CHAPTER 5

WIREFRAMES

"How do I pick the right design?"

ALL TOO OFTEN designers and developers jump headfirst into a project. Designers focus on making things look as good as possible, and developers put their focus solely on the technology platform. And both groups ignore the deeper aspects of functionality. The result is a shallow, "happy-path" design—everything looks great on the surface, but when it's time to produce, things don't fit. Text is longer than expected. Images aren't the resolution you planned for, and not all of the functionality was taken into consideration. Next thing you know, you have a sloppy, inconsistent UI, your client is unhappy, and you're out of time and money to fix it.

The good news is that all of this can be avoided with a little communication and organization.

Software is inherently complex, and picking the *right* design for your application requires some discipline. As when testing code, a UI also must be "tested" before you commit to it. Specifically, wireframes are low-fidelity schematics of the application used as a guide throughout the design process. They are another design-thinking aide that will help you think through the details of a design and uncover complexity before it's too late. More importantly, wireframes will help you ensure good communication among your teammates and clients.

In this chapter, you look at some wireframe tips that can help you do the following:

- Create feasible application designs.
- Identify problem areas before it's too late.
- Get early stakeholder buy-in and feedback.
- Get designers and developers on the same page.
- Convey advanced interaction concepts.

Debunking Wireframes

Before going further, I want to briefly debunk any biased opinions about wireframes. A lot of people debate the "right" way to wireframe. And after reading many articles and watching sessions on the topic, I can't say that I fully agree or disagree with any of the opinions out there. Most of the debate is about fidelity and roles: How much detail should wireframes show? Should they be heavily annotated? Should they show interactivity? Should developers, designers, or IAs create them? Should wireframes exist at all?

I don't have a definitive answer for any of these questions. And, honestly, I would be skeptical of anyone who did. The fidelity of the wireframes will vary greatly depending on the specific project, client, and team you're working with. I've created many sets of wireframes in the past,

everything ranging from a couple of boxes and arrows to frame-by-frame illustrations detailing exactly how the UI will work. My wires have been described as both over and underwhelming. Thus, I've experimented with varying levels of detail by dialing-up and dialing-down the detail, and I still can't say there's one be-all approach to creating effective wireframes.

So where did I end up? Let me just say that the fidelity of wireframes should be proportional to your ability to communicate the idea. In other words, if you're creating a complex app with atypical features, you'll likely have a hard time communicating your ideas. So higher fidelity wires will be necessary. And if you're creating something basic, say a simple web form, a few boxes and arrows will suffice. One caveat: If you're working with a big team or a remote team, you'll want to lean toward producing slightly more formal wires to avoid misinterpretation.

The fidelity of wireframes should be proportional to the complexity of the idea being communicated. KEY POINT

To close the loop on roles, sometimes the designer will produce the wireframes; other times, the developer or project manager will do the work. If you happen to be one of the elusive designer-developer hybrids, you'll probably do the work. I believe that the person with the clearest vision of the final product should be the one producing the wireframes. That said, the task of creating wireframes should also be shared. Everyone on the team will bring a different, valuable perspective. Also, it's much easier to change a wireframe at this point in the process than down the road when the design is fully baked.

Whoever has the clearest vision of what the product should ultimately be is the person who should be producing the wireframe. KEY POINT

Wireframes 101

At this point, you've pegged personas and audience, developed a handful of solid insights, and mapped the content. Wireframes are the next step, and there's nothing better than knowing you have all your ducks in a row before getting started. So next, take a look at your loose sketches, content, and ideas and package them up into a deliverable to get your client and teammates on board.

When Should You Create the Wireframes?

Producing wireframes usually comes after you've sketched out some possibilities for application flow (see Chapter 3) and gone through the due-diligence information architecture (see Chapter 4). Before you create wireframes, you want to have a good approach in mind for

high-level navigation and layout. Also, it's always helpful to have all of the content identified so you can plug it into the various screens. If you've checked those items off your design list, it's time to produce some wireframes.

Anatomy of a Wireframe

Some people insist on limiting wireframes to nothing more than boxes containing short descriptor labels. Others go so far as to make a grayscale version of the final product. If you work closely with a small team, sticking with a few boxes and arrows can be more efficient. But if you have a client who doesn't speak the box-and-arrow language, you may need a little more detail. Regardless of personal style, the goal remains the same: Communicate your idea clearly. To me, an effective wireframe contains the following:

- Layout
- Hierarchy
- Interaction
- Content
- Functionality

Layout

At this point, layout doesn't have to be perfect, but you do want to establish the general placement of screen elements. If you're unsure about where to start, try picking from one of the basic layouts mentioned in Chapter 4.

Hierarchy

Many applications have different levels of content: primary, secondary and ancillary . . . Arranging and sizing the content appropriately helps to create a clear, consumable experience. UIs that have good hierarchy are easily "decoded" and users quickly make decisions about what to click and where to go.

Interaction

Any gestures—for example, drag-and-drop, tap-and-hold, or hand wave actions—need to be depicted in the wireframe. If the software is interaction-heavy, creating a separate interaction-specific document can be helpful.

Content

Placeholder and *lorem ipsum* text doesn't cut it anymore. Clients are distracted by fake copy and miss the point of the design. Spend the extra three minutes to place realistic titles and descriptions in your wireframes. It's okay to use placeholders for images and other graphical elements.

Functionality

Controls such as carousels, image rotators, or context menus are dynamic by nature; show their intended functionality with state diagrams or additional callouts (see Figure 5-1).

FIGURE 5-1
Effective wireframes will show many aspects of the design vision, including interaction, layout, and custom functionality.

Are You Speaking Wireframe?

The practice of creating wireframes has become more common in the software world. Wireframes have been identified as a key communication tool among UX practitioners and also because there have been some attempts to standardize the common elements that make up wireframes. Although the aesthetic quality may vary, wireframes usually contain the following, essential elements:

- Views
- Connectors
- Conditional elements
- Annotations

Many frameworks, languages, and tools aim to standardize the creation of wireframes. For example, the Unified Modeling Language (UML) is a general-purpose modeling language usually taught in the field of software engineering. It's a full-featured modeling language for describing complex software systems. Admittedly, UML is overkill for most wireframes

scenarios, but the concepts are widespread within the developer community. Another common framework is Jesse James Garrett's Visual Vocabulary (available at `www.jjg.net/ia/visvocab`), which is a set of standard symbols used to describe high-level app structure, flows, and interactions. Depending on the level of accuracy and detail needed, you may never go beyond the basics. Do what makes sense for your project.

Views

These are the fundamental elements found in wireframes. They can represent an entire window, or they can represent a piece of content in a larger view. Pretty simple (see Figure 5-2).

FIGURE 5-2
Even with simple rectangles a page layout starts to take shape.

Connectors

Connectors are used to describe relationship and directionality among views. Placing views left, right, above, or below each other and connecting them with a line and arrowhead indicates their location within the app flow (see Figure 5-3). Double-edged arrows indicate that the relationship among views is bidirectional. And you can use a line with a single arrowhead to indicate UI elements that have specific functionality.

Conditional Elements

When presenting complex scenarios, there will inevitably be points in the UI that require users to make a decision. When the user is prompted with a choice that results in navigating to one of many pages, a diamond is used to indicate the path options (see Figure 5-4).

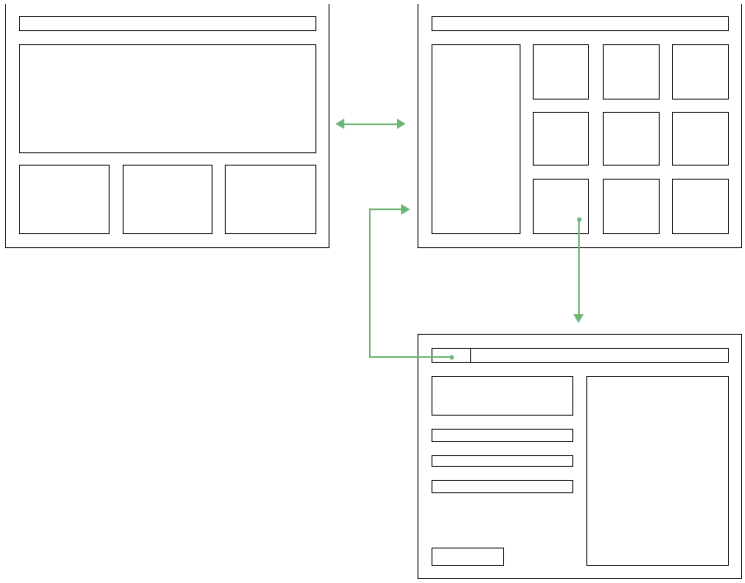


FIGURE 5-3
Basic arrows and lines go a long way in communicating the how screens connect to each other.

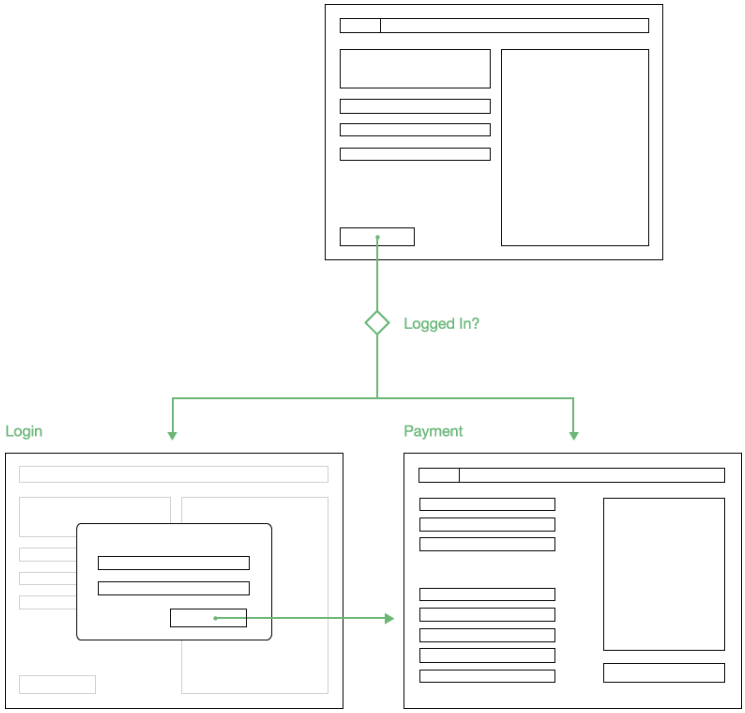


FIGURE 5-4
Conditional paths in an application flow are often distinguished with a diamond shape.

Annotations

Annotations are short concise notes that describe additional information about an element that is *not immediately apparent*. They are usually shown with a small callout containing a number (see Figure 5-5). Notes associated with the callout are listed along the side of the page. For some reason, there seems to be a trend to completely cover wireframes with annotations. Don't annotate everything — only the things that deserve additional explanation. For example, annotating the footer with a note such as "Footer containing global links" is unnecessary and will just waste your client's time. Keep it short and sweet.

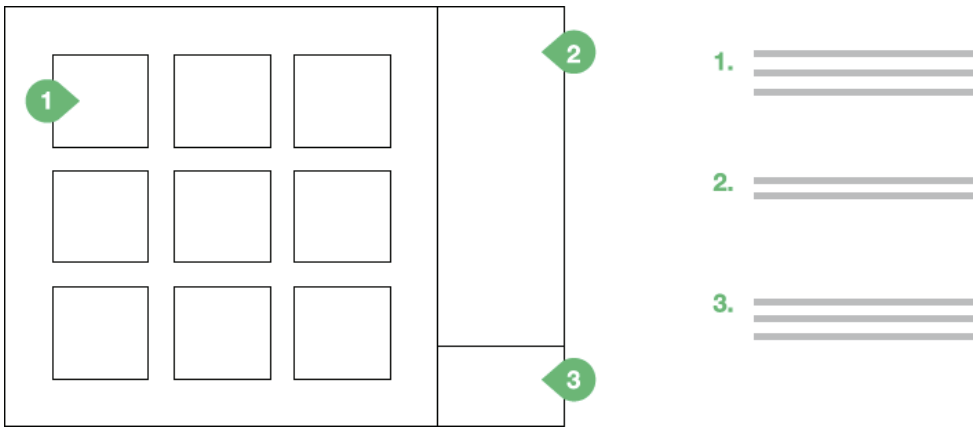


FIGURE 5-5
Annotations are usually shown with a callout containing a number.

Do's & Don'ts

If you're using wireframes as a client deliverable, sweat the details. Remember, it's your job to convince your client that time spent on design-thinking activities is valuable. Paying attention to simple things like spacing and alignment will go a long way in the overall quality of your deliverable. This section provides a short list of do's and dont's to keep in mind when creating your wires.

Do Pay Attention to Spacing

Whether you're showing a series of views or a list of items, spacing is important (see Figure 5-6). This is the *number one* thing that gets overlooked, and it makes such a big difference. Vertical and horizontal spacing should be as consistent as possible. Pick a base size for spacing and use a multiple of that size for things that need more or less space. For example, 20 pixels is a good base size; 5 pixels and 100 pixels are multiples of 20 that play nicely and that will create a nice visual rhythm.

NOTE

You find more on visual rhythm in Chapter 8.

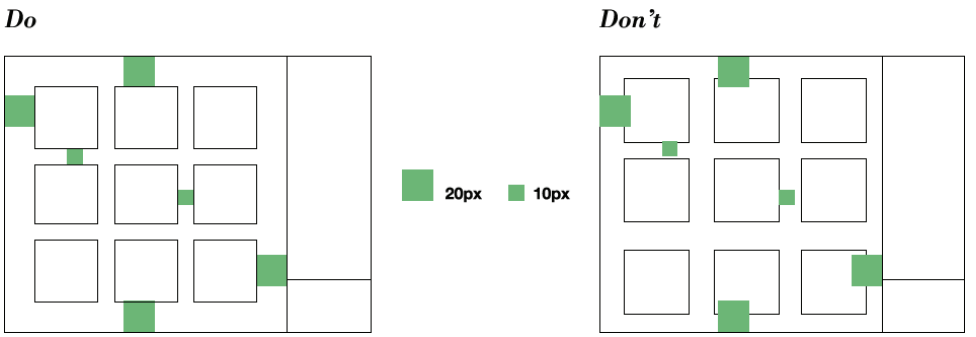


FIGURE 5-6
Being consistent with spacing in wireframes helps avoid miscommunication and gives your deliverables more professional look.

Don't Overlook Alignment

Alignment and spacing go hand in hand when creating professional-looking documents. Align both the views and the content on the page (see Figure 5-7). And don't forget to align your labels. If you decide to align a label at the upper-left, make sure you do so for all the labels.

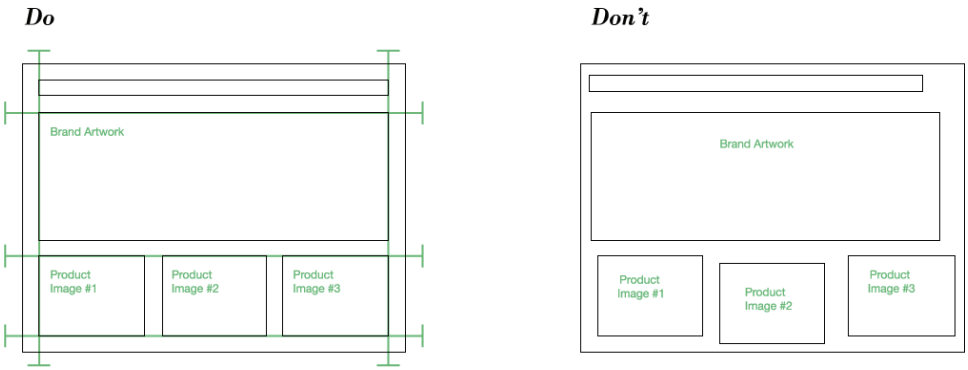


FIGURE 5-7
Ensure proper alignment and spacing when using wireframes as a deliverable.

Do Make Good Connections

Showing connections between screens or screen elements is an integral part of most wireframes (see Figure 5-8). Make sure that your connections are accurate. Use the right arrowheads to show directionality among screens. When connecting screen elements among pages, make sure they begin and terminate in the right location. Inaccurate or sloppy connections make wireframes hard to read and potentially misleading.

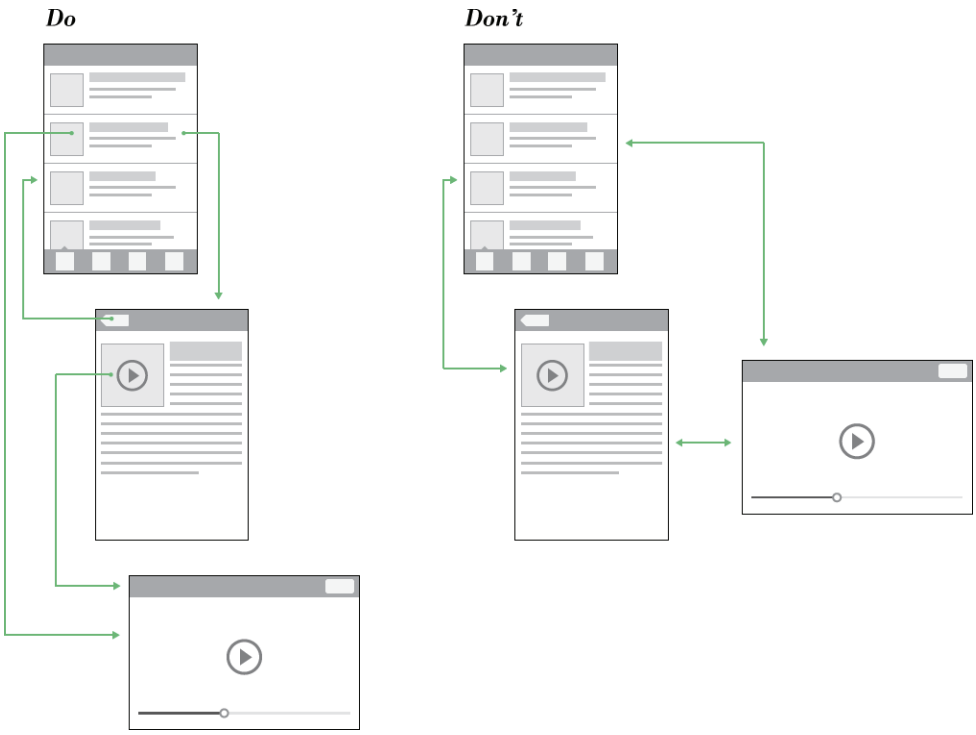


FIGURE 5-8
Show accurate connections among elements in your wireframes to avoid miscommunication.

Do Design Thoroughly

This is a biggie. In fact, if you *remember only one thing from this entire chapter, this is it*. Good wireframes anticipate how varying content will fit into the design (see Figure 5-9). For example, when creating text labels or containers, consider how longer and shorter text will look. Use real text, not placeholder or *lorem ipsum* text. Think about the use of imagery. Will all items in a list be associated with images? What is the image resolution? Will the images be stretched or pixelated? Did you think about the on/off and up/down states? What about error messages and notifications? Chances are you'll need more than one wireframe template or layout to accommodate various content scenarios. Think about it. Trust me. It will be worth it.

Don't Forget About the Edge Cases

This is another biggie. Identifying and thinking through edge cases is what separates the professionals from the amateurs. Good wireframes will show not only the common app scenarios but also the atypical and one-off scenarios (see Figure 5-10). For example, if your application has a subscription model, what will the UI look like when the user isn't logged in?

What if the app crashes? Will it resume or auto save the user's work? What happens when the app is offline? In some first-time-use scenarios, the UI will be empty; do you have good defaults and instructional cues? As you go through the wireframe process, you'll undoubtedly uncover many edge-case scenarios. This is a good thing. It's a sign that you're putting your design to the test.

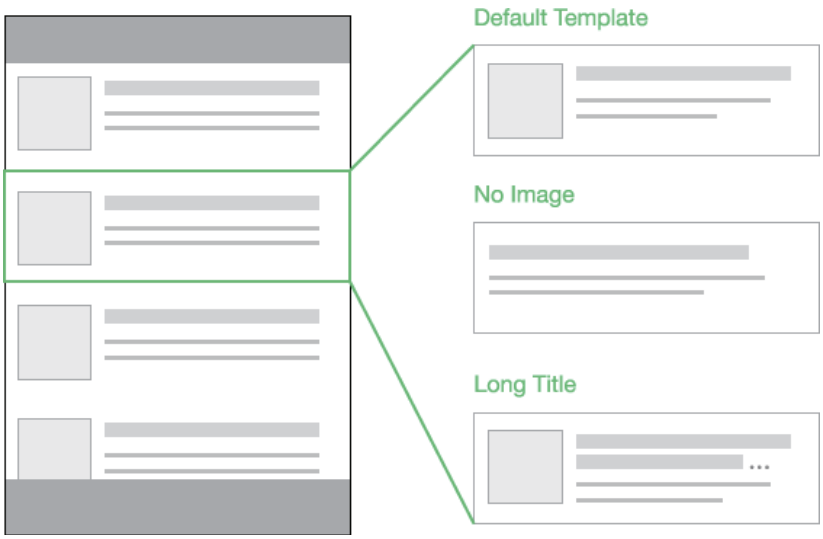


FIGURE 5-9
Effective wireframes will anticipate many content scenarios and edge cases, not just the happy path.

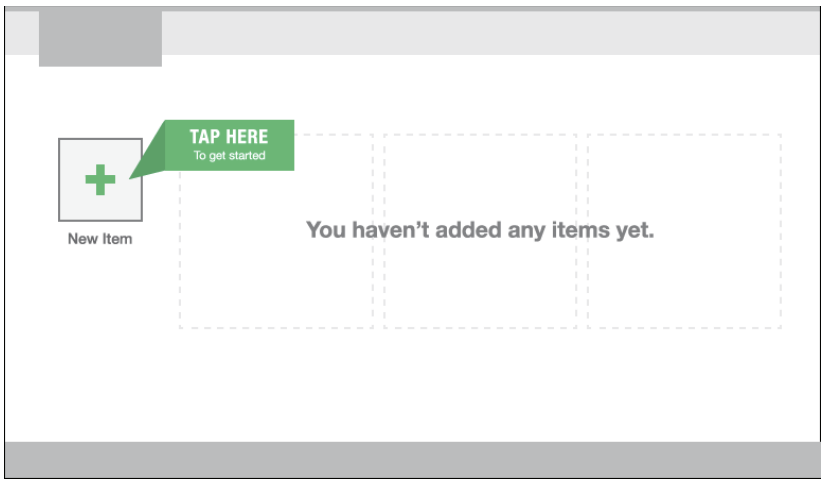


FIGURE 5-10
Be sure to design for edge cases and atypical scenarios such as blank slate and connectivity loss.

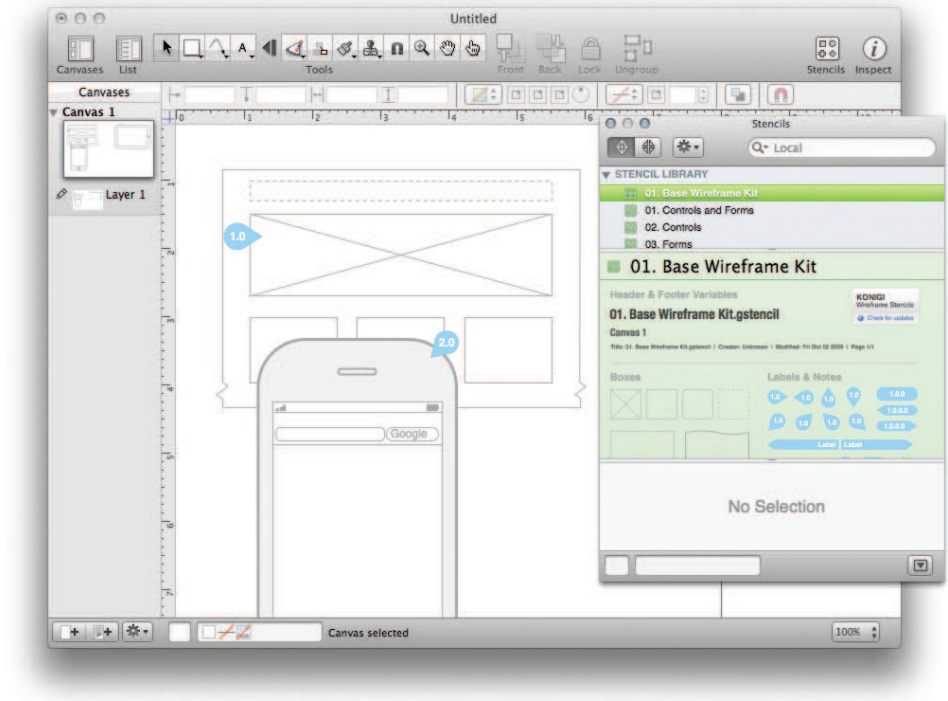
Tools for Awesome Wireframes

There are many tools of the trade when it comes to wireframes. Depending on the complexity of your project and your own artistic ability, it may be best to experiment until you find the right fit. Some tools are designed specifically for modeling UIs, whereas others are broad creative tools. Regardless of which tools you use, the goal is to produce professional-looking deliverables. Following is a list of tools that will do the trick.

OmniGraffle

This is a very popular diagram and wireframe tool. OmniGraffle touts an intuitive UI (see Figure 5-11) and supports just about every wireframe case I can think of—from simple one-page wireframes to complex multipage documents. OmniGraffle also has a rich community of user-submitted custom stencils for specific scenarios. Stencils for Android, iPhone, UI, and architecture are free to download (www.graffletopia.com/). The only drawback is that this tool is currently available only for the Mac.

FIGURE 5-11
OmniGraffle is the top choice for many wireframe tasks. *(Copyright The Omni Group)*



Adobe Illustrator and Fireworks

Adobe Illustrator and Fireworks are vector-based tools that have a ton of flexibility for creating a variety of wireframes (see Figure 5-12). Illustrator is my wireframe tool of choice

because of its custom illustration capabilities and layout support. There are also many third-party templates available for download to accelerate the creation process (a quick web search for **Adobe Illustrator UI templates** will send you in the direction). Although Illustrator and Fireworks give you the most control over your wires, they come with a hefty price tag and learning curve.

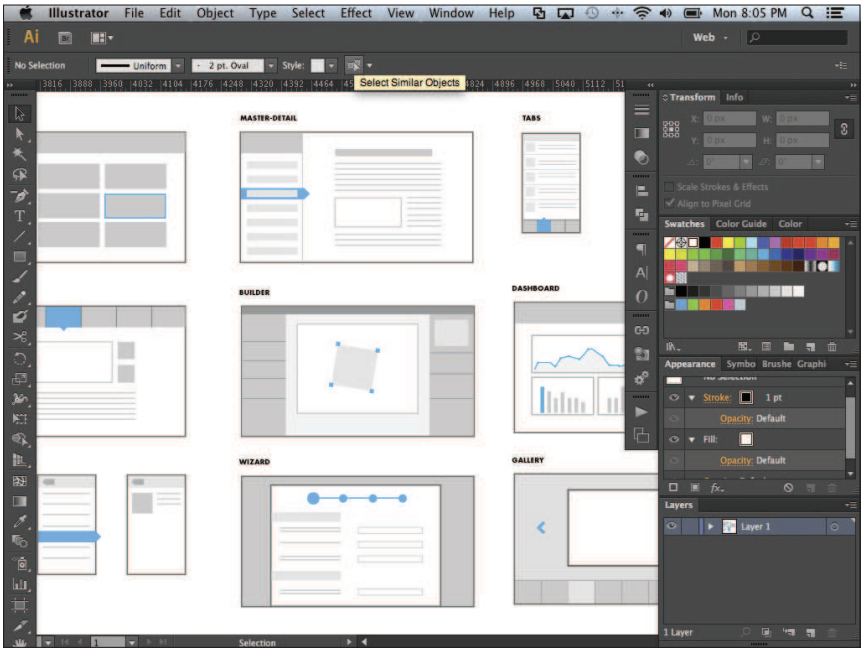


FIGURE 5-12 Adobe Illustrator is my wireframe tool of choice although it has a high learning curve.

Balsamiq Mockups

Balsamiq (see Figure 5-13) is designed specifically for UI wireframes and is available for Mac and PC and as a web-based tool (available at www.balsamiq.com/). Balsamiq comes with many UI-specific components and a bunch of community-generated components that make generating wireframes very fast. This tool has a low barrier to entry and works well for rapidly producing wireframes. I recommend this tool to anyone looking to get started with wireframes.

Microsoft Visio

Designers have ridiculed Visio for many years, mainly because of its battleship-gray components and overall bland look and feel. However, with its latest release (see Figure 5-14), Visio has received a major upgrade in the wireframing department. What's more, Visio now includes a ton of UI-specific components with its Wireframe Diagram template. I've seen some very well made wireframes generated by Visio over the past couple of years, and chances are you may already have Visio installed. If so, I recommend giving it a try.

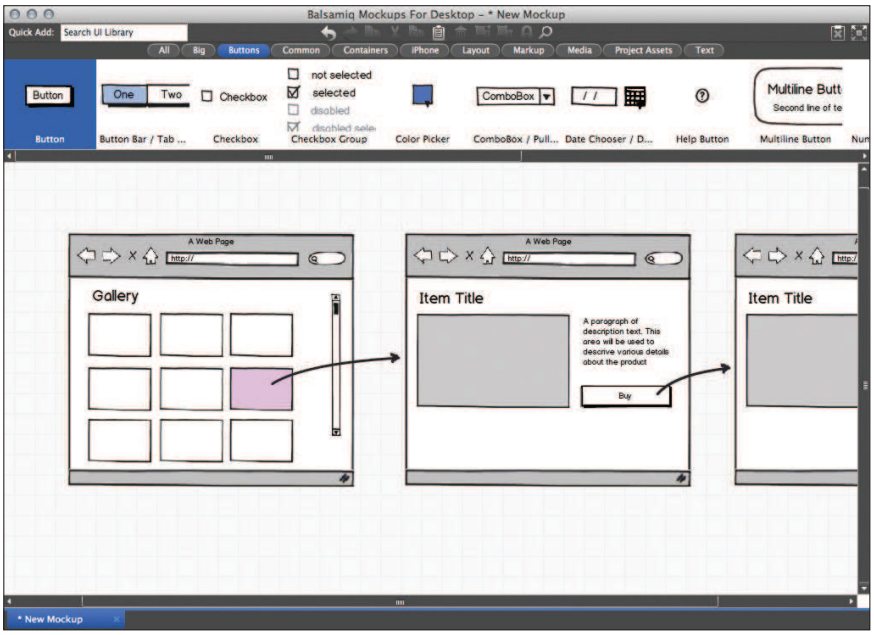


FIGURE 5-13 Balsamiq is a good wireframe option for both the entry-level designer and the seasoned practitioner. *(Copyright Balsamiq Studios, LLC)*

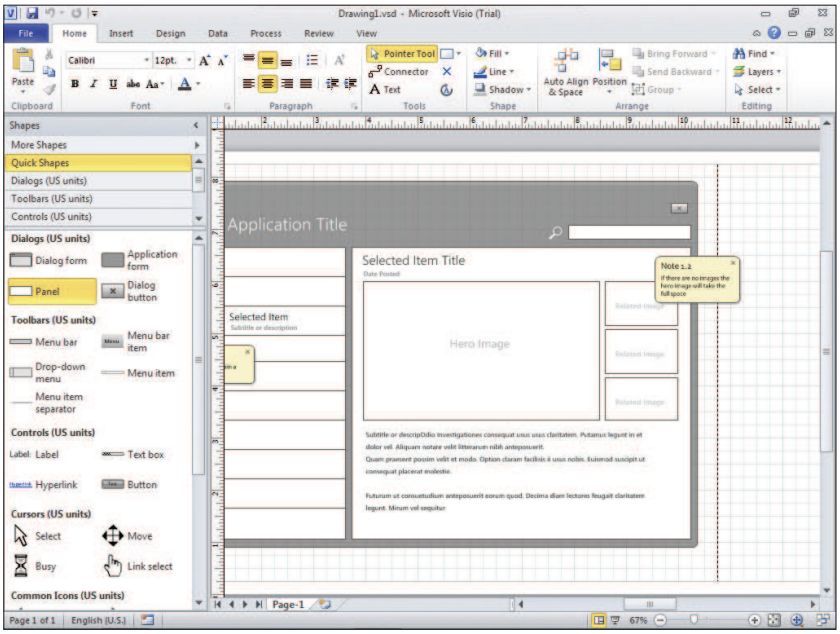


FIGURE 5-14 Visio has good support for creating wireframes and many people have the program installed as a part of the Microsoft Office suite.

Tools for Awesome-er Wireframes

What's that? Static wireframes aren't good enough for you? You want to dazzle your clients and impress your friends? Look no further. The following tools are great for expressing interactivity and complex scenarios that otherwise would be impossible to communicate with static wireframes.

Keynote and PowerPoint

Apple's Keynote and Microsoft's PowerPoint (see Figure 5-15) are generally overlooked as wireframe tools, which is unfortunate because they have great animation support. Showing click-through functionality or step-by-step progressions with these tools is easy and powerful. These tools can help your audience connect the dots for complicated UI scenarios.

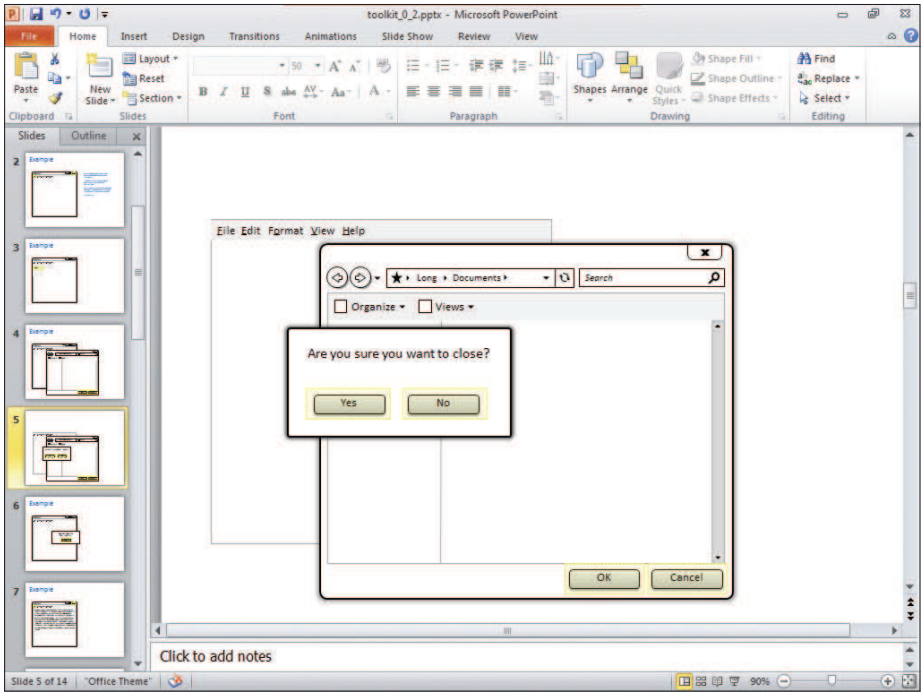


FIGURE 5-15 Presentation programs such as Keynote and PowerPoint provide basic animation support for communicating interactivity.

Adobe Edge Tools

Adobe Edge Tools (see Figure 5-16) is a suite of tools for creating interactive content. The tools are targeted for web-based content, but easily adopted for wireframe purposes. On the lower bound, these tools have great support for animation and interactivity. On the upper bound, there is a development IDE that allows you to add dynamic data and JavaScript for creating more advanced results. This tool requires a significant time commitment compared to some of the others I've mentioned, but the results are great.

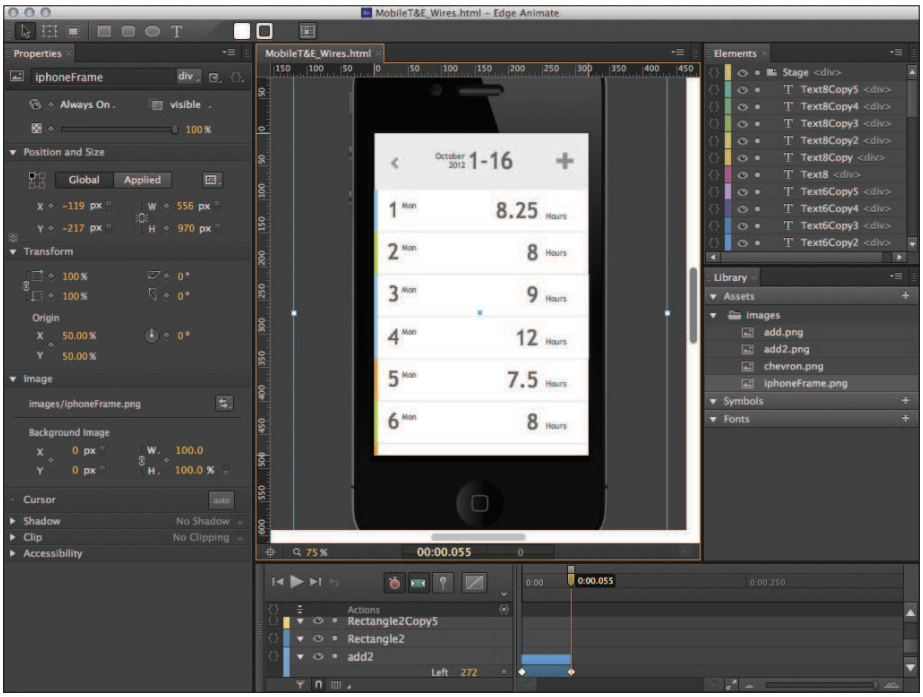


FIGURE 5-16 Adobe Edge Tools are web-focused tools but work great for showing interactivity in wireframes.

Microsoft SketchFlow

SketchFlow (see Figure 5-17) is one of my favorite wireframing and prototyping tools. This tool focuses on rapid creation and experimentation of application ideas. I use this tool to help me quickly iterate and evolve concepts with very granular control. It supports mouse interaction, animation, sound, and touch and has a built-in feedback mechanism. This tool is probably a bit techie for the average UXer, but that's the whole point: It produces atypical results that are very compelling.

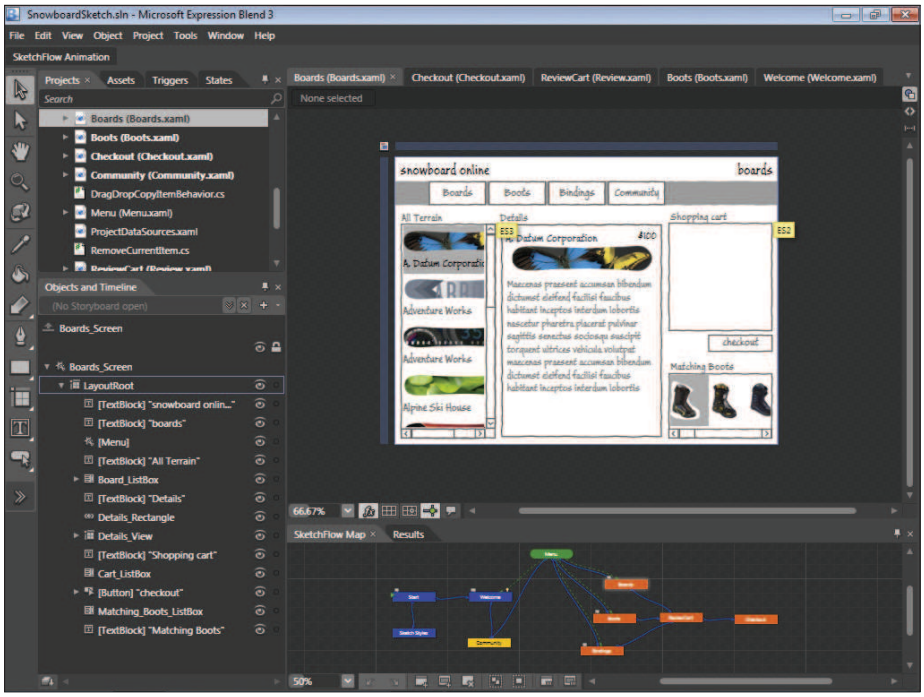


FIGURE 5-17 Microsoft SketchFlow is a great wireframe tool that has support in Visual Studio and the Microsoft Expression suite.

Wireframe Techniques

This section features a list of wireframe techniques that I use frequently. The first two are basic everyday techniques that can be used for virtually any type of digital design: websites, mobile apps, large screen apps, touch apps, and so on. These approaches are flexible. You can dial them up and down according to your specific situation. The remaining techniques are handy nuggets that are useful when you're looking to communicate complex ideas with minimal imagery.

Wireframe Technique #1: Creating the Basic Wireframe

This is the prototypical wireframe (see Figure 5-18). This technique is best used during the onset of a project because it allows you to put design elements in place quickly without getting hung up on small details. I typically use this type of wireframe as a base; then as I evolve the design, I add more layers and details. Start big with the major elements: layout, navigation, header, footer. Then start working your way through the details of each UI component. Tools like Balsamiq Mockups and OmniGraffle have a bunch of built-in templates to make this work easy and fast.

When to Use

If you have an aggressive timeline or small budget, this is the wireframe for you. The learning curve is low and thus these wireframes are very quick to produce. Because of their simplicity, you can easily modify them when integrating feedback.

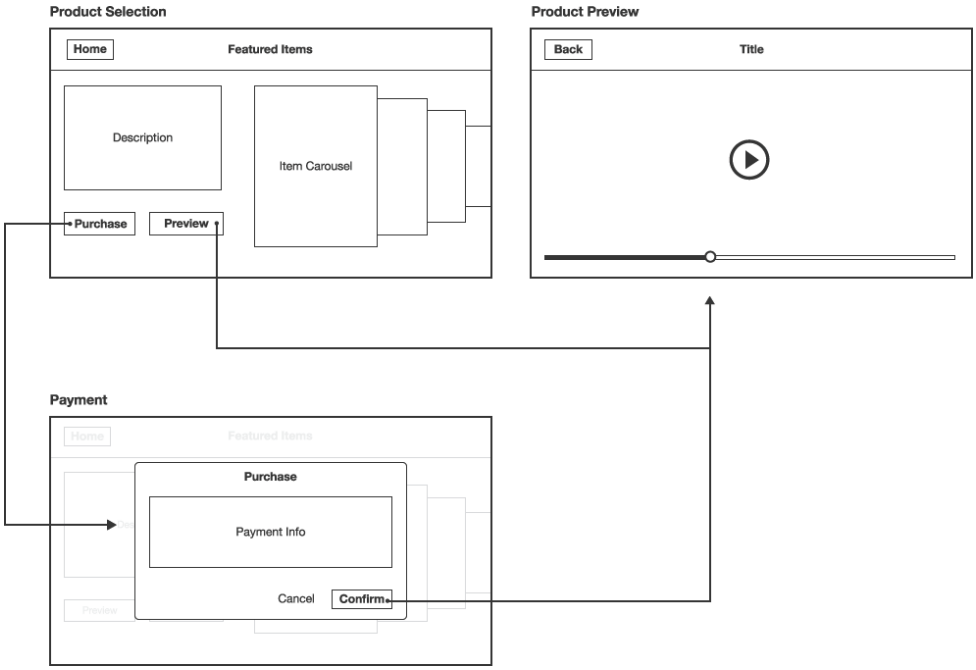


FIGURE 5-18
Even the most basic wireframe can help get teams and stakeholders on the same page..

Wireframe Technique #2: Using Shades of Gray and One Color

By simply adding one color to your wireframes, you can effectively layer notes and visual cues without distracting from the main content (see Figure 5-19). This is one of the easiest things you can do to enhance the effectiveness of your wireframes. Additionally, using some gray can help distinguish separate content areas. I first discovered this technique in the classic data visualization book, *Envisioning Information,* by Edward Tufte.

NOTE

Edward Tufte's *Envisioning Information* is a must-read for anyone interested in the field of information design.

When to Use

This style of wireframe is best suited for projects of medium-to-high complexity. Keep the color minimal, and use it to make a succinct point. UI specifications, animation notes, and annotations are all things best communicated with splashes of color.

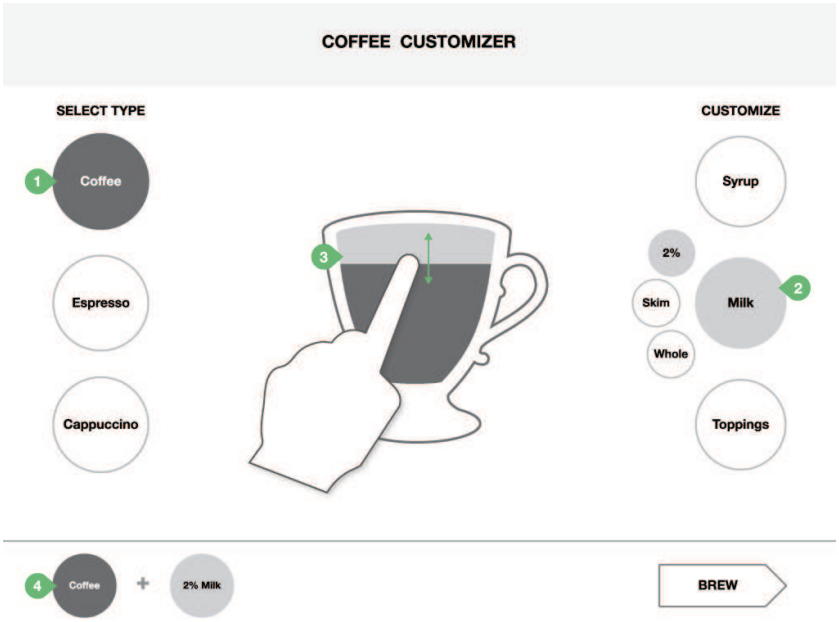


FIGURE 5-19
A splash of color and shades of gray can help keep you communicate complex functionality in a wireframe.

To illustrate some wireframe techniques, the following case study shows an example of a wireframe that was particularly challenging in the following case study. This example contains a variety of techniques used simultaneously to communicate some complex scenarios and showcases some useful illustration techniques that you will revisit individually afterwards.

Case study: A particularly challenging wireframe

Every now and again a project pops up that features new interactivity or design concepts that are challenging to communicate through static wireframes. When this happens your brain will immediately start to strategize how to best present these ideas. But it can be frustrating getting the ideas across with minimal documentation. In this case study I'll show how vacuum assembly instructions and comic books were the inspiration I needed to overcome this particularly challenging wireframe.

continued

continued

The Brief

A client asked us to create a series of interaction concepts to show stakeholders how to take advantage of an up and coming platform. The ideas needed to be forward-looking and show specific examples of how their brand could differentiate itself, appeal to consumer aspirations, and potentially leapfrog the competition—a tall order.

The Challenge

In this case, our main challenge was to create progressive but realistic ideas on a future platform. But the *real* challenge was that in a very short amount of time, we had to get the audience's attention, educate them on a brand new platform, excite them about our ideas, and do it within the bounds of a couple of 1024x768 slides.

Presenting concepts to a group of stakeholders is arguably one of the most difficult tasks in consulting. You stand in front of a bunch of clients fighting for their attention, and at any moment things can take a turn for the worst. Eyes glaze over, BlackBerry devices come out, and any chance of getting through to them is lost. Nobody likes to read slides with a ton of text and bullet points. And nobody listens to a presenter who reads from a drawn-out list of talking points. If clients don't hear anything that they think is interesting within the first 30 seconds—well, good luck keeping their attention.

The Approach

We knew we were competing against tight time constraints and short attention spans. So, our goal was to create a series of wireframes that would convey the value of our vision with little or no text. We decided to *bet* on some carefully crafted wireframes to get our ideas across.

The Solution

We explored numerous approaches for representing our concepts, and we struggled to get everything stuffed into a couple of slides. Then we stumbled upon some unlikely sources of inspiration that helped us achieve the conciseness we were looking for—furniture assembly instructions and comic books. I never noticed how efficient comics and assembly instructions are at showing complex scenarios with nothing but images (IKEA and Dyson are some of the best). So, with some new illustration techniques in hand, we finally hit the sweet spot. Figure 5-20 shows some of our concepts (slightly altered for legal reasons). The figure shows a handful of communication techniques including "frame-by-frame", "bubbles", and "magnify". You'll learn more about each technique in the remainder of this chapter. In the end, our wireframes did the trick. We were able to facilitate a lively discussion, and the client was able to envision their brand in the scenarios we presented. But, more importantly, I added some reliable tactics to my wireframe repertoire.

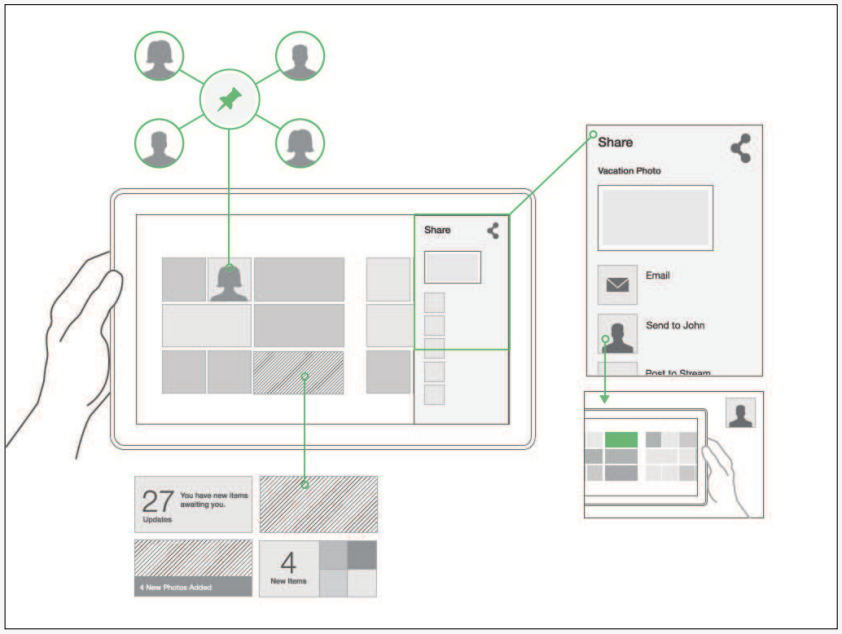


FIGURE 5-20
Applying our concepts.

Wireframe Technique #4: Using the Frame-by-Frame Approach

Showing animation and interaction concepts is challenging with static wireframes. The frame-by-frame approach (see Figure 5-21) is a clear and concise way to show a sequence of interactions with a high level of detail.

When to Use

If you're creating an application that involves specialized gestures or interactions, consider using the frame-by-frame technique. Sometimes, it's helpful to divide these illustrations into separate complementary documents. Touch gestures, drag-and-drop actions, notifications, and multitask scenarios are all good candidates for this approach. Personally, I like to reserve this approach for atypical interactions. For example, I wouldn't frame out a drop-down list to show selection. Everyone knows how drop-down lists work. A better candidate would be, say, a sketching application that supports both stylus input for drawing and touch input for rotating the page while you sketch.

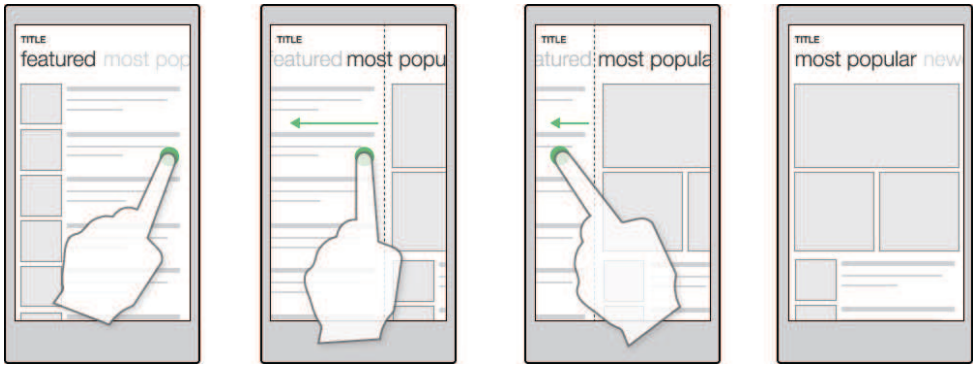


FIGURE 5-21
By using the frame-by-frame approach, you can communicate complex interactivity without heavy annotations or descriptive text.

Wireframe Technique #5: Using Bubbles

The bubble technique is commonly found in comic books to show what a character is thinking, often represented by a little cloud, or speech bubble, showing the character's thoughts. Similarly, bubbles are useful when communicating scenarios that contain background tasks or substeps (see Figure 5-22).

When to Use

The bubble is the Swiss Army knife of the wireframe world. Keep it in your back pocket, because it can serve a myriad of purposes. I often use bubbles to call out key steps in an application flow or high-level concept. Bubbles also come in handy for showing tasks that may be happening in the background of an application such as an asynchronous upload or a timeout.

Wireframe Technique #6: Magnifying Details

The *magnify* is a clever little technique for showing detail in tight spaces (see Figure 5-23). As the name suggests, the magnify calls attention to a subsection of the UI to highlight details that would otherwise be overlooked. I know what you're thinking, *Wait a minute . . . isn't the magnify just another bubble?* Yes, it is. But I use it enough that I thought it deserved its own explanation.

When to Use

This technique is most commonly associated with small devices and embedded interfaces. In general, any scenario that has a small but important feature can benefit from the magnify technique. Another handy trick is to combine the magnify and frame-by-frame techniques to show multiple states of a dynamic screen element.

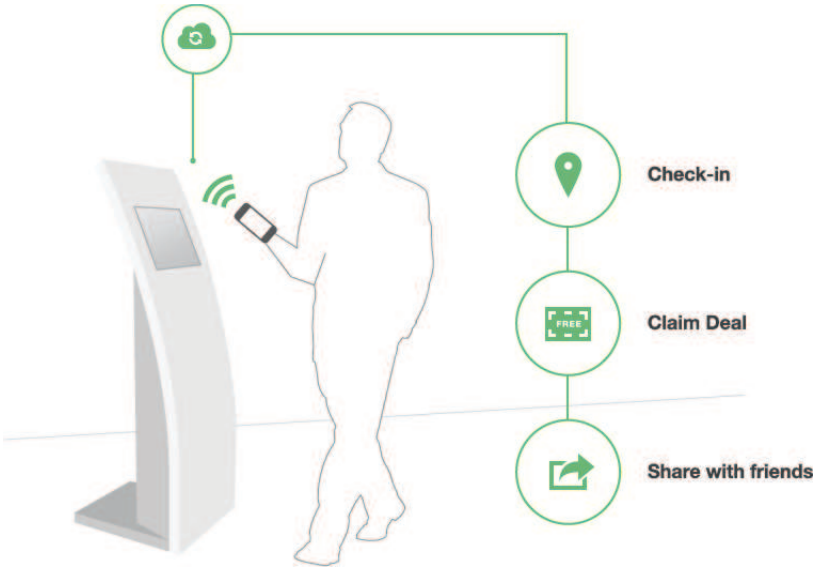


FIGURE 5-22
Amongst other things, bubbles can help frame a sequence of events by giving some insight into what is happening in the background of an application.

FIGURE 5-23
Magnifying portions of a wireframe helps show important details without having to design the entire UI.

Summary

I consider wireframes to be a design power tool because they have the capability of exciting and engaging stakeholders, and at the same time, they allow design and development resources to work closely toward the same goal. The process of creating wireframes gives you a chance to preview how your design will look and feel before you jump into a development or visual design phase. Wireframes will ultimately save you headaches down the line because they uncover complexity and opportunities early on.

The next chapter bursts you out of the perils of the static world and allows you to jump into the fast-paced society of prototyping. Prototyping is my favorite part of the design-process—I know, I say that all the time. But who doesn't like a good prototype? (Answer: Nobody! Everyone likes a good prototype.)