1

Future-Proof Survival Techniques

Tools for tomorrow’s Web

WEB DESIGN PROJECTS contain four essential components. First, you must know your environment. Then you need to plan ahead, learn to adapt, and finish the process by resolving compatibility issues as they arise. Meeting these objectives is challenging, but the work is necessary if you are to successfully future-proof your site. This chapter describes each of these components and illustrates the importance of being flexible to change your way of thinking about your design methodology, now and in the future.

Understanding the Environment

When you design a site, understanding the environment where it will exist is critical. You need to know what factors increase or decrease the chances of your site being noticed, and you need to be familiar with the tools visitors will have at their disposal when interacting with your site. Understanding what makes up the Web's current incarnation is important, as is learning what is and isn't possible (or useful) when designing for it. Before you can access any of that information, however, a number of untruths need to be dispelled.

The truth behind terminology

Throughout the Web's history, designers have become adept at assigning names to things, even if the names aren't required or deserved. Names have been assigned to specific technologies, techniques, and events that occurred long ago, and abbreviations have been created that try to encompass entire technologies. Although some of these terms (for example, HTML and CSS) do a great job at identifying an important technology, an unfortunate slew of buzzwords has been forged, leading to confusion among designers.

Here are three of the biggest offenders:

> AJAX

> Web 1.0, Web 2.0, and Web 3.0

> the Mobile Web

Although forged from the technologies it employs, AJAX (asynchronous JavaScript and XML) didn't need a shorthand name because developers were already employing these techniques in their work. The mechanism behind AJAX is a sound one, in that you can avoid page refreshes by pulling or pushing data from the server to a user's device in the background, but as far as the stability of your site is concerned, the mechanism can be fraught with problems, such as the unavailability of scripting.

Ubiquitous and future-friendly layouts cannot be obtained by jumping onboard with every new technique or technology as it arrives (as AJAX shows). No matter how popular these buzzwords become, the name of the technique is never important; what matters is the problem that the technology aims to solve and whether it, in fact, solves it. A great example of this is the Web 1.0 to 3.0 movement. The terms themselves have little meaning except to try to "mark" the Web's evolutionary progress. Yet, for all of its public appeal, it solves nothing.

Note

What makes buzzwords extra confusing is that some of them have different connotations, so they can mean different things to different people or in different situations. Web 2.0, for example, isn't just a defining era of the Web; it's also a highly recognized design trend.

Terms like *Web 2.0* have come to mean different things to different people, and often just stereotype sites as meeting a list of criteria that keeps them current. The trouble is that not all users will demand the same things and not all devices or browsers will be capable of reliably implementing the proposed features, which, as such terms imply, are critical to the evolution of the Web. In essence, not all sites require AJAX or collaboration features, and including them could damage a user's experience on your site.

I've established that AJAX can be problem for certain users and that Web 2.0 doesn't offer a firm solution to help create or maintain a stable and usable site, so the next step is to investigate what's been dubbed *the Mobile Web*. This term appeared when the use of handheld, non-desktop, web-enabled devices increased, which put pressure on designers to make their sites mobile friendly. Unlike Web 2.0, this term makes some practical sense, but the trouble begins when you try to define what actually constitutes a "mobile" device, and trying to define mobile variables just creates more questions, including these:

> If mobile just equates to a small screen, aren't laptops mobile?

> If mobile is about not being "desktop," are 100-inch TVs mobile?

> If mobile is focused on the new wave of technology, what about PDAs?

> Perhaps mobile equates to data speed, so what about dialup users?

If your aim is to make a flexible and usable layout, all that matters is that users of such devices can take advantage of your site. To achieve this goal, avoid stereotyping users' needs and situations and build real-world solutions that are flexible and durable enough to accommodate every environment, whether it's a handheld device with a touch screen attached or a desktop computer with a large display, mouse, and keyboard.

Mythology and folklore in design

In the following sections, I confront a few common myths in web design. The information in these sections will help you look beyond the old one-size-fits-all environment and begin to understand the need for layouts that flex to your users' demands. The critical thing to take away is that no silver bullets or shortcuts can ensure a stable site that'll last into the future. Instead, future-proofing your site includes balancing the needs of users with the tools you can provide.

Myth #1: Layouts can be made to appear pixel-perfect

Web designers try to make the sites they design look and feel as consistent as possible in various environments, but the idea of being pixel-perfect is flawed. By making something *pixel-perfect,* I mean trying to enforce strict viewing guidelines akin to those in the print industry, thereby making everything look the same in every situation. Because so many variables play a role in a site's rendering, situations will continue to exist where users experience some kind of limitation. Perhaps they're missing speakers for sound, or they navigate using a dodgy browser. Not all experiences are created equal.

Note

For older devices, pixel-perfect layouts were impossible from the outset. Desktops could handle feature-rich HTML and CSS layouts with plenty of complex interactive features, but traditional featurephones could handle only WML code devoid of the stylistic beauty and script-powered behavior that desktops were afforded for years.

The truth is, user experiences don't have to be identical for your site to work. You may actually want to design so that user experiences differ among platforms and make your site more usable. You might offer separate, altered experiences based on the capabilities of the different devices. (Note that a unique WML layout was compelling for older hand-held devices.) As long as your content remains visible and users are willing, within reason, to adapt their navigation techniques to interact with your site in a way that matches the requirements of their devices, you don't need to worry about precision design.

Myth #2: Designs can be considered "complete"

I'm a big believer in continued improvement, and because standards and use of sites will always be changing, based in large part on users' activities and preferences, sticking with one layout and declaring to the world "I'm finished" is . . . well, surely said in jest. Your goal as a designer is to make sure your site continues to gauge the interests of users, and although you don't want to redesign a site every week, it makes sense to iterate and improve your services regularly (as shown in Figure 1-1). As technologies and best practices change, new methods to help your visitors will appear.

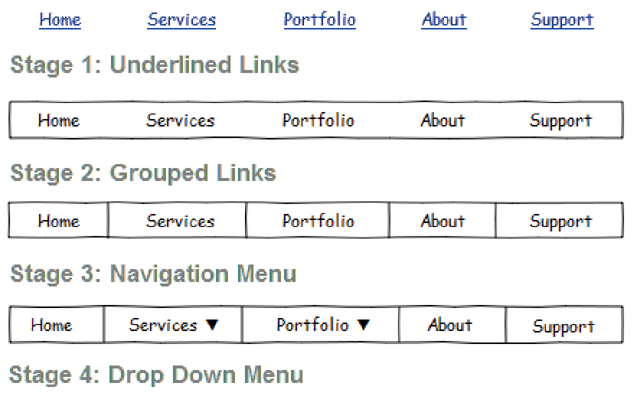

FIGURE 1-1: Iteration allows designers to continually improve their work.

The idea behind a *completed site* is that nothing can be done to make it better, which doesn't add up. Improvements can always be made and new features can always be added. Also, be sure to maintain and update the content on your site to encourage visitors to return. If you own the site that you're building, you can set it up so that iteration can occur naturally. When you're building for clients, suggest that they establish maintenance schedules and frequently improve the content of their site.

Myth #3: A design can be totally bulletproof or future-proof

Although this book's goal is to help you maintain stability in a layout and make your site as future-proof as possible, ultimately no design is immune from all that the Web can throw at it. When a site is said to be *bulletproof,* it means that the site won't fall apart under any circumstance. That a site can be bulletproof is an idealistic and unattainable notion. When a site is said to be *future-proof,* the implication is that the site will work successfully forever, across new devices and emerging platforms, all while maintaining compatibility with previous browsers and devices. In this book, I do my best to help you head toward that goal, but as much as I'd like to guarantee that goal, I can't, because the Web is far too unpredictable.

By considering the variables in this book, you can better address the concerns that designers of today's sites deal with. Keep in mind that those variables will play an important role in the Web's future landscape. But who knows what's on the horizon? In ten years, the Web may change so drastically that designers will once again find themselves building sites in new, unconventional ways. Perhaps a whole new range of variables will exist. Ultimately, all you can do is use the information you have and make the most of it.

Myth #4: Validation ensures quality and compatibility

Many Web designers make the mistake of taking validation of code as a guarantee of standards, which is why you see so many of those "Valid" buttons embedded within so many sites (see Figure 1-2). However, as you probably already know, you can have some of the best-formed code and still see quirks and inconsistencies in how a site will render among browsers and devices. This isn't to say that validation is useless because, for example, knowing how to spot bugs that could lead to quirks is important. They just aren't a silver bullet for ensuring the stability of websites.

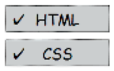

FIGURE 1-2: Validation buttons don't guarantee the quality of code or impress average visitors.

Validation is a useful tool that can help identify common flaws and mistakes that designers make when coding. Including validation in your workflow is useful, but it's just a tool. Don't consider validation programs as an alternative to or replacement for testing your work properly, and don't assume that all validation programs work equally well. Accessibility validators are notoriously bad at uncovering major failings in accessibility; manual testing is the only safe option.

Myth #5: The newer, the better — the more, the merrier

Designers often get carried away in their bid to be creative, and more importantly, they can be overly zealous about how much of a good thing their visitors will enjoy. In an effort to stay current, some designers revamp their sites regularly, when redesigning is clearly unnecessary; or they add too much information, media, or imagery to their pages thinking that substance in great quantity encourages more interaction. Of course, keeping your design and content updated is necessary; just be rational about when and how to do so.

Note

Be sure to remove clutter from your interfaces. Often, pages become stagnant and bloated as a result of mismanagement or residual features such as animation effects that you think look great but offer no real benefit to users. Overuse of design or content is a common problem, so try to keep your designs tasteful.

Incorporating the latest and greatest features can be an improvement if your goal is to improve users' experiences. If you use these features mainly to compensate for poor-quality content, you could create a real problem for visitors to your site. For each new redesign you create, just remember that your visitors' learning curves will increase because they must readapt to the new interface. The same goes for bundling more features and content on a page. Simplicity is often better than complexity. Keep in mind that adding features and too many choices may be a burden for some visitors.

Myth #6: You profile the average user or device

Design is rarely an objective art form, and as much as designers want to base their decisions on the needs of users, designers' personal biases and skewed perspectives can influence their work. For example, when designers think of a visitor, they often visualize an idealized visitor rather than one based on reality. Moreover, when designers try to profile the type of environment visitors will be using, those profiles may fail to take into account the differences between different users' experiences. The idea that an "average" user (Figure 1-3) or browsing environment exists is unrealistic.

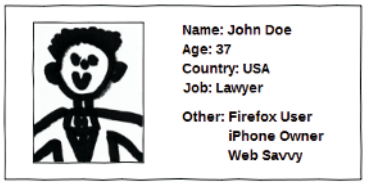
Name: John Doe
Age: 37
Country: USA
Job: Lawyer
Other: Firefox User
iPhone Owner
Web Savvy

FIGURE 1-3: Designers often use personas to group variables together, forming a browsing scenario.

The differences among the environments where your site must function can be substantial — for example, whether Flash or JavaScript are supported. The differences among users are important, too. Some users may encounter accessibility issues, and others may simply be more selective in the features they've enabled. Designing for ubiquity requires looking beyond stereotypes; instead, you need to be open and adaptable in terms of your audience. Promote equality and be flexible with whatever your site needs to function. By doing so you'll end up with a stable and usable layout.

Keeping up with the Joneses

Designing for next-generation devices poses a real challenge. After all, how can you be expected to design for something that doesn't exist or, if it does exist, hasn't gained widespread adoption? Consider how the Web works on cellphones and tablets. In a few years, the Web may work on all sorts of other unique devices, such as televisions. When you think about it, gadgets like the iPod Nano have the potential for web enablement, and it's only the size of a wristwatch, so imagine how diverse web experiences may become!

Hardware is becoming less expensive to produce, and infrastructure for web connectivity is gaining adoption worldwide (even in hard-to-reach places like Mount Everest). This situation fosters the perfect environment for ubiquity because reduced cost and low-barrier entry encourages more people to go online using devices they have handy, be it in their homes, offices, or on the move. As the number and variety of devices used on the Web increases, you have two options: Patch as new devices appear, or be generic, yet flexible, regarding usage.

When a new device gains popularity, many designers immediately patch their sites to support it, target the device for a special independent site that caters to the platform, or just ignore it. These don't seem like good options because they require you to choose what you will support and provide constant patches to the ever-growing technologies that arrive online. In some situations, but not all, a separate site might be helpful.

Reference

The debate over separate versus internal sites has been brewing for a while, leading to the idea of "One Web." Some individuals believe this principle can achieve discrimination-free usability; others believe in the stricter definition of eliminating all proprietary, single-case solutions (which means demands ensuring that everything works for everyone). Check out Opera's view of the "One Web" debate at this site: http://www.opera.com/business/oneweb/.

A better approach is to examine the symptoms, make a diagnosis, and find suitable solutions to treat the condition. It isn't the brand or model that makes a device; the components make the device. The inside of an iPhone and the code it supports (such as HTML) differ significantly from what you find in a Nokia 6610i (which supports only WML, as shown in Figure 1-4). The issue boils down to two independently built renderers doing what they can.

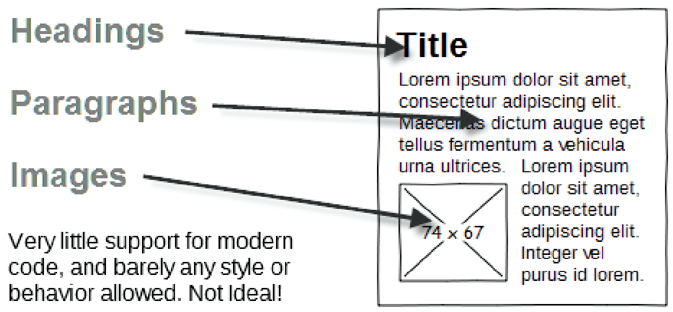
Headings
Paragraphs
Images
Very little support for modern code, and barely any style or behavior allowed. Not Ideal!
Title
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas dictum augue eget tellus fermentum a vehicula urna ultrices.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer vel purus id lorem.
74 × 67

FIGURE 1-4: When compared to today's rich and engaging HTML and CSS, WML is a real ugly character.

Ultimately, the choice to keep up with the trends or retain support for only a select few of your audience's situations is entirely up to you. It may be impractical to produce a site that is so flexible that it supports every type of product and situation without issue, and if you know what your audience requires, there's no need to go over the top, covering all possible bases. However, unless you have a good reason to do otherwise, ensuring that your site caters to as many situations as you can eliminates the need to patch your site's code in the future.

The browser wars proved that creating sites that depended on everything rendering in one way was problematic. Designers have since adopted more stringent measures for testing workflows. Currently, designers are struggling with the fact that devices and platforms force you to rethink how you present and organize content. In the future, perhaps your next tussle will be over dependency on frameworks, the continued support for deprecated code, or something else entirely. I do enjoy a good mystery—don't you?

Planning for a Successful Website

Understanding the environment you're designing for is critical when building a site, and knowing the needs of your audience is critical to a successful design. After all, recognizing potential situations beforehand allows you to make more informed, practical choices. With information about a user's environment, you can avoid making potentially costly mistakes that could reduce a site's usability, and by planning ahead, you can direct your attention to specific aspects of a design that may be more open to implementation quirks than others.

Determining project requirements

When developing for the Web, you must be able to determine the requirements and needs of a project. Often, for service providers, the demands of a project include the additional features, functionality, and layout choices that users may find helpful. For situations in which you produce work for clients, the ability to look beyond just the scope of a site and into the future needs of their businesses will help ensure that the design works for the specific audiences the clients are targeting. Because every site will be different, catering to niches is important.

The initial requirements that influence your tactics are those laid down by the users of the site. Happy visitors often result in customer loyalty, so, whenever you can, put users first. Ensuring that your users have access to sites and services regardless of the platform or device they use (which equals ubiquity) means that you can more easily get them to choose you over a competitor — and choice is a powerful motivator.

Here are some features that make for happy users:

> Consistency in a site's design

> Accessible and easy-to-use layouts

> Aesthetically pleasing designs

> Goal-oriented, useful layouts

Remember, the benefits of a ubiquitous interface extend beyond what the user sees and the devices the user uses to access your site. The number of social networks, search engines, and third-party tools connecting to your site is increasing, and it's likely that the more demanding and restrictive methods they use to view and utilize your content will become increasingly important. Just think about software like Instapaper or an RSS reader, and you'll understand how your site can be interpreted by a machine, not a human being; any errors affecting it will certainly reflect in the output.

Here are some features that make for happy robots:

> Search engines may struggle with proprietary code.

> Social networks require meaningful, contextual data.

> Browsers demand well-formed code to render pages.

If you're working for a client, you can't just plan around the needs of your users and the specific devices they use or the automated solutions that exist; you must also plan around the business or client. Clients may have certain niche requirements — if they are making an intranet, for example — or perhaps they want the added usefulness of platform-explicit applications (such as those in Android's marketplace or Apple's App Store). These days, sites encompass many more options than they used to, and every site's requirements will be different.

Note

Consider the client-user scenario as an adaptation of the "three laws" from Isaac Asimov's *I, Robot*. Sites cannot harm a user, must obey clients' orders (unless it violates the first law), and must do the same for designers (unless it violates the previous two). With this idealistic balance, the designer's priorities should be set.

The needs of a site depend on the factors described in this section. You'll probably spend as much time researching what is needed on an interface as you do building it. In a designer's ideal world, people would conform to stereotypes, devices would be standardized, and clients would jump for joy at the thought of accessibility. Unfortunately, you don't live in an ideal world, and it'll be many years before widespread compatibility and ubiquity will exist (if it comes to exist) because meeting expectations can be fraught with hurdles. Changing dogmas or perceptions takes time.

Setting goals while dodging holes

As the Web has evolved, designers have found themselves playing a superhero-like role, which you'll understand if you're a fan of fantasy and shows such as *Buffy the Vampire Slayer*. Buffy worked her way through demons, taking on increasingly dangerous and deadly foes (you can relate to this if you've coded for Internet Explorer 6), and ended up in a final showdown with the "big bad." In the show, overcoming each challenge on the path to winning the war wasn't a matter of luck or charging in blindly; it involved careful planning and research.

Because each user and situation is different depending on the type of site being built, you must carefully consider any implementation that enhances or degrades a user's experience. You need to establish primary goals to ensure that decisions are made for the right reasons. Perhaps your site will require visitors to enter some log-in details, but remember that input mechanisms can vary among web devices. Maybe a visitor will browse while on the move. These kinds of situations can trigger and affect the many variables you must consider.

The following situations affect specific factors or variables:

> HD video is affected by bandwidth and connection speeds.

> Color is eliminated if a visitor has a monochrome display.

> Mouse precision and accuracy are affected by click regions.

During the brainstorming stage, establishing where and how a site might be used is a great idea. Sites that compare prices of products are likely to be in heavy demand while visitors are on the move — for example, traveling on primary business streets and in shopping malls. The use of sites like the Internet Movie Database require specific consideration because they may be used in collaboration with cinemas, rental shops, and media retailers. Creating scenarios or profiles of these actions help you gauge targeted markets, although, of course, users browse in other kinds of situations, too.

The trick is to determine which influences and variables will affect your users; what those effects will be; how you can ensure that the interface will cater to your specific audience without negatively affecting others; and how to implement required changes in the most suitable way. Making these determinations requires a fundamental understanding of how human-computer interactions work and of the subtleties of users' devices. For example, a smartphone may be subjected to data caps and roaming charges, and an old desktop computer may have a slow or low-quality connection (see Figure 1-5). Goals must always be identified within the context of acceptable methods of interaction.

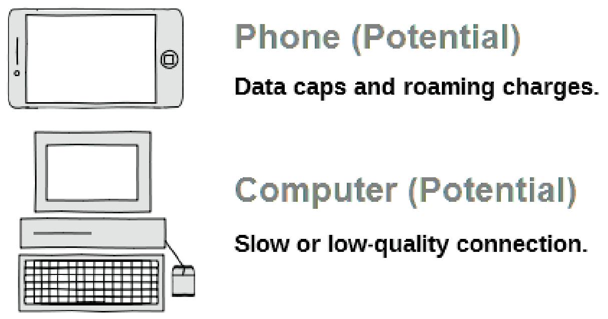
Phone (Potential)
Data caps and roaming charges.
Computer (Potential)
Slow or low-quality connection.

FIGURE 1-5: Certain situations may require you to consider how data-heavy your sites and pages are.

Dodging some of the major pitfalls in design can be tricky, especially if you've become comfortable that what you have "works." Ultimately, for any plan involving the longevity of a site, you need to observe changes in patterns of use, determine methods for improvement, and identify potential causes of concern. However, keep in mind that because the Web is constantly evolving, new standards will make providing flexibility increasingly convenient (as is the case with CSS media queries), and, as a result, your solutions can be better implemented.

Planning for implementation

Planning for the implementation of a ubiquitous site can be challenging. All too often, you'll find yourself asking a variety of questions about your audience that have, in turn, a variety of answers. For example, what screen resolutions do they use? What browsers do they use? Do they visit the site on cellphones? Does your site please or somehow irritate them? You have many design and development tools available, and with tweaks, they'll aid your ubiquity goals.

Planning ahead makes building a successful site more feasible. When envisioning your site, plan for code and a design that are well formed and as uncomplicated as possible. Set clear objectives and be willing to compromise for the sake of your users. While planning, identify where you can make implementations more accessible to and useful for your site's users, as well as related variables they will interact with. Remember to set realistic goals; otherwise, your site may fail users in some way.

Note

Treating how users will access and use your site as an afterthought is very risky. Every site relies on content and functionality; nevertheless, the basic design of the layout should always make users the top priority.

Ideally, the process of determining site-specific goals begins with competitor analysis and user testing. Next, you use wireframes, prototypes, mockups, concept sketches, and other tools to discover the specific needs of the project. If you think users may want something, don't shy away from considering it. Planning can become second nature, once you get into the swing of things. Moreover, if you determine the needs of clients or of visitors to a site, you can implement suitable outcomes, right from the start. At its heart, web design involves inspiration, iteration, formulation, and publication (see Figure 1-6).

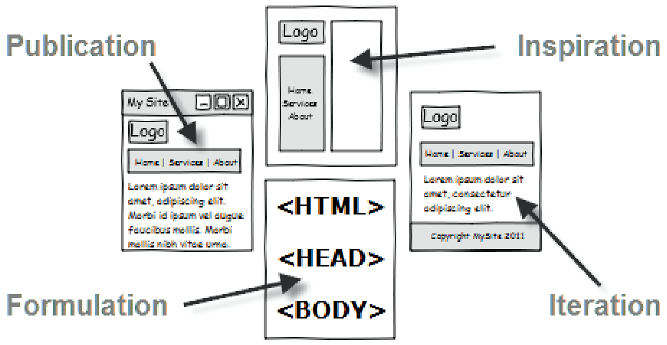

FIGURE 1-6: Inspiration, iteration, formulation, and publication are critical elements of web design.

Consider the issue of whether to offer a secondary mobile-oriented site. If you provide a separate site for visitors who are using less-capable devices, those visitors might avoid optimized environments entirely. Therefore, it's important to give them the option of returning to the "full site" (if, for example, the optimized version is slimmed down, offering users less content). Remember, having choices empowers users.

When you empower users, you give them a sense of control, enabling them to feel as though they aren't just at the mercy of a site's demands. Perhaps you deem CSS or JS a necessity. That would be fine if it couldn't be turned off, disabled, or unavailable. The best approach is to plan for the worst and hope for the best. If you make your content available to even the weakest link in the chain and, at the same time, enhance the experiences in more unique situations, you'll ensure the maximum visibility of your content.

Learning to Adapt or Evolve

You know what's going on, and you have a clever plan to provide a service that will be the envy of your competitors. Fantastic! Next on the agenda is deciding how to adapt your best-laid plans to particular environments. If you get dropped into a jungle, you don't act like you landed in Siberia. Likewise, online, you'll need a box of tricks to cope with the many different requirements a site may throw at you. Every site is different, as is its audience. Your job is to be prepared to find the answers to the difficult questions that environments can present.

Taking advantage of new technologies

Although you don't want to use every new technology just for the sake of keeping up appearances, you also don't want to let your concern over compatibility get the better of you. In an effort to appease the "old ones" (for example, Internet Explorer 6), many web designers have failed to take advantage of CSS3 (for example) purely because it creates inconsistencies with a browser's older counterpart. Although I'm all for compatibility, as I said earlier in this chapter, trying to be pixel-perfect is neither worth the effort nor possible.

Compatibility should always be possible because of the following:

> If everything is disabled, content is the one thing that remains visible.

> Many technologies, when unsupported, can have an appropriate alternative.

> Targeting specific variables allows you to offer independent fallbacks.

Going beyond the bare necessities with your code is, of course, entirely possible. If you want to provide a particular piece of functionality, make sure you have a *fallback* (alternative) for users who are less fortunate. Such functionality can work against making a site ubiquitous, but that will occur only if you fail to update the site as new and better solutions arise. Ideally, rather than restrict yourself to a limited layout, train code around issues as they appear (see Figure 1-7).

As a web designer, you have a responsibility to your clients and customers. Failings on your site raise the risk of losing visitors, even if the failings are just small, but annoying, quirks. Knowing how to write code for a site helps you understand in advance where experiences can falter, provided that you take steps to ensure that your work flows and responds appropriately to user interaction and the environment in which it's consumed. If you ignore the signs, however, issues are likely to occur and reoccur.

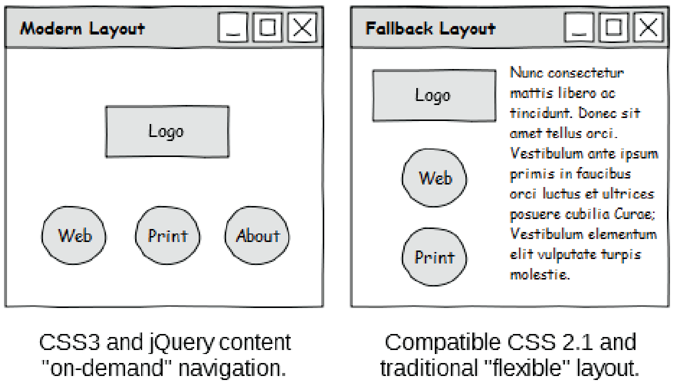
Modern Layout
Logo
Web
Print
About
Fallback Layout
Logo
Web
Print
Nunc consectetur mattis libero ac tincidunt. Donec sit amet tellus orci. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Vestibulum elementum elit vulputate turpis molestie.
CSS3 and jQuery content "on-demand" navigation.
Compatible CSS 2.1 and traditional "flexible" layout.

FIGURE 1-7: There's no shame in providing Internet Explorer 6 users with a very usable and satisfying experience.

As the use of tablets increased in popularity, web designers on the cutting edge began to investigate how this unique input method could affect interactions on sites. At first, it seemed strange that people might not be using a mouse or keyboard. Continued study is the best route to understanding any device or design variable.

You want to make your site as unique and easy to use as possible—and give users a memorable experience (don't go over the top). Designers have come to look fondly on trends (see Figure 1-8), conventions, and patterns for this very reason. Part of adaptation is moving with the times, working with your surroundings, and recognizing when views of how the Web should operate alter over the years. Maintaining a high level of awareness and staying ahead of the curve makes sense.

Home
Services ▼
Portfolio ▼
About
Support

FIGURE 1-8: Following trends isn't a bad thing, especially because they're usually based on established solutions.

Sometimes *cutting edge* or *bleeding edge* is used in reference to designers or developers who use tools that aren't yet supported by the mainstream. Although both the cutting and bleeding edge may appear as unsuitable candidates for crafting a stable site, using one or a mixture of both can be done in such a way that those who have access to the supporting tools benefit and those who don't have access have something just as fitting to use in its place.

If your competitors are going to provide support for a tool, and you look around to find that you could be the last person standing in the traditional-techniques circle, it may be time to investigate whether moving on can benefit your audience. Often, new technologies provide designers with appealing solutions that otherwise wouldn't be possible (for example, CSS sprite rollover menus). Many designers keep an eye on sites that use cutting-edge work, looking for inspiration and creative ways to polish their own skills.

Adapting your site to account for the many existing variables gives you insight into the habits of users and how they embrace technology, and it gives you the opportunity to offer them more user-centered designs. You will see an increase in the use of small screens and the removal of the barriers of fixed-width design. Also, you'll see how reducing the requirement for inputting text helps users without quick access to a normal keyboard. Creating future-proof sites is about molding platforms around experiences that can benefit every user.

Solutions for a successful layout

This section covers techniques for producing first-rate, scalable layouts. Before you can understand design variables in their entirety, you must first be conversant with the methods designers use to make layouts as flexible and future-proof as possible. This information includes making decisions about which methods you will use, why using a particular technique will benefit your audience, and which of the many layout techniques will sustain the highest levels of ubiquity.

Consideration #1: Need versus none

At first glance, it may appear a bit silly to ask, "Do I really need a flexible site?" For the purpose of this discussion, the aim isn't to question whether having a flexible site is a good thing, because clearly it is. Also, if you design flexible layouts from the outset, you will reduce the chances that users will face problems with your site later on. However, understanding the needs of your audience can tell you a lot about their specific requirements or about non-issues that may influence decisions to build or postpone the implementation (see Figure 1-9).

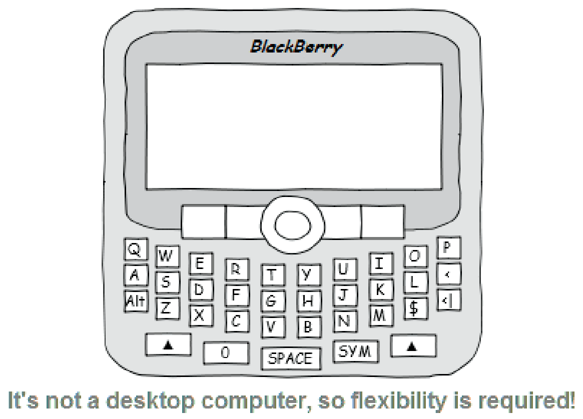

FIGURE 1-9: If a site primarily attracts users of desktop browsers, you could postpone the flexible upgrade.

If you were to produce a site purely for consumers of Apple products, you would probably question the need for a stress-testing spree to try out the site with as many emulators, browsers, and devices as possible. On the other hand, making your site as flexible as possible is important, but there isn't really much point in spending the next year and three months scaling your site to be in line with every potential variable. Let your users and their needs determine the level of flexibility and whether you can afford to cut corners.

Consideration #2: Rigid versus fluid

You can lay out content in different ways. In one camp, you have the grounded, rigid units of measurement that can cause unpleasant horizontal scrolling when the available space doesn't match the demands of the interface (think fixed designs using pixel widths). In the other camp, you have fluid designs that are pleasing, until there's too much or too little assigned space (causing occasional spillages or overflow from scrolling). In both cases, entire layouts can break if the equations don't add up.

Reference

An article I have written shows how the formats of layouts are changing. Not too long ago, you had only fixed, fluid, and elastic to contend with. Today, you have no less than ten choices! They range from units aimed at print or default preferences to complex mathematically instigated alternatives. For details on how each could affect a design's flexibility, visit this site: http://sixrevisions.com/web_design/a-guide-on-layout-types-in-web-design.

You can choose units of measurement based on compatibility (units aren't treated equally online), on a design method (such as responsive design), and even on a hybrid of one or more techniques. Making the right decision about the mechanism of layouts can play a critical role in how variables interact with a layout and, more importantly, how a page will respond when under stress. You want to base such needs primarily on the requirements of the content and then on the space required for functionality on the web page.

Consideration #3: Dynamic versus static

Dynamic and static layouts also play a part in the construction of sites. *Static designs* are those with little to no interaction, are comprised entirely of text or images, and are more focused on a read-only approach. *Dynamic designs,* on the other hand, usually include scripts, changeable content, features, and perhaps some clever code in order to boost the site's core flexibility (as shown in Figure 1-10). Both of these design types have advantages and disadvantages, and both affect a layout's core stability.

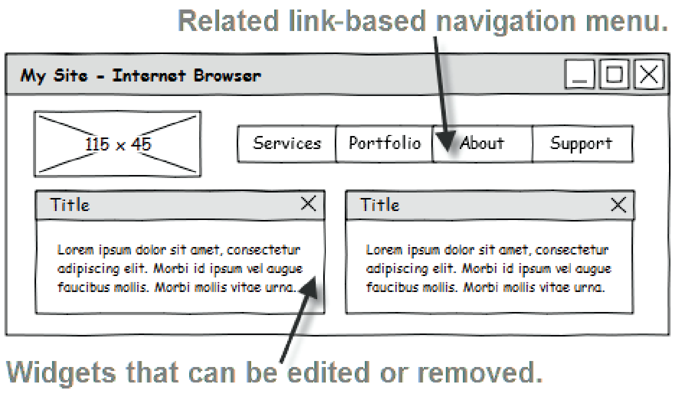
Related link-based navigation menu.
My Site - Internet Browser
115 x 45
Services
Portfolio
About
Support
Title
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi id ipsum vel augue faucibus mollis. Morbi mollis vitae urna.
Title
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi id ipsum vel augue faucibus mollis. Morbi mollis vitae urna.
Widgets that can be edited or removed.

FIGURE 1-10: You may be able to improve the flexibility of dynamic sites by structuring them around visitors' preferences.

Static sites have little going on under the hood. What you see is really what you get. The benefits of this traditional form of layout are that once you've ensured the content scales appropriately, little else beyond the visual arrangement can go wrong. With dynamic sites, you may find that if scripting becomes unavailable or interaction requires additional user involvement, trouble can occur. However, even with such concerns, dynamic sites can offer a greater level of individually oriented flexibility than static sites can, so the payoff might be worth the effort.

Consideration #4: Internal versus external

This consideration relates to how to handle alternative device requirements. Sometimes, designers choose a "one site rules all" approach and account for variables by using scripts or stylistic fallbacks. Tools such as browser-detection scripts, frameworks, and media queries allow the layout's appearance to change based on a user's needs. Although this is the best choice (requiring little added maintenance), the major catch is that it forces you to rethink a site's mechanics, based on assumed scenarios of use. Figure 1-11 illustrates the concept of a script working as a robot to "build" a site around you.

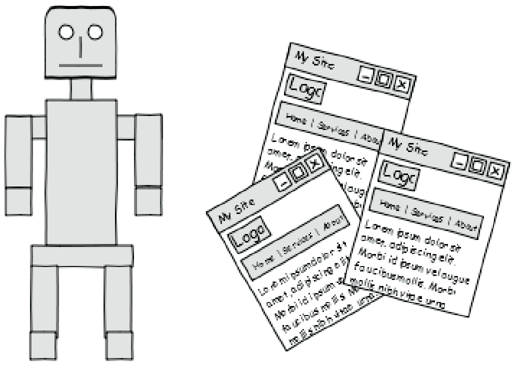

FIGURE 1-11: Scripts act like little robots, reporting on what will or won't work.

If the work of designing for the lowest common denominator isn't your cup of tea, a quick-and-dirty solution is to provide an external site that does the job, similar to what you may have seen in mobile-specific sites. In these optimized layouts, however, you'll often find that content is either "dumbed down" to reduce the pressure of the layout or condensed to make things more lightweight. These layouts, however, beg the question, "If it's not needed on a mobile site, why would you want it on the desktop?"

Consideration #5: Redesign versus realign

If you choose to accommodate various browsing environments, such as the use of specific devices or products by creating separate layouts, you must determine whether to build a new layout entirely from scratch or to realign an existing site's design (if one exists) to consider the more diverse uses being asked of it. Ultimately, situations will exist when a new and separate layout may be beneficial (perhaps for a mobile-only service that's not available for desktops), but in the vast majority of cases, keeping sites together requires less work.

Calculating whether to redesign or realign may be easy, depending on the state of the site in question. For example, if the layout is falling apart, cannot match the needs of the content, or is simply unattractive, redesign it! After all, revitalizing the layout can't make it look, work, or feel any worse to your visitors than it does in the state it's in now, right?

If you choose to redesign a site from the ground up, all your previous work may be wasted. In addition, you'll have to go through various iterations to regain the previous layout's level of flexibility before the total revamp (which means more testing and, perhaps, some secret sauce). Just realigning can be tricky, too, because you may encounter various barriers hiding under the hood, waiting for their moment to break the site or its underlying system.

Consideration #6: App versus online site

This final consideration is a quick one, and it rounds off the fundamental considerations involved in planning a site. First, my question: Should you provide your site in the form of a native, downloadable app or within the wrapper of a well-crafted, browser-based site? As with the previous issue, no one answer will work for everyone. Apps have the advantage for offline use, guaranteed rendering, and more, but because they require no compiling per device, sites require less work, if the variables are accounted for.

Note

The divide between services and applications is getting thinner by the day. You can, in many ways, rightfully claim that most apps can be built using modern web browsers, but compatibility remains a constant issue because you'll attract all sorts of platforms.

In the apps versus online debate, I have my own biases and preferences. Desktop apps are suited for situations in which access to a device's native hardware is essential. Also, desktop apps are ideally suited if you want to build a single app for a specific platform (such as iOS). In my opinion, however, beyond those exceptions, web apps are the better option: You ensure that users have the most recent version, you can build for every platform equally, and in most cases, web apps can be made to perform offline.

Beyond design: An essential business guide

Because building sites is a business for many (if you're reading this book in an attempt to improve your own quality control, this includes you), I need to offer some cautionary words about how striving to survive the Web's future may affect the way you craft or maintain interfaces, and how you bill for doing so.

First, the all-important consideration: money. Budgeting for the work you will do to bring an older, less flexible site into the "here and now" is a complex calculation. At one time, you could design sites so that they were cross-compatible by writing well-formed code (which designers often failed to do). Justifying it as cost-effective wasn't difficult because you could literally see bandwidth bills drop (moving on from messy and inaccessible table-based layouts). Now it's a different story with fewer immediate gains or losses involved (see Figure 1-12).

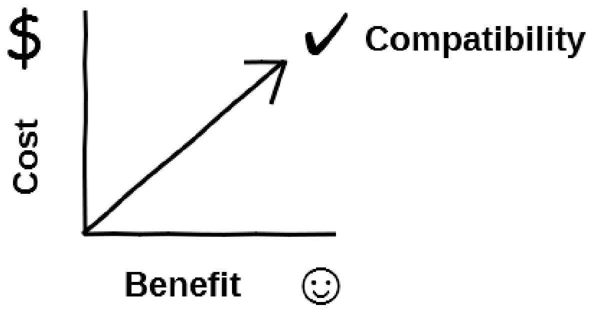

FIGURE 1-12: The extra costs of ensuring flexibility can result in a bigger audience.

Building a mobile-friendly site demands regular testing, and not on the same level as in the old days. Testing on large numbers of browsers, both desktop and handheld, can adversely affect billable hours, and testing on physical devices could potentially be very costly if you want to guarantee device-specific support. Because of the overwhelming array of possible combinations of browsers and devices, testing on all of them will be impossible, so you need to find ways around this issue in order to test as economically and accurately as possible.

Also, you must consider the increased time it could take to build such a scalable site. If a site was originally built with flexibility in mind, updating it with newer variables probably will not take as much time as it would to bring an older, less flexible site up to modern standards. Be sure to account for how long you'll assign to the test phase, which types of tests you'll run, and if you want to consider only a baseline set of commonplace variables in your package, providing less common ones as "billable extras."

Tip

If you're like me, you probably find checking off your progress helpful. By determining a set range of variables you're willing to test against and estimating how long you believe each test will take, you can plan a release cycle fairly accurately.

Furthermore, part of the process of running a business relates to how well you understand your client's niche, users, and community. Considering variables that will not affect a site's audience could be deemed wasteful. However, forgetting about or not adding support as it's needed could just as easily be deemed neglectful! It's a tightrope that you'll need to walk, so know who you'll be coding for—a flexible site is more important than ever in accurate designing!

Finally, there is the educational side of things, which could potentially create a few issues. As a designer, you know that it's good to remain vigilant and keep up with the ever-changing environment. Factoring in the effort it takes to retrain yourself (if needed) to a particular technique and making sure you have the tools required to test against variables is critical. Ideally, you'll spend as much time learning as you do coding.

Here are some ways to stay up-to-date:

> Read books, magazines, and blogs on what interests you.

> Attend web-design conferences and network with others.

> Listen to podcasts and do training via video tutorials.

> Examine the design code of others, and gain inspiration!

Running a business has all sorts of practical considerations, and it's only natural that you evaluate the need or your ability to consider everything in this book. Focusing your attention on what matters to your visitors is part of what makes a great designer. It takes a good amount of common sense and experimentation to organize your workflow in a way that benefits those who'll be affected by it.

Resolving Issues of Compatibility

There is one final, essential survival skill that all designers must have if they want to overcome the challenges of achieving compatibility. By understanding an environment, you can identify the core issues you need to address; by planning ahead, you can reduce the chances for errors; and by remaining open to adaptation, you can implement satisfactory outcomes. But every now and then, you will encounter quirks that trip you up and solutions that fall short of meeting your layouts' requirements. Knowing what to do when things go wrong will help you survive and maintain ubiquity.

Debugging for durable devices

Comprehending the complexities of the Web involves knowing how to spot errors, determining the cause of the problem, and finding a nice, clean, and workable solution. Ideally, all your sites will look and feel amazing on every medium and in every environment, but of course, things often don't work out that way. By ensuring that your sites work well for your many users, you provide a perception of professionalism and competence.

To the average person, the Web may be a bit of a mystery. People don't understand its origins and complexity (Figure 1-13), but they marvel at how it can be experienced and utilized through many mediums. Your responsibility to produce durable, bug-free experiences for users is much an extension of this. Because the Web seems so mysterious, part of its appeal is simply that it works. However, if compatibility on the Web falls apart, any illusion is lost, the failings of a site are exposed, and its elegance and beauty evaporate.

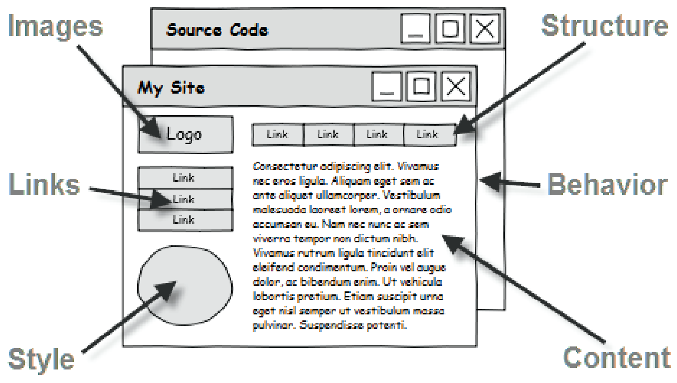
Images
Source Code
Structure
My Site
Logo
Link
Link
Link
Link
Links
Link
Link
Link
Behavior
Consectetur adipiscing elit. Vivamus nec eros ligula. Aliquam eget sem ac ante aliquet ullamcorper. Vestibulum malesuada laoreet lorem, a ornare odio accumsan eu. Nam nec nunc ac sem viverra tempor non dictum nibh. Vivamus rutrum ligula tincidunt elit eleifend condimentum. Proin vel augue dolor, ac bibendum enim. Ut vehicula lobortis pretium. Etiam suscipit urna eget nisl semper ut vestibulum massa pulvinar. Suspendisse potenti.
Style
Content

FIGURE 1-13: The average website contains many variables.

In the past, bugs in browsers have caused a number of problems for designers. However, because of the limited array of devices and circumstances in which sites were regularly viewed, issues related to a damaged experience had a minimal impact (you could fix a problem by throwing hacks at it, often resolving early browser bugs). So, the problems of maintaining a stable and ubiquitous site have always existed; it's just that the method of interaction and content absorption has dramatically changed in the visitor's favor.

Many site issues could be labeled as critical (severe), moderate, and mild, but I'm not one to place labels on quirks because what may appear as an inconsequential error to one individual may well be a catastrophic, game-changing bug for another! An essential part of designing your sites to be durable on many different devices (and within many different situations) is that you really focus on identifying flaws and solving them. Often, designers get so caught up in the debate about best practices and ideals that they may lose sight of the bigger picture.

When a problem occurs, don't conclude that you must attack it in a "nuclear-warhead" fashion. Actually, it's rare for a site to be in such a poor state that users' online experiences become totally inaccessible. Often, the quirks and issues are mild enough to simply cause irritation or confusion. On the other hand, don't put off implementing a flexible design just because it seems trivial to you, even if the fix for that quirk could have a similar effect!

You can debug code in many different ways. You have concurrent debugging that effectively involves checking and retesting your work as you progress through a series of stages. There's elimination debugging, where you find a problem and begin crippling bits of code in order to eliminate potential causes from the list (what Sherlock Homes would do if he were a programmer). Finally, there's proactive testing in which you uncover quirks or issues (plus the effectiveness of fixes) by getting users to report faults (Figure 1-14).

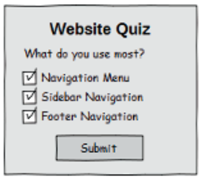

FIGURE 1-14: Proactive testing can consist of structured usability studies or simple verification tests.

Uncovering problems can involve as much investigation as any detective novel does. Throwing a site into all sorts of unique situations can also help you understand what users may be seeing on their screens and in the case of really old technology, that view can turn a classic detective novel into a horror franchise. Methods of testing vary among developers, and you probably have your own style. Just remember that you'll probably need to spend a good deal of time proactively trialing out use cases.

Cultivating customer service

If you don't have visitors to your site, it will, of course, fail. If your site isn't accessible or usable in its current form, you will lose visitors. The Web is a beacon of ubiquity and universal access, and as its facilitator and representative, your job is to help ensure that this beacon remains a reliable one over time. By empowering users with the tools they need to engage with your site, you're likely to see an increase in user activity. Also, if more people can access and use your services effectively, they're more likely to pass the link on to their friends.

All of this discussion centers on the importance of understanding your users and their needs. Also, it centers on looking beyond how individual pages or page elements appear to you and considering what such features will look like to different users, whether they're using popular tools or ones with little recognition. Your selection of features to implement will depend on necessity, so calculating your options may involve working out a "cost versus reward" ratio, prioritizing upgrade release cycles, and keeping visitors informed while engaging them in this process.

Note

An example of testing by necessity includes, for example, the justification for ensuring that a site works in older versions of Internet Explorer. Although we'd happily like to see them evaporate from the earth's atmosphere, they often stick around for a long time!

Involving your clients in the design process has never been more important. Because their satisfaction is central to achieving popularity and widespread use of the site, you'll want to take every opportunity you can to obtain meaningful feedback, useful assistance, and potentially groundbreaking ideas. The key to gaining useful feedback is trust. Be ready with open ears and mind and be honest and transparent with your processes. It also pays to thank users when they get in touch with you, even if it's uncomplimentary.

Don't be afraid to ask a community for help in testing your work or letting you know what things they'd like to see in future versions of your site. Feedback of a negative nature may seem like a miserable way to spend a day of inbox catching-up, but it's often the less-flattering stuff that has the biggest impact. If everyone says that your work is perfect, be suspicious; if some send hate mail, don't take it personally — see it as a chance to improve. Users matter in the design process, so don't neglect this mighty resource.

Here are some ways you can initiate communication with your visitors:

> Direct methods like e-mail, instant messaging, and chat rooms.
>
> Indirect methods like forums, feedback systems, or bug trackers.
>
> External solutions like social networks and review websites.

Of the different ways that you can gauge this feedback, your two primary sources of useful data will come from *quantitative* (numbers and statistics) and *qualitative* (descriptive and opinionated) research. Measuring this data is a challenge, but the benefits that they bring include faster identification of flaws and ideas to help you make your site more flexible. If users want 3D video, for example, you can implement it.

Ultimately, as with any type of community involvement, there will be disputes, and not everyone will agree on every action. Just know that, as you test to ensure compatibility and durability, you'll encounter a few bumps along the way. Finding a happy medium is something many designers do in their daily jobs with clients, users, and each other. Because going for broke and leaving users to their own devices aren't acceptable options, compromising (that is, going for adequate rather than optimal solutions) is a satisfactory alternative.

The Web: Survival of the fittest

The modern Web presents many challenges. With competitors breathing down your neck, a layout that fails to work on a range of devices and hardware, such as the simple ones of the laptop shown in Figure 1-15, represents a missed opportunity. Sites are like children, needing lots of care and attention. Nurturing unique devices will encourage return visits, and providing enough education to ensure users can react appropriately in current situations helps, too. In this ever-demanding environment, you want to think in terms of taking small steps toward achieving long-term goals.

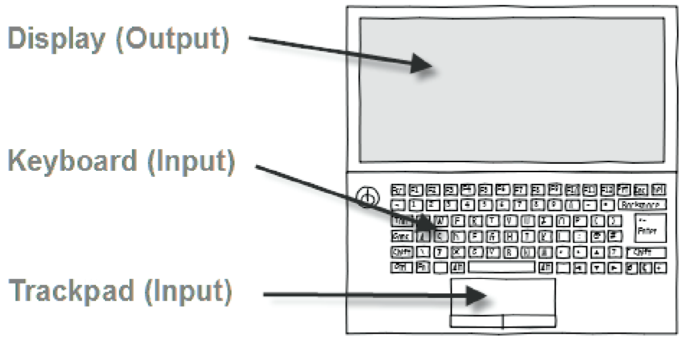

FIGURE 1-15: Devices contain input and output hardware, and your layout must work with both.

Design based on the experiences of users is an expanding area of interest, particularly with the new ideas that are pushing us toward a more sociable and useful Web. Survival isn't just a matter of ensuring that your site looks great via Internet Explorer, Firefox, Chrome, Safari, and Opera. Being future-proof depends on the devices, hardware, software, standards, and consumer variables explored in greater detail throughout this book. So, whatever you do, be sure to stay up with the times with updated and feature-rich layouts.