a powerful tool for creating interactive prototypes but you can't view Flash files on the iPhone.

Issues to Explore

You can explore almost any aspect of the user experience; it basically depends on how much time you want to put into the prototype. In contrast to static image prototypes, you can provide forms, transitions, and scrolling content. More important, given the portability of these prototypes, you can get out in the field and walk through your scenarios in context. Although this can be done with paper, the process is much easier with an interactive prototype on the device.

Challenges

Although interactive prototypes are powerful, there are still some aspects that differentiate them from the "real" experience. In particular, you will still likely need to fake the current location information, live data feeds, and the handling of interruptions (what happens when the connection is lost or disrupted?).

HOW TO DO IT

Interactive prototypes can be created with tools like Keynote or PowerPoint[11] (FIGURE 7.13), but specialized readers are required to display these on the iPhone.[12] On the other end of the spectrum are custom CSS solutions that are essentially web applications made to look like native iPhone applications. These prototypes can take a significant amount of time, though there are some tools like ProtoShare (FIGURE 7.14) that aim to simplify the process.[13] Another solution that holds promise is Briefs, developed by Rob Rhyne.[14] Briefs prototypes run on the iPhone, like actual apps, but take much less time to code and produce. Each "brief" contains a text file that references a series of static images organized into "scenes."

[11] "How to Mockup Your iPhone App with MockApp," http://mockapp.com/2009/10/12/new-video-demo-how-to-mockup-your-iphone-app-with-mockapp/.

[12] "Viewing Your MockApp Mockup on Your iPhone," http://mockapp.com/?s=goodreader.

[13] "Native iPhone Prototypes with Protoshare," http://blog.protoshare.com/2009/06/17/native-iphone-prototypes-with-protoshare/.

[14] Briefs, http://giveabrief.com/.

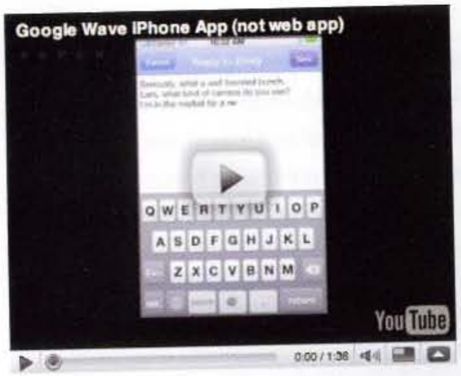

FIGURE 7.13 MockApp example of Google Wave *(Courtesy of MockApp)*[15]

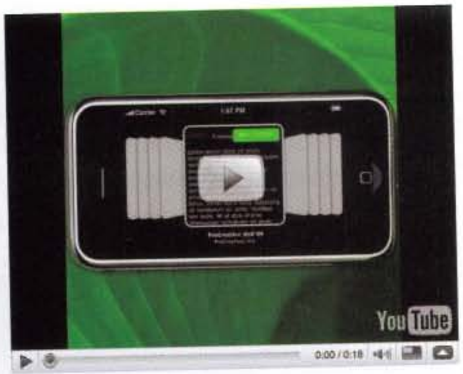

FIGURE 7.14 ProtoShare example showing cover flow *(Courtesy of ProtoShare)*

Word of Caution

As mentioned in this section, it is possible to create prototypes that mimic nearly all of your app's features and functionality. This isn't a bad thing as long as the work can be accomplished within a reasonable time frame.

If you're spending an inordinate amount of time prototyping, perhaps you've gone beyond the rapid iterative testing "tipping point." In this case, reconsider what aspects of the prototype can be faked or explained to your audience—the effort required for some user experience elements may outweigh the benefits.

VIDEO PROTOTYPES

Video prototypes are a powerful way to show app usage in context—the actors, the environment, concurrent activities, the passage of time. These contextual elements are particularly important for apps that interact with the real world: location-aware apps, remote controls, cooking aids. Additionally, immersive apps such as musical instruments or games may use video to show their apps in action. Although video prototypes can be used to elicit feedback via usability testing, they are typically created to evaluate and communicate design ideas. FIGURE 7.15 shows a screen capture from a video prototype created for a caregiver app; the full video is available online.[16]

[15] "Google Wave iPhone App Demo," http://mockapp.com/2009/10/19/google-wave-iphone-app-demo/.

[16] Eldia video, http://vimeo.com/2420799.

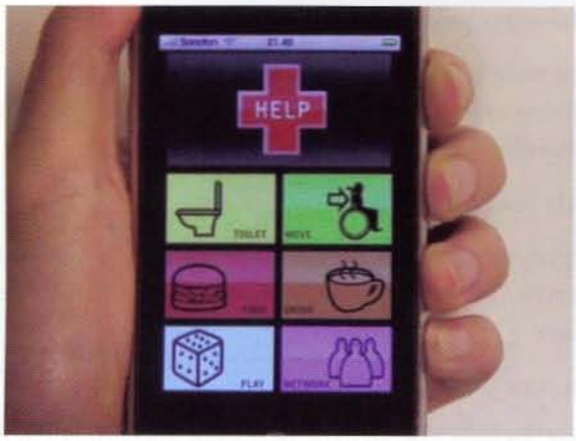

FIGURE 7.15 Prototype of Eldia app for caregivers
(Courtesy of Ujjval Panchal, Marcin Ignac, and Yu-Min Chen)

Issues to Explore

Video prototypes are an effective way to explore and document how an app works in the real world. Examples of interactions that may be captured include

- Handling of the device while performing other activities
- Other people who impact the experience
- Interruptions that may influence the user

Harder to Explore

Some issues are less suitable for a video prototype. For example, if you want to explore low-level interactions in great detail, consider creating an interactive prototype that users can walk through at their own pace.

HOW TO DO IT

Video prototypes may seem like a significant undertaking, but the process can be relatively simple:

Step 1. Develop your script.

First, you'll want to develop a script for your video. If you created scenarios as discussed in Chapter 4, it shouldn't take long to write the script. Focus on the scenarios that show how the app is used in a contextual and realistic way. Also, include an introduction to set the stage, for example, who the characters are, where are they located, what their goals are.

Step 2. Sketch storyboards.

If you have already created storyboards for your app, you'll want to adapt them based on your script. You may discover that additional screens are needed to provide a seamless user experience.

Step 3. Create your prototype.

Your prototype can be paper, electronic, or whichever medium you find most effective. Be sure your screens, overlays, and so on are ready before you start shooting video.

Step 4. Film your prototype.

You don't need to purchase an expensive high-end camera to film your prototype. At a minimum, be sure your camera has solid zoom capabilities and a good microphone. If the microphone picks up too much background noise, you may want to rent or purchase a lavalier microphone that can be clipped onto the actor's shirt. Here are some additional video tips:

- Choose realistic locations; seek permission to shoot video as needed.
- Ask the main actor to "think aloud" so viewers know the "whys" behind his or her actions.
- Use a combination of environmental views and close-up app screen views.

Step 5. Edit your video.

There are dozens of video-editing tools on the market. For the purposes of your video prototype, something basic such as iMovie should be sufficient. Some additional editing tips are the following:

- If background information is necessary, remember to start with a voice-over to set the stage.
- Try to keep the final video under five minutes.
- Use fade in/out to indicate the passage of time.

Other Types of Video Prototypes

As mentioned in the previous section, video prototypes can also be an effective way to illustrate how immersive apps like games and musical instruments work. Gogogic, an online and iPhone game developer, uses animatics in its app development process. Animatics,[17] a series of still images displayed in sequence, enable Gogogic to visualize the player experience before diving into coding. Additionally, animatics help prioritize the app requirements. According to Gogogic's CEO, Jónas Antonsson, "At Gogogic, the animatic is king" (FIGURE 7.16).

[17] "Animatics," http://en.wikipedia.org/wiki/Animatic#Animatics.

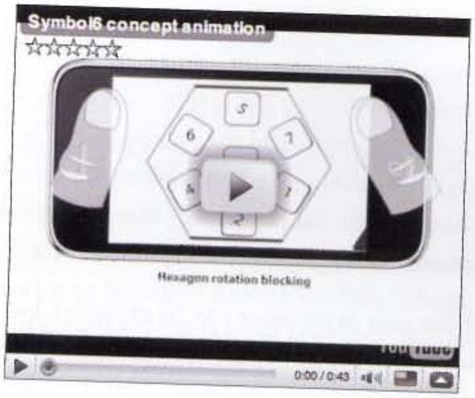

FIGURE 7.16 Animatics Gogogic developed for the game Symbol6 *(Courtesy of Gogogic)*[18]

Word of Caution

As mentioned in this section, prototyping should be a relatively rapid process. Sure, you could spend weeks shooting and editing a production-quality video. However, for prototyping purposes, you'll want to "time-box" the effort put into the video.

If the video capture and editing take more than a week, consider an alternative strategy.

Another approach is to animate your scenario with cartoon-like characters as is done with the Xsights iPhone app (**FIGURE 7.17**). Services like GoAnimate provide tools and templates to help newbies create basic animations.

FIGURE 7.17 Animation developed for Xsights app *(Courtesy of Xsights)*[19]

[18] Gogogic, "Symbol6: How We Created an iPhone Game," http://gogogic.wordpress.com/2009/02/09/symbol6-how-we-created-an-iphone-game/.

[19] Xsights video, www.xsights.com/index.php.

THE IPHONE SDK

Prototypes can be developed using the iPhone SDK within Interface Builder. While some design professionals may argue that "working code" is not prototyping, it really depends on the domain and the prototype complexity. For example, paper prototyping would be inadequate for exploring a musical instrument app. In this case, digging into Apple's Audio Library may be the most efficient way to experiment and figure out what's possible.

NOTE

Marcos Pianelli of Digital-Gourmet created a great app prototype with the iPhone SDK. You can view it on Vimeo: http://vimeo.com/5947546.

Almost any aspect of the user experience can be explored, but be careful how far you develop your prototype—if you become too invested in the design, you may be less likely to adapt it based on user feedback. Also, if your prototype isn't fully functioning, it will still be challenging to evaluate features such as location awareness and live data feeds.

HOW TO DO IT

Programming the iPhone user experience is too broad a topic to cover in this book. There are countless books and web sites on the topic. Some titles in the Addison-Wesley family that have been well received include

- *Cocoa Programming for Mac OS X, Third Edition,* by Aaron Hillegass (2008)
- *The iPhone Developer's Cookbook: Building Applications with the iPhone 3.0 SDK, Second Edition,* by Erica Sadun (2009)

Word of Caution

Once you dive into the iPhone SDK, it's typically harder to explore divergent design solutions. Prototyping within the SDK can be time-consuming, so your team may not want to "throw away" hours and hours of development work. As a result, they may be inclined to stick with the first approach and make smaller iterations. Moreover, keep in mind that the UI controls in the SDK may not be as comprehensive as you expect; many seemingly common controls are actually custom-coded.

If your coding skills are limited, you may choose a direction because it's easy, not because it's the best solution. Eventually you may have to make compromises, but this shouldn't happen on day one. In short, use the iPhone SDK for prototyping only if you've explored other options and determined that it is the most appropriate course.

Summary

Prototyping your iPhone app before coding will enable you to explore, evaluate, and communicate your design ideas. Through prototyping, you may uncover ways to improve the app concept, flows, terminology, and low-level interactions such as transitions.

This chapter discussed a variety of prototyping approaches, including paper, electronic, video, and even using Interface Builder and the iPhone SDK. The approach you choose will depend largely on the type of app, your design goals, and the project stage. Whatever you decide, keep these tips in mind:

- Try to explore divergent directions with your prototypes. It will be harder to change course once your app designs are coded.
- You don't have to prototype the *entire* user experience. Prototype only the areas that will help you explore the design issues under investigation.
- Prototypes don't have to be limited to the early design stages—use them to refine app issues that arise in the later stages. ■